Antti Peltonen

Application Delivery Controller Implementation to Cyberlab Data Center

Bachelor’s Thesis

Information Technology

May 2016

KYAMK
University of Applied Sciences

Tekijä/Tekijät	Tutkinto	Aika
Antti Peltonen	Tietotekniikan insinööri	Toukokuu 2016

Opinnäytetyön nimi

Application Delivery Controllerin implementointi Cyberlab datakeskukseen

43 sivua

Toimeksiantaja

Kymenlaakson ammattikorkeakoulu

Ohjaaja

Lehtori Vesa Kankare

Tiivistelmä

Palveluiden häiriönsietokyky, saatavuus ja turvallisuus ovat olennainen osa tämän päivän tietoverkkoympäristöä. Siksi Application Delivery Networking teknologiasta on tullut yrityksille korvaamaton työkalu parantaa juuri näitä ominaisuuksia palvelinympäristössä. Application Delivery Controller laitteet ja ohjelmistot tarjoavat yrityksille monia eri ominaisuuksia räätälöidä niiden tarjoamat palvelut juuri itselle sopiviksi.

Opinnäytetyön tarkoituksena oli asentaa F5 BIG-IP ADC:t Cyberlab palvelinhuoneeseen ja testata niiden kuormanjako-ominaisuutta. Tehdyn työn lisäksi tämä opinnäytetyö antaa tietoa kuormanjaosta ja sen eri tavoista, ADC:stä ja sen tarkemmista teknologioista sekä F5:stä.

Työssä käytettiin F5:n virtuaaliversiota 11.6 ohjelmistosta sekä 4000-sarjan laitteita. Virtuaaliversiolla perehdyttiin F5-ohjelmistoon ennen laitteiden saapumista. Virtuaaliversiota ajettiin VMware Workstationista ja koneen verkkokortteja käytettiin simuloimaan ulkoverkkoa sekä sisäverkkoa. Sisäverkon puoli oli kytketty yhteen tietokoneeseen, joka toimi palvelimena ja ulkoverkon puoli luokan verkkoon. Tässä järjestelyssä oli yhteensopivuusongelmia palvelimen ohjelmisto kanssa, mutta antoi kuitenkin kuvan siitä, miten konfiguroida kuormanjakoa sekä SSL:ää. Laitteiden saavuttua ne asennettiin palvelinhuoneeseen ja konfiguroitiin tukemaan etäyhteyttä. Jotta F5:t saivat yhteyden palvelimiin, oli VMware vSphereen tehtävä muutoksia. Muutosten jälkeen F5:t konfiguroitiin käyttämään kuormanjakoa.

Lopuksi koko verkko testattiin tekemällä palvelunestohyökkäyksiä F5:een

Työ saatiin tehtyä pisteeseen, joka vaadittiin kyseisenä ajankohtana. Kuormanjako saatiin toimimaan onnistuneesti.

Asiasanat

ADC, kuormanjako, virtuaali server

Author (authors)	Degree	Time
Antti Peltonen	Bachelor of IT engineering	May 2016

Thesis Title

Application Delivery Controller Implementation to Cyberlab Data Center

43 pages

Commissioned by

Kymenlaakso University of Applied Science

Supervisor

Vesa Kankare, Senior Lecturer

Abstract

Service resiliency, availability and security are essential in today's network environments; that is why Application Delivery Networking technology has become an invaluable tool for companies to better these qualities in their server environments. The Application Delivery Controller equipment and software offer many different customization options for companies to tailor them exactly for their needs.

The purpose of the bachelor's thesis was to install F5 BIG-IP ADC in the Cyberlab server room and test the load balancing features of the equipment. Along with the work done, the thesis offers information on load balancers and load balancing techniques, ADCs and ADC specific technologies and F5.

The work was done by using F5 BIG-IP virtual version 11.6 and 4000 series equipment. The virtual version was used to get familiar with the F5 software before the equipment arrived. The virtual version setup consisted of the software running in VMware workstation and using the host PC's network cards to simulate outside and inside VLANs. The inside was connected to a PC that was used as a server and the outside was connected to the classes network. This setup had some interoperability issues with the server software but offered some insight as how to configure load balancing and SSL. After the equipment arrived they were installed in the server room and configured to support remote session. Before the F5 could get connection to the servers, there was a need to make some changes in the VMware vSphere server environment. After the changes in the vSphere, the F5s were configured for load balancing.

Last, the whole network was stress tested by doing DDoS attacks against the BIG-IP.

The work was completed to the point that was needed in the environment at the time with working load balancing. SSL and further optimization need to be implemented in the future.

Keywords

ADC, load balancing, virtual server

Table of Contents

LYHENNELUETTELO

ADC	Application Delivery Controller
ADN	Application delivery network
CPU	Central processing unit
DDoS	Distributed denial of service
DSwitch	Distributed Switch. Virtual switch used inside the VMware vSphere web application
Gb	Gigabit
HTTP	Hypertext transfer protocol
IP	Internet protocol
KVM	Kernel-based Virtual Machine. Full virtualization solution for Linux on x86 hardware
MAC	Media access control
NAT	Network address translation
NIC	Network interface card
OS	Operating System
OSI-model	Open System Interconnection model
PC	Personal computer
SSL	Secure Socket Layer. Cryptographic protocol family
SFP	Small form-factor pluggable
TCP	Transmission control protocol
VLAN	Virtual local area network

1 INTRODUCTION

The idea for this bachelor's thesis came from the need for having strong server side protection for the Cyberlab game and a need to scatter the traffic to multiple servers. The machines that are specialized in this kind of behavior are the F5 BIG-IP equipment and software family. Two of the 4000 series machines were bought to do the work. They needed to be installed and configured for the lab environment.

Before the equipment arrived, the preliminary familiarization was done with the virtual version of the 11.6 software that was running in VMware Workstation. Even though the virtual version had full graphical support, it was not easy trying to configure it because of no earlier experience with F5 products. Luckily, the instructor had enough knowledge to get started with the configuration.

After getting more familiar with the software, a goal was set to make a virtual pool, and put a game server behind that so that it was possible to ping it from some client machine. After the ping test was completed, SSL-offloading was to be tested.

After the physical machines arrived, the real work started. The equipment was installed in the Cyberlab server room and configured relatively the same way as the virtual version. Because the virtual servers were running in VMware vSphere, there was a need to make some configuration changes to there.

After the vSphere was configured, the client got a connection to the servers. The project remained in this state for a while, until the whole Cyberlab network was stress tested.

2 CYBERLAB PROJECT

2.1 Project overview

The idea of the project is to create an environment where game design, cyber security and data center knowledge work together to create a bigger environment, what is called Cyberlab. It tries to mimic what is done in real business server environments and at the same time, offers a learning platform for cyber security and tries to constantly evolve through different kinds of tests and analytics.

The learning part of the project is done with gamification. It consists of game platforms created in the data center, where the players have a hacking OS and a target network or system. The goal is to try and get information out of the target OS and score points as set by the game designers. This could mean that getting inside the target OS or finding some relevant information through hacking and so on, would score you points.

Because the platform is built this way, there is a possibility that someone might try to break into the system itself. This is can be beneficial if the game environment is built rigid enough not to get hacked easily. Thus it can be used to determine and analyze what kind of hacking attacks are being done. The environment is mainly used for the IT-branch of studies as of now, but can be made to accommodate the needs of other branches of study as well.

2.2 Game environment architecture

The operating systems run in KVM host, which in turn is created in vSphere server cluster, as seen in figure 1*.* The users make connection to the servers through F5 BIG-IP ADC, which handles SSL-offloading, load balancing, access policies and general information security, illustrated in figure 2*.*

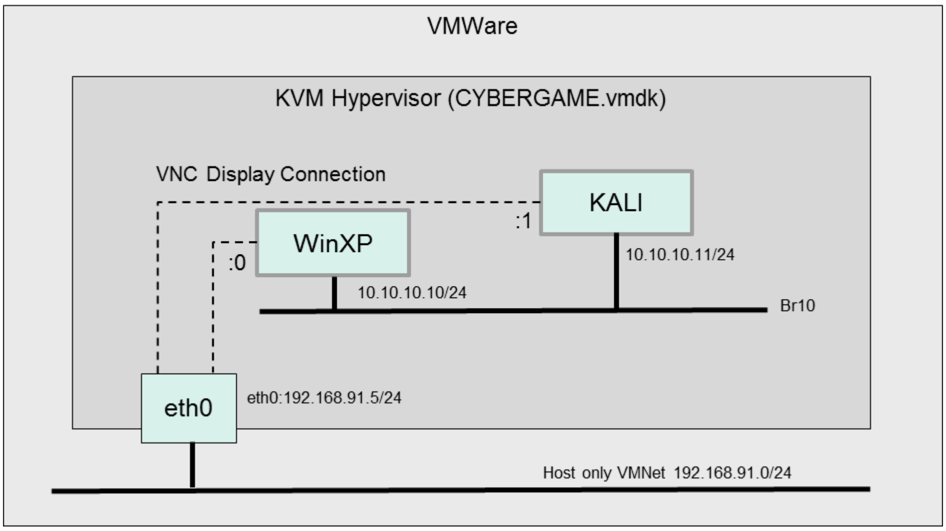

Figure 1. Game nest

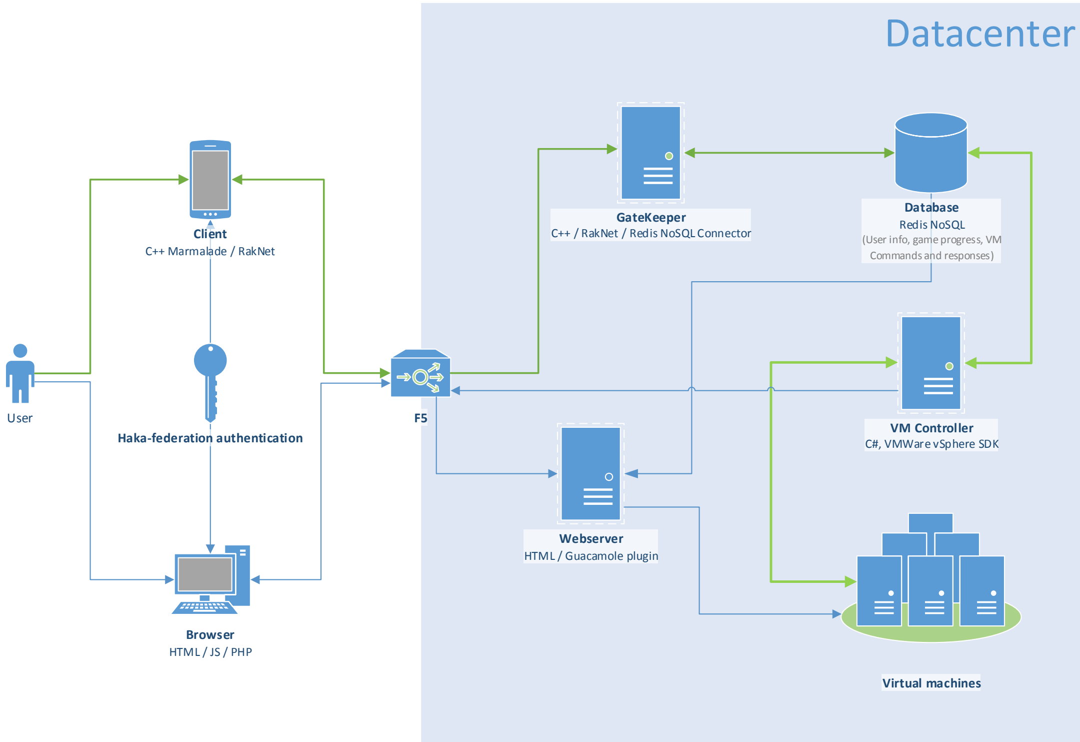

Figure 2. Cybergame connection architecture

3 LOAD BALANCING

Load balancing in network environment is a technique to distribute incoming traffic to multiple servers. It has become an important factor in today's internet architecture, as it is essential in server environment, where large numbers of users are trying to connect with the same content running in the servers. To provide all users continued access, there needs to be some sort mechanism to know when a server cannot accept any more connections and has to route the traffic to another server. The traffic can be distributed using many different styles.

3.1 Evolution of load balancers

The first load balancing solutions for content servers used the cluster-IP, as seen in figure 3. This meant that even though the individual servers in a cluster had their own IP-address, the client would connect to the cluster-IP. The selection as to which server the client would connect, would be determined by which of the servers responded to the request the fastest. The OS in the servers was responsible for the load balancing. (Salchow 2012.)

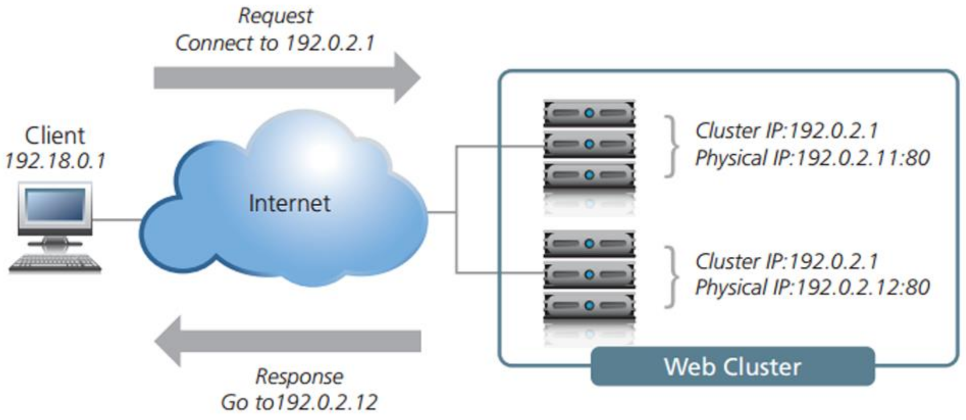

Figure 3. OS-based load balancing and cluster IP (Salchow 2012)

This kind of solution handled the load balancing well, if there was a small amount of clustered servers. When the server count rose to about 5 to 10, the communication between the servers started to impact the client connection. Also, if there happened to be any kind of disturbance in the solution used, it would impact the whole application and network that was running under it. (Salchow 2012.)

A step up from the OS based load balancing were the network-based load balancing hardware. These resided outside of the servers themselves and did not depend on a purpose built application, as seen in figure 4. (Salchow 2012.)

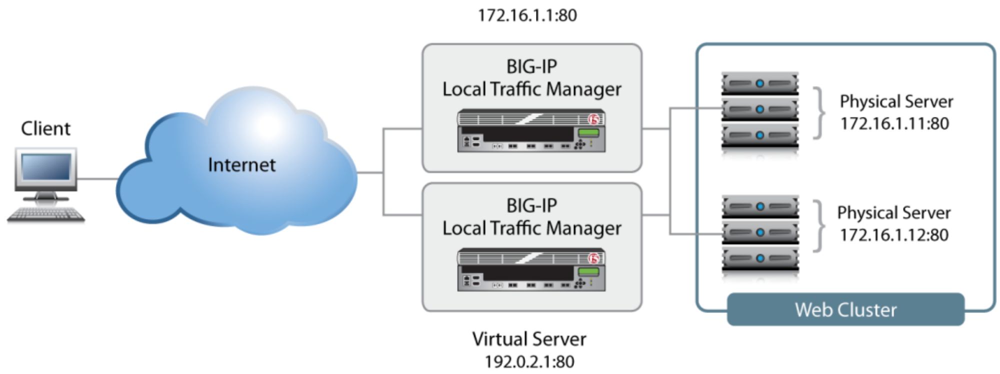

Figure 4. Network-based load balancing (Salchow 2012)

For the client to server connection the machines used virtual server addressing, similar as to how it is done by today's ADCs. The virtual server is created in the load balancers and is given its own IP-address to which clients are connecting. The load balancer then forwards the traffic to the servers, which are allocated to its use. Because the load balancers were now outside the servers themselves, it lessened the burden on the servers and the network. (Salchow 2012.)

High availability was strengthened simply because of the simpler solution. Also predictability became a new variant, because now that the equipment was outside the software itself, the traffic behavior could be monitored using response times, connection load and other variables. (Salchow 2012.)

By analyzing this information, the load balancing could be customized for the needs of the company, for example, a more heavily utilized server does not receive as many connections as another server that has lesser load, based on server response time. The ADCs are heavily based on these designs, so they can be called the founding fathers of the ADCs. (Salchow 2012.)

3.2 Load balancing methods

Round Robin

Round Robin load balancing sends the client request to all servers in a list like manner; first on the list gets it first, the second after that, and so on. The first server to announce it is available, becomes the active server for the client and is moved to the bottom of the list. (MacVittie 2009.)

This is easy to implement but has drawbacks. It assumes that all the servers on the list are up and have exactly the same load, storage and computing capacity. Round Robin has some variants that make it more viable solution to use. (MacVittie 2009.)

In Weighted Round Robin, the servers are assigned a weight number. The weight number indicates how the client requests are distributed among the servers according to assigned parameters, for example, 5:3:1 relation with three servers. This means that the server assigned with 5 gets the most of the requests, and so on. (MacVittie 2009.)

In Dynamic Round Robin, the system dynamically assigns the weight, based on a variety of parameters gathered from the real time data of the servers. (MacVittie 2009.)

Random

As the name implies, Random load balancing method directs the traffic to a random server on the list using some kind of randomized algorithm and does not use any performance or status information to choose which server the traffic is forwarded to. Not very viable in today's high demand, high availability server environments, because of the simplicity of the design. (MacVittie 2009.)

Fastest

Fastest option uses the server respond times to decide, as to which server to forward the traffic to; the fastest answering server is used for the client. It can be useful in situations, where the servers are scattered in different locations or networks. Its drawback is that a single response does not mean that it is the fastest after the response given and can so lead to congestion. (MacVittie 2009.)

Least Connections

Least Connections distributes the traffic to a server that has the least amount of connections. Works well, when the servers are all the same type; memory, computing capacity, and so on. It can cause congestion, because it does not know how long the connections are. For example, a single web page request versus multiple database queries. (MacVittie 2009.)

Observed

Observed distributes the traffic to servers based on the combined values of least connections and fastest. The server with least connections and fastest response time is ranked the highest and gets the traffic. Can cause servers to be overloaded, because it does not know what kind of workload the server is going to be having after accepting the connections. (MacVittie 2009.)

Predictive

Uses the same ranking method as observed, but also checks, if the server performance is changing. If the server performance is improving, it gets the traffic and vice versa. (MacVittie 2009.)

4 APPLICATION DELIVERY CONTROLLER

ADCs are the next evolution of load balancers and SSL accelerators. They are platforms that do not only deliver load balancing features, but application acceleration, security and resiliency. They are very beneficial for businesses, which stretch over wide areas and are not confined in single location. In this kind of scenario, there could be multiple networks, protocols and restrictions working in each branch of the enterprise. The ADC can be made to accommodate the needs of every branch and optimize the traffic for each of them, while retaining the confidentiality, integrity and maximum availability of the data. Also if the enterprise has multiple data centers and one of them is experiencing congestion or goes down, the traffic can be forwarded to another data center even if it is located in another geological location.

In these kind of equipment there are a multitude of analytics and monitoring tools for optimizing the traffic and isolating different kinds of problems that might arise. For example, a service might be experiencing some lag for unknown reason. By using the tools to benchmark the service and system before, the problem can be found much faster by comparing the data with the benchmarked data and thus keep the system available and healthy for the users.

4.1 ADC used technologies

Load balancing

Load balancing is widely used and essential for ensuring availability of services to customers and workers. See Chapter 3 for more information on load balancing techniques.

Inherent connection proxy and threat mitigation

Because the equipment resides in front of the main system, they add another layer of security for it. They can also, depending on the system, check the connection for any malcontent at any level of the OSI-model, as shown in figure 5.

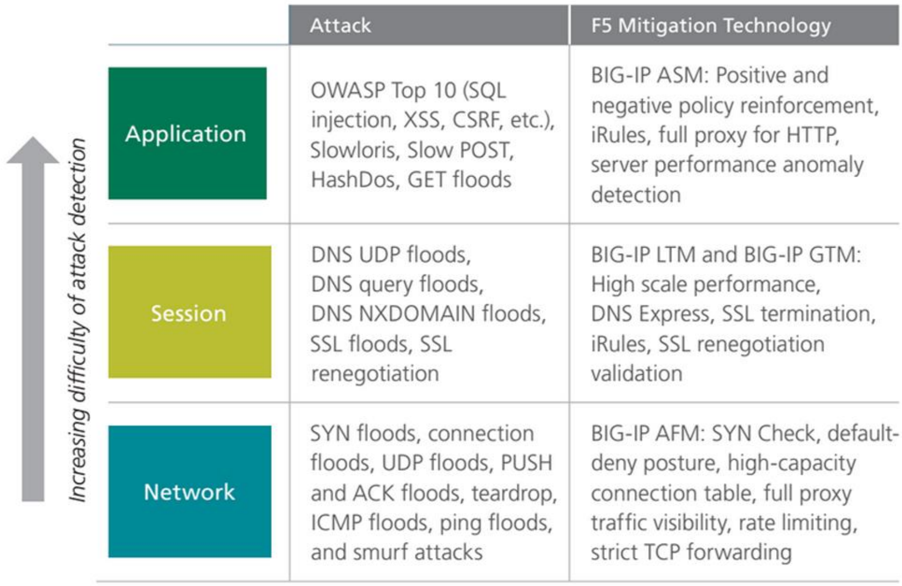

Figure 5. Overview how F5 ADCs cut the connection and can mitigate threats (Holmes 2013)

Server task offloading

The ADCs can be used as a front end for the servers to offload different kinds of server intensive tasks.

SSL-offloading/acceleration is a technique to terminate encrypted data transmission before passing the traffic to the content server, as seen in figure 6. SSL-encryption and decryption are very intense for the CPU of the server, so making the ADC do the work for it helps ease the work load of the server (KEMP).

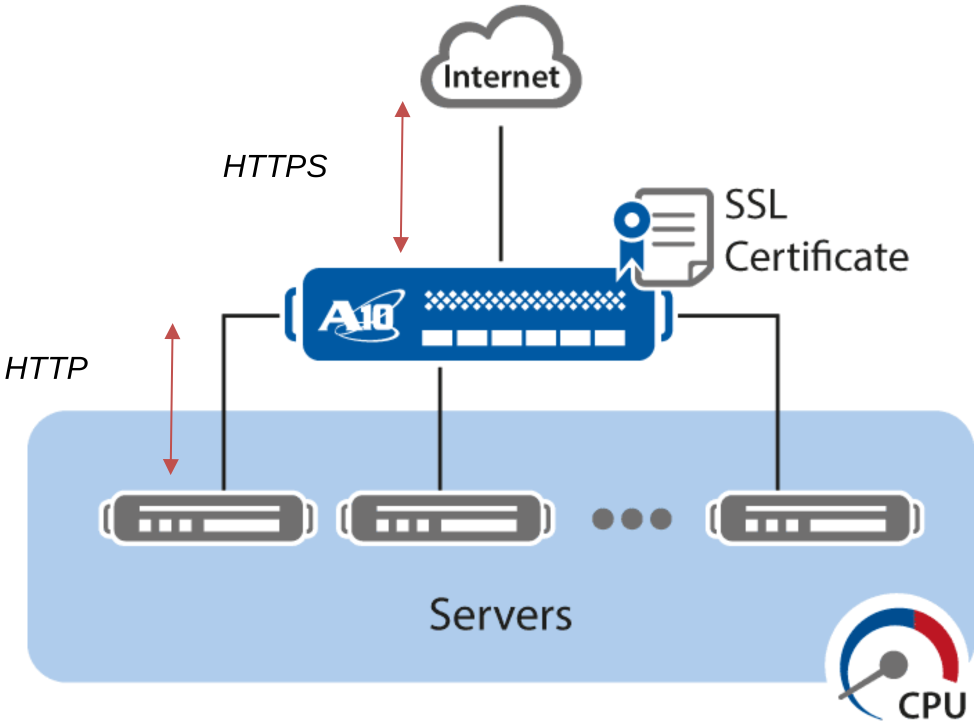

Figure 6. SSL-offloading (A10 Networks)

Before ADCs were integrated with this function, a dedicated CPU was handling the offloading task. This option was rather expensive, since there had to be one card for each server and its only function was to perform SSL. (KEMP.)

Then came the SSL accelerators, which operate in the same area as ADCs. They would be in front of the servers decrypting and encrypting the traffic. These proved to be very successful and were later paired with load balancers. (KEMP.)

TCP multiplexing is a technique to reuse non active TCP connections, as seen in figure 7. This helps to improve server capacity and performance since they do not have to close the ended connections and open new ones, but can use the open connections for other tasks or users. This can also be beneficial in design perspective as the servers can be made to take less connections than without multiplexing, thus having more resources for more important tasks or having less servers to run the hosted applications. (MacVittie 2008.)

Figure 7. TCP multiplexing (MacVittie L 2008)

Advanced services

Advanced firewall

Ensures data availability, integrity and confidentiality, using methods such as network session tracking, application awareness, attack mitigation and DDoS protection. (F5 Networks Inc. n.d.b.)

Access policy manager

Differentiates users as to which resources they can access using user and group identities; such as location, device type and security posture. (F5 Networks Inc. n.d.a.)

Application acceleration

Optimizes services and content by using HTTP-offloading and data caching, image downsizing, file size reduction and more. (F5 Networks Inc. n.d.c.)

Secure web gateway

Works together with access policy manager to ensure maximum access to services and data, while keeping the assets healthy from malware and other web-borne threats using user access management, data integrity checks with Websense and configure actions taken after an attack or infection is noticed. (F5 Networks Inc. n.d.f.)

These are just a few advanced services that the companies' products offer. There are as many solutions as there are companies.

5 F5 NETWORKS

F5 is the market leader in ADN services (F5 Networks Inc. n.d.e). Its knowledge, innovation and resources differentiate it from its competitors (Fabbi&Lerner 2014).

F5 offers hardware and software solutions for companies' data, video and voice requirements. Its products include BIG-IP ADC hardware family, which offers load balancing, advanced firewalls, carrier grade NAT and many more solutions for companies to fully customize the machines for their needs. (F5 Networks Inc. n.d.e.). Their Silverline products offer cloud based solutions, like DDoS protection on demand and web application firewall (F5 Networks Inc. n.d.d.).

The company was founded in 1996 and has grown since to become a multi-million company boasting over 65 million dollars in shares and 2.3 billion dollars in assets. (F5.)

6 GETTING FAMILIAR WITH F5 SOFTWARE

Familiarization for the coming implementation of the equipment started with virtual version 11.6 of the F5 BIG-IP platform which ran over VMware Workstation. The first task was to get a server and the F5 see each other.

Before starting to configure the F5, there was a need to make changes and create new networks to VMware Workstation. The outside interface was bridged to the PC's main NIC (VMnet0), inside to a newly installed NIC (VMnet1) and the management interface used NAT to communicate with the host PC (VMnet2), as seen in figure 8.

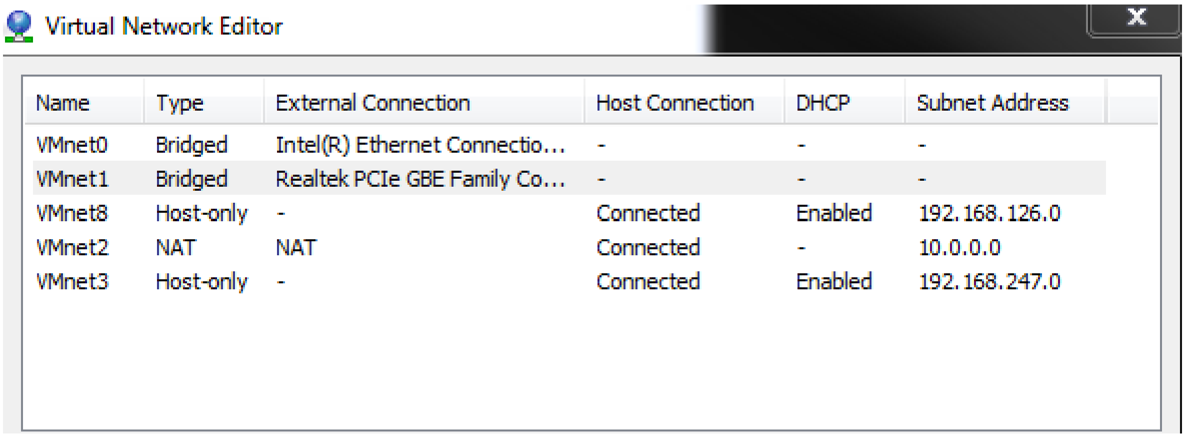

Figure 8. WMnet configuration

After that, the created VMnets were paired with the interfaces of the virtual machine. Their MAC addresses needed to be checked so that they were the same in VMware and in the F5, as shown in figures 9 through 11.

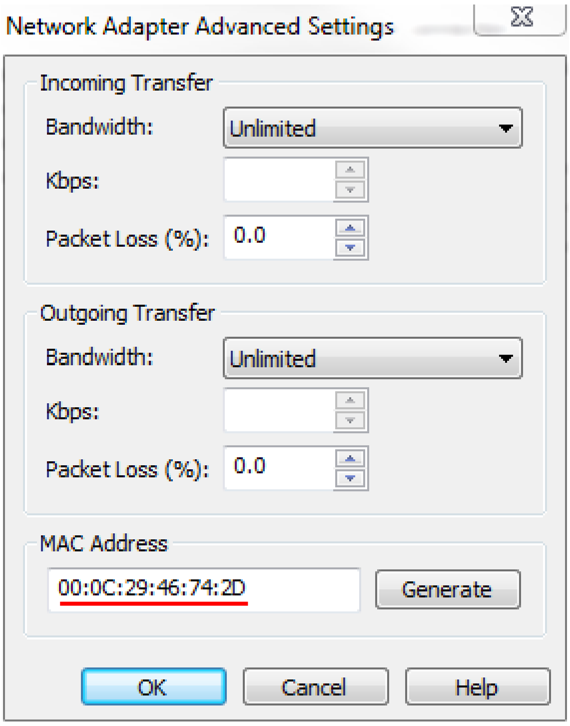

Figure 9. Checking MAC-address from VMware

```
[root@localhost:INOPERATIVE:Standalone] config # ifconfig
eth0      Link encap:Ethernet  HWaddr 00:0C:29:46:74:2D
          inet addr:10.0.0.50  Bcast:10.0.0.255  Mask:255.255.255.0
          inet6 addr: fe80::20c:29ff:fe46:742d/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:131 errors:0 dropped:0 overruns:0 frame:0
          TX packets:92 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:15077 (14.7 KiB)  TX bytes:7032 (6.8 KiB)
```

Figure 10. Making sure the F5 MAC-address is the same as in VMware

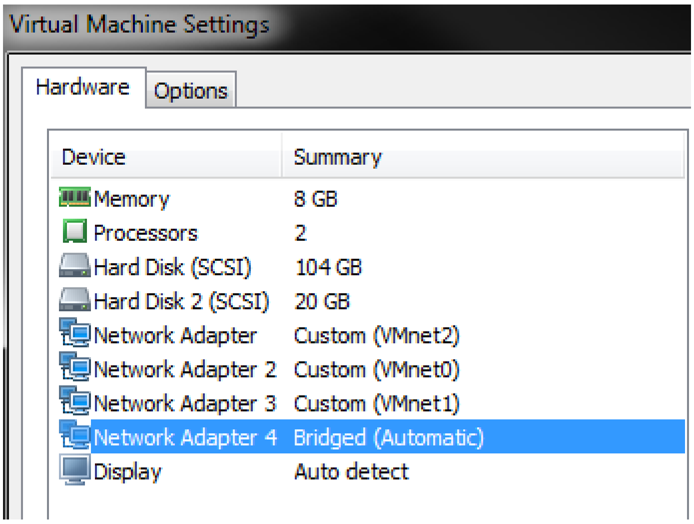

Figure 11. Network adapter alignment

Next step was to try and configure the software so, that it could get connection to a server. This proved to be rather difficult because of no earlier experience with F5 products, so a lot of trial and error was used, but finally the connection was made with the help from the instructor.

First, the inside VLAN, Self IPs and an outside route were created, as shown in figures 12 through 14.

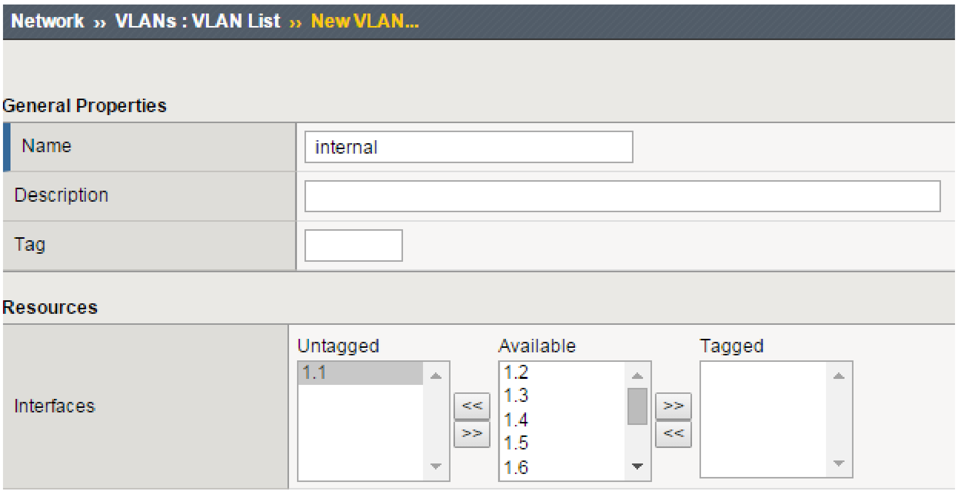

Figure 12. Creating VLAN

inside_1	192.168.0.1	255.255.255.0	internal
inside_2	192.168.0.2	255.255.255.0	internal
outside_1	10.0.0.2	255.255.255.0	external

Figure 13. Self-IP addresses

Network » Routes » New Route...

Properties

Name	GW_outside
Description	
Destination	0.0.0.0
Netmask	0.0.0.0
Resource	Use Gateway...
Gateway Address	IP Address 10.0.0.1
MTU	0

Figure 14. Configuring outside route

Then, a pool and virtual server were created for the server, as shown in figures 15 and 16.

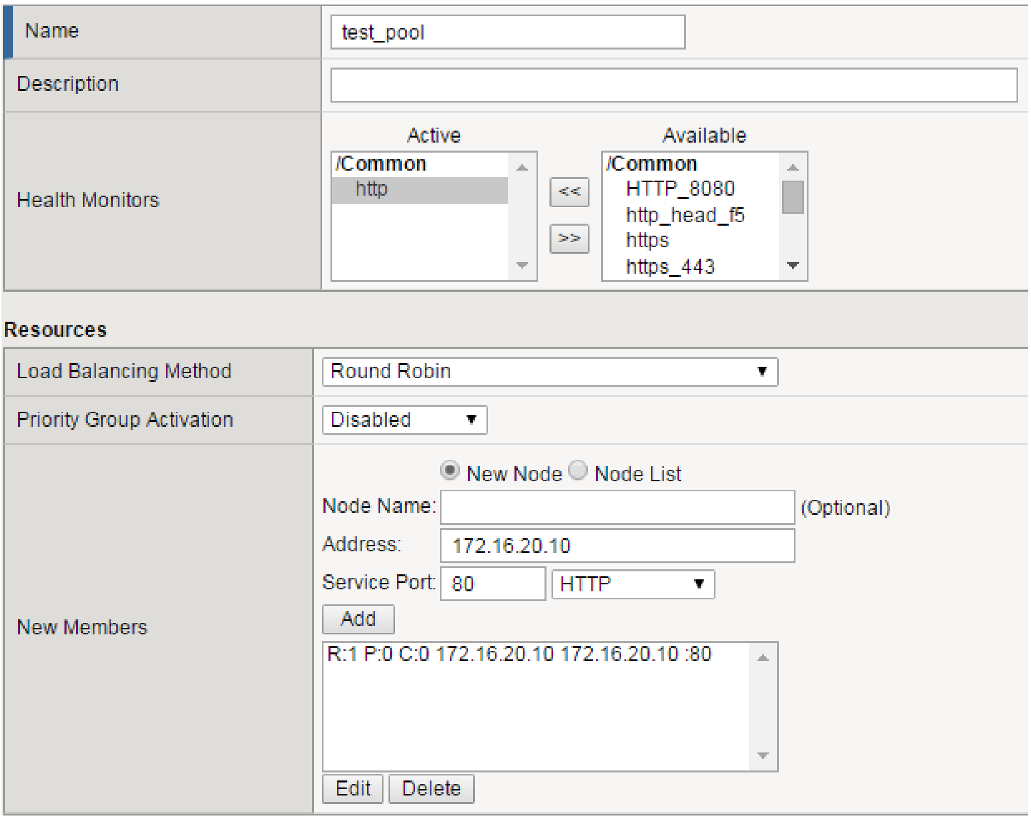

Figure 15. Creating test pool and adding a node

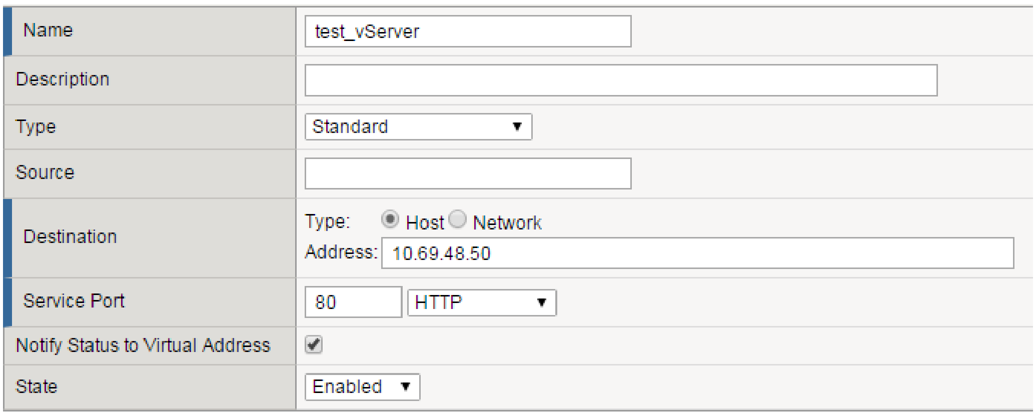

Figure 16. Creating virtual server

Last, the configuration was tested by pinging the inside VLAN server from an outside VLAN endpoint.

7 IMPLEMENTING BIG-IP TO THE CYBERLAB

The goal was to build an environment to test the load balancing of the BIG-IP. It consisted of an end machine, which simulated the user connecting from internet, the F5s, which load balanced the traffic, and the servers, which the user was trying to connect to, as seen in figure 17.

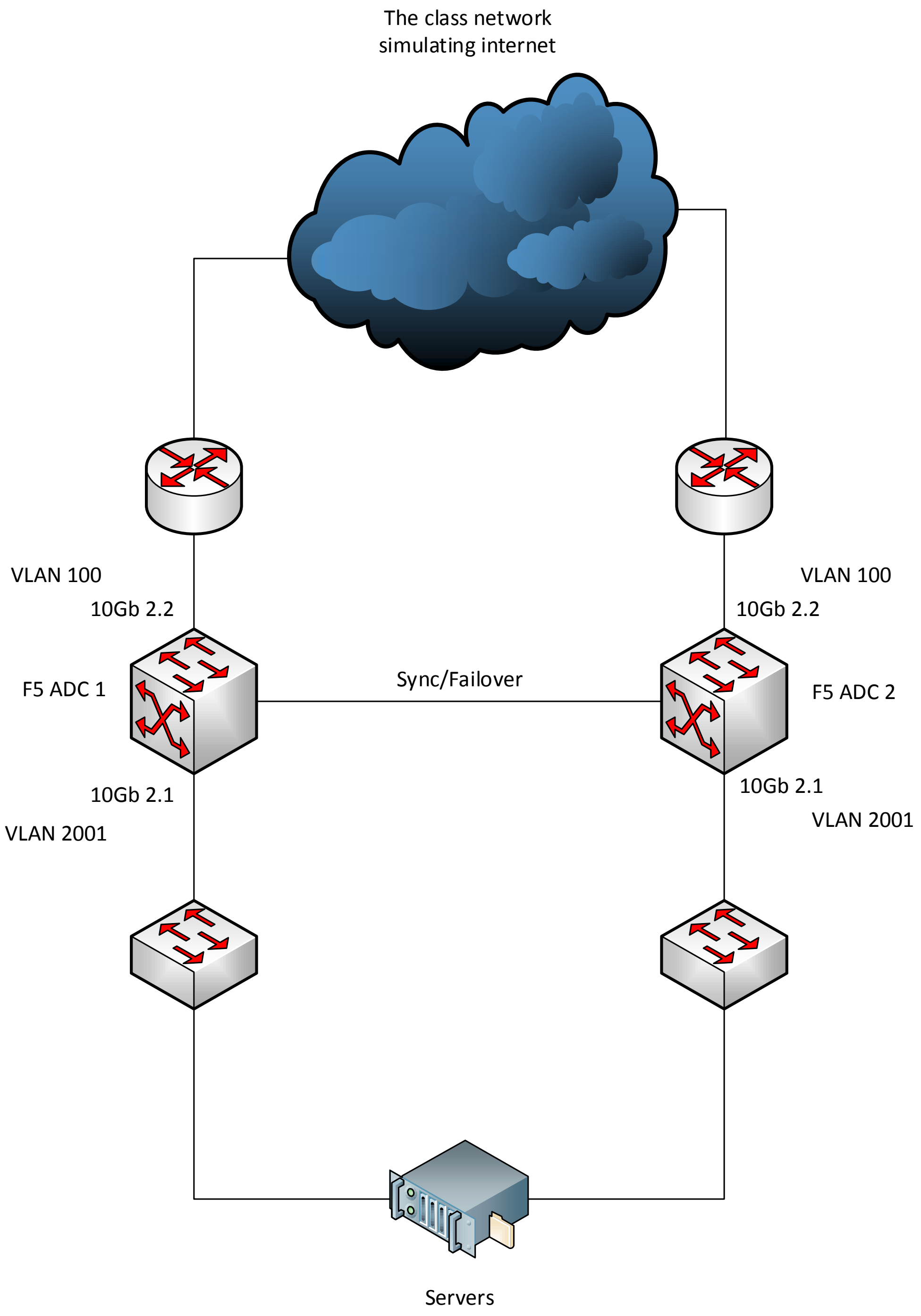

Figure 17. Project topology

7.1 Installation and initial configurations

The work started by installing the power supplies to the equipment and mounting them to the server rack. The installation was easy because the rack supports were a tool-less model. Next the default management IP-addresses were changed to temporary ones for access to the machines remotely.

The cables used were SFP copper and the interfaces that were used for the client and server connection were 10Gb ports. 2.2 ports were connected to Cisco ASR 900 routers which marked the outside and 2.1 ports were connected to a Dell switch, which forwarded the traffic to the inside network. Also the sync cable that came with the package was connected between the two F5 devices.

After getting control remotely, the machines were run through the initial configuration wizard which was prompted automatically when logging in the first time. There, the Self IPs, VLANs and outside routes were configured according to the laboratories IP-addressing.

VLANs were tagged; inside was tagged with 2001 and outside with 100.

7.2 Configuration sync

After the initial configuration was ran through, the machines were made to be in sync. This was done from the Device Management. First, a peer needed to be added. This needed the IP-address of the peer device and the configured credentials, as seen in figure 18.

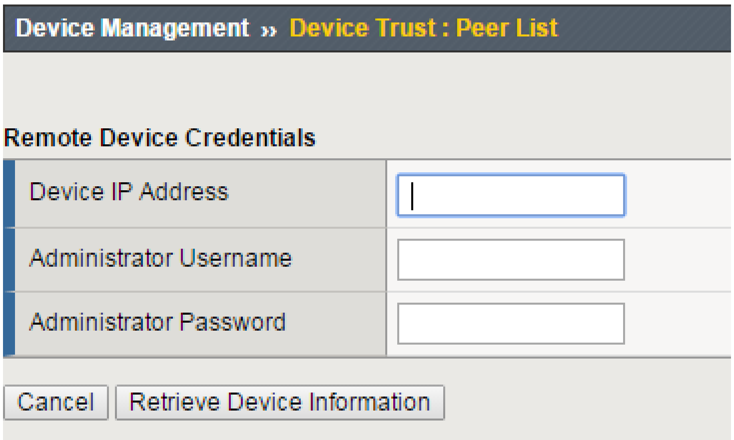

Figure 18. Adding a peer device

Next step was to configure the network failover addresses, as shown in figure 19.

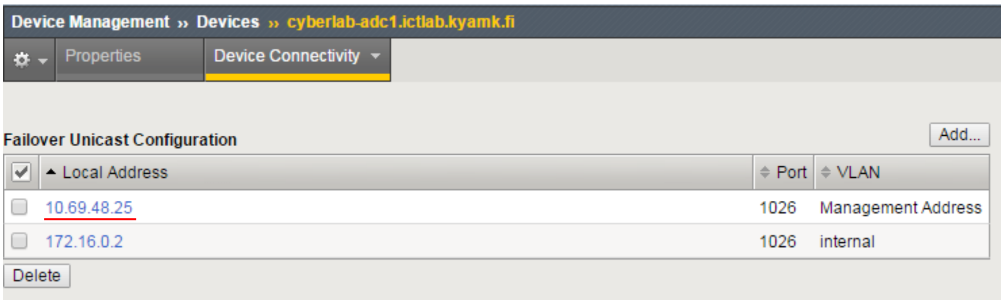

Figure 19. Adding the needed failover addresses

Because the devices said they were in sync, they were left as they were. In reality this was not the case, because the network failover options and group syncing were not configured properly. The full syncing was done at a later time, as shown in figure 20, which resulted in the devices becoming properly synced, with the other device being the primary one and in active mode and the other secondary and in standby mode.

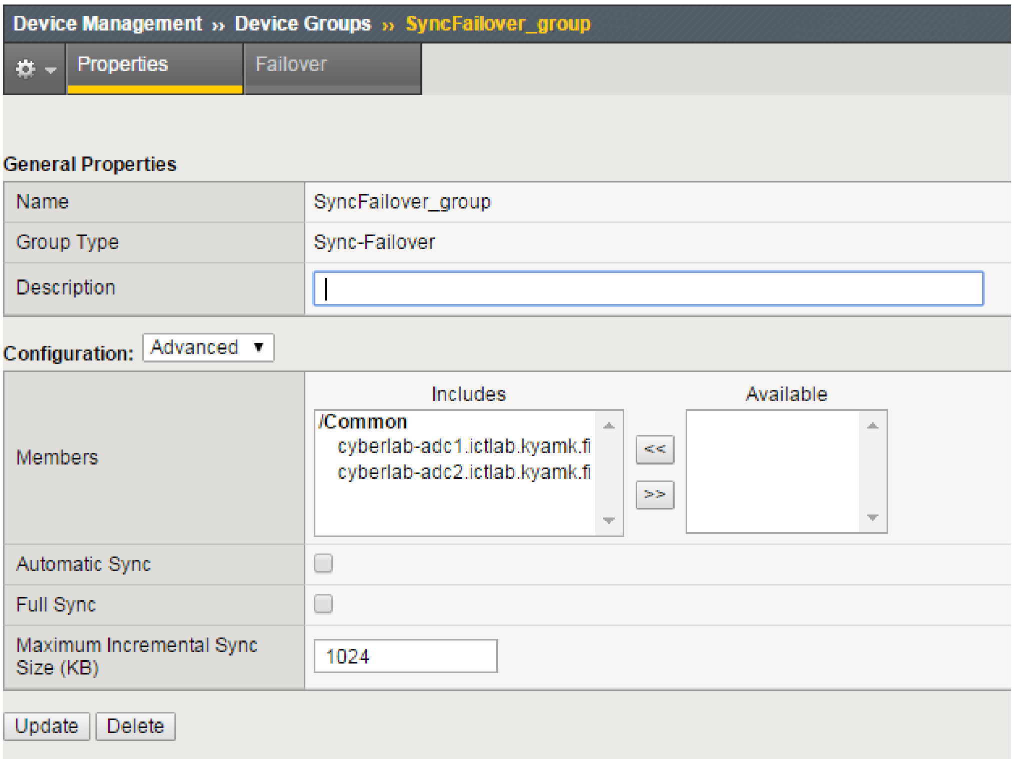

Figure 20. Device group configuration

Finally, the configured device was made to sync its configuration to the group and overwrite the configuration of the other device, as seen in figure 21. This is a step that needs caution, because if careless, it can be made to the sync configuration of a device that is not wanted

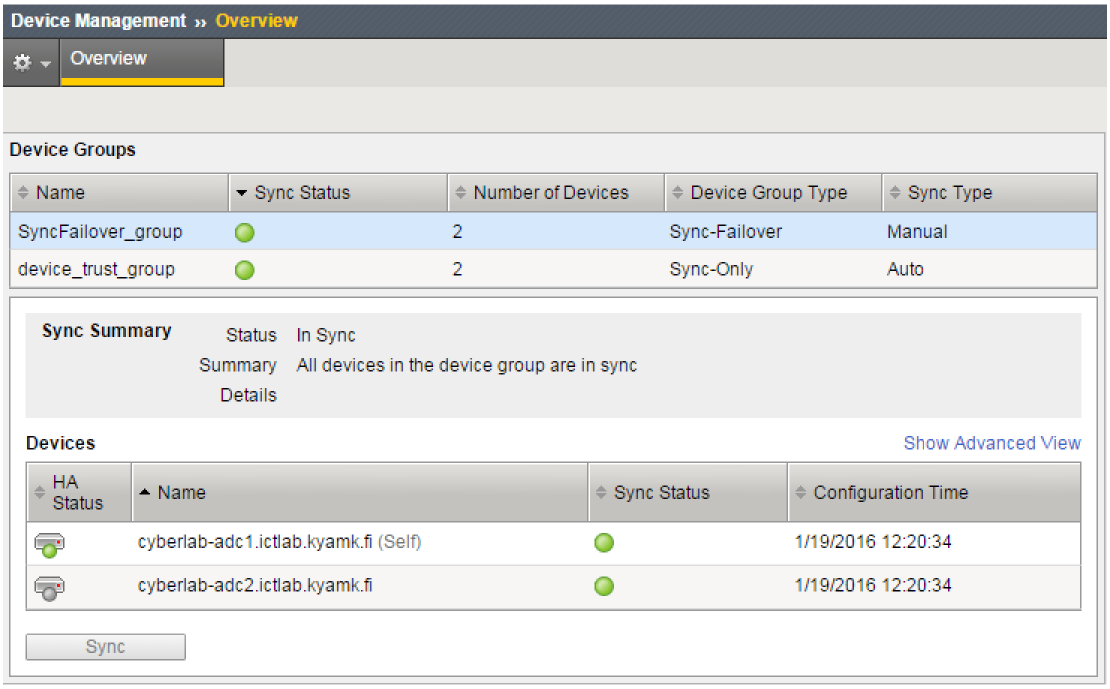

Figure 21. Syncing the main devices configuration to peer

7.3 Creating test virtual server

Before the sync, a virtual test server was made to see, if prior knowledge from the virtual version held true and the connection from a client to the server could be established.

The configuration started with creating a pool for servers and adding a host under it, as seen in figure 22.

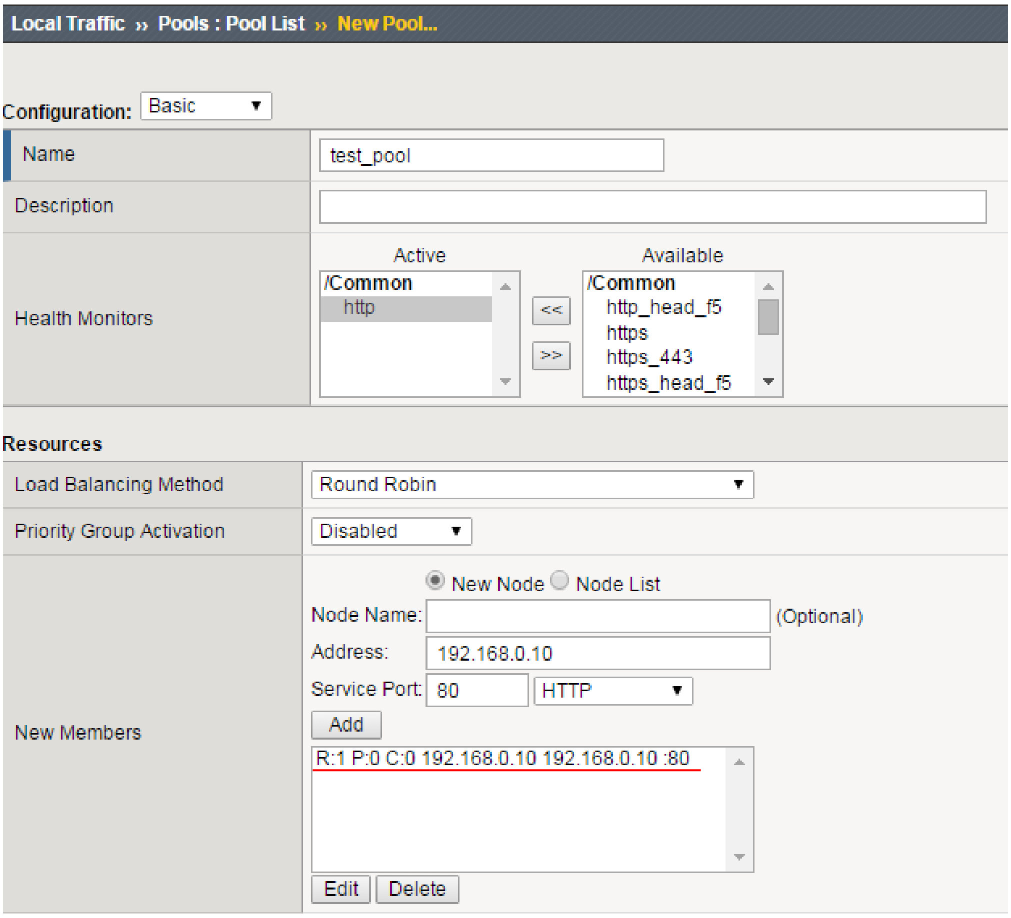

Figure 22. Creating a pool and adding a node

After the pool was configured, the next step was to create a virtual server for the pool, as seen in figure 23.

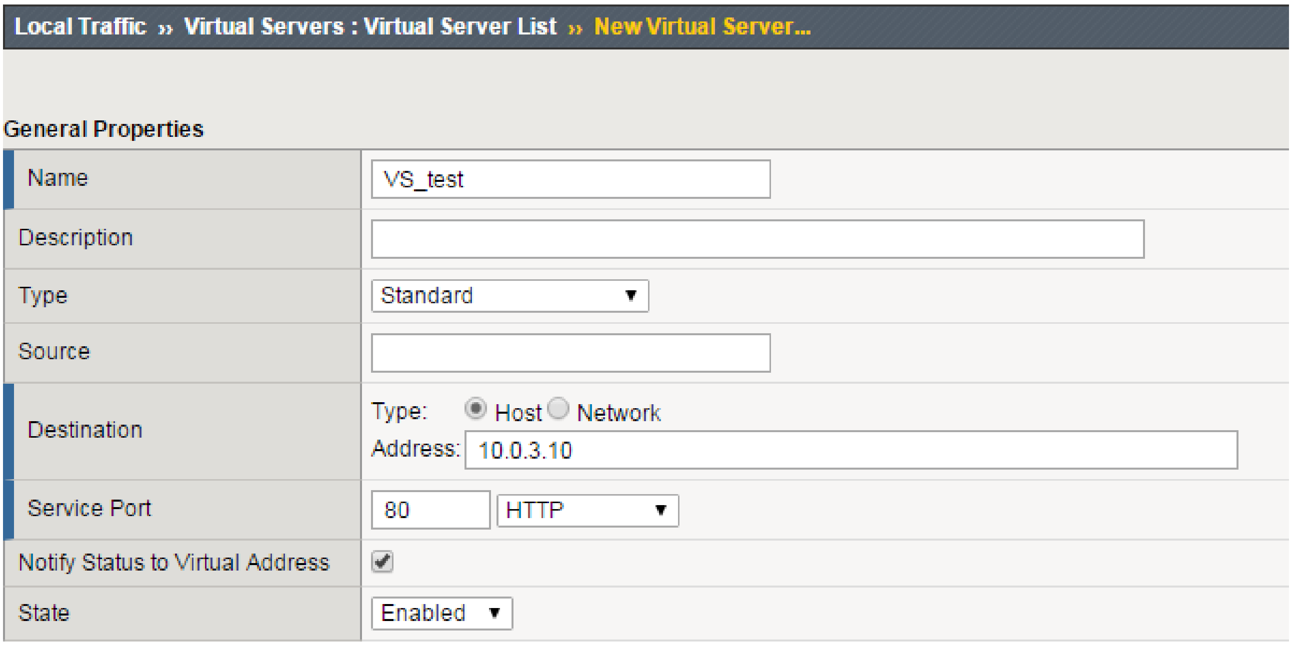

Figure 23. Creating a virtual server

The configuration options that were in the previous figure are what mattered the most. Other options, except source address translation were left as they were, even though it is best practice to use a TCP-LAN-optimized profile for the client side protocol profile and have the virtual server enabled only on the external side.

The auto map option was used as source address translation. This option changes the clients' own IP to BIG-IP's self -IP address and forwards the traffic with that address as a source to the servers.

Last, the previously configured pool was added as a resource for the virtual server, as show in figure 24.

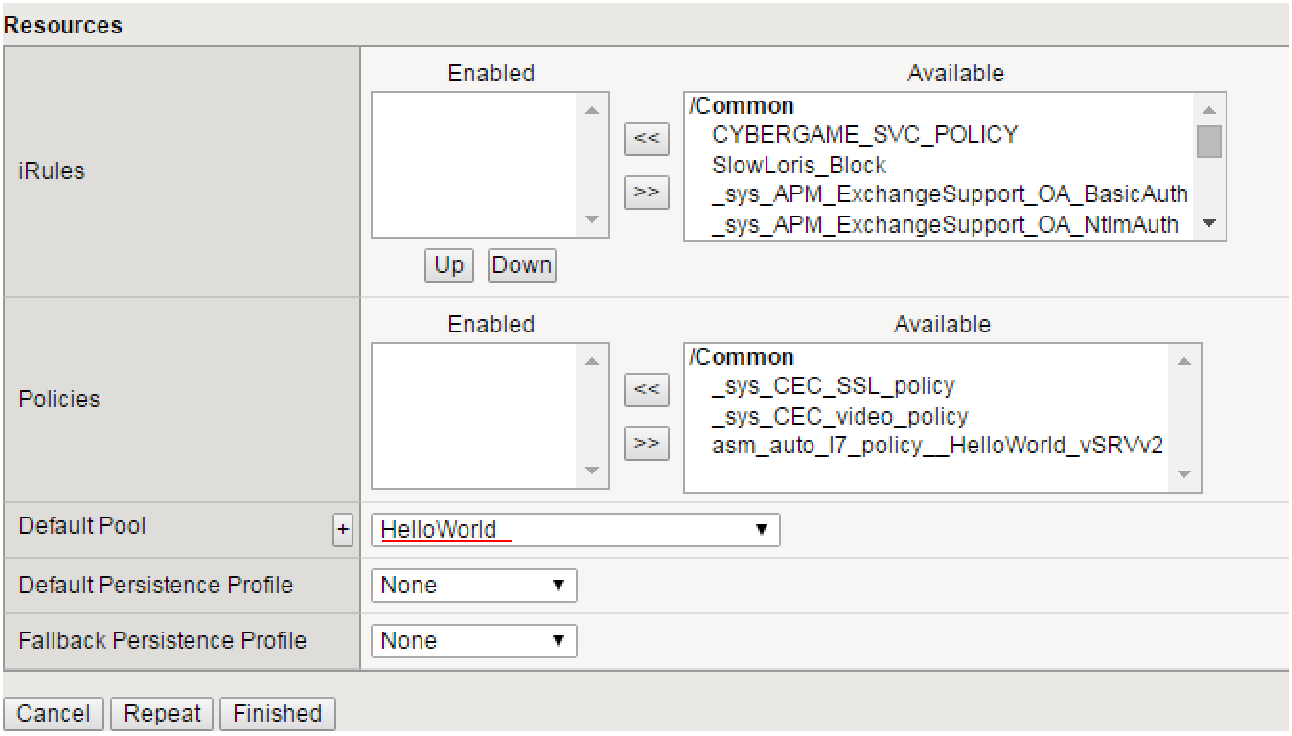

Figure 24. Adding pool as a resource to virtual server

7.4 VMware vSphere configurations

Next step was to make changes in the server cluster with vSphere web client, so that the traffic going for the servers behind BIG-IPs was isolated from the main link. This needed a creation of new distributed switch inside the server cluster, as shown in figure 25.

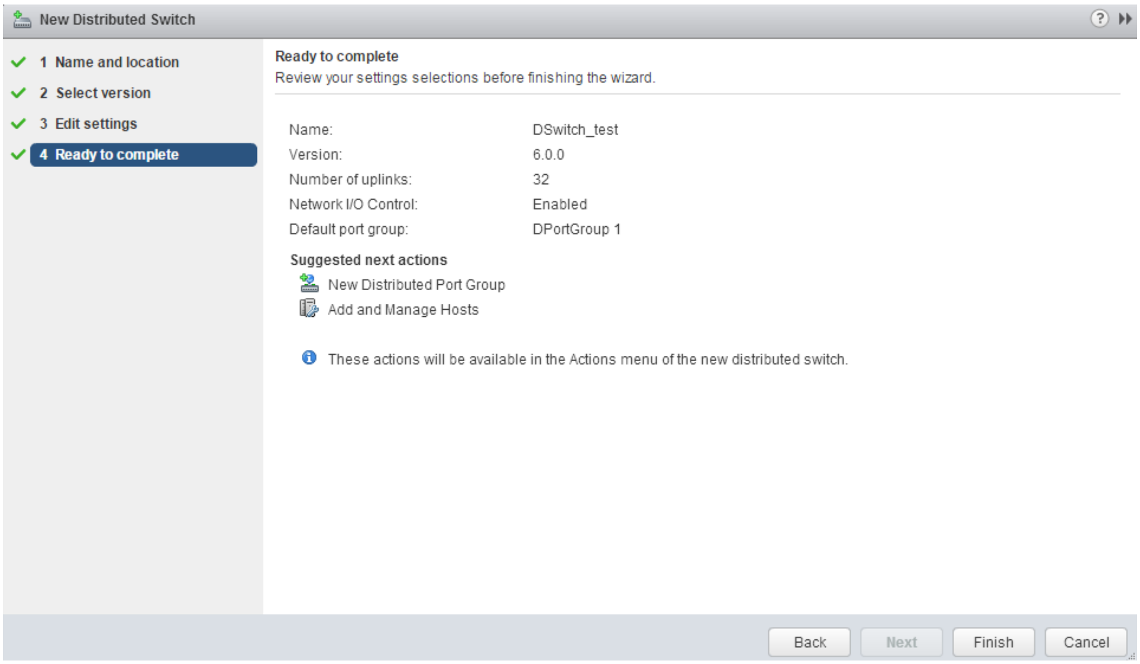

Figure 25. Creating a DSwitch in VMware web application

Next there was a need to configure new port group for the test lab, so that the DSwitch forwards the traffic to the right place. The inside was chosen to be in VLAN 2001, as seen in figure 26.

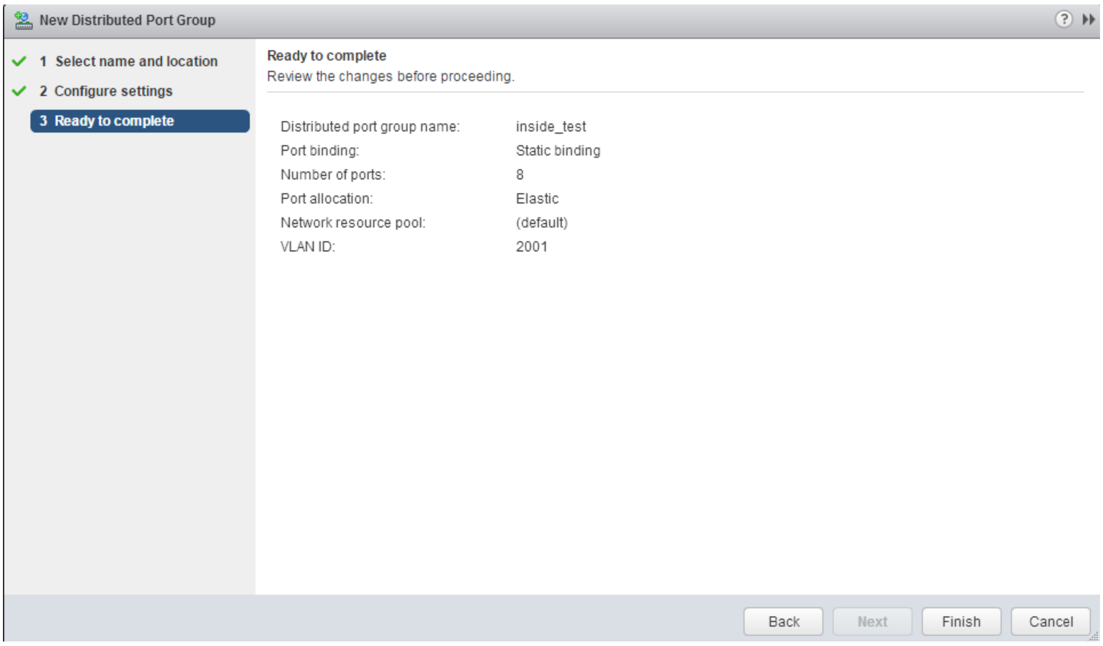

Figure 26. Port group configuration

After that, the needed server hosts were migrated under the newly created port group, as seen in figure 27.

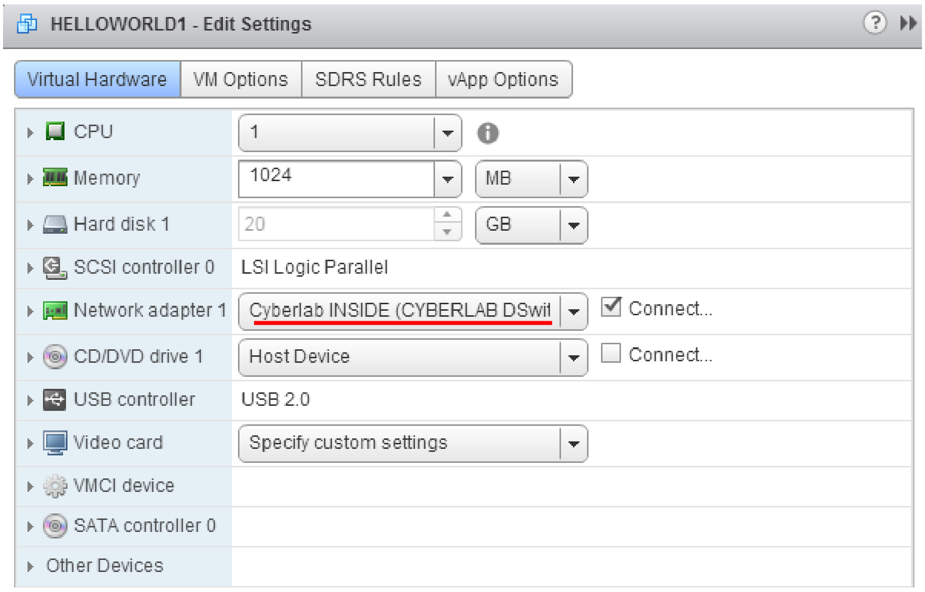

Figure 27. Adding host to port group

After the configurations were done, ping command was used to test if the server node was up and active, as shown in figure 28.

```
[root@cyberlab-adc1:Active:In Sync] config # ping 172.16.30.10
PING 172.16.30.10 (172.16.30.10) 56(84) bytes of data.
64 bytes from 172.16.30.10: icmp_seq=1 ttl=64 time=0.578 ms
64 bytes from 172.16.30.10: icmp_seq=2 ttl=64 time=0.807 ms
64 bytes from 172.16.30.10: icmp_seq=3 ttl=64 time=0.912 ms
```

Figure 28. Ping test from F5 to server

7.5 Load balance test

After the environment was tested and was deemed ready for further experimenting, a goal was set to create 3 different web servers and see how the F5 would load balance the traffic from the clients to those.

The web servers were made with Linux and a simple html index page was used to demonstrate the case. Three of these pages were created with different index pages and IP-addresses, as seen in figures 29 and 30.

```
<html>
 <head>
  <title>TEST SERV 1</title>
 </head>
<body>
<h1>Test_serv 1</h1>
 </body>
</html>
```

Figure 29. Test server index page

```
# The primary network interface
auto eth0
iface eth0 inet static
        address 172.16.20.10
        netmask 255.255.0.0
        gateway 172.16.0.1
```

Figure 30. Test server IP-addresses

After they were created, their interfaces needed to be brought down and up for the new IP-addresses to take effect. Then, they were migrated under the DSwitch inside in vSphere, so that they would get connection to the Big-IP.

Next, a pool and virtual server was created for the environment, as shown in figures 31 through 34.

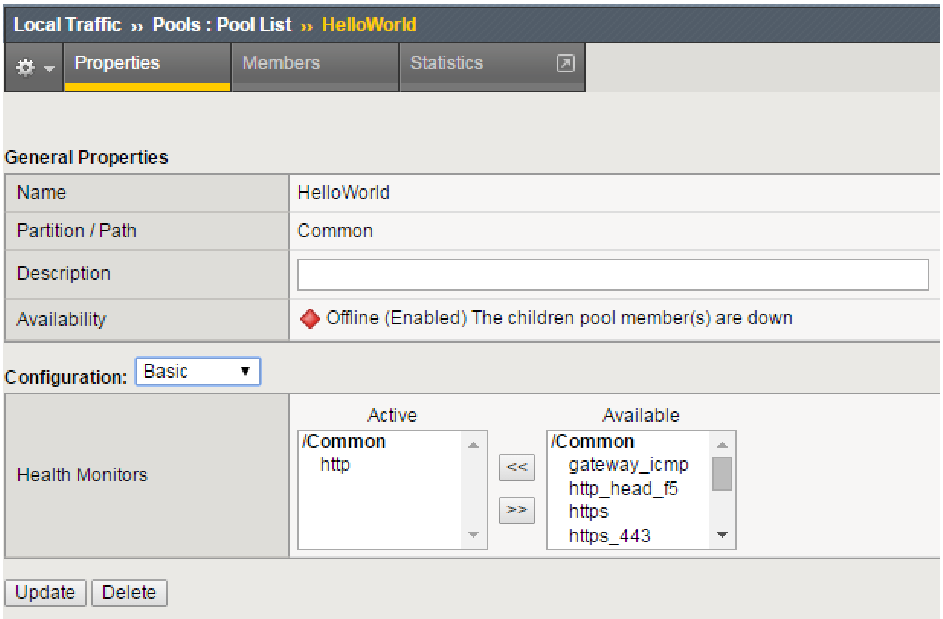

Figure 31. Pool for the web servers

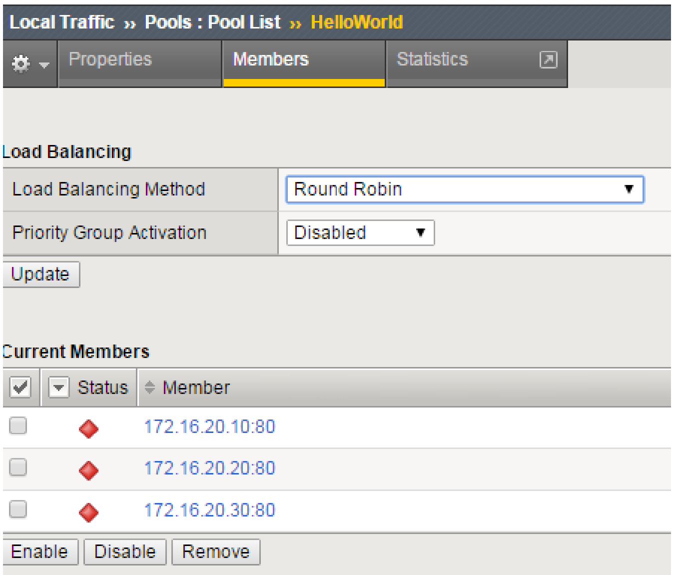

Figure 32. Servers used in the pool

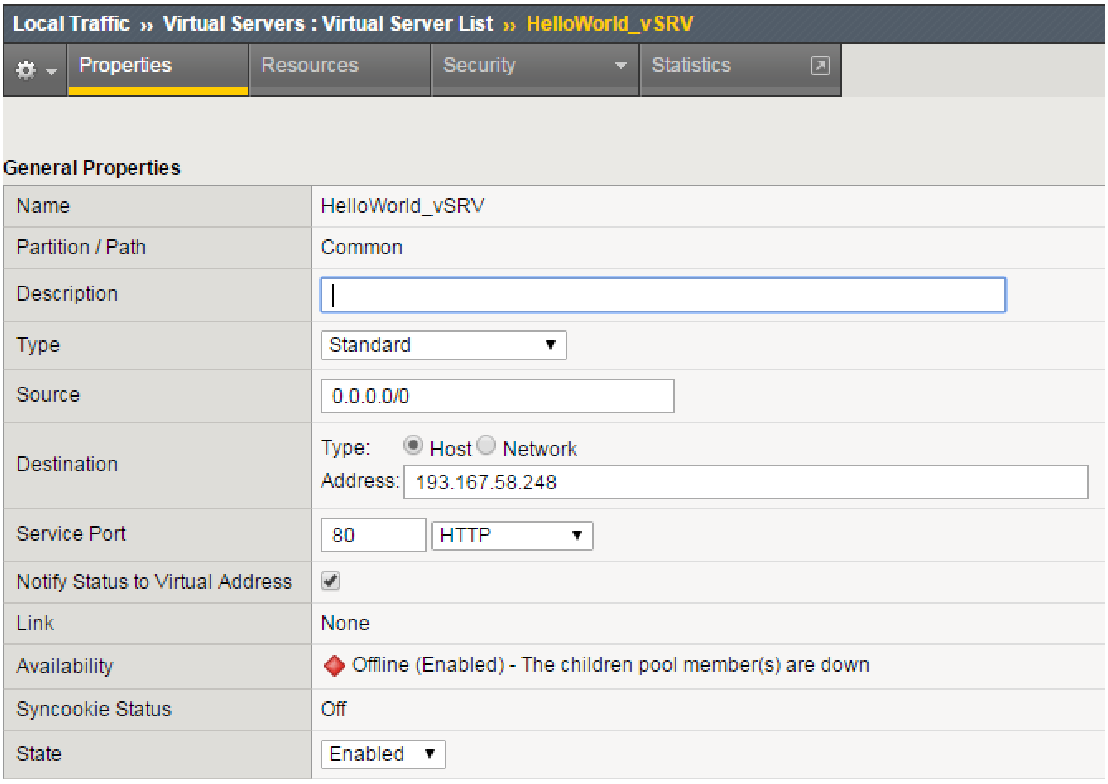

Figure 33. Virtual server configuration

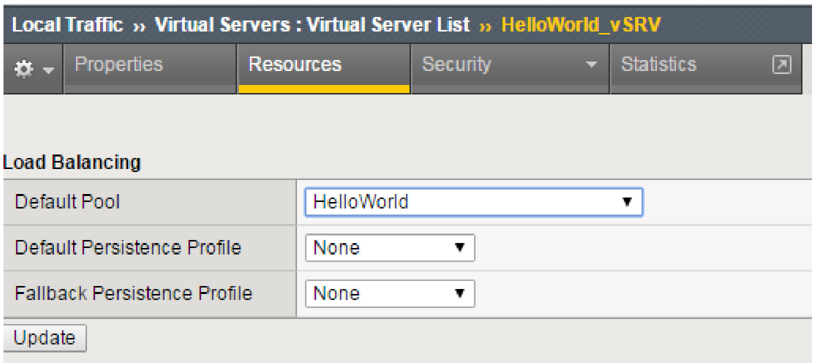

Figure 34. Adding pool to the virtual server

Finally, the system was tested by trying to connect to the virtual server IP and see if the ADC balanced the traffic to each node as the traffic of the servers increased; and it worked.

8 NETWORK STRESS TEST

After the BIG-IPs and the network were configured, a stress test of the whole environment was done by launching different kinds of DDoS attacks against the BIG-IP, as shown in figure 35.

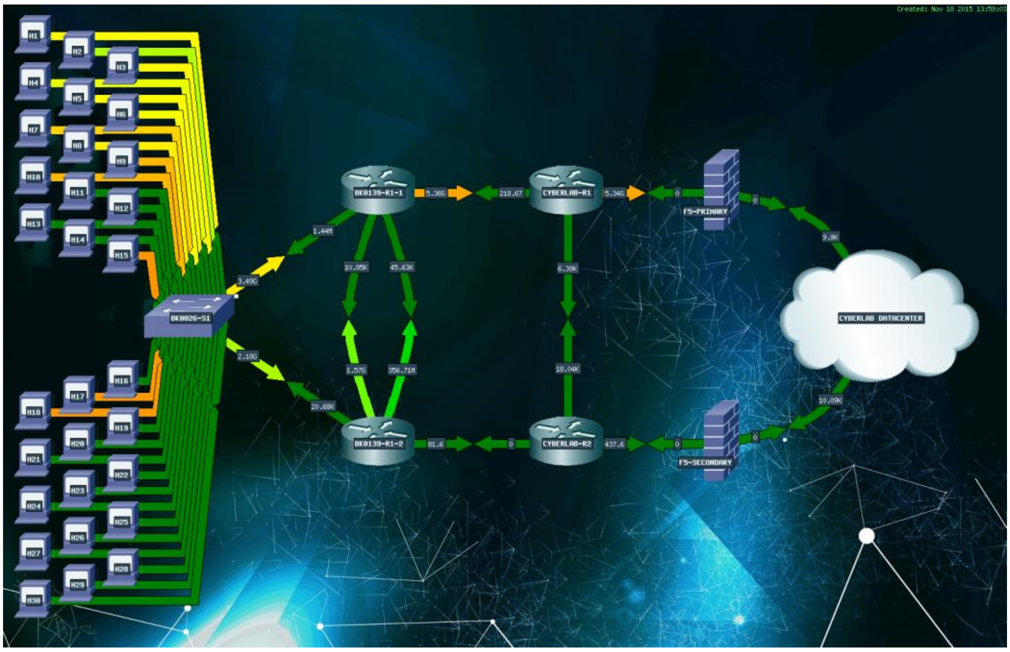

Figure 35. Stress test network

The main attacks used were TCP SYN flood and Slowloris. SYN flood uses TCP state retention to its advantage when a new connection is being established. By sending large amount of SYN packets against the target machine, it stops the target’s ability to establish new legitimate connections. (Eddy 2007.)

Slowloris uses HTTP connection timer to its advantage by sending HTTP GET request very slowly. For example, a server has 200 second timer before it closes the connection if the client does not finish the GET request. By sending a partial GET and then feeding bogus information before the timer expires, the connection stays open. Sending large amount of these requests a hacker can overwhelm the targets ability to take any legitimate connections. (Muscat 2013.)

The testing spanned two days with one day done only with students. The second day a F5 consultant came to help with the configuration of the BIG-IP.

At start, BIG-IP held fairly well against the attacks without specific configurations, but as the severity of the attacks rose, the BIG-IP started showing extreme lag and poor availability of resources.

First module used against the attacks was the DoS protection profile and under it Application Security. Also Source IP-based Rate Limiting and URL-Based Rate Limiting, were enabled as shown in figure 36.

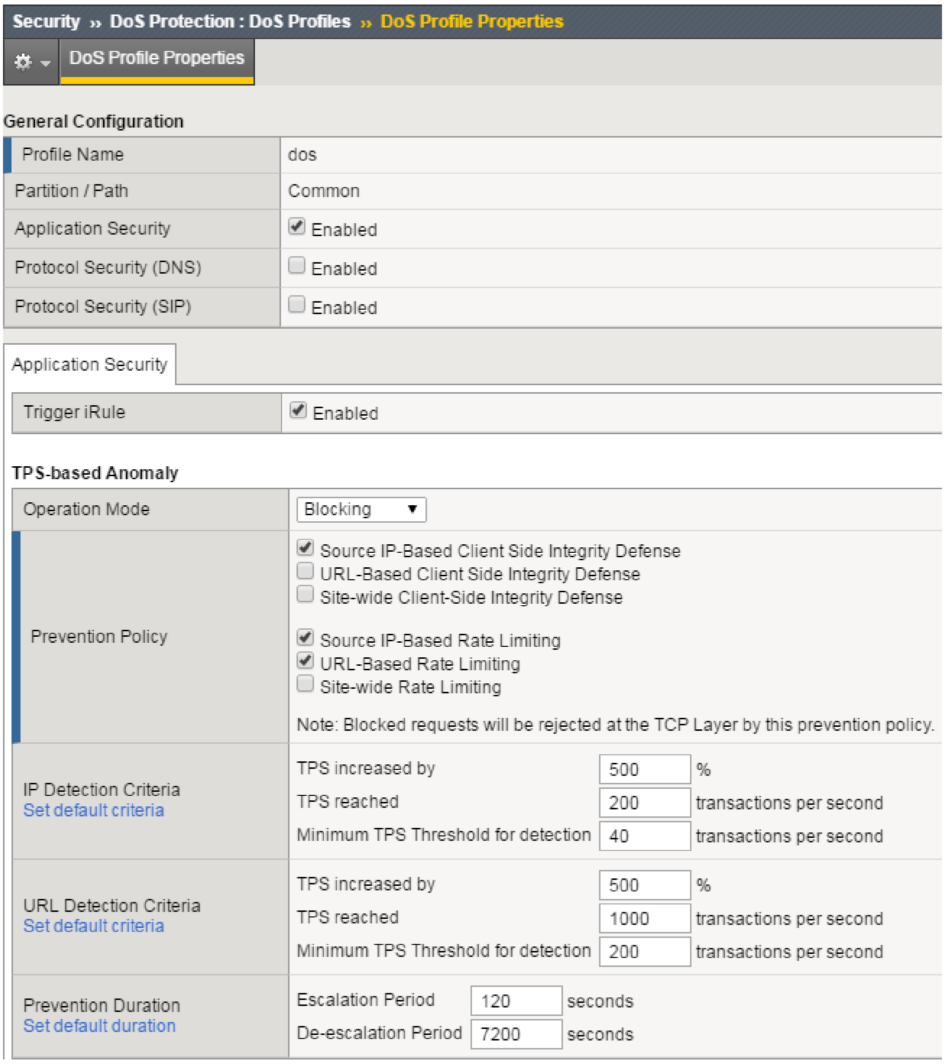

Figure 36. DoS Profile configuration

After the settings were done, the BIG-IP was able to forward traffic to the servers even though 4Gb of traffic went to it.

The next attack used was Slowloris. This stopped the BIG-IP service. Many configuration changes were tried against the attack but none were able to get the service running, until Single Endpoint Flood and Sweep values were changed, as shown in figure 37. It remained unclear this was the correct way of mitigating the attacks because while it stopped the attack it also stopped the monitoring tool used in the test. These were the last configuration changes done the first day.

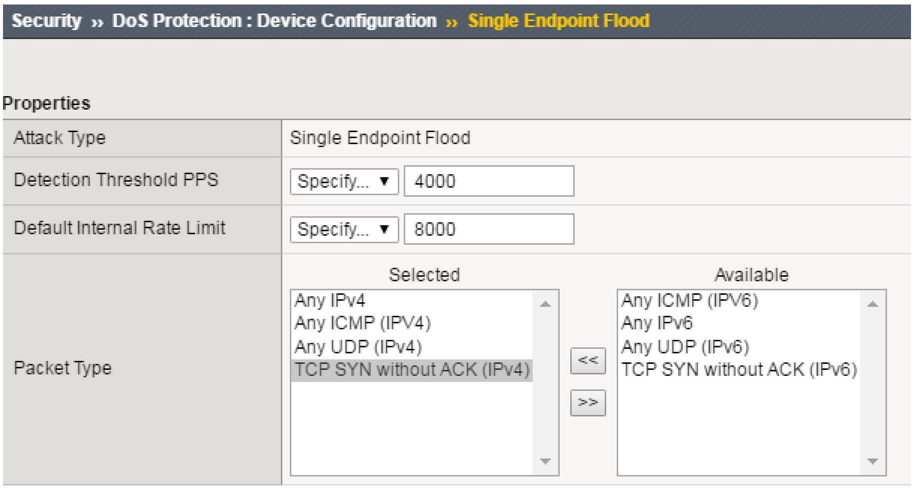

Figure 37. Single Endpoint Flood and Sweep configurations

The next day of testing started the same way as the first, with rising attack severity, but with the help of the F5 consultant, the BIG-IP was able to mitigate all of the attacks at the end. All in all the BIG-IP was deemed very effective against the attacks and fully serves its purpose in the Cyberlab environment.

9 CONCLUSIONS

The work was completed to the point that was needed at that time in the environment; with working load balancing. It provided a good learning experience as to how the ADC distributes the traffic and how to get traffic running through it.

The work done with the vSphere provided a small experience as how to configure virtual switches and distribution port groups and how to use the web application in general.

SSL-offloading was tested with the virtual version, but did not yield conclusive evidence that it worked properly. It needs to be tested with the hardware and implemented when the need arises.

The F5 iRule module was tested but because of no programming experience, was not used to its full potential. Used properly, it could be very beneficial for the Cyberlab project.

The primary modules that were used in the ADC were local traffic, network and device management, so there is much more to discover and use to optimize the F5 BIG-IP for the needs of the project and to learn of the machine in general. The F5 offers many more bachelor’s thesis ideas such as IPv4 to IPv6, firewall, packet manipulation.

The work proved very challenging at start because of no previous experience with F5 products. The virtual testing environment was not very effective because of the interoperability issues with the server software and thus created more problems at the start. Also because the F5 has so many modules and functions, a lot of time was consumed trying to find the right commands for the right functions.

SOURCES

A10 Networks Inc. n.d. SSL Offload. Available at: https://www.a10networks.com/products/ssl-offload [Accessed: January 25 2016].

Citrix. 2015. What is an application delivery controller? Available at: https://www.citrix.com/content/dam/citrix/en_us/documents/products-solutions/what-is-an-application-delivery-controller-adc.pdf [Accessed: January 23 2016].

Eddy, W. 2007. TCP SYN Flooding Attacks and Common Mitigations. The IETF Trust. 8.2007. Available at: https://tools.ietf.org/html/rfc4987 [Accessed: May 6 2016].

F5 Networks Inc. n.d.a. Access Policy Manager. Available at: https://f5.com/products/modules/access-policy-manager [Accessed: February 2 2016].

F5 Networks Inc. n.d.b. Advanced Firewall Manager. Available at: https://f5.com/products/modules/advanced-firewall-manager [Accessed: February 2 2016].

F5 Networks Inc. n.d.c. Application Acceleration Manager. Available at: https://f5.com/products/modules/application-acceleration-manager [Accessed: February 2 2016].

F5 Networks Inc. n.d.d. F5 Silverline Cloud-Based Application Services Platform. Available at: https://f5.com/products/platforms/silverline [Accessed: February 2 2016].

F5 Networks Inc. n.d.e. Investor Relations. Available at: https://f5.com/about-us/investor-relations [Accessed: February 2 2016].

F5 Networks Inc. n.d.f. Secure Web Gateway Services. Available at: https://f5.com/products/modules/secure-web-gateway [Accessed: February 2 2016].

Fabbi, M. & Lerner, A. 2014. Gartner. Magic Quadrant for Application Delivery Controllers 11.10.2014. Available at: http://innetworktech.com/wpcontent/uploads/2014/11/Magic-Quadrant-for-Application-Delivery-Controllers.pdf [Accessed: February 5 2016].

Holmes, D. 2013. Mitigating DDoS Attacks with F5 Technology. F5 Networks Inc. 1.31.2013. Available at: https://f5.com/resources/white-papers/mitigating-ddos-attacks-with-f5-technology [Accessed: February 8 2016].

KEMP Technologies. n.d. Round Robin Load Balancing. Available at: https://kemptechnologies.com/load-balancing/round-robin-load-balancing/ [Accessed: January 10 2016].

KEMP Technologies. n.d. SSL Acceleration & SSL Offloading Solutions. Available at: https://kemptechnologies.com/solutions/ssl-acceleration-solutions/ [Accessed: January 25 2016].

Maakuntahallitus. 2014. Kyberturvallisuusosaamisen ja liiketoiminnan kehittäminen 12.15.2014. Available at: http://kokoushallinta.ekliitto.fi/djulkaisu/kokous/2014193-10.PDF [Accessed: November 20 2015].

MacVittie, D. 2009. Intro to Load Balancing for Developers – The Algorithms. F5 DevCentral 3.31.2010. Available at: https://devcentral.f5.com/articles/intro-to-load-balancing-for-developers-ndash-the-algorithms [Accessed: January 15 2016].

MacVittie, L. 2008. 3 Really good reasons you should use TCP multiplexing. F5 DevCentral 10.14.2008. Available at: https://devcentral.f5.com/articles/3-really-good-reasons-you-should-use-tcp-multiplexing [Accessed: January 25 2016].

Muscat, I. 2013. How To Mitigate Slow HTTP DoS Attacks in Apache HTTP Server. Accunetix. 10.8.2013. Available at: http://www.acunetix.com/blog/articles/slow-http-dos-attacks-mitigate-apache-http-server/ [Accessed: May 6 2016].

NGINX. n.d. What is round robin load balancing? Available at: https://www.nginx.com/resources/glossary/round-robin-load-balancing/ [Accessed: January 10 2016].

Salchow, K Jr. 2012. Load Balancing 101: The Evolution of Application Delivery Controllers. F5 Network Inc. 4.23.2012. Available at: https://f5.com/zh/resources/white-papers/load-balancing-101-the-evolution-to-application-de [Accessed: January 10 2016].

FIGURE LIST