University of Amsterdam

System & Network Engineering RP1

Defending against DNS reflection amplification attacks

February 14, 2013

Authors:
Thijs Rozekrans
<Thijs.Rozekrans@os3.nl>

Javy de Koning
<Javy.deKoning@os3.nl>

Supervisor:
Matthijs Mekking
<Matthijs@nlnetlabs.nl>

Abstract

The goal of the research described in this paper is to find out if the proposed mechanisms to defend against a DNS amplification attack are effective. The decision is made to focus on Response Rate Limiting (RRL) and determine the effectiveness of this mechanism against current and future attacks. In order to determine the effectiveness of RRL a repeating (ANY) query attack, which is currently the most popular attack, is simulated. This basic attack is followed up by four more sophisticated attacks. The effectiveness of RRL is measured by comparing the DNS servers in- and outbound traffic with and without RRL activated. When analyzing the results it becomes clear that the effectiveness of RRL decreases when the attack becomes more sophisticated. Because RRL is ineffective against a more sophisticated attack, another proposed defense mechanism is briefly discussed called DNS dampening. The results show that this mechanism is effective against sophisticated attacks but is missing some essential features which makes it impractical to use in a live environment. The main conclusion is that RRL is a proper defense against current amplification attacks, but it is not effective against future more sophisticated attacks.

Contents

1. Introduction

A DNS amplification attack is a type of distributed denial of service (DDoS) attack that takes advantage of the fact that a small DNS query can generate a much larger response. An attacker can direct a large volume of network traffic to a victim's system by initiating relatively small DNS queries. The attacker spoofs the IP address of the victim to reflect the network traffic using the DNS server. This makes it difficult to trace the attacker.

At first, resolvers were used for reflecting traffic but the internet community has recently seen an increase of amplification attacks that making use of authoritative name servers. One reason for this is that more resolver operators are following the access restriction guidelines from RFC5358[3], greatly reducing the chance that their name server can be used in a reflection attack. Attackers now making use of authoritative name servers which, by design, cannot follow these guidelines.

These DDoS attacks are becoming more sophisticated, making it hard for packet filters to catch the traffic. This urges to push the filtering towards the name server, so that the defense mechanisms can also become more sophisticated.

Response Rate Limiting (RRL) is currently the only practical defense mechanism available for filtering in the name server. There is a patch for BIND and the proposal is implemented by NLnet Labs for NSD. RRL is in a state where the effectiveness has not been exhaustively researched yet.

Research regarding the effectiveness of the available RRL implementation can help organizations responsible for the authoritative name servers to make decisions on what measures to take against amplification attacks. Since those organizations involuntary contribute to the attack, it is unclear in what way organizations can be held responsible if their name servers are abused for an attack. Since these attacks are appearing more often, the need for a defense mechanism becomes more important.

1.1. DNS amplification attack

In order to launch a DNS amplification reflection attack the attacker needs to perform two tasks. First the attacker spoofs the address of the victim. This is the reflection part, it will cause all the reply's from the DNS server to be directed to the victim's server. This can easily be done since in UDP no handshake (like in TCP) is being done between the client and the server. Secondly the requester searches for responses that are several times bigger than the request. The attacker achieves an amplification factor because the response is many times larger than the request. The amplification can even be larger when DNSSEC is used, because of the signatures used the size of the response increases.

Now the attacker is ready to perform the attack. The attacker sends a stream of small queries originating from a group of infected computers (referred to as a botnet) to one or multiple authoritative DNS servers. The DNS servers will then reply to the resolver. However, because the attacker spoofed the address of the victim, all the traffic is directed to the victim.

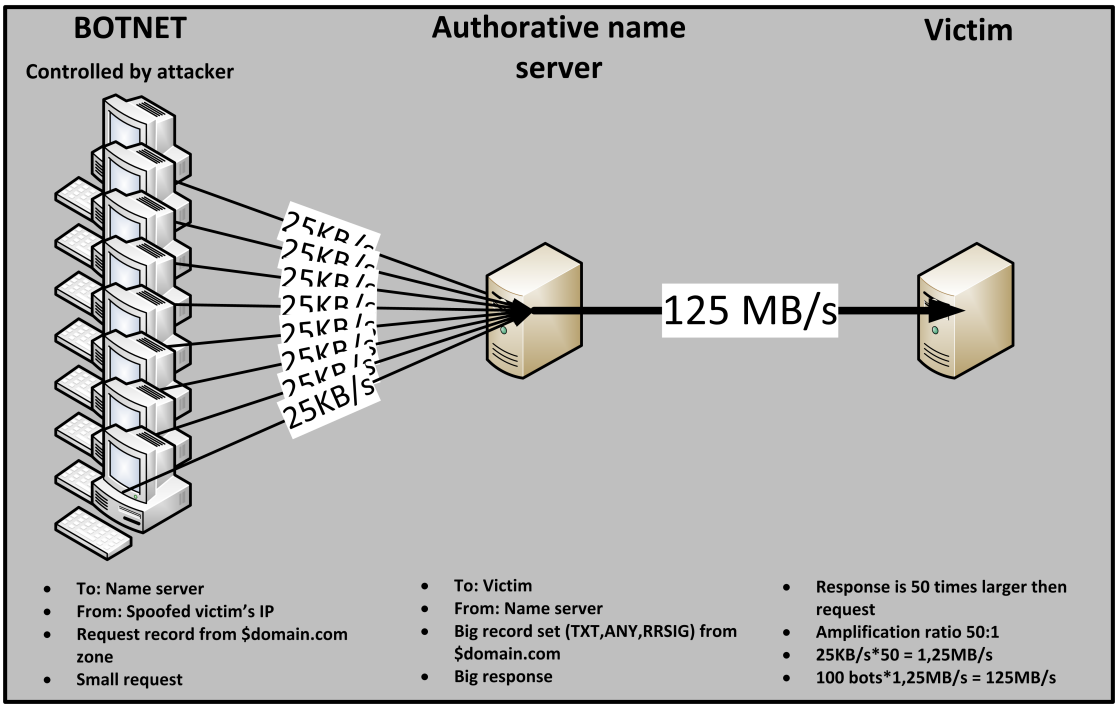


Figure 1-1: This example represents a bot-net containing 100 machines sending requests at a rate of 25KB/s and the name server responses with 125MB/s of traffic which is directed to the victim. This results in an amplification factor of 50:1

The victim gets overloaded with the amount of traffic send to it and possibly can not make use of the internet connection anymore. Not only the bandwidth can be exhausted but also the resources on the clients machine. The clients machine can be so busy processing the incoming traffic that is exhaust the resources, this could lead to a halt of the clients machine. So a DNS reflection amplification attack could lead to two types of Denial of Service.

1.2. Research questions

This research will focus on properly defending against these types of attacks, which results in the following research question:

"What measures can be taken to defend against DNS amplification attacks on authoritative name servers, and what is the effectiveness of one of Response Rate Limiting?"

Sub-questions

In order to completely answer the research question, it can be split into the following sub-questions:

- Which defense mechanisms are currently available?
- How can the effectiveness of RRL be measured?
- What can we do to prevent false-positives?
- How do we classify traffic that is used for an attack?
- Which RRL configuration parameters are the most effective?

The defense mechanisms that are currently available will be discussed in section 3. Section 4 will explain the method that was used to measure the effectiveness of RRL. Measurements and the classification of attack traffic are shown in section 5. Recommended configuration settings and a method to prevent false positives is presented in section 6.

1.3. Related Work

Research has, amongst others, been done by Randal Vaughn & Gadi Evron in 2006[1] and by Duane Wessels in 2007[2]. However, both researches focus on amplification attacks on recursive name servers, the proposed solutions[3] are not applicable on authoritative name servers.

2. Attacks

This section describes the most common DNS amplification attacks and their characteristics. The attacks can be divided into three types, repeating queries, varying queries and a distributed attack. All attacks require the ability to spoof IP address using UDP. Another requirement is that the response is larger than the request to create the amplification. This is often achieved by querying for ANY, DNSKEY, NS or RRSIG records.

For the examples presented in this section a DNS setup with different zone files is used. A more detailed description of these zone files is included in appendix C.

2.1. Repeating query attack

A repeating query attack will request the same record (set) over and over again. Usually an ANY query is used. An ANY query returns all the records for a specific domain name regardless of the record type. When sent to a recursive server, the server will only return the records that it has cached. The server will have to reply, regardless of available recursion. This is currently the most common attack because the ANY request usually returns a large collection of resource records, creating a high amplification ratio.

```
1 jkoning@prague:~$ dig @localhost prague.studlab.os3.nl ANY +
     stats +dnssec | grep -i "size"
2 ;; MSG SIZE  rcvd: 3837
```

Listing 2-1: ANY Query response size.

```
1 11:15:13.707111 IP localhost.45176 > localhost.domain:
     28919+ [1au] ANY? prague.studlab.os3.nl. (50)
2 11:15:13.707311 IP localhost.domain > localhost.45176:
     28919* 18/0/7 SOA, RRSIG, NS ns2.prague.studlab.os3.nl.,
     NS ns1.prague.studlab.os3.nl., RRSIG, A 145.100.104.57,
     RRSIG, MX mail2.prague.studlab.os3.nl. 20, MX mail.prague
     .studlab.os3.nl. 10, RRSIG, TXT "v=spf1" "ip4
     :145.100.104.57" "mx:mail.prague.practicum.os3.nl" "a:sub
     .prague.practicum.os3.nl" "mx:sub.prague.practicum.os3.nl
     " "-all", RRSIG, NSEC, RRSIG, DNSKEY, DNSKEY, RRSIG,
     RRSIG (3837)
```

Listing 2-2: TCP dump output.

From the example above notice that the request is 50 bytes and the response is 3837 bytes. This results in an amplification factor of $\frac{3837}{50} \approx 77$. These attacks are usually easy to detect because the exact same queries are being re-used. This is uncommon behavior because the recursive DNS server is expected to cache the records. If ANY requests are blocked the attacker might switch to RRSIG, DNSKEY or any other query that generates a large response.

2.2. Varying query attack

When attackers notice that the attacks mentioned in section 2.1 are being mitigated they may have to switch to a more sophisticated attack. The varying query attack sends queries for varying domain names to the DNS server. By sending different unique requests the attacker tries to generate unique responses making the attack traffic less obvious and therefore harder to detect and counter. Even if the domain name cannot be resolved, an NXDOMAIN response is returned. An NXDOMAIN response is an error message indicating that domain name does not exist. When DNSSEC is configured the server also returns one or more NSEC(3) records which are used to prove that the

requested domain name does not exist. Together with RRSIG signatures this generates a big amplification as well.

```
1 jkoning@prague:~$ dig @localhost random390931.prague.studlab
     .os3.nl A +stats +dnssec | grep -i "size"
2 ;; MSG SIZE  rcvd: 1153
```

Listing 2-3: Response size with NSEC.

```
1 14:50:23.863425 IP localhost.58708 > localhost.domain:
     11083+ [1au] A? random390931.prague.studlab.os3.nl. (63)
2 14:50:23.863665 IP localhost.domain > localhost.58708: 11083
     NXDomain* 0/6/1 (1153)
```

Listing 2-4: TCP dump output.

From the example above notice that the request is 63 bytes and the response is 1153 bytes. This results in an amplification factor of $\frac{1153}{63} \approx 18$. The response is not as big as with the ANY request but it is still disturbing. When NSEC3 is used the amplification ratio is even larger.

```
1 jkoning@prague:~$ dig @localhost random390931.prague.studlab
     .os3.nl A +stats +dnssec | grep -i "size"
2 ;; MSG SIZE  rcvd: 1604
```

Listing 2-5: Response size when using NSEC3.

```
1 10:47:09.923850 IP localhost.47262 > localhost.domain:
     30527+ [1au] A? random390931.prague.studlab.os3.nl. (63)
2 10:47:09.924226 IP localhost.domain > localhost.47262: 30527
     NXDomain* 0/8/1 (1604)
```

Listing 2-6: TCP dump output.

The results above show that the amplification factor in this specific case grows, to $\frac{1604}{63} \approx 25$ when using NSEC3.

2.3. Distributed attacks

The attacks mentioned in 2.1 and 2.2 could be further enhanced by distributing the attack traffic over multiple DNS servers. An example could be an attack that is targeting multiple DNS servers that are hosting a lot of signed zones. By distributing the queries over zones and servers the attacker can try to stay hidden from the proposed defense mechanism mentioned in section 3.4. These 'intelligent' attacks are making it more difficult to detect and counter attack traffic.

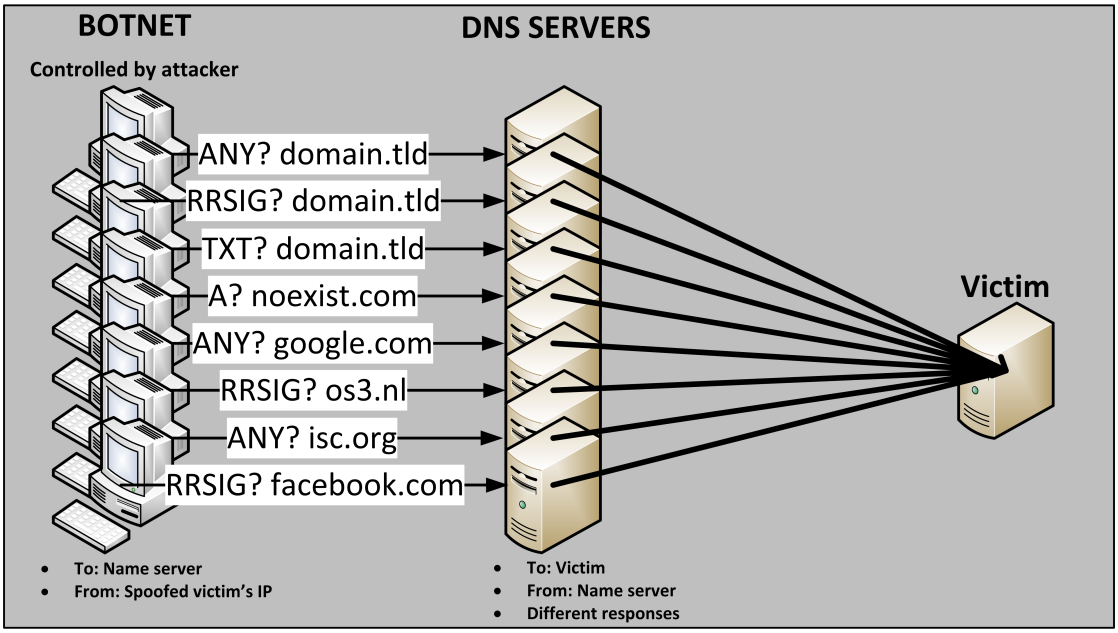


Figure 2-1: Example of a more advanced DNS amplification attack.

3. Defending against amplification attacks

The purpose of the presented solutions is to protect the innocent victims from a DNS amplification attack caused by spoofed UDP queries. Apart from the defense mechanisms described in this section a couple of other solutions have been proposed. Orphan Detection[8] and Stream Control Transmission Protocol[9] will not be covered in this paper. SCTP is suggested to replace UDP in the DNS protocol which is impractical because it will introduce significant side effects. A proper defense mechanism will have to remove the amplification or the reflection factor, while continuing to service legitimate requests.

3.1. Firewall

An obvious example of a defense mechanism against an amplification attack is the use of a firewall. Most infrastructures already have a firewall installed and it can easily be configured to block specific packets or IP addresses. A firewall can be configured to block all ANY requests. Since most environments do not rely on these queries this would cause little harm. The main drawback of this approach is that it would probably cause false positives and block legitimate traffic as well. Another drawback is that attackers can easily switch to other DNS queries that cause large amplifications like RRSIG, DNSKEY etc. If the attack becomes more sophisticated another defense mechanism is needed.

3.2. BCP38

An amplification attack is a type of DDoS attack that relies on a spoofed IP address. The main reason for this is that without IP address spoofing an attacker cannot let the DNS server reflect traffic to the victim. A spoofed address also makes it hard to trace back the attacker.

BCP38[5] is a mechanism which allows routers to check the validity of an IP address. A customer is assigned an IP address by the internet service provider (ISP). The ISP can then check incoming packets for a valid IP address. If the IP address does not match the the range that the customer should have, the traffic can be dropped. This mechanism prevents spoofing IP addresses which reside outside the range of the customer or the ISP depending on where BCP38 is implemented. If BCP38 was to be implemented by the majority of ISP's throughout the internet, it will prevent IP addresses from being spoofed.

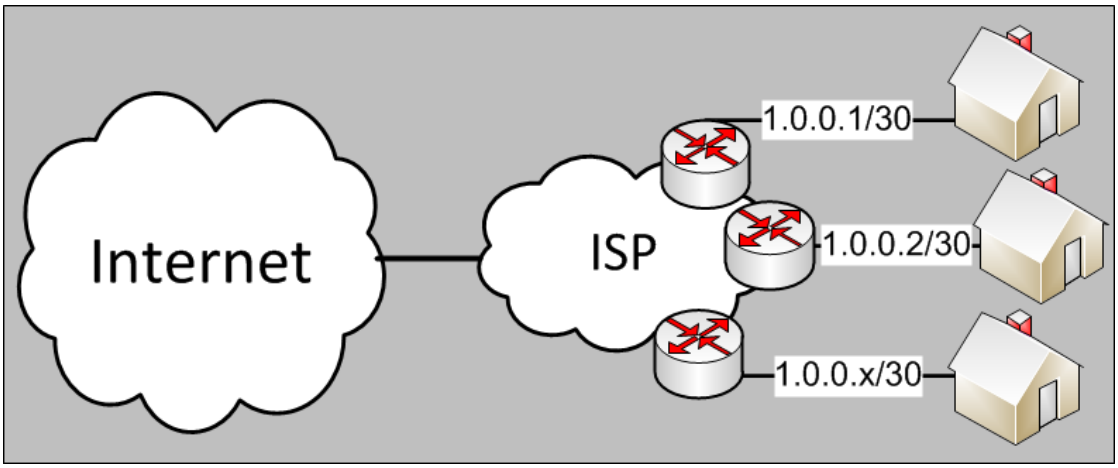


Figure 3-1: Overview of the BCP38 solution

Figure 3-1 gives an example of a BCP38 implementation. On one side the ISP is connected to the internet, on the other side it is connected to their customers. The ISP has several IP prefixes which it can assign to their customers. If the ISP has BCP38 implemented on their customer routers, only packets originating from their customers range would be allowed to pass. If IP addresses do not match the range the router will drop the packets. This costumers (who might be part of a botnet) are unable to spoof IP addresses outside their range.

3.3. DNS dampening

DNS dampening[10] is based on BGP route flap dampening[11]. The basic idea behind DNS dampening is to collect penalty points per requester based on the query type, the size of the response and other parameters. If the penalty points reach the configured limit, dampening will start. While in dampening state the server drops all queries from

the spoofed IP address. Penalty points will be decreased exponentially over time. When the penalty points drop below a secondary limit dampening will stop and the server will start to process requests again. The storage required to keep state of the clients must be in memory only for fast processing.

Incoming requests are classified and penalty points are assigned based on the factors mentioned above. Every query receives penalty points. For example, an ANY request gets assigned 100 points because these queries are often used in attacks and rarely used for legitimate requests and another 100 penalty points are added when a query is repeated with the same query ID. For legitimate requests the query ID is a randomly chosen 16 bit value. Duplicate query ID's should only occur on a regular basis during attacks. Finally additional penalty points are assigned based upon the size of the response. The drawback of this approach is that it provides no mechanism to counter false positives. This could lead to a legitimate client to be blocked from the DNS system at all.

3.4. RRL

Response Rate Limiting[12] is a mechanism for limiting the amount of unique responses returned by a DNS server. This can limit the effectiveness of a DNS amplification attack by dropping responses that exceed the configured rate limit.

Rate limiting works as follows:

- When the server generates a response to a DNS query, the requesters IP addresses are grouped into buckets. By default IPv4 addresses on the same 24 subnet are put into a single bucket. The same counts for 56 IPv6 addresses.
- The wild-card, zonename or query name is stored together with the IP address and the boolean error indicator (response code: REFUSED, FORMERR or SERVFAIL).
- The server uses <(IP), (NAME), error-status> to decide how to act upon a request. This information is used as the state.
- If the amount of unique responses exceed the configured limit, the server drops requests for a specific IP or network. The server can also ask the requester to retry using TCP instead (called SLIP).
- Note that the server is only looking at the given responses and ignores the amount of incoming requests.

When using RRL the victim might still notice it is under attack, because it receives DNS responses for which no request was sent out for a limited time. An attacker might also be able to circumvent this defense mechanism by distributing it's attack over a large number of DNS servers, to stay under the RRL limits of the DNS servers.

3.5. Summary

Several defense mechanisms have been discussed, some more promising than others. Although firewalls are available everywhere and the rules are simple to implement, they can only guard against very basic attacks. BCP38 would be the ultimate solution because it will prevent IP address spoofing, unfortunately it is unlikely to be implemented throughout the whole internet in the near future. DNS dampening seems to be a very promising defense mechanism. However, in it is current stage there is no mechanism available to avoid false positives. Most attacks that are currently detected on the internet use repeating queries. RRL could be effective by limiting repeating responses, and might be an effective solution. This research will focus on RRL because this mechanism looks the most promising.

4. Research method

In order to completely answer the research question, real-world attacks have to be simulated. The repeating ANY attack is currently the most encountered attack on the internet. Therefore it is relevant to know how effective RRL is against this kind of attack. Attacks are abusing all kinds of authoritative DNS servers: TLD's, webhosters and organizations hosting their own "small" zone. Defense mechanisms currently in use specifically defend against repeating query attacks. When more DNS servers mitigate these attacks it is only a matter of time before attackers are going to find and use more sophisticated methods. A more sophisticated attack which can be thought of is querying for varying domain names and possibly combining this with querying different record types.

A TLD name server usually has one large zone file containing a large collection of domain names. For example the name server responsible for all .nl domain names contains more than 5 million[1] individual domain names in total. Before an attacker can perform a varying query attack, some information about the domain names present in the zone is required. An attacker could easily index domain names by using a webcrawler or using some type of dictionary attack. By creating 5 attacks, that differentiate in the amount of resolvable domain names, a wide selection of attack scenarios is simulated. The first attack results in 0% resolvable domain names and 100% NXDOMAIN responses, the last attack results in 100% unique resolvable domain names and 0% NXDOMAIN responses. The attacks will differ in 0%, 25%, 50%, 75% and 100% resolvable domain names. Details can be found in appendix C.1.

Compared to a TLD a "web" hosting company usually hosts a large collection of smaller zones. For example the largest web hosting company called GoDaddy manages 54 million domain names [2].

Because the hardware available in the lab setup is not capable of hosting 54 million domain names, 1000 individual zone files are created in order to imitate a hosting company. A script is used to generate the zones, the details can be found in appendix C.3. The reason that hosting companies are good amplification targets is the variety in domain names, by querying a different domain name each time an unique response is returned. Because limited time is available for this project the measurements will focus on a TLD like DNS server. The same attack will also be tested against a "small" zone . This zone type contains less than 20 resource records. Details can be found in appendix C.2.

This research focuses on measuring the effectiveness of RRL and finding the optimal parameters. Focus will be on the SLIP parameter because this parameter directly effects the change for false positives and the amount of outbound traffic. In order to replicate a realistic amplification attack, every scenario makes use of DNSSEC signed zones. The research is done in the lab of NLnet Labs which they made available for this research. The lab consist of the following three servers:

[1] `https://www.sidn.nl/,28/1/2013`

[2] `http://www.godaddy.com/newscenter/about-godaddy.aspx?ci=9079`

- **Attacker**, hathi.nlnetlabs.nl;
- **DNS server**, kaa.nlnetlabs.nl;
- **Victim**, balou.nlnetlabs.nl.

Because the limited amount of hardware available to imitate a bot-net to generate the traffic, packet capture (pcap) files are created, modified and replayed to simulate an attack. The IP address of the player is replaced by the IP address of the listener. The amplification attack is executed by replaying modified pcap files from the player to the name server. BIND is used as the name server daemon. The reason for using BIND is that it is by far the most widely used DNS software on the Internet[3]. The name server will then respond to the request and sends the responses to the listener. In- and outbound traffic is measured on the DNS server using a network monitoring tool named Cacti, this tool can easily be set up and a graphical representation of the measurement can be created. A graphical overview of the lab environment can be find below and a detailed description of the scripts and applications used can be found in appendix D.4 and D.

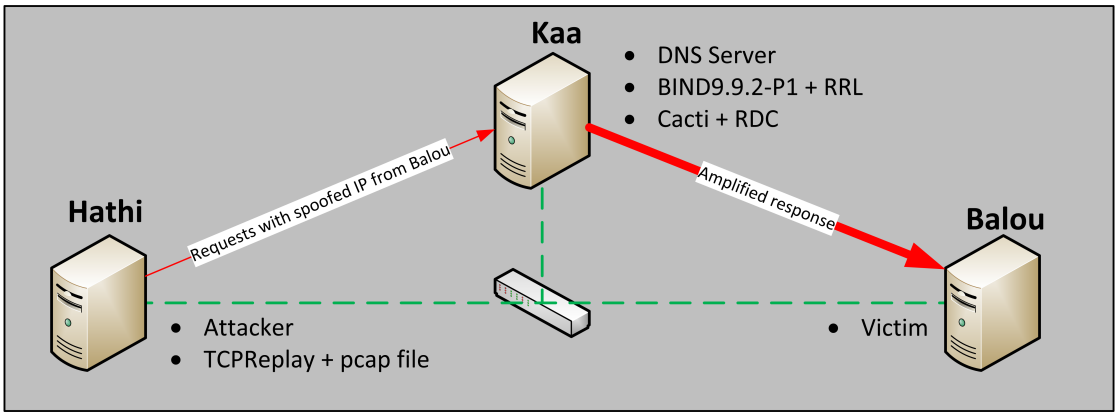


Figure 4-1: Graphical overview of the lab environment

[3] https://www.isc.org/software/bind

5. Measurements

The goal of this research can be divided into two sections. The first one is to find out how effective response rate limiting is. The second part focuses on finding the best parameters to set for the environments described in section 4. To understand the results of this research it is important to understand the following parameters:

- **RESPONSES-PER-SECOND** is a limit on identical responses. The default limit of zero specifies an unbounded limit to turn off rate limiting in a view or to only rate limit NXDOMAIN or other errors. All responses to a specified network with IPv4-PREFIX-LENGTH (Default 24) or IPv6-PREFIX-LENGTH (Default 56) are assumed to come from a single DNS client.
- **NXDOMAINS-PER-SECOND** by default the limit on NXDOMAIN errors is the same as the responses-per-second value, but it can be set separately.
- **ERRORS-PER-SECOND** The maximum amount of error (REFUSED, FORMERR and SERVFAIL) that can be returned to the requester.
- **WINDOW** When any per-second limit is exceeded the source can be penalized for up to WINDOW seconds.
- **IPv4-PREFIX-LENGTH** All responses to a network block with a given prefix length are assumed to come from a single DNS client. This parameter sets the prefix length for IPv4.
- **IPv6-PREFIX-LENGTH** All responses to a network block with a given prefix length are assumed to come from a single DNS client. This parameter sets the prefix length for IPv6.
- **SLIP** When the RRL mechanism is activated and queries are being dropped, a small response claiming that the response would have been truncated is sent randomly once per SLIP query. This allows a legitimate client to reconnect over TCP. Because the SLIP response is approximately the same size as the request it is unattractive for abuse.
- **MAX-TABLE-SIZE** RRL needs to keep state of the unique responses in order to be able to assign penalties. This entry sets the maximum amount of entries (called state blobs) which can be stored at the same time. This should be set to the product of the window size and maximum queries per second. 10000 state blobs should take about one megabyte of server memory. $MaxQPS * Window = Tablesize$.
- **MIN-TABLE-SIZE** Since growing this table has a cost, an operator might decide to start with a larger than default size table.

Not all parameters will be researched in depth because some have a limited influence on the effectiveness of RRL. The RRL parameters will need to be tailored for each specific environment. A recommendation for the configuration will be given in section 6.2.1. The effectiveness of RRL is highly depended on the SLIP parameter because it directly affects the chance for false positives and the amount of outbound traffic. In the following sections the effectiveness of RRL with differing SLIP settings will be measured in the scenarios mentioned in section 5.

5.1. Repeating query attack measurements

Section 2.1 mentioned that repeating query attacks often target authoritative name servers. In this section a similar attack is simulated against a small zone, a TLD-like zone and a DNS server that hosts multiple zones like a hosting provider. The details of these zone configurations can be found in appendix D.

5.1.1. Single Zone

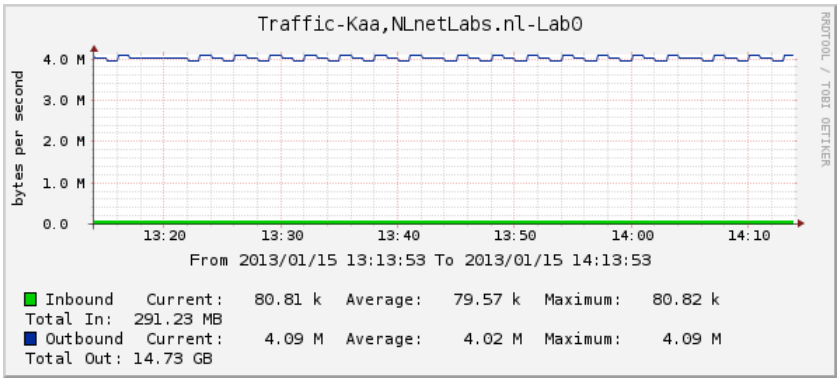
To answer the behavior of the RRL during an attack can best be explained using the following table format which is used table 5-1. The first column shows the SLIP setting. The second column shows the chance of a possible false positive. This is calculated based upon the SLIP setting. A response is send once every SLIP query. Therefore a request has an one out of two chance to get a response with a SLIP setting of two. The relation between the SLIP setting and TCP responses will be further explained in 6.2.4. The third and fourth columns show the in- and outbound traffic which are measured using Cacti. The last column shows the chance that a legitimate client reconnects over TCP. This value is based upon the assumption that a client does 3 retries before aborting.

For the first measurements a repeating ANY attack was sent to the DNS server hosting a single zone. The server returns all the records and signatures it has for the requested domain name. This results in a high amplification ratio.

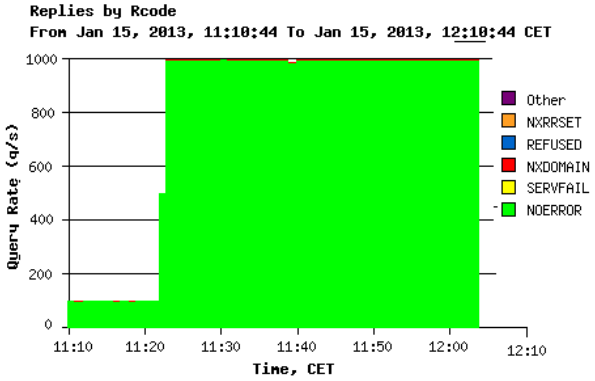
Figure 5-1 shows that 1000 incoming ANY queries per second for this specific zone results in 80.81KB/s inbound and 4.09MB/s outbound traffic. When RRL is enabled, with the default settings, the outgoing traffic quickly dropped to 39KB/s. This means that the traffic is no longer amplified. Table 5-1 shows the effect of the different SLIP parameter settings that have been researched. When RRL is enabled with a SLIP setting of 1 the outbound traffic is approximately equal to the inbound traffic (ratio 1:1) because every rate limited request will get a response with the TC bit set. This TC response is approximately the same size as the request. If the SLIP value is increased the amount of outbound traffic will decrease because less TC responses are sent out. When the SLIP value increases legitimate clients have a lower chance to receive a response because a TC bit is sent out once every SLIP query, meaning that the client has a lower chance of reconnecting over TCP. The chance for a client to reconnect over TCP is less then 66% when using a SLIP setting bigger than 2. Because SLIP is used to provide legitimate clients a chance to reconnect over TCP a setting above 2 is not recommend.

SLIP	False positives	In	Out	Amp. ratio	TCP responses
Slip 1	0%	80KB/s	81KB/s	≈1:1	100%
Slip 2	50%	79KB/s	39KB/s	≈1:0.5	87,5%
Slip 3	66.6%	79KB/s	26KB/s	≈1:0.3	66%
Slip 5	80%	80KB/s	16KB/s	≈1:0.2	49%
Slip 10	90%	80KB/s	8KB/s	≈1:0.1	27%

Table 5-1: ANY attack targeting a DNS server hosting a small zone.



(a) Average inbound and outbound traffic per minute.



(b) Incoming DNS queries per second. The attack was changed from 100 QPS to 1000 QPS at 11:20.

Figure 5-1: Repeating ANY Attack at 1000 queries per second without rate limiting.

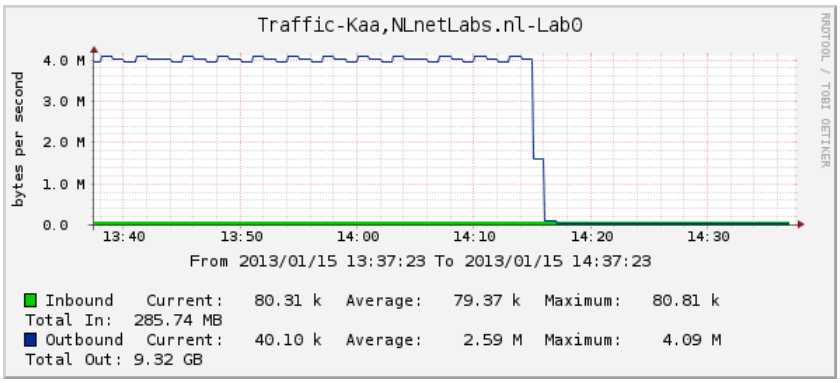


Figure 5-2: An incoming repeating ANY attack on a server hosting a single small zone. RRL with SLIP=1 is enabled at 14:15. At 14:18 SLIP is bumped up to 2.

5.1.2. TLD

The same attack is executed against a DNS server hosting a TLD like zone.

RRL	In	Out	Amp. ratio
Disabled	73KB/s	4.01MB/s	1:54.93
Enabled (SLIP=1)	73KB/s	73KB/s	1:1

Table 5-2: An ANY attack on a TLD like zone.

Since the ANY attack just repeats the same request over and over again it will result in the same response every time. As soon as RRL is enabled, the outbound traffic drops as expected. In all the scenario's that have been tested, RRL is an effective defense mechanism against the ANY attack.

5.2. Varying query attack

The type of attack explained in this section is more sophisticated than the repeated (ANY) query attack explained above. In order to perform a varying query attack, the attacker generates random queries or needs to gather information about the domain names hosted on the authoritative name server. An attacker can abuse NSEC records to gather information about a certain zone. NSEC records are used to prove a name does not exist by pointing to the previous and the next record. When NSEC records are used the attacker can perform a so called zone walk to index a zone. To prevent zone walking NSEC3 records have been introduced. IETF published RFC5155 [7] explaining NSEC3 in detail.

5.2.1. Varying query attack on TLD

A TLD name server usually has one large zone file containing a large collection of domain names. ANY queries are sent out for a variety of domain names of the zones. In the following section five different zones for a TLD are attacked, each varying in the amount of existing domain names. All the attacks take place with 1000 ANY queries per seconds.

5.2.2. Attack 1 (0% existing domain names)

The first zone generated simulates an attacker which does an attack by which all request result in an NXDOMAIN. From table 5-3 similarities can be seen with the measurements from an ANY attack. From the results can be concluded that RRL is effective against an attack targeted against non existing domain names. RRL get triggered because every request results in the same response and thus stored in the same bucket. Since RRL is effective during such an attack adjusting the SLIP value influences the amount of outbound traffic. As can be seen from table 5-3 the amount of outbound traffic drops as the value of SLIP increases. However it is not recommended to increase the SLIP value

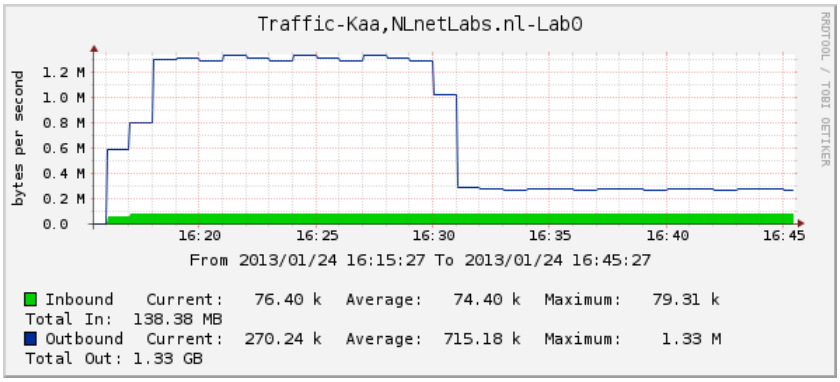
above two, the chance for a legitimate request to successfully gets a response over TCP is lower than 66%.

SLIP	False positives	In	Out	Amp. ratio	TCP responses
Slip 1	0%	77KB/s	78KB/s	≈1:1	100%
Slip 2	50%	79KB/s	38KB/s	≈1:0.5	87,5%
Slip 3	66.6%	79KB/s	26KB/s	≈1:0.3	66%
Slip 5	80%	78KB/s	14KB/s	≈1:0.2	49%
Slip 10	90%	78KB/s	8KB/s	≈1:0.1	27%

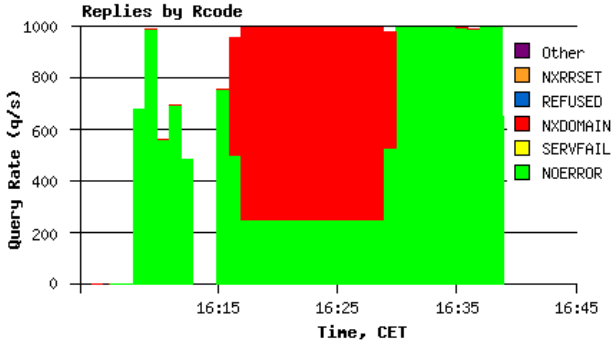
Table 5-3: Varying query attack on a TLD zone with 100% non-existing domain names.

5.2.3. Attack 2 (25% existing domain names)

The next zone file contains 25% existing domain names, this attack caused the server with RRL disabled to sent responses at a speed of 1350KB/s resulting in an amplification ratio of $\frac{1350}{79} \approx 17.1$.



(a) Average inbound and outbound traffic per minute.



(b) DNS incoming queries per second

Figure 5-3: Varying query attack (25%) at 1000 queries per second. RRL with SLIP=1 is enabled at 16:30.

SLIP	False positives	In	Out	Amp. ratio	TCP responses
Slip 1	0%	79KB/s	278KB/s	≈1:3.5	100%
Slip 2	50%	79KB/s	249KB/s	≈1:3.2	87,5%
Slip 3	66.6%	79KB/s	239KB/s	≈1:3.0	66%
Slip 5	80%	78KB/s	227KB/s	≈1:2.9	49%
Slip 10	90%	79KB/s	218KB/s	≈1:2.76	27%

Table 5-4: Varying query attack on a TLD zone with 25% existing and 75% non-existing domain names.

Table 5-4 shows that outbound traffic is approximately 278KB/s when RRL is enabled with a SLIP setting of 1. The inbound traffic is only 79KB/s, meaning that the attacker still achieves a small amplification. Because 25% of the responses return an existing unique domain name they are not grouped together by the RRL algorithm. Only the NXDOMAIN responses are grouped together and get rate limited. The NXDOMAIN answer is larger because three NSEC3 records are required to prove the denial of existence. The amount of outbound traffic decreases when the SLIP value gets increased. RRL manages to decrease the amplification ratio from ≈ 17 to ≈ 3 but is unable to fully deflect the attack. Since the amount of non-existing domain names is relatively high, increasing the SLIP value decreases the amount of outbound traffic.

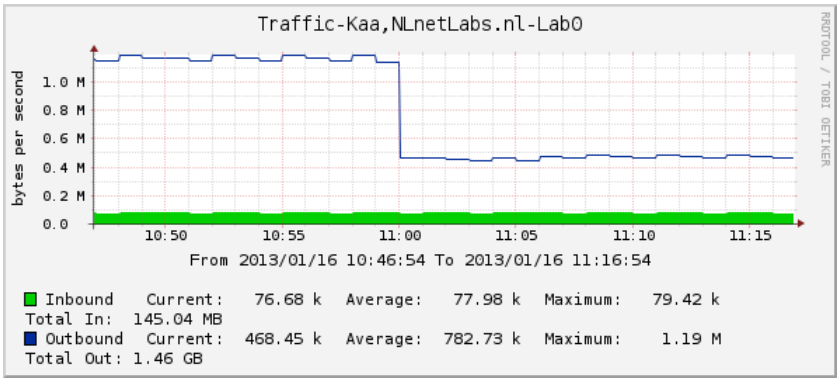
5.2.4. Attack 3 (50% existing domain names)

With RRL disabled the '50%' attack caused the server to send responses at a speed of ≈1170KB/s resulting in an amplification ratio of $\frac{1170}{79} \approx 14.8$. This ratio is slightly lower then the 25% attack, because the NOERROR answer from the DNS server is smaller then an NXDOMAIN answer.

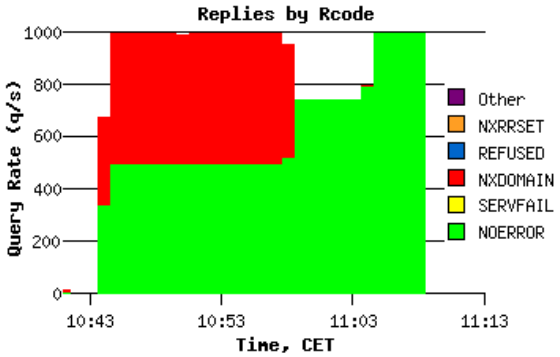
Table 5-5 shows an attack that returns 50% existing and 50% non-existing domain names. When rate limiting is enabled the outbound traffic dropped from 1170KB/s to 470KB/s. However, the inbound traffic is 77KB/s which still results in an amplification ratio of $\frac{470}{77} \approx 6.1$. If the SLIP value is increased the outbound traffic slightly decreases. Compared to measurements of the 25% attack the amount of outbound traffic is doubled. The reason for the increased traffic is that 50% of the responses are unique and therefore not rate limited. RRL manages to decrease the amplification ratio from ≈ 14.8 to ≈ 6 in

SLIP	False positives	In	Out	Amp. ratio	TCP responses
Slip 1	0%	77KB/s	468KB/s	1:6.0	100%
Slip 2	50%	78KB/s	455KB/s	1:5.83	87,5%
Slip 3	66.6%	78KB/s	450KB/s	1:5.76	66%
Slip 5	80%	79KB/s	451KB/s	1:5.71	49%
Slip 10	90%	80KB/s	448KB/s	1:5.60	27%

Table 5-5: Varying query attack on a TLD zone with 50% existing and 50% non-existing domain names.



(a) Average inbound and outbound traffic per minute.



(b) DNS incoming queries per second

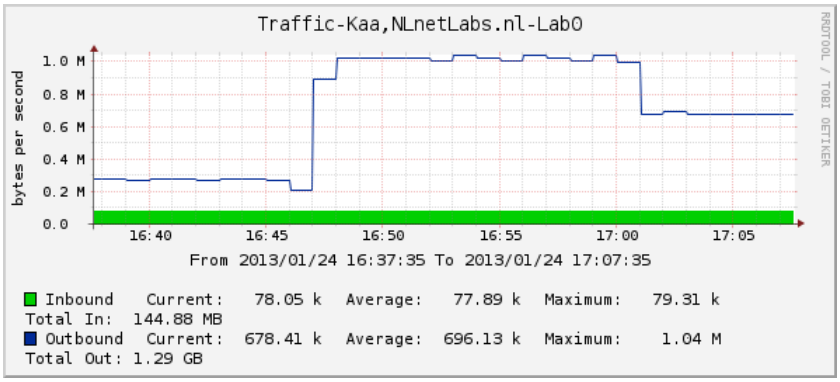
Figure 5-4: Varying query attack (50%) at 1000 queries per second. RL with SLIP=1 is enabled at 11:00.

this scenario. In this case the SLIP value decreases to influence the amount of outbound traffic, because the effective of RRL decreases as well.

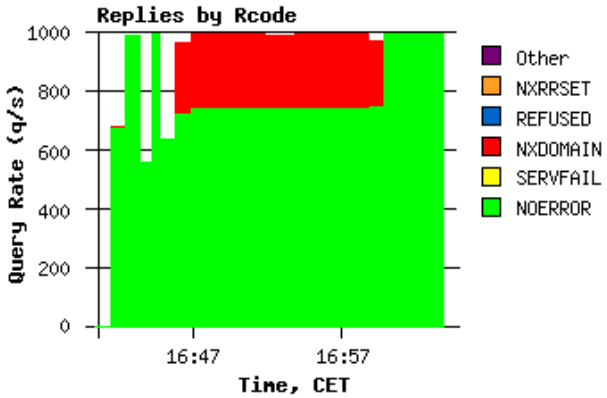
5.2.5. Attack 4 (75% existing domain names)

The '75%' attack caused the server to send responses at a speed of ≈1060KB/s resulting in an amplification ratio of $\frac{1060}{79} \approx 13.4$ when RRL is disabled. This ratio is lower than all previous attacks because the NOERROR answer from the DNS server is smaller then an NXDOMAIN answer.

The amplification ratio with RRL enabled has increased because RRL only limits the NXDOMAIN responses, which exist of only 25% within this zone file. With RRL the amplification ratio is decreased from ≈ 13.4 to ≈ 8.7. Due to the variety in existing domain names the effect of RRL and SLIP is limited.



(a) Average inbound and outbound traffic per minute.



(b) DNS incoming queries per second

Figure 5-5: Varying query attack (75%) at 1000 queries per second. RL with SLIP=1 is enabled at 17:01.

SLIP	False positives	In	Out	Amp. ratio	TCP responses
Slip 1	0%	79KB/s	689KB/s	1:8.72	100%
Slip 2	50%	78KB/s	680KB/s	1:8.72	87,5%
Slip 3	66.6%	79KB/s	677KB/s	1:8.57	66%
Slip 5	80%	79KB/s	673KB/s	1:8.52	49%
Slip 10	90%	79KB/s	665KB/s	1:8.42	27%

Table 5-6: Varying query attack on a TLD zone with 75% existing and 25% non-existing domain names.

5.2.6. Attack 5 (100% existing domain names)

In a zone with 100% existing domain names, which imitates an attack where the attacker knows all domain names for which the name server is responsible, all request will be replied with an unique response.

RRL	In	Out	Amp. ratio
Disabled	80KB/s	891KB/s	1:11.14
Enabled	80KB/s	891KB/s	1:11.14

Table 5-7: Distributed attack on a TLD zone with 100% existing domain names

There is no difference seen in amplification with RRL disabled or enabled. Table 5-7 and 5-9 shows that enabling RRL in this scenario does not have any effect on the amount of traffic. This means that RRL is not triggered at all using this attack type. If RRL is not triggered it does not make a difference if the SLIP value changes.

5.2.7. Varying query attack on a single small zone

A varying query attack is done on the same small zone file used in the repeated ANY attack to see how effective RRL is during this type of attack. As can be seen from the table below the DNS traffic is dropped from almost 1.5 MB/s to 82KB/s, this results in an amplification ratio of 1. In this situation RRL is effective because there are not enough records in a small zone for the varying query attack to generate sufficient unique responses. Since the responses are repeated RRL can successfully group the responses and limit the outbound traffic. This type of attack cannot target a single small zone because the attack requires a high amount of unique records to return unique results. Increasing the SLIP value in this case will result in the same values seen in the repeated ANY query attack in 5-1.

RRL	In	Out	Amp. ratio
Disabled	83KB/s	1.48MB/s	1:17.83
Enabled	82KB/s	82KB/s	1:1

Table 5-8: Varying query attack on a ”small” zone

5.2.8. Varying query attack on hosting company configuration

The same attack is done on an imitated hosting company configuration, containing 1000 individual ”small” zones. Table 5-9 shows that the in- and outbound traffic is approximately equal whether RRL is enabled or disabled. In this case the variety in domain names is large enough to create unique responses within the window size of RRL. The behavior of the hosting company configuration can be compared with an attack done on a TLD like zone. In both cases RRL is unable to deflect the attack when all domain names used in the attack are resolvable. Again changing the SLIP setting where RRL has no effect does not make any difference.

RRL	Inbound	Outbound	Amplification ratio
Disabled	80KB/s	2.40MB/s	1:30.00
Enabled	81KB/s	2.40MB/s	1:29.63

Table 5-9: Distributed ANY attack on 1000 individual "small" zones

5.3. Distributed attack

To increase the amount of traffic, an attacker can use multiple name servers instead of just one. For each extra name server used the amount of traffic which is sent to the victim multiplies. Due to the fact that the effectiveness of RRL has been extensively measured on a single server and due to the limited amount of hardware available within the lab environment, no further effort has been put into researching a distributed attack. Distributing an attack over multiple name servers can not only increase the traffic but it could potentially prevent all proposed defense mechanisms from triggering.

5.4. RRL impact on server load

The impact on server resources has also been briefly examined. Because these measurements fall outside of the scope of this research they can be found in appendix E.

6. Results

6.1. RRL effectiveness

The measurements in the previous section show that the effectiveness of RRL depends on the design of the attack. When an attacker is able to hit a higher ratio of existing domains more unique responses are generated. This causes RRL to distribute the responses over different buckets.

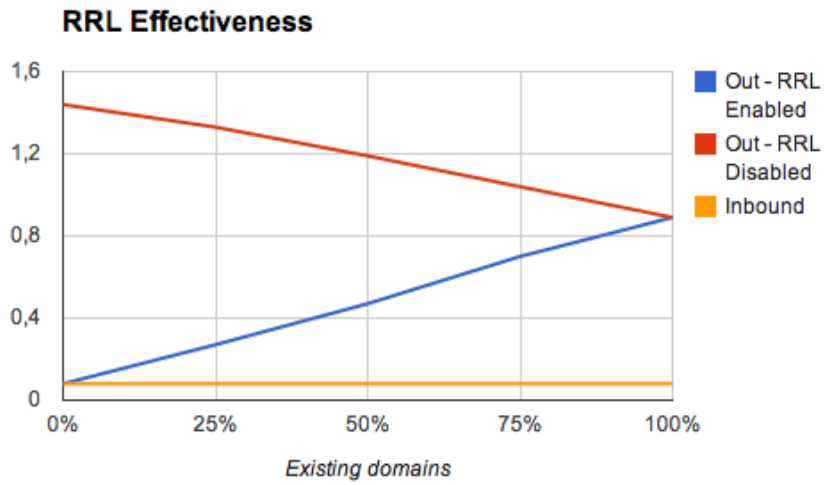


Figure 6-1: Effectiveness of RRL

Figure 6-1 shows that, when RRL is disabled, the outbound traffic decreases when more unique responses (existing domains) are returned. The reason for this behavior is that the NXDOMAIN response is larger in size then a response containing an A record and its signature. The NXDOMAIN response is larger in size because, in most cases, 3 NSEC3 records are required to prove the denial of existence. Figure 6-1 also shows that the effectiveness of RRL decreases to 0% when the attacker only uses varying queries that result in unique existing domain names. When unique responses are returned RRL will distribute the responses over different buckets. As long as the buckets don't reach the per second limit RRL will not be triggered to limit responses and the attacker can execute an amplification attack.

Figure 6-2 shows that when the SLIP setting is increased, the chance to encounter false positives also increases. Because SLIP is designed to provide legitimate clients with a chance to reconnect over TCP it is not recommended to configure a value higher than two.

6.2. Recommended RRL settings

In this section the recommended configuration settings are discussed for RRL. Keep in mind that there is no global setting that suites every scenario. These settings are highly depending on the environment and the load of the server. Testing the configuration in

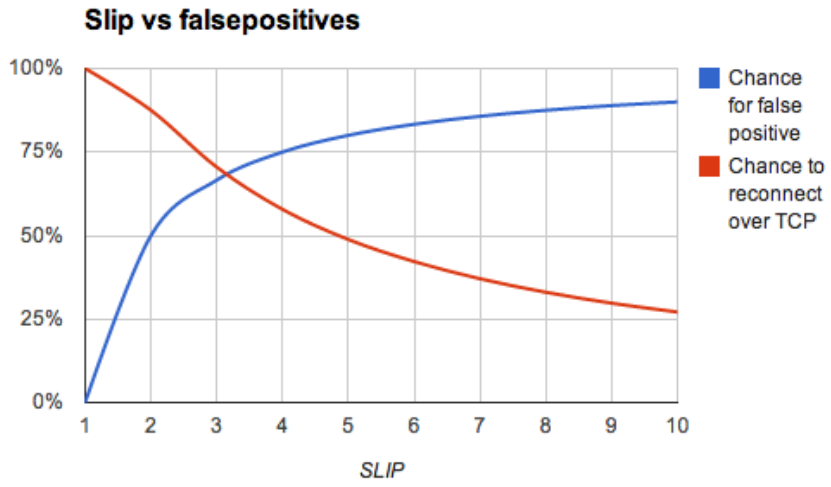


Figure 6-2: Chance of false positive with different SLIP

"log-only" mode is highly advised. Using this parameter the behavior of RRL on the name server can be analyzed.

6.2.1. Per-second settings.

RRL allows to rate limit NOERROR, NXDOMAIN and error responses on a per second basis. These settings can be adjusted individually. To determine which parameter setting suits best for a specific environment it is recommended to do the following:

- Monitor and analyze your responses per second on the size of prefix length defined;
- Find the largest response on the name server;
- Determine the maximum amount of outgoing traffic to a single IP-address;
- Calculate the maximum amount of responses-per-second allowed.

As an example, after analyzing the traffic and determining that, under normal behavior, the maximum amount of responses per second sent to a /24 IPv4 subnet is 2. The largest response has a size of 3916 bytes. Limiting the outgoing traffic to 0.25Mbit/s is determined as acceptable because the chance is small that this causes any harm to a single network. To calculate the max responses-per-second setting the maximum acceptable outgoing traffic is divided by the largest response: $\frac{0.25Mbit/s}{3916Bytes} \approx 8$. The outcome will always be larger then the monitored maximum. The following formula can be used to calculate the maximum responses per seconds:

$$\frac{(Maximum\ acceptable\ outgoing\ traffic)}{(Largest\ response\ found)} = (maximum\ responses\ per\ second)$$

The formula shows that the maximum acceptable outgoing traffic is divided by the largest response found on the server.

6.2.2. Window

This setting defines how many seconds a client will be rate limited after an attack has stopped. RRL uses a credit system that is described below.

- The responses-per-second setting is equal to the maximum amount of credits a client can have;
- $0 - (Window * responses - per - second)$ = the maximum amount of negative credits a client can have;
- The responses-per-second setting is also equal to the amount of credits a client receives per second;
- When the amount of credits is negative responses are dropped.

This window should be kept small to revert to normal service as soon as possible after an attack. A window of 5 seconds should be sufficient in most cases. Be aware that if the windows size is decreased any further this could cause on/off behavior when the incoming traffic is inconsistent.

6.2.3. IPv4 and IPv6 prefix length

Client IP addresses are grouped into buckets because RRL is designed to protect distant networks from amplification attacks. By default these buckets are equal to the size of a /24 networks which contain up to 256 host addresses. For IPv6 the default is /56 which also relates to 256 sub-networks. Decreasing the prefix length will increase the chance for false positives on a busy DNS server because more client addresses are grouped together. If there is no special reason to deviate from the default settings, it is recommended not to change those parameter settings.

6.2.4. SLIP

A DDoS victim, while under attack, may not be able to contact the DNS server which is used for the attack because the DNS server will drop the traffic. It is not possible for a server to distinguish legitimate traffic from illegitimate due to UDP source-address spoofing. If traffic for a network is being dropped, a TC flag can be returned once per SLIP queries. The client takes the TC flag as an indication that it should retry over TCP, which only legitimate clients will do. Setting the SLIP value to one will prevent all possible false positives, SLIP will respond to every dropped response with the TC bit set, making sure that legitimate queries are can be answered over TCP. Because this TC response is approximately the size of the request an attacker will not achieve any amplification. Setting SLIP to two will sent a TC response to 50% of the request. This gives a victim a fair chance to communicate with the DNS server while being under attack. Assumed a client will retry three times in order to get a response from the DNS

server the chance of communicating is 87.5%. $(100 * (1 - (0.5 * 0.5 * 0.5)))$. This results in an amplification ratio of 1:0.5, cutting the attack traffic in half.

6.2.5. Table-size

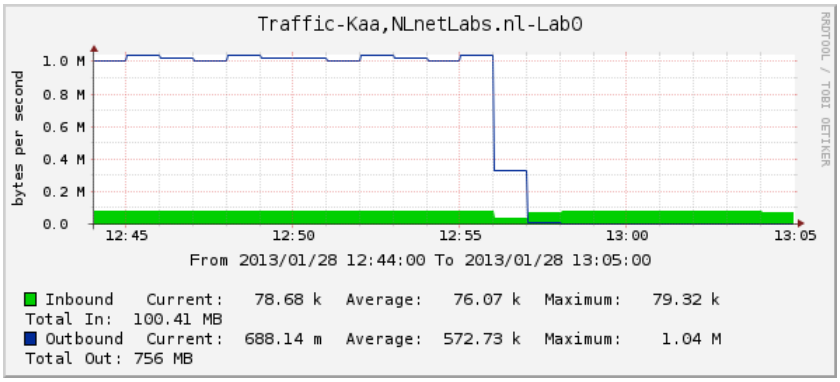
The max-table-size sets the maximum number of state blobs the server will maintain. This setting should be set for the worst case scenario where all queries need to be maintained because the responses are unique. To calculate the value take the maximum queries per second and multiply this value by the window size. 80 Megabyte of server memory is consumed, this is seen in the measurements from the previous chapter for 1.000.000 state blobs. If a maximum of 100.000 queries per second is processed with a windows size of 5 this will result in a max-table-size of 500.000.

The min-table-size parameter sets the initial size of the state-blob table at start-up time. Because growing this table will consume resources it is advised to set this to a higher value. Taking the maximum queries per second under normal operation and multiplying this by the window size should be a good baseline for this setting.

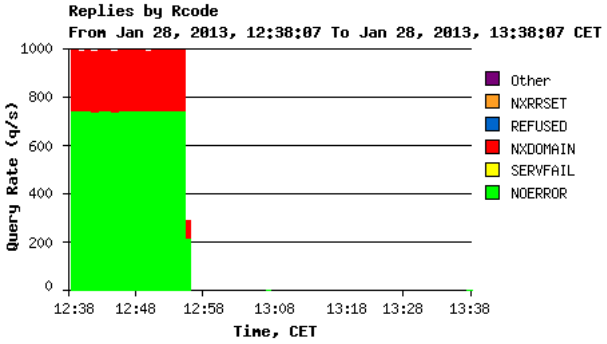
7. Dampening

The results presented in section 6 show that RRL is unable to properly defend against distributed attacks. For this reason another available defense mechanism mentioned in section 3 is briefly researched called DNS Dampening. There is a DNS dampening patch available for BIND which is designed and created by Lutz Donnerhacke. Due to limited amount of time and the scope of this project DNS dampening will not be covered in dept. Because the mechanism assigns penalty points to spoofed clients instead of grouping the unique responses together in buckets it should be more effective against distributed attacks.

7.1. Measurements



(a) Average inbound and outbound traffic per minute.



(b) DNS responses queries per second

Figure 7-1: Dampening is activated at 12:55.

Figure 7-1 shows the result of a varying query attack with 75% existing domain names. DNS dampening is activated at 12:56 and the traffic is almost instantly dropped. Other scenarios where briefly tested as well and every attack was successfully countered by DNS dampening.

7.2. DNS dampening impact on server load

The impact on server resources has also been briefly examined. Because these measurements fall outside of the scope of this research they are attached as appendix E.

7.3. Drawbacks

Although DNS dampening looks very promising based upon the measurements, there are a few drawbacks to using it in production. The most significant drawback is that their is no mechanism in place to counter false positives. When a client or network is limited, no legitimate traffic from that client or network is possible. This provides an attacker with an easy way to prevent DNS requests from being answered to a specific client. Another drawback is that it is currently not possible to tailor the DNS dampening parameters to the environment. DNS dampening in its current form is aggressive, therefore extra care should be taken when implementing it.

8. Conclusion

DNS amplification attacks rely on two concepts. The first one; when using UDP the IP address can easily be spoofed causing traffic to be reflected by the DNS server. The second one; DNS requests are small while responses can be multiple times the size of the request causing the amplification. DNSSEC further contributes to the amplification of DNS traffic because it requires the DNS server to include signatures causing the size of the response to grow.

The amplification of illegitimate responses can be limited by implementing RRL on authoritative name servers. When responses are duplicated within the window size or if multiple errors or NXDOMAIN responses are returned RRL will be triggered. RRL can prevent false positives by setting SLIP to 1, this will have the DNS server to respond to every dropped query with the TC bit set giving a legitimate client a chance to retry over TCP. The chance for a client to reconnect over TCP is less then 66% when using a SLIP setting of 3 or higher. Because SLIP is used to provide legitimate clients a chance to reconnect over TCP a setting above 2 is not recommend. RRL is very successful in mitigating basic attacks like the standard repeating ANY attack. However, the effectiveness decreases when the attack gets more sophisticated. When an attacker uses a large set of varying queries only the NXDOMAIN responses will get limited and an attacker might still be able to amplify the traffic. If the attacker manages to find enough existing domain names, the queries can be distributed over these names, resulting in unique responses which are not limited by RRL. When using DNSSEC on authoritative name servers NSEC3 should be implemented instead of NSEC. This prevents an attacker from indexing the zone by doing a zone walk and then distribute the attack over all the records in the zone. RRL is not effective against a fully indexed varying query attack that only queries for existing domain names.

Another defense mechanism called DNS dampening can be successful in mitigating more sophisticated distributed attacks. DNS dampening looks like a promising defense mechanism, but is missing some essential features. It does not have a technique implemented to counter false positives, which makes it easy for an attacker to block a server for legitimate clients.

Unfortunately there is no easy way to stop DNS amplification attacks without any side effects. Until the major network vendors make source address validation (BCP38, Ingress filtering) a default setting, reflection/amplification attacks will remain an issue. Response Rate Limiting is the best approach to eliminate the current form of attacks. When attacks become more sophisticated in the future, RRL will lose its effectiveness and the need for a different defense mechanism will rise.

9. Future work

If future attacks get more sophisticated, it will be possible to bypass RRL. DNS dampening is a promising defense mechanism against DNS reflection attacks which could help in this situation. However, in the current state it is missing a feature to prevent false-positives. A similar feature like SLIP will need to be developed for DNS dampening to make it a practical solution.

When attacks get distributed across many DNS servers it might even be possible to prevent DNS dampening from triggering. As an addition to the current defense mechanisms available, new solutions will need to be researched. In order to prevent a DNS reflection amplification attack either the reflection or the amplification should be removed. A promising solution for removing the reflection is already available, BCP38 prevents IP addresses from being spoofed, therefore making reflection attacks impossible. A method for a faster adoption of BCP38 could be researched in the future as well.

A. Abbreviations

BIND	Berkeley Internet Name Daemon
DNS	Domain Name Server
IP	Internet Protocol
KB	kilobyte
MB	Megabyte
NXDOMAIN	Non Existing Domain
QPS	Queries per second
RP1	Research Project 1
RR	Resource Records
RRL	Response Rate Limiting
RRSIG	Resource Records Signature
TC	Transport Control
TLD	Top Level Domain
UvA	Universiteit van Amsterdam

B. Bibliography

[1] **Randal Vaughn & Gadi Evron**, "DNS Amplification Attacks", 2006. http://packetstormsecurity.com/files/download/44824/DNS-Amplification-Attacks.pdf

[2] **Duane Wessels**, "Tracing a DNS Reflection Attack", 2011.

[3] **J. Damas**, "RFC5358", 2008. http://www.rfc-archive.org/getrfc.php?rfc=5358

[4] **Georgios Kambourakis, Tassos Moschos, Dimitris Geneiatakis and Stefanos Gritzalis**, "A Fair Solution to DNS Amplification Attacks", 2007. http://www.cs.columbia.edu/~dgen/papers/conferences/conference-07.pdf

[5] **P. Ferguson, D. Senie**, "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing", 2000. http://tools.ietf.org/html/bcp38

[6] **NLnet Labs**, "DNS Response Rate Limiting as implemented in NSD.", 2012. http://www.nlnetlabs.nl/blog/2012/10/11/nsd-ratelimit/

[7] **B. Laurie, G. Sisson, R. Arends and D.Blacka**, "DNS Security (DNSSEC) Hashed Authenticated Denial of Existence", 2008. http://tools.ietf.org/html/rfc5155

[8] **Georgios Kambourakis, Tassos Moschos, Dimitris Geneiatakis, Stefanos Gritzalis**, "Detecting DNS Amplification Attacks", 2007. http://www.cs.columbia.edu/~dgen/papers/conferences/conference-08.pdf

[9] **Multiple authors**, "Stream Control Transmission Protocol, RFC2960", 2000. http://www.ietf.org/rfc/rfc2960.txt

[10] **Lutz Donnerhacke**, "DNS Dampening", 2012. http://lutz.donnerhacke.de/eng/Blog/DNS-Dampening

[11] **C. Villamizar, R. Chandra, R. Govindan**, "BGP Route Flap Damping, RFC2439", 1998. http://www.hjp.at/doc/rfc/rfc2439.txt

[12] **Paul Vixie, Vernon Schryver**, "DNS Response Rate Limiting", 2012. http://ss.vix.com/~vixie/isc-tn-2012-1.txt

C. Scenarios

C.1. Scenario 1: TLD DNS server

To represent a TLD domain, a zone file that contains 1000 sub-domains has been generated. Every sub-domain has 2 NS records with 2 glue (A) records attached.

The figure below show the zone file used to imitate a TLD DNS server before it got signed:

```
1  $TTL    3600 ; minimum TTL
2  @       IN SOA tld. hostmaster.nlnetlabs.nl (
3  2013011000 ; <serial in YYYYMMDDnn>
4  3600       ; refresh every hour
5  1800       ; retry after 30 minutes hour
6  21600      ; expire after 6 hours
7  3600       ; negative cache is 1 hour
8  )
9
10 $include Ktld.+010+09458.key ;ksk
11 $include Ktld.+010+60204.key ;zsk
12
13 @        IN NS   ns1.tld.
14 @        IN NS   ns2.tld.
15 @        IN MX   10 mail.tld.
16 @        IN MX   20 mail2.tld.
17 @        IN TXT v=spf1 ip4:213.154.224.109 mx:mail.zone.tld
      a:sub.zone.tld mx:sub.zone.tld -all
18 @        IN A    213.154.224.109
19 ns1      IN A    213.154.224.109
20 ns2      IN A    213.154.224.109
21 mail     IN A    213.154.224.109
22 mail2    IN A    213.154.224.109
23
24 1.tld. IN NS ns1.1.tld.
25 1.tld. IN NS ns2.1.tld.
26 1.tld. IN NS ns3.1.tld.
27 ns1.1.tld. IN A 213.154.224.109
28 ns2.1.tld. IN A 213.154.224.109
29 ns3.1.tld. IN A 213.154.224.109
30
31 ...
32 ...
33 ...
34
35 999.tld. IN NS ns1.999.tld.
```

```
999.tld. IN NS ns2.999.tld.
999.tld. IN NS ns3.999.tld.
ns1.999.tld. IN A 213.154.224.109
ns2.999.tld. IN A 213.154.224.109
ns3.999.tld. IN A 213.154.224.109
```

C.2. Scenario 2: Authoritative DNS server with a single zone

The zone file used to represent a "regular" authoritative DNS server contains the following resource record types: SOA, NS, A, MX, TXT, NSEC, DNSKEY and RRSIG.

The listing below shows the used zone file before it got signed, this way it is clear to see what type of records are used.

```
$TTL    3600 ; minimum TTL
@       IN SOA zone.tld. hostmaster.nlnetlabs.nl (
2013011000 ; <serial in YYYYMMDDnn>
3600       ; refresh every hour
1800       ; retry after 30 minutes hour
21600      ; expire after 6 hours
3600       ; negative cache is 1 hour
)

$include Kzone.tld.+010+24783.key ;ksk
$include Kzone.tld.+010+37817.key ;zsk

@        IN NS   ns1.zone.tld.
@        IN NS   ns2.zone.tld.

@        IN MX   10 mail.zone.tld.
@        IN MX   20 mail2.zone.tld.

@        IN TXT v=spf1 ip4:213.154.224.109 mx:mail.zone.tld
   a:sub.zone.tld mx:sub.zone.tld -all

@        IN A    213.154.224.109
ns1      IN A    213.154.224.109
ns2      IN A    213.154.224.109
mail     IN A    213.154.224.109
mail2    IN A    213.154.224.109
```

C.3. Scenario 3: Hosting Provider

In order to imitate a zone from a web hosting company, a script[4] is used to create the zone files showed below:

```
1tldzones. 3600 IN SOA ns1.1tldzones. postmaster.1tldzones.
   1000 1200 180 1209600 3600
1tldzones. 3600 IN MX 10 mail.1tldzones.
1tldzones. 3600 IN NS ns1.1tldzones.
1tldzones. 3600 IN NS ns2.1tldzones.
1tldzones. 3600 IN A 192.0.2.1
mail.1tldzones. 3600 IN A 192.0.2.1
ns1.1tldzones. 3600 IN A 192.0.2.1
ns2.1tldzones. 3600 IN A 192.0.2.1
label1.1tldzones. 3600 IN A 192.0.2.1
label2.1tldzones. 3600 IN A 192.0.2.1

...
...
...

999tldzones. 3600 IN SOA ns1.999tldzones. postmaster.999
   tldzones. 1000 1200 180 1209600 3600
999tldzones. 3600 IN MX 10 mail.999tldzones.
999tldzones. 3600 IN NS ns1.999tldzones.
999tldzones. 3600 IN NS ns2.999tldzones.
999tldzones. 3600 IN A 192.0.2.1
mail.999tldzones. 3600 IN A 192.0.2.1
ns1.999tldzones. 3600 IN A 192.0.2.1
ns2.999tldzones. 3600 IN A 192.0.2.1
label1.999tldzones. 3600 IN A 192.0.2.1
label2.999tldzones. 3600 IN A 192.0.2.1
```

[4] http://svn.opendnssec.org/trunk/testing/zonegen.pl

D. Lab setup

NLnet Labs made their test environment available containing three server to imitate an amplification attack. Below a description of each server used in the lab environment.

D.1. Hathi (Attacker)

In stead of using a bot-net which normally is being used in an attack, a packet capture (.pcap) file is used to send the multiple DNS request to the authoritative name server. Creating a pcap file can be done using Tcpdump which can capture all the traffic seen on an interface and stores this in a pcap file. After the different pcap files had been created, the source addresses were changed changed using the a script:

```
1 [root@hathi /home/matje/rp1/scripts]# cat rewrite
2 TCPREWRITE="/home/matje/rp1/bin/tcprewrite"
3
4 echo $TCPREWRITE -D 0.0.0.0/0:192.168.0.11 -S
     0.0.0.0/0:192.168.0.1 -C \
5  --infile=test.pcap -o nlnetlabs-queries.pcap
6
7 $TCPREWRITE -D 0.0.0.0/0:192.168.0.11 -S
     0.0.0.0/0:192.168.0.1 -C \
8   --infile=test.pcap -o nlnetlabs-queries.pcap
```

Listing D-1: TCP rewrite example.

This imitates the spoofed IP address of the victim which would usually be used to request DNS queries at an authoritative name server. Because the name server finds the address of the victim in the request, responses are sent to the victim as well.

The pcap files are sent to the server using a tool called Tcpreplay. This tool can sent packets using a premade pcap file. The amount of packets sent per second can be adjusted.

D.2. Kaa (DNS Server)

The most recent version of BIND (9.9.2-2.1) is installed as the name server daemon including the Reponse Rate Limiting patch. This patch is developed by Vernon Schryver and Paul Vixie and not yet officially implemented in BIND.

A different zone file is created for each of the environments mentioned above, in order to create a zone file which can be compared with a TLD, ZoneGen is used. This tool automatically generates zone files according to the parameters entered.

D.3. Balou (Victim)

A third server is used as the victim. This server receives all the DNS responses from the authoritative name server. The amount of incoming traffic is measured on this server, this way the amplification can be calculated.

D.4. Monitoring tools

D.4.1. Cacti

The inbound and outbound traffic passing the lab interface on the DNS server is measured using a network monitoring tool called Cacti[5]. Cacti is a complete network graphing solution designed to use RRDTool's data storage and graphing functionality.

D.4.2. DSC

The DNS requests and responses are logged using DSC. DSC[6] is a system for collecting and exploring statistics from busy DNS servers. It currently has two components:

- **Collector**: The collector process uses libpcap to receive DNS messages sent and received on a network interface.
- **Presenter**: This component receives XML datasets from collectors. Data is presented in a web browser.

D.4.3. HTOP

The resource consumption of the BIND daemon is measured using HTOP [7]. HTOP is a basic interactive process viewer for Linux.

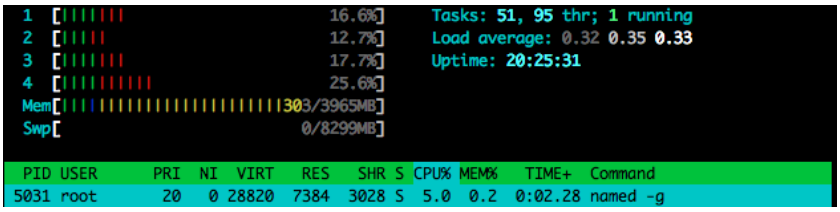
[5] http://www.cacti.net/
[6] http://dns.measurement-factory.com/tools/dsc/
[7] http://htop.sourceforge.net/

E. Impact on server load.

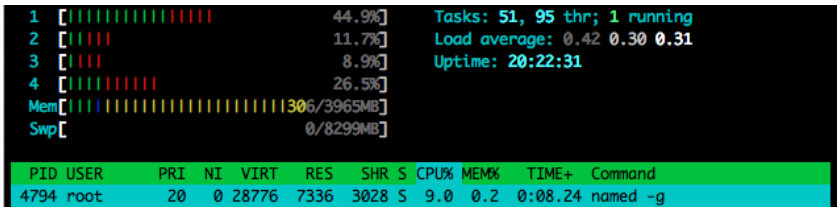
E.1. RRL impact on server load.

NLnet Labs posted on their blog[6] that RRL puts minimal extra load on the server. Measurements performed on NSD show that RRL adds an additional 4% load on the servers resources while the server is not under an attack.

Additional measurements on BIND show that when the server is abused for an amplification attack RRL can actually reduce the amount of system resources consumed. Figure E-1 shows that the CPU utilization is 4% higher without rate limiting enabled



(c) Server load with RRL enabled.



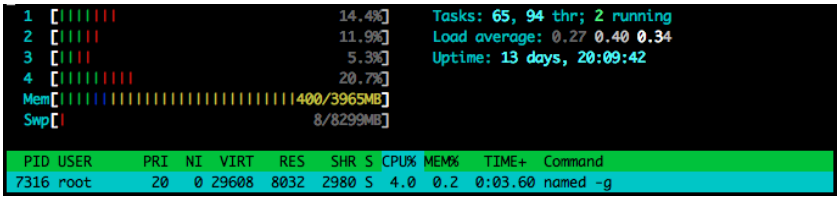
(d) Server load with RRL disabled.

Figure E-1: The server load while the server is under attack with 1000 ANY queries per second.

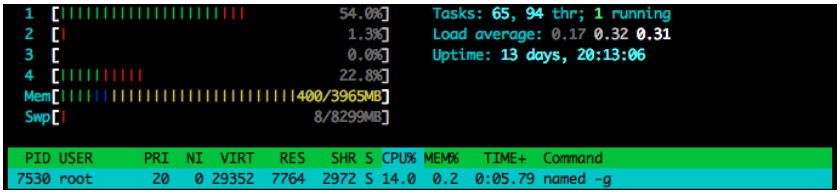
when the server is under a repeated ANY attack at a rate of 1000 queries per second. When RRL is triggered the server does not need to create and send the responses, causing the decrease in server resource consumption. The state table that RRL requires is completely loaded in the servers memory, putting a little extra load on the servers memory. When the state table is increased to keep 1.000.000 states it consumes an additional 80-100 MB of memory. The table size required differs per scenario. The TLD obviously requires a larger state table then a server hosting a small zone. A method to calculate the size of the required state table is given in section 6.2.5.

E.2. DNS dampening impact on server load

DNS dampening can reduce the amount of system resources used when the server is abused for an amplification attack. The reason for the decrease in load is that the server does not have to generate responses. The requests do not get handled by the DNS server while the penalty points exceed the DNS dampening limit.



(a) Server load with RRL enabled.



(b) Server load with RRL disabled.

Figure E-2: The server load while the server is under attack with 1000 ANY queries per second.

Figure E-2 shows that the CPU utilization is 4% higher without rate limiting if the server is attacked at a rate of 1000 queries per second.