Interested in learning more about security?

SANS Institute
InfoSec Reading Room

This paper is from the SANS Institute Reading Room site. Reposting is not permitted without express written permission.

# The Achilles Heal of DNS

One of the four categories of Denial of Service (DoS) attacks list by Scambray, McClure, and Kurtz is "Routing and DNS attacks."1, refers to attacks which corrupt the information these systems use to perform their functions. Information Poisoning, though more general, is a more accurate term for categorizing these types of attacks . It is also more inclusive of attacks such as ARP Poisoning which employ similar tactics and are possible because of a common vulnerability. Each of the protocols associated with these attac...

Copyright SANS Institute
Author Retains Full Rights

AD
Take control of vulnerability threats Now!
Request a Free Trial
Secunia Stay Secure

# The Achilles Heal of DNS

Christopher Irving
Security Essentials Version 1.2e

## Introduction

One of the four categories of Denial of Service (DoS) attacks list by Scambray, McClure, and Kurtz is "Routing and DNS attacks."[1] This refers to attacks which corrupt the information these systems use to perform their functions. Information Poisoning, though more general, is a more accurate term for categorizing these types of attacks. It is also more inclusive of attacks such as ARP Poisoning which employ similar tactics and are possible because of a common vulnerability. Each of the protocols associated with these attacks either completely lacks or has very poor methods of authentication. Attackers capitalize on this weakness to undermine the trust relationship between two systems. This paper will attempt to illustrate consequences of this deficiency. Buffer overflows and other attacks on specific software that implement DNS will not be covered.

## Background

The Basic function of the Domain Name System (DNS) is to translate domain names, which are easy for humans to remember, into the numerical IP address which are simpler for computers. It is a distributed database in which local administrators have control over segments that contain the information about their domain. Name servers are the software programs that implement the database and respond to queries by clients. Clients, which can be either a host's resolver or another name server, query the name servers to learn the domain name to IP mapping.[2] The DNS protocol defines the message format a client uses to query the name server database and the format of the name server's response.

When a client wishes to know the IP address of a particular host's domain name it sends a query to the name server. If the name server is authoritative for that domain name (in which case it will always know answer) or if it contains the information in its cache, it will answer to the query. What happens when the name server doesn't know the answer depends on a couple of factors; the query type and the name server's configuration. Most resolvers always send recursive queries. As long as the name server is configured to accept them, it will recursively resolve the domain name. The originally queried server now becomes the client of the authoritative name server. When it receives the answer from the authoritative name server, the original name server passes it on to the resolver. It also stores a cached copy of the response for a limited period of time. When subsequent queries are made for the same domain name it will use the information in its cache to generate a response instead of doing more recursion. If the domain name doesn't exist or the name server is configured to refuse recursive queries, an error will be returned to the resolver.[3] For the purposes of this paper it will be assumed that all name servers accept recursive queries.

## Information Poisoning

© SANS Institute 2003, As part of the Information Security Reading Room. Author retains full rights.

As stated above, DNS is a mechanism for resolving domain names to IP addresses. Client systems wishing to know the IP address of a machine it would like to contact use a resolver to query a DNS name server. The name server sends back a reply containing the IP address to the client. Unfortunately, the DNS protocol has virtually no authentication method built into it. There is nothing in the protocol that provides a means to ensure that the requesting client is who it says it is or that the replying name server is a real name server. The message headers of a DNS query and response do contain a 16-bit identification field but it is mostly used for matching queries with responses. If an attacker can successfully predict future values of the identification number, he could fool a DNS client into accepting a false reply as the real one. The client could be a host resolver, but there is a much greater potential for harm if the client is another name server doing recursive resolution.

Prediction of the identification number can be trivial. Older versions of BIND and other client software have been known to use sequentially increasing numbers for the identification field.[4] If an attacker is able to determine the ID of one query, he could easily guess the identification number for the next and subsequent queries. When the next query is sent the attacker sends an impersonated response using the predicted sequence number and containing the false information. As long as the impersonated packet arrives before the response from the real server, the resolver will match the identification number of the reply with that of the query and accept it as the official answer.

An intelligent attacker won't simply spoof the identification number of the DNS query; he will also spoof the address fields in the IP packet to make it appear as though the response actually came from the real name server. By doing this it will make it more difficult for the victim to identify the source of the false responses. This is possible since the machine he is attacking from is completely under the attacker's control and he could presumably craft packets in anyway he desired. Therefore the attacker will alter the IP datagram so that the source address is that of the real server and modify the query ID in the DNS message so that it appears to be the real reply. When the packet from the real server arrives, the DNS client software will assume it has already completed the transaction for that query ID and simply drop it on the floor.

By providing false information to the requester the attacker has kept the requester from accessing the information they want, thus creating a Denial of Service. Even worse, the attacker could redirect the victim to any host he wanted. The scope of the DoS depends on what the client was that was doing the query. If the query came from the resolver on a host machine, then more than likely only one person will be affected. This really isn't a very interesting scenario. What if the query was sent to another DNS server doing a recursive lookup? The false response would then be stored in the querying server's cache until the entry's time to live period expired. Attackers will often set the Time to Live field to a very high number so that it will remain in the cache longer (of course this is DNS TTL field, which is completely different from the IP TTL field.) This is commonly referred to as DNS cache poisoning. Any subsequent queries for this record to the

© SANS Institute 2003, As part of the Information Security Reading Room. Author retains full rights.

poisoned server will yield the bogus information. The scope of the Denial of Service would increase significantly. Every client, not just a single user, of that name server would be affected by the false information.

By now the reader is probably thinking that this all sounds good in theory, but how does the attacker actually find out the query ID sequence? And once the attacker can predict the ID sequence, how does he use that the poison the DNS cache. It works something like the following:

First the attacker configures a resolver to point to the victim's name server. He then uses the resolver to query the victim's name server for information about a host on his own domain, or any domain where he can sniff the network traffic destined for the name server. The attacker may need to do this several times depending on how difficult it is to determine the sequence used to generate the query ID. Even in the simplest case, where the ID is incremented sequentially, the attacker would have to do this twice. Once he can accurately predict the query ID, the attacker will query the victim's name server for information about hosts in other domains. When the victim's name server does a recursive lookup to find out the correct information, the attacker uses the predicted query ID number as described above to provide a false response. To ensure that the falsified response is received before the response from the real authoritative name server, the attacker may simultaneously DoS the real authoritative name server.

Newer versions of BIND and other software have changed their implementations to randomize the identification number in an attempt to combat this attack. However, this doesn't completely solve the problem. If the attacker was capable of sniffing the packets that contained the queries, prior knowledge of the identification number really isn't necessary. All the information that the attacker needs would be in the sniffed packet. The only requirement of the attacker is that he is able to get his packet to arrive before the response from the real server. Again, a very determined attacker would also be committing a Denial of Service attack against the real DNS server so that it was incapable of responding, or at least could not respond as quickly. However, this scenario is much less likely and if an attacker could sniff packets sent to a victim network, the victim would have a lot more to worry about than poisoned DNS information.

## Expanding the Scope

In these cases the scope is still rather limited to one record at a time and one server at a time. Perhaps only one company would be unable to reach the systems in a single domain or even several domains. The impact on the Internet community as a whole, in this case, would be small. It would be much more effective if an attacker could poison the authoritative source of an organization's DNS records. In some situations just such an attack is possible. Many companies these days have a second party act as their external authoritative DNS server. Although the second party is registered with the Internic as being the authority for DNS information, control of the information is actually kept by the company. The second party's DNS server is a slave and transfers all its

© SANS Institute 2003, As part of the Information Security Reading Room. Author retains full rights.

information from the organization's real primary server. This is known as invisible primary. It is also susceptible to an information poisoning attack.

Instead of attempting to spoof the response to a single query, the attacker attempts to hijack the transfer of an entire zone between the primary DNS server and the slave acting as the authoritative answer. Since there is little difference in the DNS message format between a query and a zone transfer, the attack is similar to those described above. One of the biggest differences between the message types is that query messages are sent via UDP while zone transfers use TCP. The impact of poisoning the information of a Domain's authoritative DNS server would be much greater. Eventually every system wishing to know the translation information for that domain will query the poisoned server. The false information will be propagated out to every DNS server that does a lookup on the poisoned domain. No one would be able to reach the victim domain and the attacker could redirect traffic to wherever he wanted.

The difference between this attack and the cache poisoning attack is subtle but important. In the cache poisoning attack only those systems which use the poisoned name server for DNS resolution are affected. When the database of a domain's published authoritative name server is poisoned everyone looking for an authoritative answer for information about the victim domain is affected.

## Using DNS Servers for Bandwidth Consumption DoS Attacks

In April of 2000 CERT (The Computer Incident Response Team) released incident note IN-2000-04. The note describes the use of DNS name servers to execute bandwidth consumption denial of service attacks. Although it shares many of the same characteristics as Smurf, Fraggle and other attacks which use IP spoofing, it doesn't utilize broadcast addresses to achieve amplification. Instead it uses the fact that DNS response messages may be substantially larger than DNS query messages.[5] Not including the IP headers or the UDP headers, a query message is approximately 24 bytes (depending on the size of the variable length Query Domain Name field). A response message could easily triple that size. This is illustrated in figures 1 and 2. Figure 1 shows a packet for a DNS query. The size of the entire packet is given in the Packet size field of the Ethernet header, 79 bytes. The length of the UDP segment is 44 bytes which means the size of the DNS query message is indeed 24 bytes (44 – 20). Compare this to figure 2 which depicts a packet for a DNS response. It has a total packet size of 206 bytes of which the response message makes up 152 bytes.

Figure 1.

```
ETHER: ----- Ether Header -----
ETHER:
ETHER: Packet 14 arrived at 10:50:9.49
ETHER: Packet size = 79 bytes
ETHER: Destination = 0:0:c:7:ac:14, Cisco
ETHER: Source = 8:0:20:a2:e3:bc, Sun
ETHER: Ethertype = 0800 (IP)
ETHER:
IP: ----- IP Header -----
IP:
IP: Version = 4
IP: Header length = 20 bytes
```

© SANS Institute 2003, As part of the Information Security Reading Room. Author retains full rights.

```
IP: Type of service = 0x00
IP: xxx. .... = 0 (precedence)
IP: ...0 .... = normal delay
IP: .... 0... = normal throughput
IP: .... .0.. = normal reliability
IP: Total length = 65 bytes
IP: Identification = 47370
IP: Flags = 0x4
IP: .1.. .... = do not fragment
IP: ..0. .... = last fragment
IP: Fragment offset = 0 bytes
IP: Time to live = 255 seconds/hops
IP: Protocol = 17 (UDP)
IP: Header checksum = 985b
IP: Source address = 10.2.32.65, host.mydomain.com
IP: Destination address = 10.2.2.1, ns1.mydomain.com
IP: No options
IP:
UDP: ----- UDP Header ----
UDP:
UDP: Source port = 34465
UDP: Destination port = 53 (DNS)
UDP: Length = 45
UDP: Checksum = D7E0
UDP:
DNS: ----- DNS Header -----
DNS:
DNS: Query ID = 23865
DNS: Opcode: Query
DNS: RD (Recursion Desired)
DNS: 1 question(s)
DNS: Domain Name: www.someoneelse.com.
DNS: Class: 1 (Internet)
DNS: Type: 1 (Address)
DNS:
```

Figure 2.

```
ETHER: ----- Ether Header ----
ETHER:
ETHER: Packet 17 arrived at 10:50:9.59
ETHER: Packet size = 206 bytes
ETHER: Destination = 8:0:20:a2:e3:bc, Sun
ETHER: Source = 0:10:b:49:0:a0,
ETHER: Ethertype = 0800 (IP)
ETHER:
IP: ----- IP Header ----
IP:
IP: Version = 4
IP: Header length = 20 bytes
IP: Type of service = 0x00
IP: xxx. .... = 0 (precedence)
IP: ...0 .... = normal delay
IP: .... 0... = normal throughput
IP: .... .0.. = normal reliability
IP: Total length = 192 bytes
IP: Identification = 52070
IP: Flags = 0x4
IP: .1.. .... = do not fragment
IP: ..0. .... = last fragment
IP: Fragment offset = 0 bytes
IP: Time to live = 254 seconds/hops
IP: Protocol = 17 (UDP)
IP: Header checksum = 8680
IP: Source address = 10.2.2.1, ns1.somedomain.com
IP: Destination address = 10.2.32.65, host.somedomain.com
IP: No options
IP:
UDP: ----- UDP Header ----
UDP:
UDP: Source port = 53
```

© SANS Institute 2003, Author retains full rights

```
UDP: Destination port = 34465
UDP: Length = 172
UDP: Checksum = 1B5B
UDP:
DNS: ----- DNS Header -----
DNS:
DNS: Response ID = 23865
DNS: AA (Authoritative Answer) RA (Recursion Available)
DNS: Response Code: 0 (OK)
DNS: Reply to 1 question(s)
DNS: Domain Name: www.someoneelse.com.
DNS: Class: 1 (Internet)
DNS: Type: 1 (Address)
DNS:
DNS: 1 answer(s)
DNS: Domain Name: www.someoneelse.com.
DNS: Class: 1 (Internet)
DNS: Type: 1 (Address)
DNS: TTL (Time To Live): 1200
DNS: Address: 10.1.28.62
DNS:
DNS: 3 name server resource(s)
DNS: Domain Name: someoneelse.com.
DNS: Class: 1 (Internet)
DNS: Type: 2 (Authoritative Name Server)
DNS: TTL (Time To Live): 1200
DNS: Authoritative Name Server: ns1.someoneelse.com.
DNS:
DNS: Domain Name: someoneelse.com.
DNS: Class: 1 (Internet)
DNS: Type: 2 (Authoritative Name Server)
DNS: TTL (Time To Live): 1200
DNS: Authoritative Name Server: ns2.someoneelse.com.
DNS:
DNS: Domain Name: someoneelse.com.
DNS: Class: 1 (Internet)
DNS: Type: 2 (Authoritative Name Server)
DNS: TTL (Time To Live): 1200
DNS: Authoritative Name Server: ns.someoneelse.com.
DNS:
DNS: 3 additional record(s)
DNS: Domain Name: ns1.someoneelse.com.
DNS: Class: 1 (Internet)
DNS: Type: 1 (Address)
DNS: TTL (Time To Live): 1200
DNS: Address: 10.1.28.43
DNS:
DNS: Domain Name: ns2.someoneelse.com.
DNS: Class: 1 (Internet)
DNS: Type: 1 (Address)
DNS: TTL (Time To Live): 1200
DNS: Address: 10.1.28.35
DNS:
DNS: Domain Name: ns.someoneelse.com.
DNS: Class: 1 (Internet)
DNS: Type: 1 (Address)
DNS: TTL (Time To Live): 1200
DNS: Address: 10.1.28.46
DNS:
```

Just as in other IP spoofing attacks, the attacker falsifies the source address field in the IP datagram to be that of a host on the victim network. Using the spoofed address, a DNS query for a valid resource record is crafted and sent to an intermediate name server. To increase the amplification effect the query is usually directed at a non-authoritative name server for the resource record. Thus the name server, as long as it is configured to, will do a recursive lookup to resolve the query. Recursive lookups add significantly to the

© SANS Institute 2003, As part of the Information Security Reading Room. Author retains full rights.

amount of data returned in the DNS response message. The attacker will continually send the query to the intermediate name server with all the responses going to the victim network. Potentially, an attacker could consume the entire bandwidth of a T1 by generating just a few thousand responses.

## Prevention

So how can such attacks be prevented? The best answer to that question is to fix the problem of DNS's lack of any real authentication. One solution that could currently be implemented utilizes existing technology. IPSec provides a means of authentication, via its Authentication Headers, and can also provide the additional benefit of encrypting the data transfer. The difficulty with this solution, of course, is that it would require a tunnel between every client and every name server on the Internet.

Another method would be to fix the problem at its source; modify the protocol to add some kind of authentication. In fact, there is just such an effort currently being worked on in the Internet community. DNSSEC is an addition to the DNS protocol, which provides a method of digitally signing the resource records in a response message. This authenticates the identity of the name server as well as validating its responses. Now, whenever a name server sends a response it includes a previously generated digital signature of the resource records. To verify the digital signatures the name server also sends a zone's public key which itself is signed by a higher level domain. Three new resource records are defined by DNSSEC to provide this functionality. The SIG resource record contains the digital signature, KEY contains the public key, and NXT contains the next host's name.[6]

"No attempt has been made to include any sort of access control lists or other means to differentiate inquirers."[7] DNSSEC provides only one-way authentication. Only the responses from a name server are digitally signed, there is no authentication of the client. Thus DNSSEC does not prevent an attacker from using a name server for bandwidth consumption DoS attacks. A possible improvement would require both the client and the server to use digital certificates. Besides providing a means of authenticating the two, the client and server could encrypt the messages using two-way SSL. However, the designers of DNS feel that since the information it provides is public, DNS doesn't need to include this extra precaution. They leave it as an issue to by handled by IPSec.[8]

BIND version 9.x, which is the most current version of the BIND software, does provide support for DNSSEC. However a large portion of existing name servers are still using an older version of BIND. It will take some time before DNSSEC can provide an effective solution. Every name server and client would have to be updated to a version that supports DNSSEC.

## Conclusion

All of the attacks discussed thus far use a technique known as spoofing. The DNS poisoning attacks use a more sophisticated version of this method know as a man-in-the-

© SANS Institute 2003, As part of the Information Security Reading Room. Author retains full rights.

middle attack. Spoofing is possible because of the lack of authentication in the DNS protocol. Other protocols suffer from the same weakness and are also vulnerable to information poisoning attacks. New security extensions to DNS have been designed to correct this problem. Yet, until DNSSEC has been completely accepted by the Internet community, lack of authentication will still be a problem and poisoning attacks will still be possible.

---

[1] Joel Scambray, Stuart McClure, and George Kurtz, Hacking Exposed: Network Security Secrets & Solutions, 2nd ed. (Berkeley: Osborne/McGraw-Hill, 2000), 485-7.

[2] Paul Albitz and Cricket Liu, DNS and BIND, 3rd ed. (Sebastopol: O'Reilly & Associates, 1998), 4.

[3] Ibid, 27-28.

[4] Sinead Hanley, "DNS Overview with a Discussion of DNS Spoofing", 6 November 2000, URL: http://www.sans.org/infosecFAQ/DNS/DNS.htm (1 August 2001).

[5] "Cert Incident Note IN-2000-04", 15 January 2001, URL: http://www.cert.org/incident_notes/IN-2000-04.html (1 August 2001).

[6] Vivian Burns, "DNSSEC and BIND9", 11 November 2000, URL: http://www.sans.org/infosecFAQ/unix/BIND9.htm (1 August 2001).

[7] "RFC 2535", March 1999, URL: http://www.ietf.org/rfc/rfc2535.txt?number=2535 (1 August, 2001).

[8] Ibid.

© SANS Institute 2003, Author retains full rights

© SANS Institute 2003, As part of the Information Security Reading Room. Author retains full rights.

Last Updated: June 18th, 2013

# Upcoming SANS Training

**Click Here for a full list of all Upcoming SANS Events by Location**

| | | | |
|---|---|---|---|
| **Digital Forensics & Incident Response Summit 2013** | **Austin, TXUS** | **Jul 09, 2013 - Jul 16, 2013** | **Live Event** |
| **SANS London Summer 2013** | **London, GB** | **Jul 09, 2013 - Jul 16, 2013** | **Live Event** |
| **SANS Rocky Mountain 2013** | **Denver, COUS** | **Jul 14, 2013 - Jul 20, 2013** | **Live Event** |
| **SEC528 SANS Training Program for the CompTIA® New Advanced Security Practitioner (CASP) Certification** | **Washington, DCUS** | **Jul 22, 2013 - Jul 26, 2013** | **Live Event** |
| **SANS Mumbai 2013** | **Mumbai, IN** | **Jul 22, 2013 - Jul 27, 2013** | **Live Event** |
| **SANS San Francisco 2013** | **San Francisco, CAUS** | **Jul 29, 2013 - Aug 03, 2013** | **Live Event** |
| **SANS Boston 2013** | **Boston, MAUS** | **Aug 05, 2013 - Aug 10, 2013** | **Live Event** |
| **SANS SEC 560: Network Penetration Testing @ Bangalore 2013** | **Bangalore, IN** | **Aug 05, 2013 - Aug 10, 2013** | **Live Event** |
| **Critical Security Controls Summit** | **Washington, DCUS** | **Aug 12, 2013 - Aug 18, 2013** | **Live Event** |
| **Industrial Control Systems Security Training - DC** | **Washington, DCUS** | **Aug 12, 2013 - Aug 16, 2013** | **Live Event** |
| **SANS Virginia Beach 2013** | **Virginia Beach, VAUS** | **Aug 19, 2013 - Aug 30, 2013** | **Live Event** |
| **SANS Thailand 2013** | **Bangkok, TH** | **Aug 19, 2013 - Aug 31, 2013** | **Live Event** |
| **SANS Melbourne 2013** | **Melbourne, AU** | **Sep 02, 2013 - Sep 07, 2013** | **Live Event** |
| **SANS Capital City 2013** | **Washington, DCUS** | **Sep 03, 2013 - Sep 08, 2013** | **Live Event** |
| **Network Security 2013** | **Las Vegas, NVUS** | **Sep 14, 2013 - Sep 23, 2013** | **Live Event** |
| **SANS Canberra 2013** | **OnlineAU** | **Jul 01, 2013 - Jul 13, 2013** | **Live Event** |
| **SANS OnDemand** | **Books & MP3s OnlyUS** | **Anytime** | **Self Paced** |