DNS Security

Imran Khan
Email: imrkh883@student.liu.se
Supervisor: David Byers, davby@ida.liu.se
Project Report for Information Security Course
Linköpings universitetet, Sweden

Abstract

The Domain Name System is a distributed database that allows convenient storing and retrieval of information and resources records. It has been extended to provide DNS security extensions (DNSSEC) mainly through public key cryptography. In this report I have described the common attacks affecting the Domain Name Systems performance and the way how modern Domain Name Systems are secured. The different kinds of DNS security flavors and their working also described in detail. The DNSSEC subset proposed is presented and analyze from different point of view.

1. Introduction

To fully understand the strategy of DNS security [1] there is a well- known case of DNS spoofing need to be considered. In July 1997, during two periods of several days user around the internet who typed "*www.internet.net"* into their web browsers thinking they were going to the InterNIC's web site instead ended up at a web site belonging to the AlterNIC. How'd it happen? Eugene Kashpureff then affiliated with the AlterNIC, had run a program to "poison" the caches of major name servers around the world, making the belief that "*www.internic.net's*" address was actually the address of the AlterNIC web server. The web site that user reached was plainly the AlterNIC's not the InterNIC's. Imagine users typing in their credit card numbers and expiration dates is a more swear case.

1.1 What is DNS?

DNS system provides a mechanism of conversion with double functionality [2]: It translates both host name to IP addressees and IP addresses to host names. It has three major components:

- The first category contains:
  - **The Domain Name Space** and
  - **The resource record,** that are specifications for a tree structured name space and the data associated with these names.
- **Name Servers** are server programs which maintain the information about the DNS tree structure. A name server may cache information about any part of the domain tree, but in general it has complete information about a specific part of the DNS. This mean the name server has authority for that sub domain of the name space-therefore it will be called authoritative.
- **Resolvers** are the server programs that extract the information from name servers in response to the client requests.

1.2 DNS Security Threats.

It is known that DNS is weak from several aspects [2]**.** Using the Domain Name System we face the problem of trusting the information that came from a non authenticated authority, the name based authentication process, and the problem of accepting additional information that was not requested and that may be incorrect.

"Many of the classic security breaches in the history of Computers and computer networking have had to do, not with fundamental algorithm or protocol flaws, but with implementation errors. While we do not intend to demean the efforts of those involved in upgrading the Internet protocols to make security a more realistic goal, we have observed that if BIND would just do what the DNS specifications say it should do, stop crashing, and start checking its inputs, then most of the existing security holes in DNS *as practiced* would go away." - Paul Vixie, founder of ISC and main programmer of BIND.

1.2.1 Denial of Service Techniques

In DoS attacka a legitimate user is prevented from using a service through some illegal means e.g. by flooding a network to increase network traffic load or disrupting connection between two machines. Basically there are three modes of DoS attack [3].

1.2.1.1 Consumption of scarce resources

Generally the computers and the network itself need resources to operate properly if these resources are not available accordingly than the overall functionality gets

affected. E.g. DoS attack against network connectivity, bandwidth consumption and consumption of Other Resources like printers, tape devices, deny of service from other limited resources important to the operation of organization are all lies under this category.

1.2.1.2 Destruction or Alteration of Configuration information.

Alteration or destruction of the configuration information may result in partially or totally breakup of operations and thus an attacker may stop a user from using computer or network.

1.2.1.3 Physical Destruction or Alteration of Network Components.

This mode of attack includes the threat against physical security including all the components of computer and the network.

Like all internet resources DoS attack is also a threat for DNS servers. It is possible to send a large number of queries to DNS servers from spoofed sources to raise a condition of DoS so that the server's network uplink becomes congested or the DNS server response time becomes severely degraded.

One dangerous technique similar to the one used in Smurf attack. In this technique a DNS server can be used for amplification of attack traffic by creating small request packets to generate large responses from the queried server e.g. request for a zone transfer for small request large reply. If a name server allows zone transfers from just about anyone, it is possible for an attacker to spoof a large number of small zone transfer requests using source addresses on a specific victim network. In this case the DNS server will then amplify the traffic sent to it as it returns the significantly larger zone-transfer reply packets to the alleged requestor.

Another potential DoS attack related to DNS may lead due to the nature of recursive lookups. If sending a large number of requests for domains guaranteed not to be cached at a particular name server. In this case for each small query packet sent, the resolving name server will have to perform at least one recursive lookup per packet. This could lead to severe service degradation if a coordinated attack could be launched from numerous sources in a Distributed Denial of Service (DDoS) scenario.

1.2.1.3 Recent DNS Denial of Service Attacks.

The following are the two most popular DNS attacks that occurred in past few years.

i. DDoS Attack (February 7, 2007)

In February 2007, five of the root names servers are affected by a DDoS world wide, two of which stops responding to the queries of up to 90%, according to the RIPE NCC (Réseaux IP Européens Network Coordination Centre)[4][5]. The attack started at 10:30 UTC and lasted for five hours. None of the root name server was crashed. Also the internet service was not disrupted due to the working of other name servers including RIPE NCC managed-k root.

The botnets (malicious softwares) responsible for these attacks were originated form Asia-Pasific region but it was published most about South Korea. The term botnet mean here is a collection of software "rebots" or "bots" that runs automatically and collectively to flood the targeted systems. The botnets are actually collection of compromised computers also called zambie computers running softwares , usually installed using Trojan horses, worms and backdoors under a common command and control infrastructure.

The term botnet can also be used to refer any group of bots such as Internet Relay Chat (IRC). The botnet controllers establishes their own infrastructure for creating the kind of attacks (flooding to name servers) .They organize, e.g., IRC servers, command and controls servers, set of protocols to communicate etc.

Internet relies on the thirteen root name servers deployed world wide and they are organized as A to M. To ensure stability and availability the control of these thirteen name servers are not held by a single organization.

ii. DDoS Attack (21-Oct-2002)

This attack was occurred in October 2002 and lasted for one hour [6][5]. This was the second major failure of root name servers after one happened in April 1997 due to technical problem and affects the whole internet service badly instead of particular websites. All thirteen root name servers were affected simultaneously.

The attack volume was 50 to 100 Mbits/sec per root servers, with a total volume of 900Mbits/sec. The attack traffic was contained ICMP, TCP SYN, fragmented TCP and UDP. The source was randomized and generated automatically in a particular network at the time of attack.

Impacts of Attack.

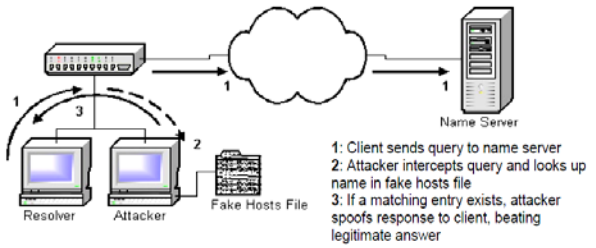
Some root name servers were continuously unreachable in many part of the internet world due to heavily created congestion in the network while others were responding continuously to their queries. This is due to the successful overprovisioning of host resources. Many valid queries were unreachable to some root name servers and hence were not responded. Several root name servers were continuously reachable from all monitoring points for the entire duration of the attack due to the successful overprovisioning of the network resources. Although the attacks was present for one hour but there was not any report for end user error condition. There was a minor

delay for some lookups, this due to the efficient design of DNS protocol.

1.2.2 Local Query Interception and Response Spoofing (DNS Hijacking)

A user submits many queries to a server; these queries are recursive queries that should fail, cause the server to do the more work. It is possible for an attacker to intercept the queries and beat the name server's response by sending a spoofed response with their own information. This kind of attack can occur if an attacker can see the DNS queries on the network being sent by clients to a DNS server. This attack results in a race condition if the attacker resides on the same LAN as a victim. As the legitimate server may not be on the same LAN or may need to perform a number of recursive queries to return a result, which will slow it down considerably.

Response spoofing is a DNS attack that involves intercepting and sending a fake DNS response to a user. This attack forwards the user to a different address than where he wants to be [7]. DNS hijacking is effective if the attacker can observe the victim DNS query traffic. In most case the DoS attack to DNS server is unnecessary as the fake DNS reply usually come before the true one from the DNS server. However, the attacker needs to be close to the victim or the DNS server so as to observe the DNS query traffic. Man in middle attack is an example of DNS hijacking. Figure below describes the DNS hijacking scenario [8].



Local DNS Hijacking Scenario

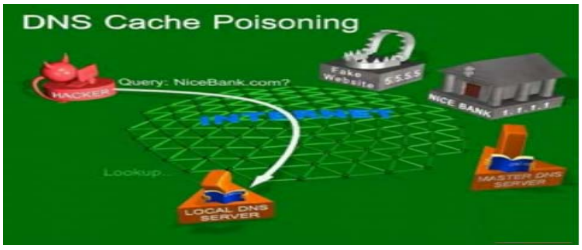
1.2.3 DNS Cache Poisoning

Cache poisoning attacks whereby the cache of the DNS is deliberately contaminated by an attacker. This is done by using DNS Transaction ID predication or Recursive queries. This attack is more dangerous as the attackers do not need to be positioned near the name server to observe the replies. In case of DNS cache poisoning it is possible for an attacker to make a legitimate DNS server to cache falsified information, which the attacker will supply. The figure [9] below describes the scenario of cache poising very simply:

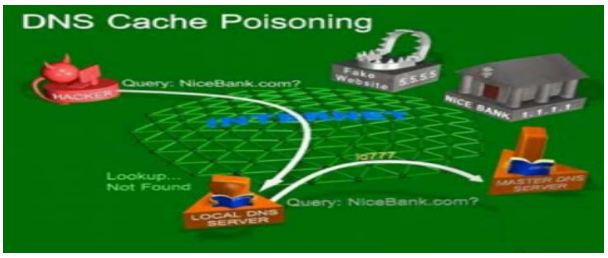
A user types a website into the browser and asks the DNS server for the address and then server takes the user to the desired website.

Local DNS server makes it faster; they store the addresses in cache so that, the requests don't go to the internet every time. If the request is not in the cache the local DNS server forwards the request to the internet's DNS. Cache poisoning attack is:

1. A hacker sends a request to a local DNS [9].



2. The query is then forwarded to the internet's DNS [9].



3. And the attacker then floods the Local DNS with fake reponses [9].

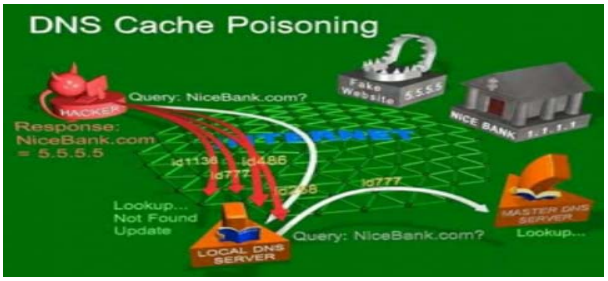


Figure 1. Cache Poisoning attack Scenario

The local DNS server finds the malicious site in its cache and forwards the user to the malicious site.

In more technical terms the above scenario can be described as [10] a DNS query is sent over the connectionless UDP protocol. With each request a UDP response is associated via the source and destination host and port (UDP properties), and via the 16 bit transaction ID value. Assuming that an attacker knows that a DNS query for a specific domain is about to be sent, from a specific DNS server/resolver, the attacker can trivially predict the source IP address, the destination IP address and the destination UDP port (53 – the standard UDP port for DNS queries). The attacker needs additional 2 data items – the source UDP port, and the DNS transaction ID, to be able to blindly inject his/her own response (before

the target server's response – typically DNS server use the first matching response and silently discards any further responses).

Deficiencies causing attacks

There exist some deficiencies in the DNS protocol and defects in common DNS implementations that facilitate DNS cache poisoning attack. The following are examples of these deficiencies and defects [11]:

Insufficient transaction ID space

It is possible for an attacker to attempt to successfully predict the transaction ID field (consist of 16 bits) described in the DNS protocol specification. On average an attacker required 32,768 attempts to successfully predict the ID. If smaller number of bits for this transaction ID are selected than an attacker require fewer attempts to predict the ID.

Multiple outstanding requests

Multiple requests for the same resource record (RR) is a vulnerability caused by some implementations of DNS services. This vulnerability generates multiple outstanding queries for that RR. As a result of this vulnerability, it is possible for an attacker to apply a 'birthday attack' technique to dramatically improve the probability of a successful DNS spoofing attack. When performed against a caching name server, this can result in cache poisoning.

1.2.4 Follow-On (Enabling) Attacks

It is important to illustrate the situations that could arise in case of attack on DNS in any form.

The attacker can make an effective DoS attack against both the requesting party as well as the service provider by making the selected destination pointing to an offline or nonexistent address.

An attacker can make a fake site and redirect the legitimate user to this malicious site e.g. in case of online banking website if an attacker successfully redirect the user to the fake site then he/she can steal the user's confidential data like passwords, credit card numbers etc.

It can also be case, in which an attacker could proxy connections and serves as a man-in the- middle to capture all the data exchanged between the client and the bank's website, including login information, etc, and would not even need to construct a fake site.

2 DNS Security.

Protecting these kind of attacks require security [1][8]. DNS security comes in several flavors- the queries, responses and other messages your name servers sends and receive.

You can secure your name server, refusing queries, zone transfer requests, and dynamic updates from unauthorized addresses, for example. You can even secure zone data by digitally signing it.

2.1 TSIG

BIND 8.2 introduced a new mechanism for securing DNS messages called *transaction signature, or* TSIG for short. TSIG uses share secrets and a one- way hash function to authenticate DNS messages, particularly responses and updates. TSIG is relatively simpler to configure, light weight for resolvers and name servers to use, and flexible enough to secure DNS messages (including zone transfer) and dynamic updates.

With TSIG configured, a name server or updater adds a TSIG record "signs" the DNS message, providing that the message's sender had a cryptographic key shared with the receiver and that the message was not modified after it left the sender.

There is no provision that has been made to distribute the share secret keys. It is up to the Network Administrator that he configures the Domain Name Server and client using some kind of mechanism known as sneakers-net until a secure automatic mechanism for key exchange is available.

2.1.1 One-Way Hash Functions

It is also called a cryptographic checksum or message digest that computes a fixed-sized value based on arbitrary input. This is calculated by a mathematic formula call One-Way Hash Function. TSIG provide authentication and data integrity by using it. The output depends on the each and every bit of output, if there is change in a single bit in input the resulted output will also change. It is computationally infeasible to reverse the function and find an input that produces a given hash value.

TSIG uses a one way hash function called MD5. In particular it uses a variant of MD5 called HMAC-MD5. It generates a 128-bit hash value that depends not only on the input but also on a key.

2.1.2 The TSIG Records

TSIG is a "meta-record" that never appear in zone data and is never cached by the resolver or name server. A signer adds TSIG records in a DNS message and the receiver removes the record and verifies it before doing anything further. A TSIG record is calculated over the entire DNS message means that the resulted hash value in calculated on the entire DNS message, and additional data are fed into the HMAC-MD5 algorithm to generate the hash value. The hash value is keyed with a secret share between the signer and verifier. That proves that the DNS

message is signed by the holder of a share secret and that it was not modified after it.

2.1.3 Configuring TSIG

There are one or more keys which are configured on either end of the transaction before using the TSIG for authentication.
For example, if we want to use TSIG to secure zone transfer is between the master and slave name servers for movie.edu, we need to configure both name server with common key:

```
Key-terminator-wormhole.movie.edu. {
  Algorithm hmac-md5;
  Secret "skrkc4twy/cIgIykQu7JZA==";
};
```

terminator-wormhole.movie.edu. is the name of the key and is encoded in the DNS message in the same way as the domain name. The TSIG RFC 2845 suggests, name the key after two hosts that use it and it also suggests that use different keys for each pair of hosts. If the keys are not same at both sides of the system it will generate the error message like:

```
Nov 21 19:43.00 wormhole named-xfer [30326]: SOA
TSIG verification from server
[192.249.249.1], zone movie.edu: message has BADKEY
set (17).
```

Algorithm is now hmac-md5. The secret is base 64 encoding of the binary key. BIND 8.0 and BIND 9.0 introduces dnssec-keygen for generating the base 64 – encoded key. Key generated method using dnssec-keygen is:

```
# dnssec-keygen –a HMAC-MD5 –b 128 –n HOST
terminator-wormhole.movie.edu.
Kterminator-wormhole.movie.edu.+157+28446
```

The option –a take the argument name of the algorithm that is HMAC-MD5 and use with the key, -b take the length of the key as its argument that is 128-bits long. –n takes an argument HOST, the type of key to generate. The last argument is name of the key.

2.1.4 Using TSIG

Once the configuration has been done successfully with TSIG keys, we should then configure them using these keys. BIND 8.2 and later version uses TSIG to secure queries, responses, zone transfer and dynamic updates.
The work in the configuration is to configure the server statement's key sub statement, which tells a name server to sign queries and zone transfer requests sent to a particular name server. This server substatement, for example, tells the local name server, wormhole.movie.edu, to sign all such request sent to 192.249.249.1(terminator.movie.edu) with the key terminator-wormhole.movie.edu.

```
Server 192.249.249.1 {
Keys {terminator-wormhole.movie.edu. ;};
} ;
```

Now, on terminator. movie.edu, we can restrict zone transfers to those signed with the terminator-wormhole.movie.edu key:

```
Zone "movie.edu" {
        Type master;
        File "db.movie.edu";
        Allow-transfer {key terminator-
wormhole.movie.edu. ;};
};
```

Terminator.movie.edu also signs zone transfer, which allows wormhole.movie.edu to verify it.
Similarly dynamic updates are also restricted using TSIG by using the allow-update and update-policy substatement.

2.2 Securing Name Server

BIND 4.9 introduced several important security features that help to protect name server [1]. These features are particular important if name server is running one the internet, but they are purely useful on internal name servers. Here we will discuss the following:

2.2.1 BIND Version

The BIND versions using to protect your name server also is a critical and affects your name server's security. All versions before BIND 8.2.3 are susceptible for various kinds of DNS attacks. There is another issue related to the security: if an attacker know which version of BIND you are using then he can make attack according to that. Some earlier version of BIND name server replies to client with the information that was enough to now about the BIND name server version. BIND versions 8.2 and later address this problem in their implementation. The syntax of the reply query of these recent versions is, for example:

```
Options {
        Version "None of your business"
    };
```

But the message is still a tip that there is latest version of the BIND is in practice.

2.2.2 Restricting queries

The idea behind DNS was to make information available for all over the internet to the desired users. In the very earlier version of the BIND, administrator has no way to

look up names on their name server. BIND 8 and 9 *allow-query* sub statement so that you can apply IP address-based access control to queries. This also allows to access particular zone's data.
It allow which ip address is allowed to send queries to the server.

Restricting All Queries

The global form of allow-query substatement looks like this:

```
        Options {
                address_match_list;
};
```

So to restrict the name server to answering queries from three different networks are for example:

```
        Options {
        allow-query {192.249.249/24; 192.253.253/24;
         192.253.254/24;};
};
```

Restricting queries in a particular zone

BIND 8 and 9 allow using access control list to a particular zone. The format of this would be like, for example:

```
acl "HP-NET" {15/8 ;}

Zone "hp.com" {
        type slave;
        file "bak.hp.com";
        masters {15.255.152.2;};
        allow-query {"HP.NET"};
};
```

Any kind of authoritative server, master or slave can apply access control list. Zone-specific access control is more permissive and always takes precedence over global access control lists. If zone-specific access control list is not implementing then global access control will be applied.
In BIND 4.9 this functionality is provided by the secure_zone record. It collectively limits queries for individual records and zone transfer also. The major drawback of BIND 4.9 is that it is used only for authoritative zones. There have no mechanism for restricting who can send your server queries for data in zones your server is not authoritative for. To use secure_zone includes one or more special TXT records in the zone data on the primary master name server. The TXT include:

```
        Address: mask
```

Or

```
        Address: H
```

In the first form, address is the dotted-octet form of the IP network which you want to give access the particular zone and mask is a network mask of that network.
In the second form address is that particular IP address to which you want to give access to zone and H is equivalent to the mask 255.255.255.255; each bit in the 32-bit address is checked. Similarly BIND 4.9 also increases the load of writing very much queries to restrict access to information for particular hosts in the network. Each host is separately restricted like:

```
        secure_zone    IN    TXT    " IP address:mask "
```

2.2.3 Preventing Unauthorized zone transfer

Although it is import to limit who can query your name server but it is also important to ensuring that only slaves name servers can transfer zone from your name servers. Remote hosts can only look up records for domain names they already know. If ensuring is not define well then any remote user can transfer zone data and can list all records in the zones.
BIND 8 and 9 allow-transfer substatement and 4.9's xfrnets allows implementing access control lists on zone transfers. Allow-transfer restricts particular zone when used as a zone substatement, and restricts all zone transfers when used as options substatement. It takes an address match list as arguments. In BIND 8 and 9 zones transfer is allowed from any IP address by default and hackers can easily take the advantages of it, they can transfer the zone from the slave servers. Therefore allow-transfer property must be disabled for it by ensuring allow-transfer {none}.
BIND 8 and 9 allow applying a global access control list to zone transfer. This make it possible to implement zone transfers that don't have explicitly defined access control list defined as zone substatements. For example to limit all zone transfers to internal IP addresses:

```
        Options {
        allow-transfer {address; address; address};
    };
```

2.2.4 Running BIND with Least Privileges

Running a network sever such as BIND as the root server can be dangerous. This often happens in implementations of BIND. If hackers find flaws in the system and get access to it, then he can enjoy the root users privileges and exploits accordingly. This will allow them to execute command, read and write files to perform his desired functions.
BIND 8.1.2 and later versions allows changing the user and group privileges the name server uses to run. This is known as least privilege for that particular configured server: the minimum set of requirements it need to complete the job. It also include an option to *chroot ()* the name server.
The command line options that allow these features to implement are:
-u specifies the username the name server changes to after

starting, e.g., named –u bin.

-g specifies the group or group id the name sever changes after starting, e.g., named –g other.

-t specifies the directory for the name server to chroot() to.

2.2.5 Split-Function Name Servers

Name servers perform two functions: answers remote name servers iterative queries and other answer local resolver's recursive calls. If the separation has to be made for these two name servers than the risks of attacks can be reduced efficiently. There are two types of separation can be made.

Delegated name server Configuration

These name severs appears in the NS records delegate zone to name servers who take care of the nonrecursive queries on the internet. For this it must be assured that the name server must not be receive any recursive call. It could also be configured to response nonrecursively even on recursive calls.

Resolving name server configuration

Unlike delegating name server, resolving name servers can not restrict recursive calls. So some configuration is to make to allow the recursive queries. Name servers are configured to response queries from their own resolver name servers and deny any other query which is not from our own IP addresses.

BIND 8 and 9 allow this that which IP addresses can send queries to our network. BIND 4.9 allows this via the secure_zone TXT record.

2.3 DNS and Internet Firewalls

The DNS was not designed to work with internet firewalls. It's a testimony to the flexibility of DNS and of its BIND implementation that you can configure DNS to work with, or even through, an internet firewall [1]. Despite that it also requires a deep knowledge of DNS and BIND's most obscure features.

2.3.1 Internet firewall software

In order to configure BIND with firewall it is important to know about the capabilities of current firewall. Because firewall's capabilities influence the choice of DNS architecture and determine how you implement it. The two most implemented firewall softwares are:

Packet filters

Packet filtering firewalls operates at network layer and transport layer of TCP/IP protocol stacks (layer 3 and 4 of OSI network layer Model). Packets are routed based on the packet-level criteria like transport protocols (TCP or UDP), Source and destination IP address, source and destination ports.

In the context of DNS, packet filtering firewalls can be configured so that it can selectively allow internal network systems and the host on the internet to communicate. Some packet-filtering firewalls also allow the arbitrary numbers of name servers to query at the internet but does not allow vice versa. All router based internet firewalls are packet-filtering firewalls. Chechpoint's Firewall-1,Cisco's PIX, and Sun's SunScreen are popular commercial packet- filtering firewalls.

Application gateway

Application gateways firewalls operate at the application layer of OSI reference model. They sense the application protocols in the same way, a server for that particular application would. An FTP application gateway, for example, can make the decision to allow or deny a particular operation.

The major drawback when working with the application-based gateways is that they handle only TCP-based application protocol. And off course DNS uses UDP-based, and there is not application gateway for DNS. As a result your internal host will not be able to directly interact with the name server at the internet.

2.3.2 Internet Forwarders

Internet forwarders take the responsibility to communicate between the internal networks hosts and rest of the internet. They limit the danger of bidirectional DNS traffic. In any application gateway firewalls, the only host that can communicate with the name servers at the internet is Bastion host, as depicted in the Figure below.

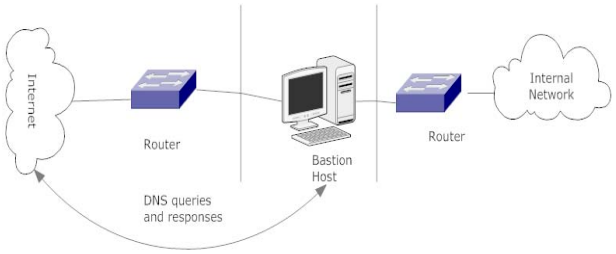


Figure 2.1. A small network, showing the bastion host

When an organization has a larger architecture and have a few name servers inside the network, packet-filtering firewalls can be used. The firewalls administrator can configure it so that small set of internal name servers can communicate with internet name servers. The figure below shows this scenario. All the internal name servers can query to internet name server without doing any major configuration.

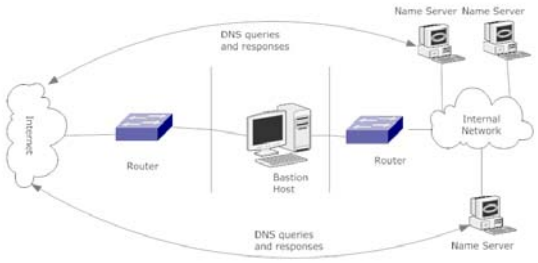


Figure 2.2. A small network, showing select internal name server

Drawbacks of Forwarders

If a corporate has a large business and have a business spread over continents with thousand of hosts and many of the name servers also, further more all of the organization's name severs don't have direct access to internet and relying only on the forwarders to resolves all the queries and connection to the internet can introduce the following disadvantages.

1. Single point of failure

If the forwarder fails, the resolvers could not be resolves internet domain names and internal domain names.

2. Concentration of load

Forwarders always has to accommodate a huge load balance due to huge network and a lot of name servers and because the queries are recursive etc.

2.3.3 Internal roots

Internal root severs solves the problem of scalability by implement as many as possible internal root name servers. Inside of the organization they just know about the namespaces of their own network.

Implementing this architecture there are certain benefits of distributed the load, redundancy and efficient resolution. But it is not without its cost, there will need a lot of efforts to configure to many internal roots name servers.

Therefore if an organization has very large networks and hosts, than implementing many of them as roots name servers as forwarders could be a good solution.

2.3.4 A Split Namespace

Unfortunately BIND does not support automatic filtering of zone data. Many organizations create split namespaces manually, in which the only internal hosts know about the real namespace and the translated versions of it that is called shadowing would be available to the rest of the internet. Shadowing namespaces performs mapping of name-to-address and address-to-name of those name servers that are accessible through the firewalls.

3 The DNS Security Extensions

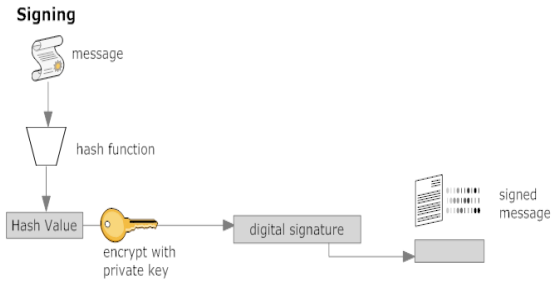
TSIG is well suited to securing the communications between two name servers and between an updater and a name server [1] [12]. However it won't protect if one of the names severs is compromised. The most common way to deal with key management problems like these is to use public key cryptography.

3.1 Public key cryptography and digital Signatures

In public key cryptography two keys are used for encryption and decryption of the message, e.g., public key and private key and an asymmetric algorithm is used to exchange the keys. When a user wants to send the message to the recipient, he encrypts the message with the public key and then sends the encrypted message to the other counterpart. If the recipient has kept his private key private then only he would decrypt the message. As a response the recipient can also encrypt the message by using his private key and send it to someone. If the receiver succeeded to decrypt it by attempting it by the public key, and the sender also did not reviled his private key to anyone then he will perform his task successfully. It also proves that the message is not decrypted in transit.

Encrypting large amount of data with an asymmetric algorithm is very slow and time consuming than encrypting with the symmetric encryption algorithm. But when public key encryption is used for authentication, not for privacy then the whole message's hash function is to be taken and instead of whole message to be encrypted, the hash value is encrypted using private key that represent the whole message. Then the digital signatures are attached to the hash value to get the sign message.

The receiver of the message can also verify the message by decrypting the digital signature with his/her public key to get the one hash value. Meanwhile he can also run the message to his/her own copy of the hash function. If the hash values are match, then message is authenticated. This whole method of signing and verifying is described in the Figure below:



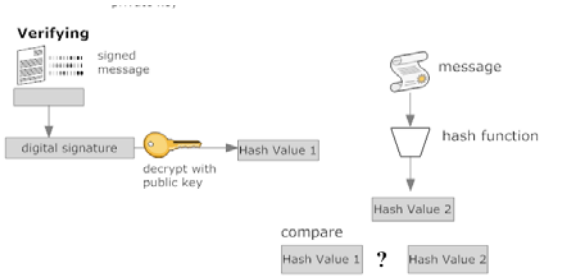


Figure 3. Signing and verifying a message

3.2 The key record

In DNS Security Extension or DNSSEC the key record is used to advertise the public key of a zone that will be attached to domain name of that particular zone. The private key of the zone must be stored somewhere in a file of a name server's files system. The key record is not only limited to store the zone's public key but many other cryptographic key can also be stored in it.

3.3 The SIG record

As the key record is used to store the zone's public key, then a new record to store the private key's signature is needed. Therefore SIG record is used to store the digital signatures of private keys on an RRset, which is a group of resource records that have the same owner class and type. The RRset class accommodates many of records types and saves time.

3.4 The NXT record

The next record solves the problem of signing negative responses. If there receive a query to look up domain name that does not exists in the secure zone's area, then if the zone were not secure it will simply response with a message "no such domain name exists " in the response code. These response codes are signed by the NXT record.

NXT record also bridges the gap between two consecutive domain name systems, so that which domain name comes after the other. To maintaining the order of different domain names is an issue that always need to taken seriously.

3.5 DNSSEC and Performance

DNSSEC does not come without its cost, it increases the size of DNS messages and as a result its demand for more computation power and resources from name servers for signing zone's data. Following are the consequences of these effects:

- Larger messages are a huge load for resolvers and domain name systems and requires processing when TCP in place as it already more resource intensive than UDP.
- Verify zone data also takes time and slow the resolution process.
- Larger zones mean larger memory consumption and processing power.

BIND 8 can not fulfill these requirements to signing the secure zones as it require more then the BIND 8 offers.

BIND 8 motivate towards the development of new and more capable of DNS server, and take part in the development of BIND 9.

4 Conclusions

The Domain Name systems are very critical service providers and every day we rely on it for our different tasks. The origin of DNS is very long before even when computer networks are not being used for commercial application, e.g., e-commerce. DNS vulnerabilities are appearing frequently as DNS interaction are increase. The vulnerabilities I have described in this report are even not new but are good guide to understand the attacks that can be made against DNSs. The need of authenticating during zone transfers and between resolving name servers and clients will eventually necessitate the need of wide spread DNS Security Extensions. The DNSSEC is a great achievement towards DNS security with the development BIND 9. Although DNSSEC requires huge computation powers and resource to implements it services, is still being implementing rapidly due the advancement in network equipments, storage devices and processing equipments. When implemented properly, offers the highest level of security and reduces network traffic. In addition, it reduces storage requirements and enable efficient mutual authentication.

References

[1] DNS and BIND, Help for System Administrators by, Paul Albitz & Cricket liu (4th edition) O'REILLY 2001.

[2] DNS Security, Antonio Lioy, Fabio Maino, Marius Marian, Daniele Mazzocchi Dipartimento di Automatica e Informatica Politecnico di Torino Torino (Italy), Terena Networking Conference, 22-25 May 2000.

[3] CERT/CC Denial of Service Attacks, http://www.cert.org/tech_tips /denial_of_service.html, April 23, 2009.

[4] RIPE NCC, May 3, 2009, http://www.ripe.net/news/global-root-server.html.

[5] DDoS Attacks on Root Nameservers, http://en.wikipedia.org/wiki/Distributed _denial_of_service_attacks_on_root_ Nameservers, 4 May 2009.

[6] DoS Attacks, 5 May 2009, http://d.root-servers.org/october21.txt,

[7] DNS Spoofing Techniques, April 26, 2009,

http://www.securesphere.net/download
/papers/dnsspoof.html.
[8] Practical Domain Name System Security: A Survey of Common Hazards and Preventative Measures by Nicholas A. Plante. College of Computer and Information Science Northeastern University, Boston MA, 2003.
[9] DNS Cache Poisoning Attacks, May 1,2009, http://www.checkpoint.com/defense/advisories/public/dnsvideo/index.html.
[10] BIND 9 DNS Cache Poisoning by Amit Klein. http://www.trusteer.com/files /BIND_9_DNS_Cache_ poisoning.pdf, May 4, 2009.
[11] US-CERT Vulnerability Note, May 3, 2009, http://www.kb.cert.org/vuls/id/800113.
[12] A New Approach to DNS Security (DNSSEC) Giuseppe Ateniese, Department of Computer Science and JHU Information Security Institute, Johns Hopkins University, 3400 North Charles Street, Baltimore, MD 21218, USA, 2001 ateniese@cs.jhu.edu..
Stefan Mangard Institute for Applied Information Processing and Communications (IAIK)
Graz University of Technology, Inffeldgasse 16a
8010 Graz, Austria stefan.mangard@iaik.at