HÖGSKOLAN SKÖVDE

Institutionen för kommunikation och information
Examensarbete i datavetenskap 30hp C-nivå
Vårterminen 2008

A case study of handling load spikes in authentication systems

Kristjon Sverrisson

Titel

Examensrapport inlämnad av Kristjon Sverrisson till Högskolan i Skövde, för Kandidatexamen (B.Sc.) vid Institutionen för kommunikation och information. Arbetet har handletts av Marie Gustafsson.

Datum

Härmed intygas att allt material i denna rapport, vilket inte är mitt eget, har blivit tydligt identifierat och att inget material är inkluderat som tidigare använts för erhållande av annan examen.

Signerat: _____

A case study of handling load spikes in authentication systems

Kristjon Sverrisson

Abstract

The user growth in Internet services for the past years has caused a need to re-think methods for user authentication, authorization and accounting for network providers. To deal with this growing demand for Internet services, the underlying user authentication systems have to be able to, among other things, handle load spikes. This can be achieved by using loadbalancing, and there are both adaptive and non-adaptive methods of loadbalancing.

This case study compares adaptive and non-adaptive loadbalancing for user authentication in terms of average throughput. To do this we set up a lab where we test two different load-balancing methods; a non-adaptive and a adaptive.

The non-adaptive load balancing method is simple, only using a pool of servers to direct the load to in a round-robin way, whereas the adaptive load balancing method tries to direct the load using a calculation of the previous requests.

Key words: UDP applications, loadbalancing, AAA, radius

Table of contents

1 Introduction

When the United States' National Science Foundation started the first TCP/IP wide area network in 1983 there were few who could predict the growth of the new network. During its first eleven years of existence the network it served as a backbone for universities that wanted to connect to each other and it grew at a high rate. A good indication of this is a study of the growth of hosts between 1981 and 1991, when the number of hosts increased from 213 to 727,000 (Lottor, 1992).

More recently the growth began to transform into the public life and dial up connections for domestic homes became common and resulted in a higher rate of growth than we expected.

As a result, many computers on the Internet that were carrying out services for users have faced increasing load and more demands in regard to responsiveness and reliability. This problem is discussed by Kima, Kimb and Kongc (2006) where they express concern whether the growth of broadband access can be handled by current authentication systems.

With the increased growth in internet usage, we have come to a limit where a single server simply cannot handle the traffic load generated in extreme situations. We will try to see how effective two different load balancing methods are.

To achieve this we will create a lab to simulate the two differend methods and compare the results.

It should be noted that in this paper there are quite a few acronyms and abbreviations. These will be introduced when first mentioned but can also be found in an appendix.

2 Background

2.1 Different ways of connecting users to the Internet

As the number of Internet user grows, the multitude of services offered to connect home users, offices and institutes to the network has multiplied. The most common method in many western countries is the broadband connection, such as the digital subscriber loop (DSL). The DSL connections are based on old legacy telephone wires, working on a different frequency band than regular telephones. This enables the possibility for the user to have both a working phoneline and a working Internet connection at the same time, without interference between these. This presents a new kind of situation for the companies resposinsible for providing Internet access since previouly most residential users used a telephone line to connect to the Internet. When not using the internet they disconnected the dialup connection, as most telephone companies charge by the minute.

When connecting to a broadband connection, most users have a residential router, which connects the home network to the Internet, actings as a gateway between the two networks. This router takes care of routing between the networks and connecting to the internet when needed. Many of these routers do not disconnect the broadband connection, even when it is not being used for internet.

Figure 1 describes each link from a computer to the Internet gateway. A home computer connects to the residential router through a wiresless or wired home network. The residential router is connected to a telephone line, through which it communicates with a DSLAM. The DSLAM and NAS are connected to a common network. This enables the residential router to create a PPPoE tunnel to the NAS.

A Network Access Server (NAS) is a gateway that connects several users to a local network/resource. A NAS can be a small router which accepts dialup users connecting through a telephone line or a router serving thousands of users connecting through high speed connections.

In the early days of the Internet, a NAS might have served 4-16 users connecting with a 14.4kbps modem. As demand for Internet connections increased, the concentration of dialup users per NAS increased. A NAS from the 2005-2008 era, connecting DSL or broadband users to the Internet might have as many as 3,000 to 10,000 users according to Metz (1999).

A typical NAS is meant to serve 7,000 to 10,000 users. However, as user traffic increases, more NAS systems have to be included in the setup (the current limiting factor of each system is the capacity of the routing processor).

A NAS can experience problems as any other computer, resulting in a reboot of the server. In such an event, all its users are disconnected and must reconnect to regain connection to the Internet.

If a glitch (for example software bug or configuration errorr) were to occur in a NAS serving 10,000 users, causing it to reboot, the Remote Authentication Dial In User Service (RADIUS) server would have to reply to almost all of them reconnecting in a matter of minutes after the NAS becomes available after a reboot. This bottleneck has been an increasing problem for most large Internet service providers the past years.

The technology used to connect the users residential router to the NAS is based on the Digital subscriber line access multiplexer (DSLAM). The hardware enables legacy phone wires to be used to achiev higher bitrate by using a different frequency band than normal phone calls use. Each DSLAM handles several aggregation cards, which each support up to 48 users. A typical DSLAM might have as many as 600-800 users. When a connection is enabled between the residential router and DSLAM, regular Point-toPoint Ethernet (PPoE) is used to initiate a link between the residential router and the NAS.

2.2 Validating users for service

A commonly used system that was created to handle authentication, authorized and account (AAA) for Internet dialup connections is the RADIUS protocol. This protocol is created to authenticate users that are connecting to a network, such as the Internet. Each time a home user connects to the remote network the protocol is used before initializing a connection.

The protocol implements the following:

- Authentication: Validate that a user is providing the correct credentials, usually a username and a password.
- Authorization: confirm that the user has valid access to the service that he is requesting. This can be used for instance to control which users should be able to connect through a broadband connect, dialup connection or via mobile.
- Accounting: Keep track of usage of a given resource such as downloaded data through a dialup connection or connection time. This information is often used for billing and information on what user is behind a dialup connection at any given time.

The protocol was first described by Livingston Enterprises in 1991 and later published as an RFC document[1], refered as RFC2058 by Rigney et al. (1997).

[1] The Request For Comment (RFC) documents are the official publication channel for the Internet Engineering Force (IETF), Internet Architecture Board (IAB) and the community of computer network researchers.

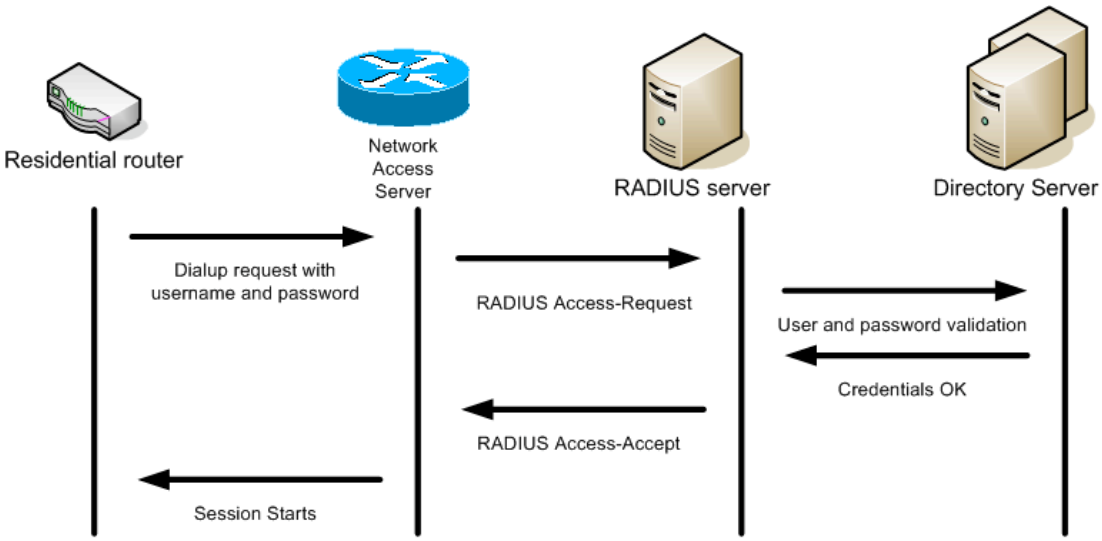


Figure 2 Example of an authentication of a user. Each arrow represents a query to a service or a reply to to the query.

The protocol is implemented using the User Datagram Protocol (UDP) instead of the Transmission Control Protocol (TCP), even though the TCP protocol makes up for the majority of internet traffic. According to Famenkov, et al. (2003) this can be as much as 75%. The reason for the common usage is that the TCP protocol is connection-aware. This means that a computer initializes a connection to a second computer or server, and keeps the connection open until all data that is to be sent has completed. The protocol keeps track of all packets that are sent and validates that every packet is received and the data contained in the packet is unaltered. If a TCP packet is lost during its course through the network, the receiving computer requests that it is sent again.

However, this mechanism can be quite expensive, as for each transmission needed to be sent through the network, there is overhead. This overhead is in the connection establishment, as there is a three-way handshake to initialize the connection. Also there is overhead for the operating systems in question, as for each TCP connection there are resources that have to stored, such as a private buffer queue for each connection.

Also, for some applications, such as video streaming, a certain amount of lost packages do not matter for the functionality of the stream. The end user may never notice that som of the packages never arrived. For these applications the UDP protocol might be of more interest, since the overhead with UDP packets is far less. Being the second most used protocol on the Internet, with about 22% of the transmitted packets, according to Famenkov, et al. (2003), it is sufficient for many applications.

The UDP protocol is state-less, which means that two computers communicating with each-other have no active session and have no way of controlling if the other computer is functioning correctly. This means that if a computer on a network sends a packet to a second computer, and for some reason the packet is dropped, neither computer will know that the packet was never received correctly.

But as the protocol is state-less, the overhead is reduced by the TCP handshake sequence and the resources that have to be provided by the operating system. This is quite positive for protocols such as RADIUS and Domain Name System (DNS) lookups.

There are other protocols that have been defined for the use in AAA services, thanks to active development in the area of user authentication. This is described by Kima, Kimb and Kongc (2006), where they express concern about the usage growth compared with the network service providers' current solutions.

The Terminal Access Controller Access-Control System (TACACS) protocol was introduced in RFC 1492 by Finseth and the University of Minnesota. The protocol is based on TCP as it is deemed by many to be more reliable than UDP.

DIAMETER was developed by a collaboration of individuals and companies and published as RFC 3588 (Calhoun et al.), released in September 2003. It is intended to be the successor to the RADIUS protocol, thought it is not backwards compatible.

As the protocol is more recent than both RADIUS and TACACS it has not been implemented in the most common NAS systems..

2.3 Basic load balancing

In the face of the growth of computer applications for the past years, the IT community has worked on several methods to share computer load among several computers. One of the most used methods is load balancing. For example, a web service can be spread out on several servers to reduce the load on each server. An example of this can be seen in figure 3, where many different users may be requesting a service from a single server. This server can be distributing the service load between many different backend servers.

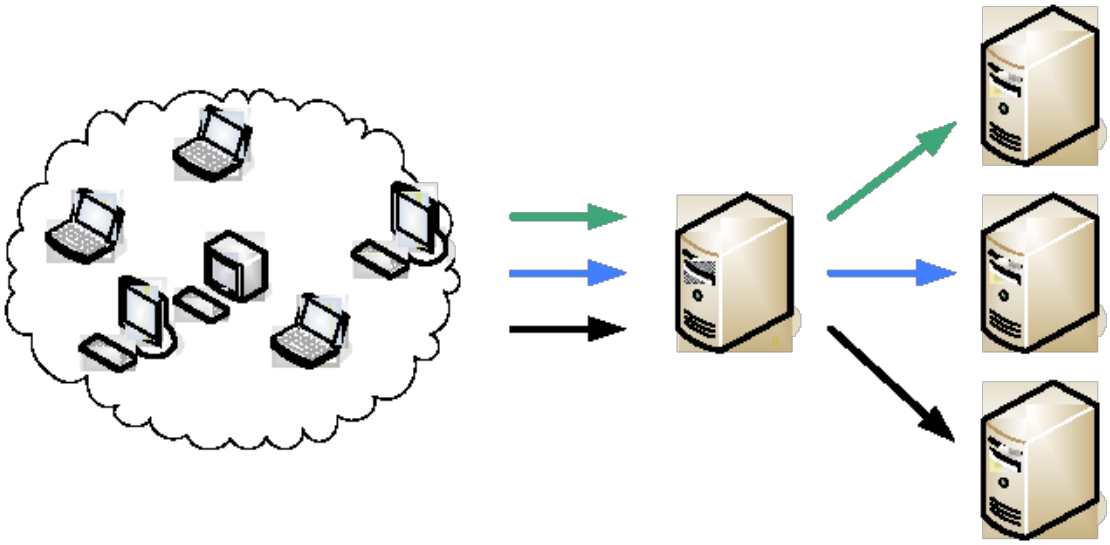


Figure 3 describes how many computers can be sending requests to a single server, which then forwards the requests to other servers, which process it. This reduces the load of the first computer, since it only has to forward the request.

This method involves having a single entity, such as a server, receive all requests from clients and distribute these to multiple servers. The first, (receiving) computer does not process the data, only forwarding it to a second server, which handles the request. By using this method the load of handling the request is moved to the second computer, while the first computer is free to receive more requests.

The solution can in many cases be extended; for example if a web site is distributed on three servers with a load balancer, a fourth or fifth server can be added without alterations to the three original servers. The load balancer is the only entity that knows of all servers involved. Load balancing has more benefits, such as being able to take one of multiple servers out of the active load balancing group, for maintainance or replacement.

As discussed by Balasubramanian, et al. (2004), there are two generic methods of load balancing: adaptive and non-adaptive load balancing.

2.3.1 Non-Adaptive load balancing

The term Non adaptive load balancing means that the load balancer is unaware of the state of the servers that it forwards requests to.

An example of such and method is the round-robin load balancer, where the load balancer keeps a list of all servers in its group, and iterates between the servers.

2.3.2 Adaptive load balancing

When a load balancer is adaptive, it has knowledge of a measurable entity at run-time, so that it can actively direct more requests to servers that for example have less processor load, or a better response time.

A comparission between the two load balancing methods is found in Figure 4, where we can see the two domains. In both cases we have the same amount of requests being generated and sent to the load balancer. How ever the load balancer spreads the load very differently based on which method it uses. In the adaptive load balancing method new requests are rather sent to the least loaded server. The non-adaptive load balancing method simply spreads the load evenly between the two servers.

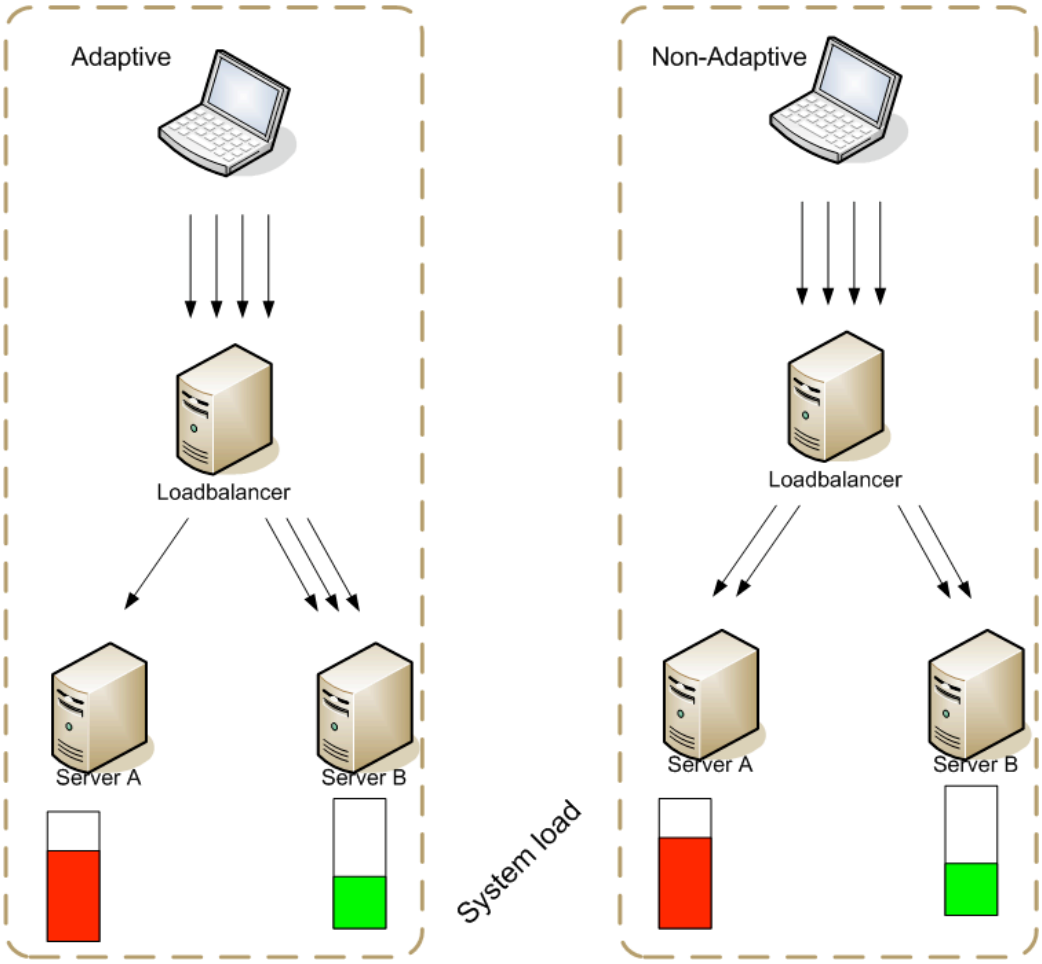


Figure 4 describes the difference between adaptive load balancing and non-adaptive loadbalancing.

3 Problem statement

The extreme load that can be created when a single NAS fails and all users must reconnect can cause great problems for many AAA servers. To handle this and other scenarios where extreme load is expected on the authenticating server, many vendors have implemented load balancing mechanisms.

RADIUS relies on UDP for transmission over the network for several reasons as Rigney, et al. (1997) describe. Using UDP simplifies the process as the NAS only needs to send a single UDP packet to the server, instead of opening and handling a TCP connection.

However there are problems affiliated with using UDP. As the protocol is stateless, there is no guarantee of delivery of the package. For this reason a NAS might have to retransmit an authentication query if it receives no reply. Cases where this might be necessary can be if the UDP package is lost on-route, if there is a malfunction in the software or if the system simply takes too long to respond. This presents no problems in normal cases as there is a very small chance of it happening to a second (or third) query sent.

This does however present a problem when there is a flood of authentication requests such as happens when a NAS is rebooted and the RADIUS server cannot reply to all of the queries in a timely fashion. In this case, the problem increases as the NAS retransmits the query for each authentication after the initial timeout and the loads on the RADIUS server increases.

The underlying problem is that the load-balancer mechanism is not efficient enough. For loadbalancing network computers, both adaptive and non-adaptive methods can be used..

We can see in the casestudy of Iceland Telecoms systems that the current system can perform badly at times. The system uses an adaptive load balancing mechanism that is based on the average service request time of the previous 10 queries. This mechanism makes it possible for the load balancer to send more requests to a server that is not as loaded as the other server, as it should be able to deliver results quicker.

In this study we will attempt to examine the different ways to direct authentication requests to a RADIUS server. We will try to find an optimal way to maximize the throughput of the load balancer in order to obtain the highest possible throughput.

4 A case study of a AAA setup

For several years Iceland Telecom has been running an AAA service called **Radiator** that implements the RADIUS protocol. This service is responsible for awaiting authentication and accounting enquiries from a Network Access Server and validating that the user credentials are correct and which sort of access the user is valid for.

A requirement for most companies with a large user base is a good database to contain the user information. The lightweight directory access protocol (LDAP) provides an interface to such a database. It derives from work made by telecom companies in the 80's which was directed towards creating a directory structured data storage method. This later became the X500 standard, specified by a number of RFC documents. (G. Carter, 2003)

The information stored in the directory server is managed in a hierarchical way, resulting in a "tree" structure. Figure 5 presents an example of such a structure:

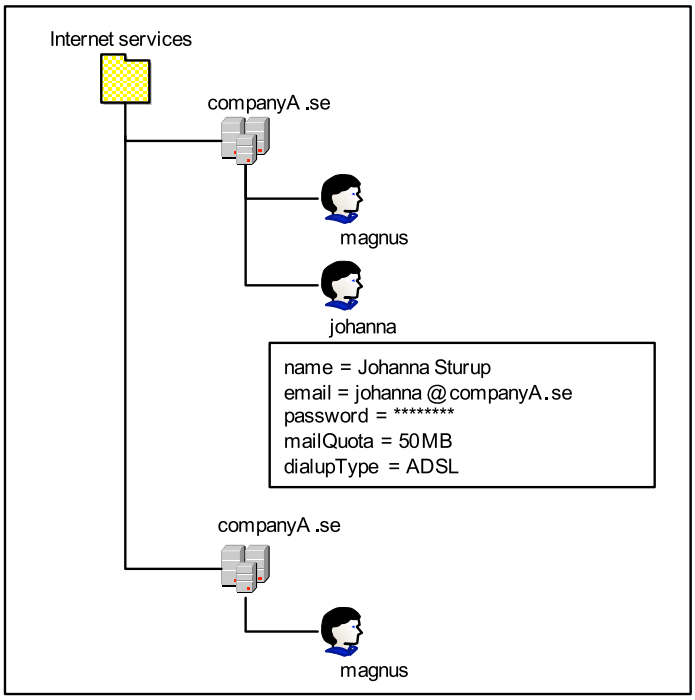


Figure 5 describes how a LDAP server might hold information in a tree like structure.

The Radiator system has proved to be reliable and flexible, which is extremely important, as Iceland Telecom acts not only as an ISP but also acts as a provider for other companies as well, allowing others commercial partners access to the DSL system in order to give end-users access to different ISP and companies.

However, in recent years the number of concurrent users in the system has sky-rocketed as the use of dialup-routers in residential homes has increased. These devices handle the dialup connection for the users, and usually stay connected at all times. Before these routers became available, a user would have to use a Dial-Up Connection in his/her computer if the user wanted access to the Internet.

One of the results of this change in users' behavior is that the load and performance requirements of the ISP NAS and RADIUS servers have increased. If a NAS were to reboot or an error would occur that disconnected all users, a large portion of these customers would attempt to reconnect as soon as they lost network connection. This causes extreme load spikes to the RADIUS server, which has proven to be hard to control.

4.1 Radiator system structure

Iceland telecom uses a distributed system for its RADIUS service, emphasizing load balancing. The system consists of three servers; one delegator and two proxies. The NAS sends its requests to the delegator which validates that the request is correctly formed and then forwards it to a proxy. In order to share the load, the delegator has a mechanism for estimating the server load on each proxy, which is based on the last query access time. The proxy is responsible for validating the user credentials, which are stored in a LDAP server.

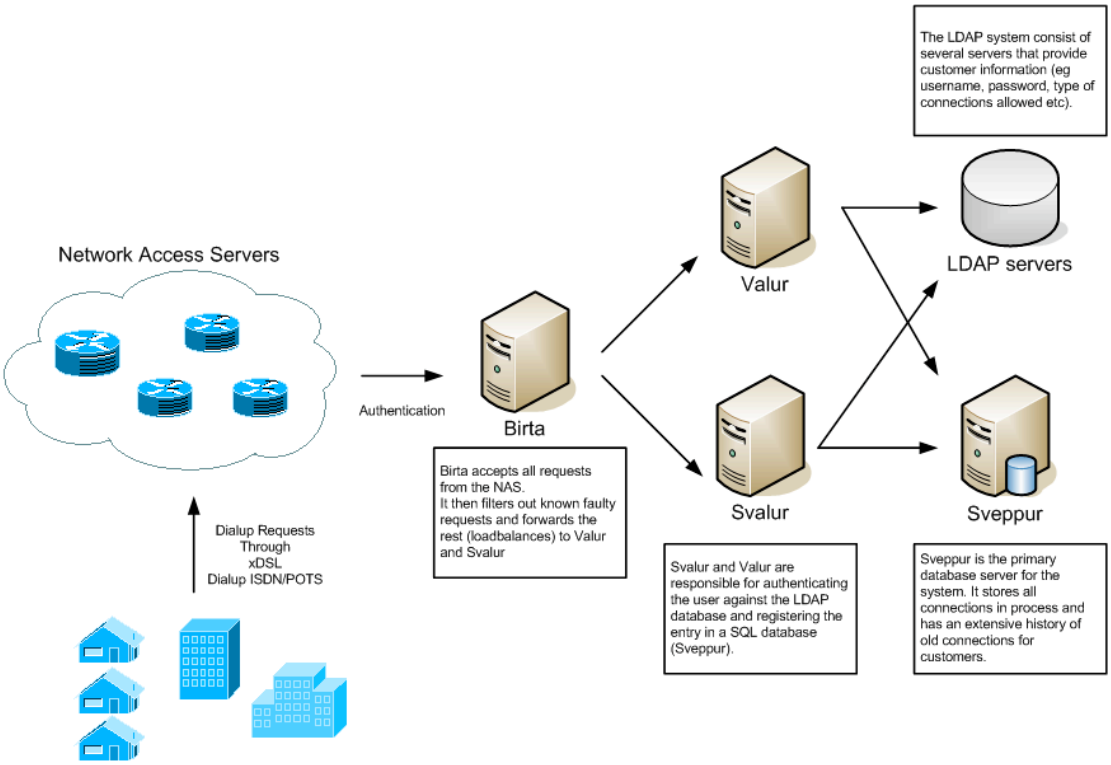


Figure 6 describes how a NAS contacts a single server to authenticate users. The single server then forwards the query to other servers which are responsible for the real authentication process.

Besides the servers that handle the basic radius authentications there are several load-balanced LDAP servers and a MySQL server. The MySQL server keeps track of all attempts to connect to the radius system, all current connections and a history of dialup connections. User passwords or other authorization requirements are never stored anywhere but in the LDAP server. These are pivotal in the system as law agencies such as the police might need the information at a later time. The database is quite large; about 20GB currently and the tables must be truncated regularly. All servers are running Red Hat Enterprise Linux.

The users connecting through the system are mostly home-users and small companies. The Network Access Servers consist mostly of Cisco equipment.

4.2 RADIUS load balancing

All authentication/accounting queries are sent to one single delegator (Birta) which controls that the query is correct (e.g., the query contains a username and that the username is in the correct realm[2]). Since the control mechanism used is simple, and no external controls are needed, it should be able to control and forward a large number of queries.

When the load balancer has validated that the query is indeed correct, it forwards it to one of the proxies that are responsible for authentication of the users against the local user-database, using a combination of the response time of the previous 10 requests and an estimate of the servers processing power.

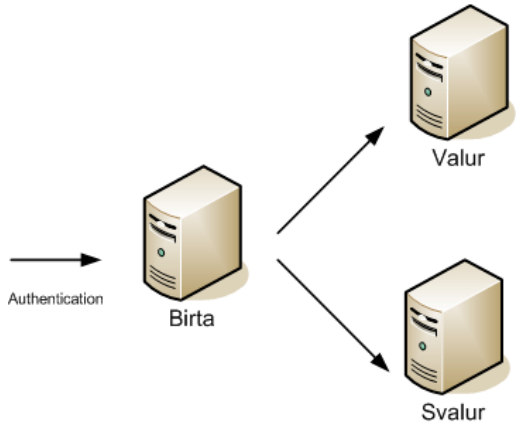


Figure 7 describes a single load balancing authentication.

The load balancing mechanism has a twofold purpose; it divides the load between the two servers and also it notices if a server becomes unavailable. In the case that one server becomes unavailable (the server fails to respond to a certain amount of queries) all requests are forwarded to the other server. The unavailable server is periodically checked to see if it is available again.

[2] Realm is a community or group of users, such as a domain @companyA.com or @CompanyB.org

4.3 Example of a load balanced authentication

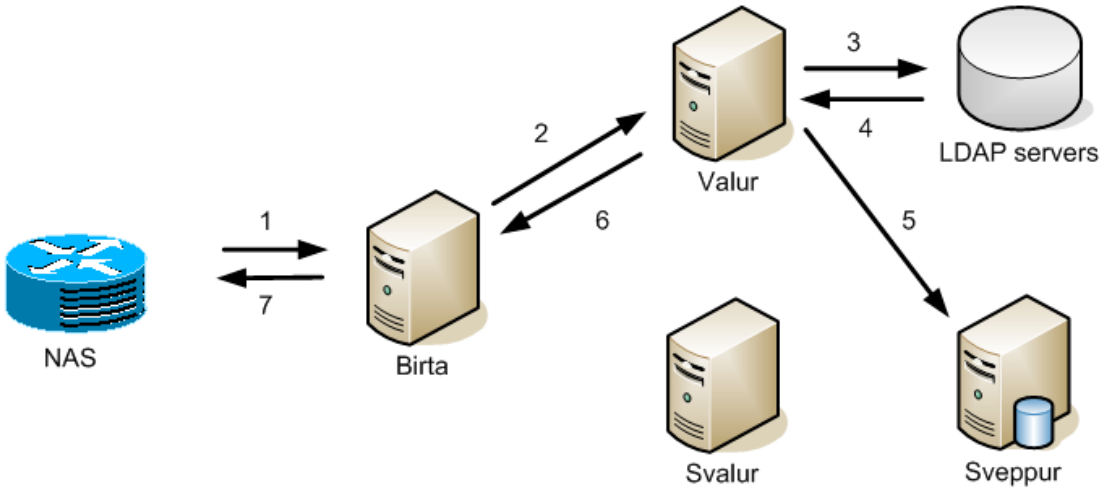


Figure 8 describes by step how each part of the load balanced authentication works.

1. Authentication request is sent from a NAS to the load balancer (birta).
2. The load balancer validates the request and forwards it to the backend servers (svalur & valur) if it is valid.
3. Valur authenticates the request against the LDAP servers to verify that the user credentials are correct and that the user is valid for accessing the system.
4. The LDAP server replies a correct or faulty reply to the request.
5. Backend server writes a log entry to the SQL database (Sveppur) for the authentication request.
6. Backend server sends a validation message to the load balancer, indicating that the request is valid and which IP address the user has been assigned.
7. The load balancer signals the NAS that the user is valid

5 Method

We propose a lab to experiment with two methods of loadbalancing, namely an adaptive load balance method and a non-adaptive method. Such a study would provide insight in the advantages or disadvantages of using the adaptive- and non-adaptive load balancing mechanisms. The result of this study is to provide insight in the advantages and disadvantages of using an adaptive load balancing mechanism versus an non-adaptive mechanism.

5.1 Lab setup

Seven servers will be used in the lab, three servers that will authenticate UDP authentication requests, one server that will provide loadbalancing and three servers that will generate/simulate the UDP authentication packets.

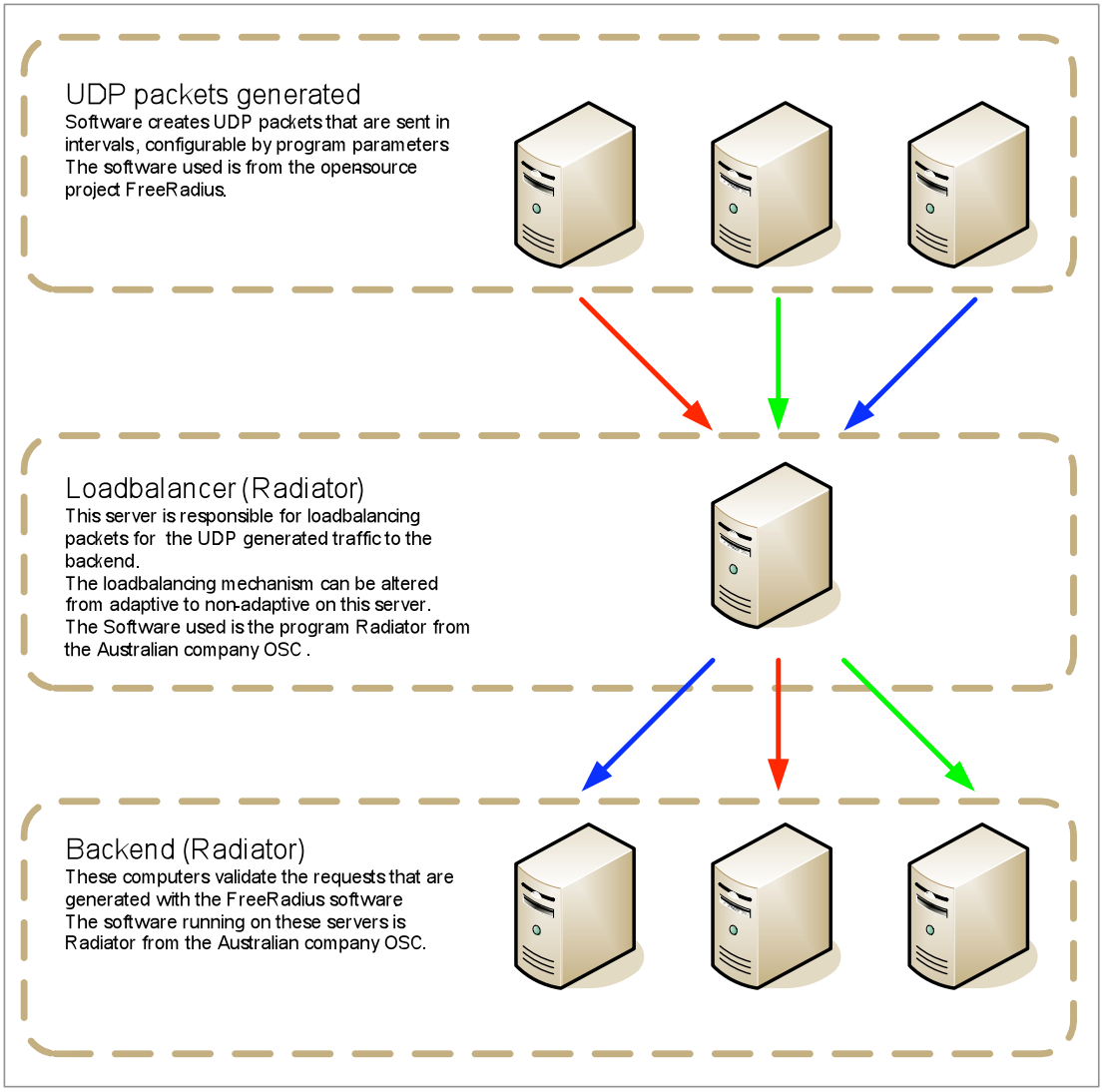


Figure 9 describes the lab setup for this study. The setup consists of three layers, the backend which authenticates requests, the load balancer which spreads the load to the backend and the generator servers which create authentication requests.

5.1.1 Backend (be)

The backend consists of the servers running radius authentication software from an Australian company named Open System Consultants (OSC). The codebase for this project is availible, but the software has to be purchased. All three servers are identical, running Debian GNU/Linux 4.0 with a bare minimum of services running.

5.1.2 Load balancer

The load balancer mechanism used is a built-in part of the Radiator software solution from OSC. It provides basic load-balancing alternatives, some of which are not interesting for this study, as they provide overhead or functionality that can be hard to compare.

This study will use the following load balancing mechanism:

- Round robin (non adaptive): Most elementary way of load balancing, namely a simple switch between servers. The first request goes to the first server, the second goes to the second server, etc.
- LoadBalance (weighted, adaptive): An adaptive load balancing method, that uses the average response time of the last ten requests to determine which server should receive the next request. This provides a way to send the next request to a server that has the least load, where the average response time of the previous ten requests was less than the other servers in the pool.

Other loadbalancing mechanisms provided by the Radiator software are:

- VolumeBalance (non adaptive): Uses a static-weighted system to load balance between servers that do not perform identically. Not of interest to this study as we use servers that are identical.
- HashBalance (non adaptive): Uses a hash system to ensure that multiple requests for the same user goes to the same server. Not interesting since the mechanism is used for other authentication methods.
- LdapRadius (non adaptive): Used to implement a dynamic setup of “backend servers”. A list is created in an LDAP server that contains all servers that are available for authentication requests. For each request that is sent to the loadbalancing server an LDAP query is made to check which backend server to use. Not interesting because of the added overhead and not providing any added functionality to the study.

The server running the load balance software is a RedHat 4.6

5.1.3 UDP request generation (ge)

For generating the UDP authentication requests, we have a mix of RedHat 4.6 and Debian GNU/Linux servers. All servers have a kernel from the 2.6 tree.

The software used is from the open-source project FreeRadius, the program is called radclient. It is a program that provides a way to generate radius requests from a file (or stdin), and send those out in controlled ways.

The main parameters used for this program are:

- Count (-c)
  Provides a way to send the same packet a number of times.

- Parallell (-p)
  Send a number of packets from a precompiled file in parallell.
- Limit (-n)
  Force the application to send a specific number of packets per second.

5.2 The simulations

The load simulations are run in a series of synchronised runs. So that when all the generate servers create load, the traffic is synchronised and starts generating UDP traffic at the same time. This is achieved by using the crontab service in Linux and having all computers synchronised to a network time service (NTP).

While the load tests are running the only changes made on the backend servers is that after each run, all radius servers are restarted, to remove any possibility that there is residue from the previous tests.

The only configuration changes that are made for the authenticating system, between simulations, are that the central load balancing method.

5.2.1 Benchmark test

The first tests are benchmark tests, run on each of the backend servers, to ensure that all reply to requests in a similar manner.

The test setup is as follows: Each generating server (ge1, ge2 & ge3) will try to authenticate 1,000 users' against each of the backend servers (be1, be2 & be3). The tests are synchronized, so that all 3 generating servers start authenticating at the same time, in total authenticating 3,000 users.

5.2.2 Adaptive test

The simulation is as follows: Each generating server (ge1, ge2 & ge3) will attempt to authenticate first 100, 1,000 and 10,000 users against each of the backend servers (be1, be2 & be3). The tests are synchronized, so that all three generating servers start authenticating at the same time, in total authenticating 300, 3,000 and 30,000 users.

Each test is repeated 3 times.

5.2.3 Non-adaptive test

The test setup is as follows: Each generating server (ge1, ge2 & ge3) will try to authenticate 1,000 users against each of the backend servers (be1, be2 & be3). The tests are synchronized, so that all three generating servers start authenticating at the same time, in total authenticating 300, 3,000 and 30,000 users.

Each test is repeated 3 times.

5.3 Measurements

To measure how long each authentication takes, we have to measure the total time to a certain amount of authentications. We use the total time divided by the number of requests to get the average authentication query time.

6 Simulation results

This chapter is divided into subchapters for each phase of the testing. The first section presents a basetest meant to control that the performance on all the backend servers is equal. The second section discusses the adaptive load balancing mechanism. The third section discusses the non-adaptive load balancing method.

6.1 Benchmark tests

Table 1 and figure 16 show the results individually to provide insight and confirm that the servers perform in a similar way.

	be1	***be2***	***be3***
simulation 1	63,85 ms	63,30 ms	63,49 ms
simulation 2	63,20 ms	63,25 ms	63,41 ms
simulation 3	63,05 ms	63,61 ms	63,75 ms
average	*63,37 ms*	*63,39 ms*	*63,55 ms*

Table 1 describes the base test. Each number represents a single authentication request response time.

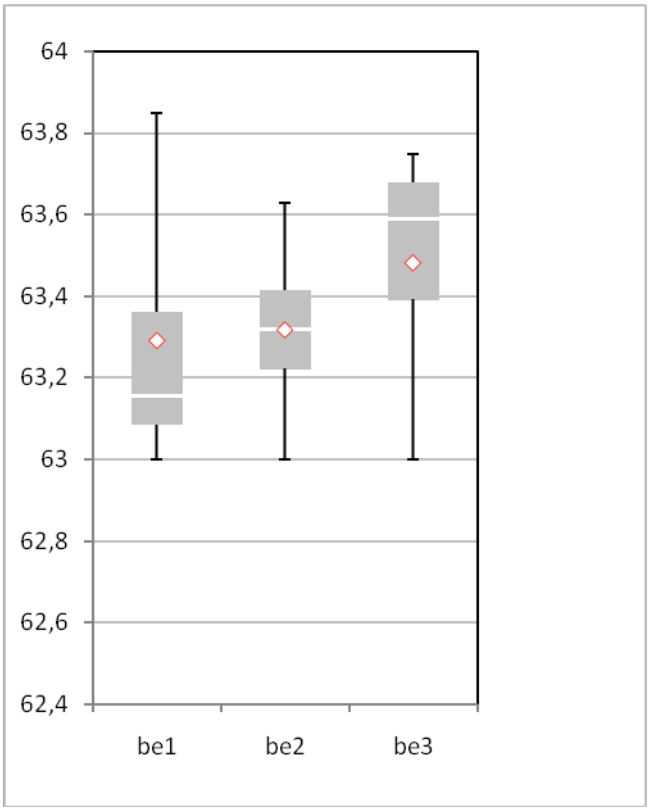


Figure 10 show the average response time per request for the basetest. This is the fastest time a single server can respond to a request, when the server has no load at all and is fed the requests slowly.

6.2 Adaptive test

The adaptive load balancing mechanism in the Radiator software attempts to send more load to the server that is the least loaded. The method to determine server load is based on the average response time. So if we look at two servers A and B, with an average service of 140ms for server A and 120ms for server B, server B will be chosen for the next packets that arrive. This should mean that server A's response time decreases, and it gets chosen for the next packets. If, for any reason server A has a problem or is overloaded, the packet flow will slowly be directed to server B while server A is recovering.

	300	*3000*	*30000*
simulation 1	165,8 ms	178,1 ms	179,1 ms
simulation 2	163,3 ms	179,6 ms	176,9 ms
simulation 3	166,8 ms	179,6 ms	178,0 ms
average	*165,3 ms*	*179,1 ms*	*178,0 ms*

Table 2 is the result of 27 simulations on the adaptive load balancer. Each number represents the average request time for a single authentication.

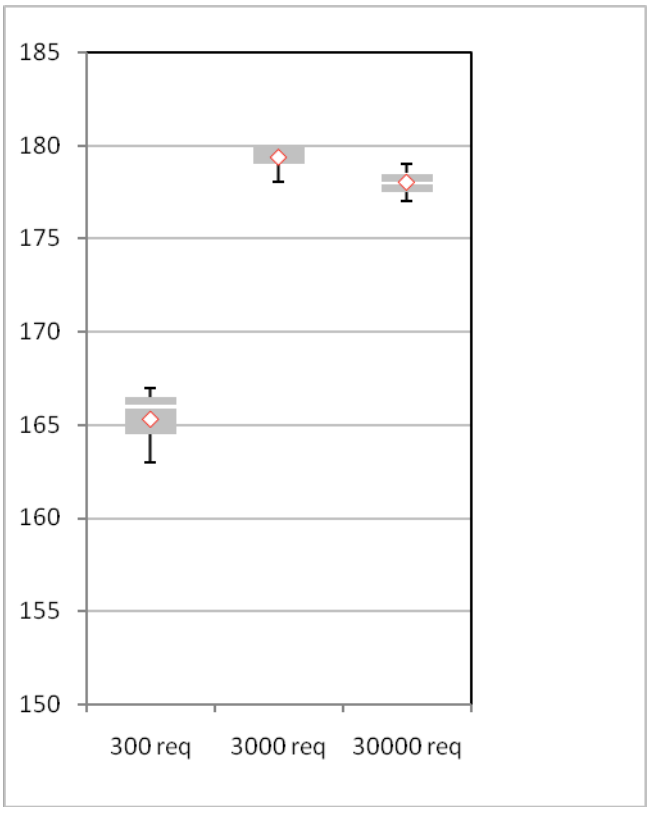


Figure 11 shows the results of tests on the adaptive load balancer. Each box represents the results from 3 tests of 300, 3000 and 30000 authentications. The average authentication request time for 300 authentications was 165ms, 3000 authentications was 179ms and 30000 authentications was 178ms.

6.3 Non-adaptive test

The non-adaptive load balancing method uses a simple round-robin method to choose the next server. The program keeps a pool of all servers that are operating, for instance A, B, C. If the last packet served went to server A, the next packet that comes will be sent to server B. If server B gets into a critical situation where its performance

decreases, the packetflow will not slow down, possibly causing even more degredation of service.

	300 requests	*3000 requests*	*30000 requests*
simulation 1	146,4 ms	145,8 ms	158,4 ms
simulation 2	141,5 ms	147,0 ms	160,7 ms
simulation 3	146,3 ms	145,7 ms	155,6 ms
average	*144,7 ms*	*146,1 ms*	*158,2 ms*

Table 3 is the result of 27 simulations on the non-adaptive load balancer. Each number represents the average request time for a single authentication.

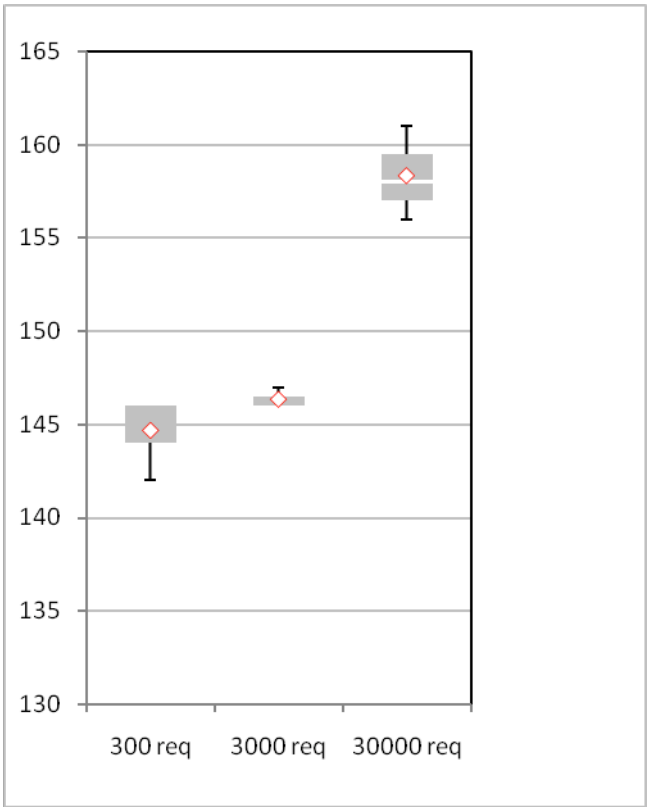


Figure 12 shows the results of tests on the non adaptive load balancer. Each box represents the results from 3 tests of 300, 3000 and 30000 authentications. The average authentication time for 300 authentications was 145ms, 3000 authentications was 146ms and 30000 authentications was 158ms.

6.4 Discussion and analysis

The results show that the round-robin algorithm provides a faster way of loadbalancing data. It is a simple way to load balance data between identical servers, which sole purpose is to handle specific services. However there might be other considerations that need to be taken into consideration, such as other services that might cause load on the server.

These simulations are a part of a quite narrow study, which only intent was to control the speed of the two loadbalancing methods. In a real-world environment, more variables must be taken into consideration. In many cases, the added cost of service time for requests on the adaptive load balancing method might be quite acceptable.

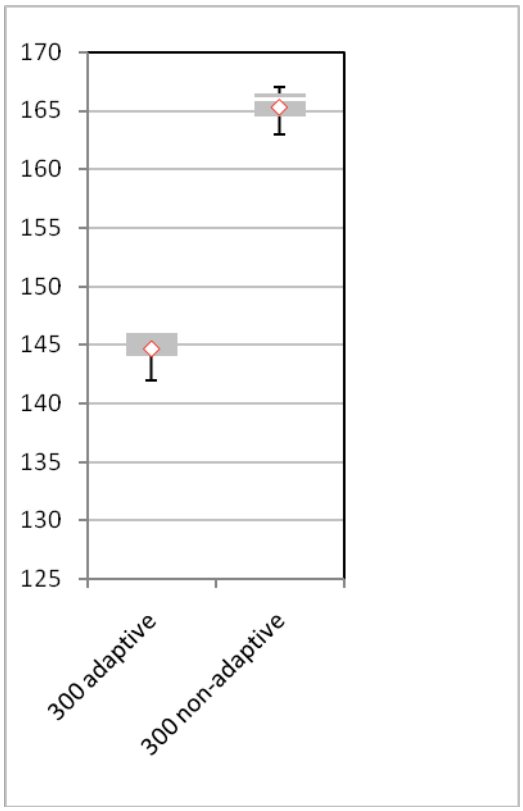


Figure 13 describes the difference in the average responsetimes of adaptive and non-adaptive load balancing methods with 300 requests.

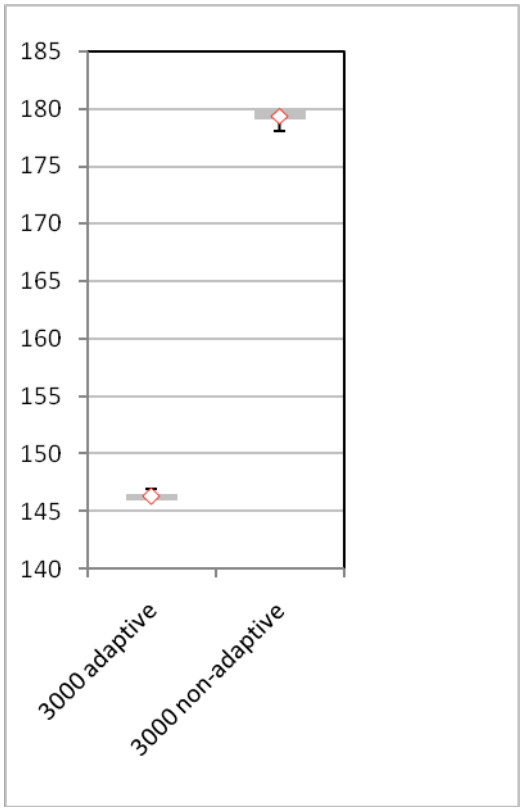


Figure 14 describes the difference in the average responsetimes of adaptive and non-adaptive load balancing methods with 3000 requests.

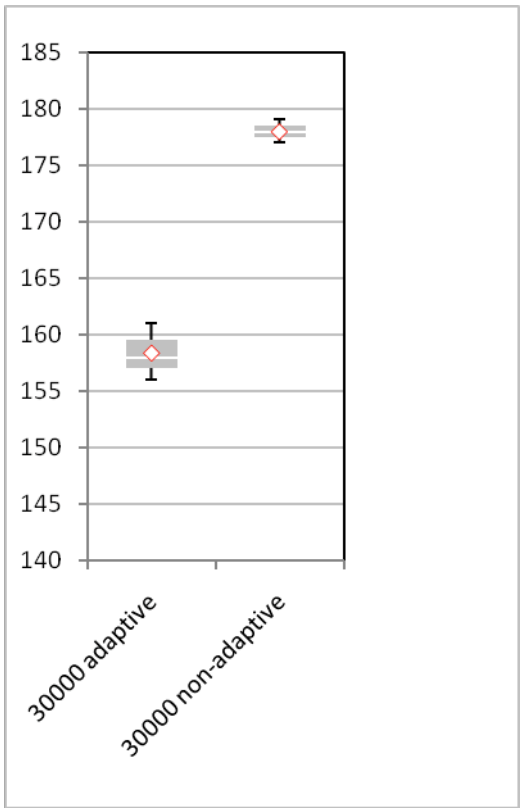


Figure 15 describes the difference in the average responsetimes of adaptive and non-adaptive load balancing methods with 30000 requests.

7 Conclusions

In this paper we have discussed a problem of increased load in authenticating servers. This problem can be addressed by employing a load-balancing method and sharing the load to several servers. To test this method we created a lab consisting of 7 servers. We installed software to simulate radius authentications and radius software to load-balance and reply to the requests. The load-balancing mechanism was changed to compare an adaptive algorithm to a non-adaptive.

As can be seen in figure 16 the non-adaptive methods of load balancing are better in all cases. The reason for this is probably the added time taken to compute the average response time to determine which server should receive the next request.

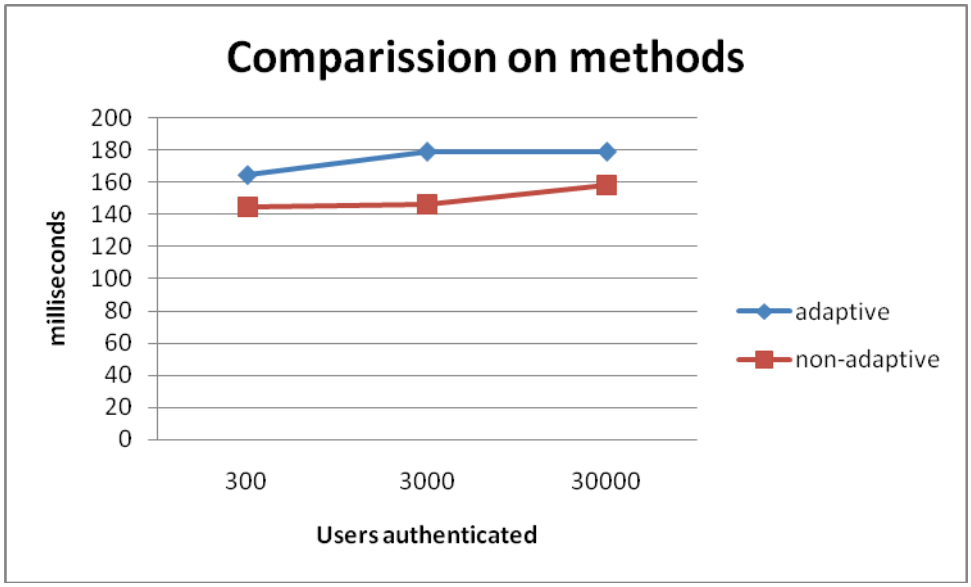


Figure 16, comparission between loadbalancing methods

In both cases, as load was applied for longer periods, the algorithms perfomed a little worse.

7.1 Future work

A very important aspect of the different loadbalancing methods is that the non-adaptive way is extremely efficient. There is no overhead as is the case with the adaptive method. An interesting study would be to change the adaptive load balancer and try to enhance the algorithm. A simple way to do this is to change the algorithm that measures the last requests to something less. This might be enough to shorten the service time of the adaptive load balancing method enough to outperform the round-robin method.

Another interesting aspect would be to receive more information on each authentication packet, such as the query time of each packet. This would enable a more thorough description of the examples that are mentioned in the study.

In all cases in this study, we have simulated a burst flow. That is, we send packets as fast as we can, from 1-3 different servers. This is done to test the functionality of the load balancer under maximal load. An interesting aspect of this study would be to create a new test that tests the functionality of the load balancers for each packet. That is measure the request time for each packet under different load balancers and different bursts.

References

Balasubramanian J., Schmidt D. C., Dowdy L., Othman O. (2004), Evaluating the performance of middleware load balancing strategies. *Procedings of the 8th IEEE Intl. Enterprise Distributed Object Computing Conference (EDOC).*

Calhoun P., Airespace inc, Loughney J., Nokia, Guttman E., Sun Microsystems, Zorn G., Cisco Systems, Arkko J., Ericsson (2003), *RFC 3588 Diameter Base Protocol,* Internet Engineering Task Force.

Carter G. (2003), '*LDAP system Administration'*, O'Reilly & Associates, p 4 – 8.

Comer D. E. (2006). *Internetworking with TCP/IP: Principles, Protocols, and Architecture*, Prentice Hall 2006.

Coulouris G., Dollimore J., Kindberg T. (2001), *Distributed Systems: concepts and Design*. Addison Weasly.

Famenkov M., Keys K., Moore D., Claffy K. (2003), Longitudinal study of Internet traffic in 1998-2003, CAIDA Tech Report.

Kima M., Kimb S., Kongc A. J. (2007), High performance AAA architecture for massive IPv4 networks, *Future Generation Computer Systems, 23:2*, 275–279.

Lottor K., (1992). *RFC1296 Internet grow*th 1981-1991.

Metz, C. (1999), *AAA Protocols: authentication, authorization and accounting for the internet*. IEEE Internet Computing, 3(6), Nov pp 75-59.

Postel J. (1980), *RFC768 User Datagram Protocol*. Internet Engineering Task Force.

Rigney C., Livingston, Rubens A., Merit, Simpson W., Daydreamer, Willens S. (1997), *RFC2058 Remote Authentication Dial In User Service (RADIUS)*. Internet Engineering Task Force.

Glossary

AAA
Authentication, Authentication and Accounting, A concept commonly used for authentication servers.

DIAMETER
A AAA protocol developed in 2003.

DNS
Domain Name System A system to resolve computer host names and IP addresses.

DSL
Digital Subscriber Line A method to provide broadband connections to consumers.

DSLAM
Digital Subscriber Line Access Multiplexer The equipment that enables DSL connections.

IAB
The Internet Architecture Board is a committee that is handles oversight of the technical and engineering development of the Internet.

IETF
The Internet Engineering Task Force developes and standardizes Internet standards.

ISP
An Internet Service Provider, is a provider of Internet service.

LDAP
Lightweight Directory Access Protocol is a protocol for querying directory services such as a X500 database.

NAS
A Network Access Server is a gateway between networks. In the concept of this report it enables end users to access the Internet.

NSP
Network Service Provider a provider of network service. This can be access to the Internet or any other network such as a company intranet.

NTP
The Network Time Protocol is a protocol developed to synchronize clocks between computers.

PPP
The Point to Point Protocol is a data link protocol used to establish a direct connection between two computers.

PPoE
The Point-to-Point Protocol over Ethernet protocol is a method of encapsulating PPP packets inside Ethernet.

RADIUS
The Remote Authentication Dial In User Service is a AAA protocol.

RFC
The Request For Comment (RFC) documents are the official publication channel for the Internet Engineering Force (IETF), Internet Architecture Board (IAB) and the community of computer network researchers.

TACACS
Terminal Access Controller Access-Control System is a AAA protocol.

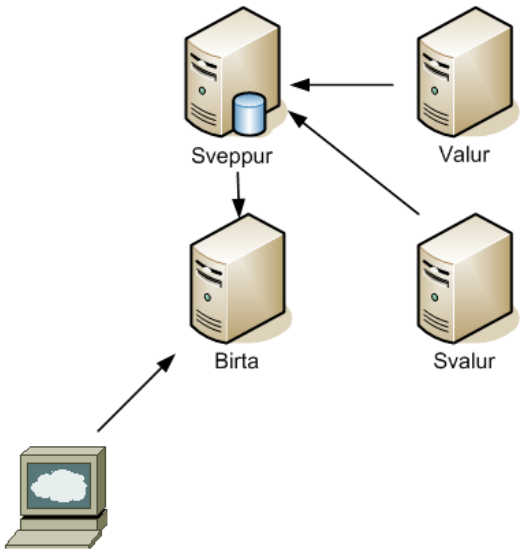
TCP
Transmission Control Protocol is one of the fundamental transmission protocols on the Internet. The protocol is statefull.

UDP
User Datagram Packet is one of the fundamental transmission protocols on the Internet. The protocol is stateless.

Appendix A: Radiator database structure

This appendix depicts the database structure for Iceland Telecoms solution. The database server used by the current setup is MySQL

Database flow structure and replication method



The database structure in the system consists of 2 MySQL servers, Sveppur serving as a dedicated primary server, which Valur and Svalur access directly. Each request (Authentication request or Accounting information) that Svalur and Valur receive is stored in the SQL database using insert and update commands. The option of using a single master SQL server is to avoid the complicity of having to implement an active Master-Master cluster, where there are two active servers and changes can be sent to any one of them. In order for this to work the servers must have a internal mechanism that synchronizes changed data.

The replication is native to the MySQL server and is based on the master server keeping track of all changes to the primary database (Sveppur). The slaves then fetch changes that have happened to the master server from their previous state.

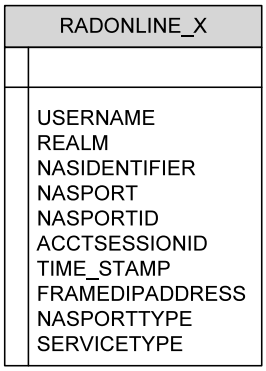
All tables on Sveppur are replicated to Birta, which acts as a secondary database server, used by external applications that need access to data in the radius system (e.g. user management and helpdesk personal). With all external SQL read (select) request moved from the main database server, load is reduced and access times for Svalur and Valur should be optimal as all applications that do not have to change the database use Birta.

Table structure

The main database tables are:

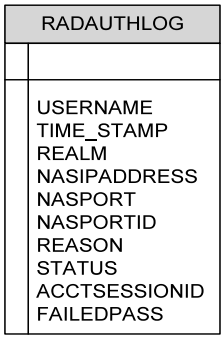
RADONLINE	x	5 MB
RADAUTHLOG	1413794	215 MB
RADIUS_FULL	27819782	4.223 MB
ACCOUNTING	5215336	6.896 MB

Currently connected users



The table which stores all currently connected users is called RADONLINE. It keeps track of all basic connection statistics; such as the username, his realm, the assigned IP address, which NAS the user is connected through, which hardware address the connection resolves from, the session ID (assigned by the NAS), timestamp of the connection and what type of connection the user uses.

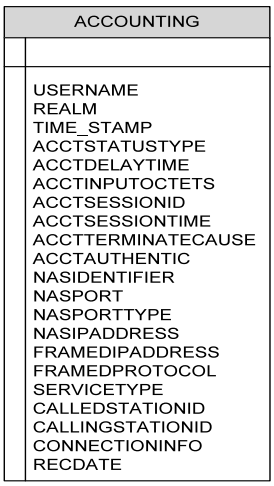
Authentication attempts



The table which stores all authentication requests attempted is RADAUTHLOG. Each time a user attempts to authenticate to the system an attempt is stored in this table (even if it is unsuccessful).

This table stores the username used to authenticate, a timestamp for the attempt, the users realm, the NAS involved, the hardware address of the user, reason for rejecting authorization (if any) and a session ID.

Radius accounting



Radius accounting packages are stored in the table ACCOUNTING. The NAS are responsible for sending accounting information for each user connected at the time the user logs in, update information at regular intervals and finally a stop packet when the user disconnects. These packets include information such as the username, status type of connection, time of sign in, information about the transmissions of the user, session ID, and information about how the user is connected.

Session information

Session information is stored in a table called RADIUS_FULL. These records consist of whole user sessions, where a single record in the database represents a user session. Not all sessions in the table are completed however, as users that are still online haven't disconnected yet.

This table is created and maintained by software designed and implemented internally at Síminn.