The State of the Art in DNS Spoofing

U. Steinhoff, A. Wiesmaier, and R. Araújo

Department of Cryptography and Computeralgebra
Technische Universität Darmstadt
Hochschulstr. 10; D-64283 Darmstadt, Germany

Abstract. The DNS is responsible for resolving human-readable domain names to numeric IP addresses. It is a protocol designed in the early days of the internet, and features only weak security mechanisms. The purpose of this paper is to give an overview of the threats on the current system for domain name resolution. We describe the DNS system and possible motivations for attackers. In the following we describe the different attacks, and discuss their success chances and possible countermeasures. We include an overview of affected versions of different DNS servers, and discuss their distribution in the internet. Lastly we give a summary on the risk of DNS spoofing.

Keywords. DNS Spoofing, Attacks, Countermeasures, IT-Security

1 The Domain Name System

The DNS is responsible for resolving human-readable domain names to numeric IP addresses. It is a protocol designed in the early days of the internet, and features only weak security mechanisms. In the last years have been several security flaws discovered in the protocol and its specific implementations. These enable attacks in different ways. This paper gives an overview over the different threats to the DNS and their success chances. The main focus of this paper is on attacks on DNS by external sources from the internet, not on attacks which need access to the communication in the local network. The threats on DNS communication in the scope of the local network (e.g. man-in-the-middle, sniffing) are similar to the well known security threads on every other plain text protocol without cryptographic authentication.

1.1 Introduction to DNS

The domain name system (DNS) is a distributed system used for the resolution of domain names to IP dresses and vice versa, and for storing additional information associated with the domain names. The DNS is organized in a hierarchical set of DNS servers, with the root servers on top. Each DNS server holds information regarding its domain including the name servers beneath it. It is called authoritative for this zone.

There are two kinds of resolution requests a DNS server has to answer. There are recursive questions ("Please give me an answer, even if you have to ask others for this."), in which the DNS server is asked to resolve the request by starting an iterative resolution process. If the asked server is not authoritative for the asked information it forwards the request to another DNS server. Then there are the non-recursive questions ("Please give me an answer, or tell me who could know about it."), which are answered only, if the server has the required information. Otherwise it delegates to a DNS server lower in the hierarchy. Normally recursive questions are only answered for hosts within the LAN. All other hosts are only allowed non-recursive queries about the servers authoritative zone.

To reduce the amount of traffic in the resolving procedures of domain names, the applications on a user's computer do not directly communicate with the DNS servers on the internet. The operating system features a resolver component which handles the DNS resolution for the applications, typically by forwarding the requests to a DNS server in the local network as shown in Figure 1. This server resolves the recursive query by iterative queries in the DNS, starting at the root servers, down to the authoritative DNS server for requested domain name (shown in Figure 2). The resolver as well as DNS server reduce the traffic by caching previous resolutions. A typical resolution request is processed as follows:

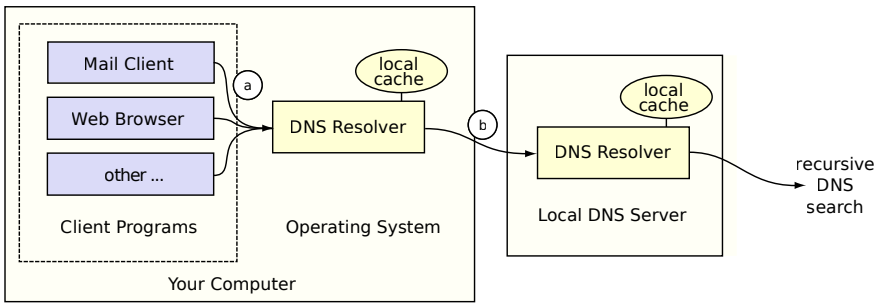

Fig. 1. Typical DNS resolution (based on [1])

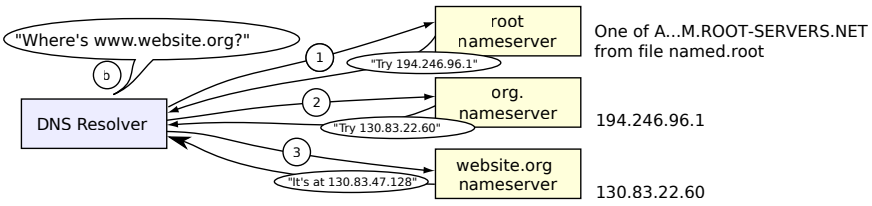

Fig. 2. Resolving a recursive query (based on [1])

- For example the browser needs to know the address of www.website.org. A resolution request is sent to the local resolver. (a)
- The local resolver looks up, whether the resolution of www.website.org is already in its cache. In this case the query is answered directly. Otherwise the resolver sends a recursive resolution request to the local DNS Server. (b)

– If the desired information is not in the local DNS server's cache it starts an iterative lookup of the domain name as follows:
  (1) The lookup for www.website.org starts at one of the 13 root servers of the DNS. Every DNS server holds a list of these servers. The root server delegates to one of the DNS servers for the de top-level domain (TLD).
  (2) In the next step the local DNS server queries this server for the resolution of www.website.org. The DNS server of the org-TLD delegates to one of the DNS servers of the website.org domain, e.g. name.server.website.org.
  (3) Lastly the local DNS server queries this name server, which is authoritative for the whole website.org and receives the IP address of www.website.org.
– At last the IP address is returned in an answer to the resolver of the users computer.

Of course this is only a simplified example, especially in huge local networks, divided into several subdomains, the resolution is not so straightforward. Good sources for in-depth information, including how to configure an own DNS server, are the books DNS&BIND by Albitz and Liu [2] and DNS on Windows Server 2003 by Allen [3]. A collection of RFCs regarding the DNS can be found on dns.net [4]. In the following we will concentrate on the 3 most common DNS Servers. These are BIND, the "standard" name server available for a broad range of OS including Unix and Windows; the MicrosoftDNS, integrated in Windows server operating systems; and DJBDNS, a secure, modular name server for Unix. All other name servers have only a small market share, as to be seen in Section 7.4.

1.2 Criteria for Validation of DNS Packets and Impact of RFCs

The DNS is normally based on the connectionless UDP-Protocol. Connection oriented TCP-Connections are used for queries and answers that are bigger than 512 byte. Additionally TCP is used for zone transfers which replicate the DNS data among the DNS Servers of a zone. The use of UDP simplifies the construction of spoofed packets for the attacker in contrast to the TCP, which employs an additional (randomized) sequence number. As discussed in Phrack Issue 62 [5], nine obvious criteria could be used for direct validation of an incoming DNS reply. However, when adhering to a liberal receiving behavior and the RFCs for DNS, UDP and IP, only 3 of the criteria could be applied for validating an incoming DNS reply. These remaining criteria are the destination port on the client side, correct Transaction ID and the timing constraint, that a spoofed packet in an attack needs to arrive before the reply of the legitimate server.

1.3 External Recursion

Nearly all of the external attacks are much easier, or even only possible at all, if the victim answers recursive questions from the internet. When recursion is enabled for queries from outside the local network, an external attacker can

trigger a DNS request from the attacked server to an arbitrary name server under his control. This provides the attacker with valuable information about the name server status and gives him the possibility to send a reply with chosen data. If recursion is not allowed the name server has to be triggered by the attacker in less reliable other ways as e.g. URLs or images in emails or websites. Or the attacker has to attack the server blindly which significantly decreases the probability of success.

1.4 Split DNS

A split DNS configuration increases the security even more. Instead of filtering external recursive queries by configuration and/or firewall, two DNS Servers are used with no shared DNS cache. One for the address resolution of the internal hosts (recursive, port 53 only internally accessible) and another on replying to the queries from the internet, only giving out the necessary addresses (recursion disabled). The servers are completely independent, there is no shared DNS cache. This also offers the possibility to hide the internal DNS structure from the internet.

2 Motivation and Consequences

Monetary profit is the main motivation behind DNS Spoofing attacks, as computer crime is getting more and more professional, discussed e.g. by Leyden [6] and Murtagh [7]. An attacker could gain money directly through pay-per-click programs or by fishing passwords, as well as by generating long-term revenue by installing ad/spyware. Also the use of the victim's computer in botnets [8] for DDOS attacks and sending spam is a possible source of profit for the attacker. Besides this "Web Spoofing" attacks DNS Spoofing affects all applications which rely on DNS resolution and could help an attacker for intrusion into a network/server.

2.1 Pay-per-Click

As online advertisement is often payed for views and clicks, an attacker gains profit by redirecting popular websites to an website with lots of advertisement and redirection schemes using JavaScript, e.g. as discovered by LURHQ [9].

2.2 Maleware Deployment

Trojans and spyware are often deployed by exploits in the user's browser software or tricking the user to accept the installation of browser plugins. With the redirection of popular trusted websites (e.g. from a big company) the attacker could gain a lot of traffic for a malicious website. He could even mislead a careful user to accept the installation of malicious software. Liston [10] shows an example for contamination with spyware due to exploits in the Internet Explorer. SANS [11] encountered such an attack in combination with DNS Spoofing in Spring 2005.

2.3 Phishing

Another threat by DNS Spoofing attacks is the utilization in Phishing attacks, sometimes also referred to as Pharming. With the help of DNS Spoofing large-scale Phishing attacks would be possible, either by attacking the client's resolver itself or poisoning the DNS Server of an ISP. Instead of tricking the users in emails to give out their passwords e.g. for online-banking and ebay, the attacker redirects the users to its servers, which imitate the original websites. For a normal user it's hard to tell if he is connected to the original site, as the right address is shown in the browser and he relies on the correct name resolution. Even in case of additional security mechanisms, such as SSL, many users tend to discard warning messages or do not check for a secured connection at all. Albeit there is no known case of pharming yet, it seems likely that future DNS cache poisoning exploits will be used in that way. Additionally an incorrect DNS resolution has impact on protocols besides http that rely on the name resolution, such as ftp, pop3, imap and smtp. Due to DNS Spoofing an attacker can redirect and record the login attempts, and gather the data of the user's accounts. Also the sniffing of outgoing mails by redirection over an prepared server is possible.

2.4 Authentication by Hostname

Tools like rlogin and rsh offer authentification schemes for remote access using the client hostname. The resolution of IP Adresses to hostnames relies on the correct information in the DNS System. Injecting false information in the DNS allows the attacker to exploit this trust relationship by masquerading as a trusted host. This attack and possible countermeasures are described in [12]. Due to several security risks and lacking cryptographic protection, the use of the rtools is generally superseded by ssh.

3 DNS Spoofing Attacks without IP Spoofing

In this Section spoofing attacks will be discussed, which need no IP spoofing and are possible because the lacking authentication and ambiguity of protocol description.

3.1 General: Manipulated Name Servers, Order of Resolution

The operator of a name server is able to configure its name server with false authoritative zones. With a false entry for authoritative data, a user's resolution request will be answered with the servers local data, instead of starting a resolution request and replying with the authoritative data of the legitimate DNS server. Without the use of cryptographic methods the user cold not determine the integrity and authenticity of the received name resolution. The technique of manipulating authoritative entries and more advances techniques for filtering DNS queries are sometimes also applied to restrict the internet access for users

in companies and by governments, e.g. in China [13] and Düsseldorf [14]. For slightly advanced users these blockades are easily circumventable by the use of alternative name servers and proxies. Another possible security risk in some environments is also the resolution order of the requests, such that a request for *www.bank.de* to the local name server in zone domain.com will first determine, if *www.bank.de.domain.com* exist and returns this address, instead of looking up *www.bank.de* externally.

3.2 Glue Records

The DNS Protocol allows to send additional records in a reply, intended to ease the resolution for standard queries for the authorative name server (NS-Type). NS queries are typically answered by names, not addresses. In the resolution process for a domain *www.domain.com* with authoritative name server *ns1.domain.com* its not possible to get the IP Address of *ns1.domain.com* without a additional glue record in the NS answer of the root server. The definition of the additional records is up to the owner of the DNS Server, so any attacker with an authoritative name server could spread forged records.

The following scenario shows two possibilities to poison the DNS cache on vulnerable servers. These are old BIND Versions before 1997 as well as Microsoft Servers up to Windows 2000 SP3 in the standard configuration, as they don't check for credibility of additional Records. WinNT SP4 to Windows 2000 SP2 could already filter out non-secure additional records, but this was not enabled by default (see Microsoft Knowledge Base [15]). There are two ways for the attacker to poison a DNS cache by using glue records, direct or indirect.

In the following examples the attacker tries to redirect *www.bank.com* to his server *6.6.6.0* with an authoritative name server *ns1.hacker.com* for the zone hacker.com. To poison the cache of the victim, he has first to trigger a question from the victims DNS resolver to *ns1.hacker.com* e.g. for www.hacker.com. In the direct case *ns1.hacker.com* sends the answer shown in the left column of Table 1. If this answer is taken in the victims DNS cache, all following questions within the TTL for *www.bank.com* are resolved to *6.6.6.0*, caused by the the 2nd and 4th line. A variant of this attack is not to spoof a single address, but the responsible name server of a address. The attacker has to delegate www.hacker.com to the authoritative name servers for bank.com, but provides for those the IPs of name servers controlled by himself. For the strategy the answer would be the one shown in the right column of Table 1. If this answer is taken in the DNS cache, all resolution requests for the bank.com domain will be directed to the servers under the attackers control and can be answered with arbitrary DNS information.

To eliminate these attacks, the DNS server has to check the credibility of an additional record before taking them into cache. It must discard all information about bank.com from other sources than bank.com, .com and the root servers. Instead of using the additional records, the resolver starts a subquery itself for the addresses of authoritative name servers situated outside the initially asked

Direct poisoning	Indirect poisoning
www.hacker.com NS ns1.hacker.com	www.hacker.com NS ns1.bank.com
www.hacker.com NS www.bank.com	www.hacker.com NS ns2.bank.com
ns1.hacker.com A 6.6.6.6	ns1.bank.com A 6.6.6.7
www.bank.com A 6.6.6.0	ns2.bank.com A 6.6.6.7

Table 1. Cache Poisoning

zone (general description in MS Knowledge Base [16]). BIND adopted such a rule much earlier in 1997, but the weakness was already described in 1993 by Schuba [12].

An analysis of a related new security risk was published in April 2005 by SANS/ISC [17], after discovering ongoing exploits in March 2005. As reason for the security hole turned out, firstly, that a Microsoft DNS Server relies always on the glue-filtering of a forwarding server. In settings with a forwarding server it applies no filtering itself, but accepts additional glue records regardless its settings for discarding them. Secondly, BIND 4 and 8, if used as forwarding server, as well as Windows up to WinNT4 SP3 (respective Pre-W2K SP3 in standard configuration) execute no filtering. Therefore setups with a Microsoft DNS Server using a non-filtering forwarding server were susceptible to cache poisoning. Mostly small to mid-size companies were affected, as this setup is a typical configuration when using an internal DNS/Active Directory-Server which is not processing DNS requests for external addresses itself, but forwards it to the ISP's DNS-Server. A cache poisoning weakness of Symantec products, discovered at the same time by SANS [17], seems to be caused by the same scheme, but no exact information is available from the manufacturer.

4 DNS Spoofing Attacks with IP Spoofing

The attacks in this Section all base on sending spoofed replies before the legitimate reply reaches the asking computer. If the legitimate reply arrives after the faked reply, it is discarded. The requirements a packet has to meet to be accepted are discussed in [5]. For an external attacker it is hard to guess the correct TID and destination port in time. A possibility to gain more time for sending packets with different TIDs to different ports to the victim is a simultaneous DoS attack on the legitimate DNS server. If the attack is unsuccessful, the attacker has to wait the time defined by the TTL (typically 8–24h), because the legitimate answer is stored in the DNS cache of the victim. Also popular names are very likely to be answered from the cache, making an attack impossible, as the victim DNS server does not start a resolution process.

4.1 Sequential IDs

The TID is the main security mechanism in the DNS protocol and should be randomized to complicate attacks. All common up-to-date DNS servers randomize the TID, but old unpatched BIND Versions up to 4.9.6 respective 8.1.1 (both

fixed in 1997) and some Windows NT4 versions did just sequentially increment the TID. This made an attack easy, especially on DNS servers with recursion enabled. The attacker just needed to trigger a request to a name server under his control, to determine the current TID. For a successful cache poisoning he could send in the attack just few spoofed packets just with the following TIDs. The resolver in WinXP SP1 had the same flaw, making a blind attack directly against the clients possible, as after startup there are only few choices for the port and the TID starts with 1. With a **random TID** the chances to send the right TID are only **1/65535** ($1/2^{16}$) per packet as the TID has a range of 16bit. For a 50% chance on a successfull attack the attacker needs to send on time more than 3MB of data(at a packetsize of 100 Bytes) to the victim, before the legitimate answer arrives. If the server is using also random sourceports this chance is lowered even more by 2^{16}-1024, as the attacker has also to guess the right port in the range of 1025 to 65535. Overall the chance with **random TID and port** is **1/4.227.858.432** ($< 1 / 0{,}98*2^{32}$). This increases the amount of data to be sent in time for a 50% chance to nearly 200GB.

4.2 BIND Birthday Attack

This attack affects DNS Servers with recursion turned on, running unpatched Versions of BIND4 $<$ 4.9.11 and BIND8 $<$ 8.2.7 respective $<$ 8.3.4. In 2002 it was discovered by Sacramento [18] and US-CERT [19], that these versions start k multiple resolution processes for k concurrent incoming resolution requests. This behavior increases the chance of a successful attack seriously. For a success chance of 50% only estimated $k = \frac{1+\sqrt{1+8t*log(2)}}{2}$ (Buchmann [20], chapter 4.3) concurrent requests and spoofed packets are needed. The probability $p = 1 - e^{\frac{-k(k-1)}{2n}}$ calculates with a k of 300 and the $n = 65535$ possible values for the TID to slighly more than 50%. Therefore the attacker needs to send only **15 KB of resolution requests** (at approx. 50 Bytes each) to the vitim for a **50% chance** in connection with **30KB of spoofed replies** right in time before the legitimate server answers. This data could be send even over a standard ADSL connection within 1–2 seconds. For **700 concurrent requests** the chance for a successful attack is **over 97.25 Percent**, compared to slightly over 1 percent for a normal attack server with random TID without port randomization. With port randomization the chance with 700 requests is at 0.006% with birthday attack and 0.000017% without. The different probabilities regarding the number of spoofed replies are shown in Figure **??**. Current BIND versions are not vulnerable to this attack anymore. BIND without external recursion was not vulnerable by this attack, as the attacker was not able to trigger the simultaneous resolution processes.

4.3 PRNG weakness

A new threat for the DNS protocol (and many other internet protocols) is the weakness of many current Pseudo Random Number Generators (PRNG) against

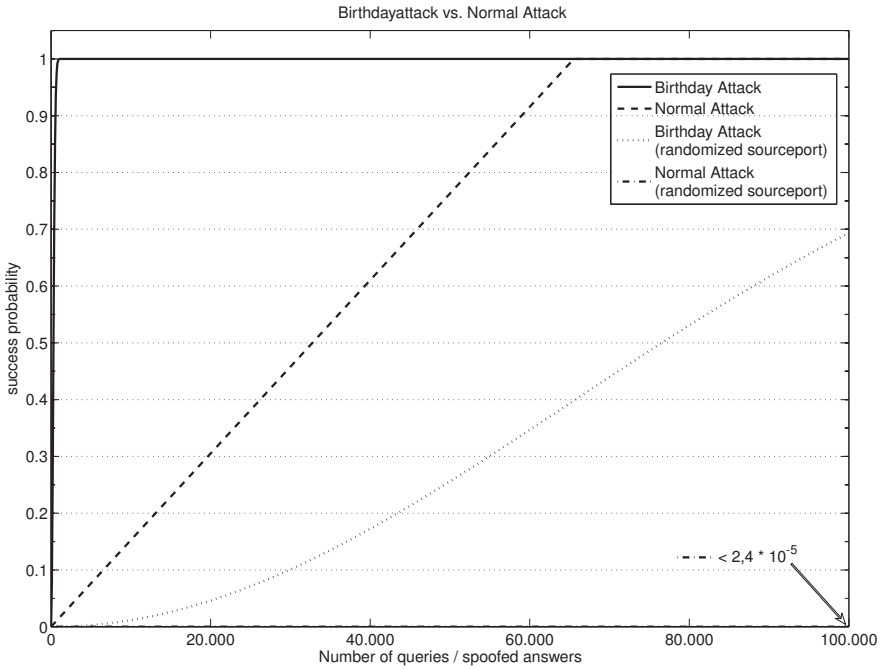

Fig. 3. Birthday attack vs. normal attack

Phase Space analysis. Michal Zalewski [21] described this weakness first in 2001 for the random TCP sequence numbers of different operating systems, but further investigation by himself [22] and Steward [23] showed that the random transaction number generation of most DNS servers is also vulnerable. Especially the PNRG employed in BIND8 is predictable. With analysis of 100000 TIDs and knowledge of the previous TIDs is the following TID foreseeable with 100% certainity. This would make a spoofing with just one packet possible. BIND 4, using the same code base as BIND 8, shares this weakness. BIND 9 does not come with an own PRNG, but uses the one of the operating system. On Linux 2.4.19 the chance of a successful attack with 5000 packets on a BIND 9 (no port randomizing) is at 20%, compared to 7,6% with a normal spoofing attack. DJBDNS has a slighly weaker PRNG with 30% success chance at 5000 packets to guess the TID, but is still more secure due to port randomization by default. The Microsoft DNS Server is not discussed in detail by Zalewski, but at least the NT4 pre-SP6 PRNG seems quite weak. Also some of the resolver libraries, MS as well as Linux(glibc2.1.9) employ a weak PRNG.

5 Counter Measures – Securing a DNS Server

This Section is mainly summarizing the possibilities for securely operating a DNS-Server e.g. in company network. It can only give a short overview, detailed information is available e.g. in Albitz and Liu [2], Allen [3] and Bauer [24].

For an end user, which is using a foreign DNS server, it's hard to improve his security. A careful use of protocols which rely on DNS information and, wherever possible, the use of secured and authenticated connections (e.g. HTTPS, Secure POP/SMTP, SSH) are recommendable. The cryptographic security extensions for the DNS protocol are still under development, and are not likely to be ready for use in near future.

5.1 Software, Versions and Security Patches

A DNS Server is a crucial point in the network infrastructure. It should be secured by an adequate firewall setup. Security fixes should be applied without delay. In general the use of alternative name server software such as DJBDNS should be evaluated, as BIND does not support the use of random source ports and has shown to be quite bug-prone in the past, as argued by the DJBDNS developer Bernstein [25]. DJBDNS is written with special focus on security and works as split DNS by design. DJBDNS is only available for Unix/Linux. If using BIND, the use of the latest BIND 9.3.2 is recommendable, instead of older BIND4 (officially deprecated) and BIND8. BIND9 is completely rewritten and contains substantial changes and new configuration possibilities against the DNS Spoofing attack vectors. A detailed, neutral comparison of BIND and DJBDNS can be found in Bauer [24]. For servers running Windows the use of a different DNS Server than MicrosoftDNS is complicated, because of its connection to the Active Directory Services.

5.2 No External Recursion

Most of the attacks on a DNS server are harder, if not impossible (e.g. birthday attack), if the name server does not allow recursive requests from the internet. In many cases recursion cannot be turned off completely, because it needs to serve legitimate resolvers in the local network or is used as a forwarder. In this case the accepted queries, and allowed hosts and network interfaces should be restricted, as supposed e.g. by Liu [26]. For glue attacks this complicates the triggering of resolution requests to a name server under the attackers control.

The spoofed packet with the right TID and port needs to reach the victim before the legitimate answer. When external recursion is not allowed the attacker does not know about the right moment and used port for sending spoofed packets to the victim. This is a big obstacle for a successful DNS spoofing.

The DNS server survey by Moore [27] in 2004 nevertheless shows that more than 80% of the DNS servers using BIND have external recursion turned on.

5.3 Split DNS

A Split DNS setup works with at least two seperate DNS servers. One DNS server acts as authoritative server for its zone for external hosts. It is not used as resolver or forwarder. Thus, recursion can be turned off completely. A second DNS

server acts as the resolver for the internal hosts, accepting recursive questions from the internal network. The internal resolving server can be protected by a firewall. The external server can be configured with recursion completely turned off. Thus, it only answers queries about its zone and never starts a resolution by itself. Furthermore a split DNS setup gives easy control over the externally available information regarding the internal network. DJBDNS is build as split DNS by design, *tinydns* for external requests and *dnscache* for resolution of internal recursive requests. The components can also be used separately, allowing the combination of other name server software with the secure resolver of DJBDNS.

5.4 Port Randomizing

As seen in Section 4, the use of a random source port for sending DNS requests decreases the probability of a successful attack by nearly 2^{16}. Unfortunately only few name servers are supporting this concept. As no version of BIND and MicrosoftDNS does support this concept, use of an alternative DNS servers is necessary for utilizing port randomization. Examples are dnscache, the resolver component of DJBDNS, and MaraDNS [28].

5.5 Concurrent Resolution Requests to Different Servers

The comparison of concurrent resolution requests is a technique which allows the detection of poisioned DNS caches. Albeit not utilized in known DNS server software, for end users a tool for Windows is available by NGsec [29]. It offers counterchecking of name resolutions. The disadvantage of such a detection scheme is the increased DNS traffic and the possibility of false positives by DNS load balancing. Some other tools offer only limited security for specific sites, e.g. ebay toolbar. They verify against a build in hostlist and/or check for HTTPS connections and valid certificates, as described by Bachfeld [30].

5.6 DNSSEC

Security extensions for DNS, which use cryptographic authentication have been discussed for several years. The first RFC for DNSSEC (RFC2535, [31]) was released in 1999. The first complete implementation was included in BIND 9.3. Due to its complexity, flaws in the key handling, high demands to the hardware and ongoing refinements DNSSEC is not deployed widely until today. RFC2535 was replaced by the RFCs 4033 [32], 4034 [33], 4035 [34] in March 2005. In October 2005 the Swedish ".SE" zone is the first TLD which employs DNSSEC. Detailed information on DNSSEC can be found on dnssec.net [35]. D.J. Bernstein [36], the author of DJBDNS, is harshly criticizing the concepts of DNSSEC.

6 Other threats to the DNS System

6.1 Homoglyphe

Internationalized Domain Names (IDN) enable the use of special, and non-latin characters in domain names, which are partially very similar to latin characters.

This offers a new possibility for cheating a user with a browser which supports the display of the IDN. For example paypal.com is hardly to distinguish from the domain paypal.com, written with a cyrillic "a", in "punycode" encoding xn-pypal-4ve.com. It is even possible to acquire a valid SSL certificate for such a domain, as shown in the advisory by Johanson [37]. In reaction to these spoofing possibilities, all major browsers have disabled the display of IDNs and show the raw punycode of the URLs instead.

6.2 Man in the Middle

An attacker within the local network who is able to intercept or sniff the network traffic of the victim, is naturally able to modify DNS packets or send DNS answers faster than the legitimate server. This attack on the DNS traffic is only one of many other (and more effective) possibilities for a local attacker. The obstacles of such a local attack are mainly the redirection of network traffic in switched networks with techniques like ARP poisoning, not the spoofing of DNS traffic. Ready to use tools like dnsspoof from the dsniff suite [38] are available. The use of tools which sniff the victims network traffic can be of greater importance in networks which use shared media, like public WLAN access points.

6.3 Server Intrusion

Another possible source of DNS cache poisoning are attacks on the DNS Server software itself or the underlaying operating system. For example many versions of BIND are susceptible to buffer overflows, which possibly allow the execution of arbitrary code (see summary of vulnerabilities from ISC [39]). The Windows operating systems as well as Linux/Unix have shown severe vulnerabilities in the past, which allowed to gain full control over the server. On such a compromised server the attacker could affect the DNS resolution and zone data at will, harming internal and external users of the DNS server.

7 Affected Versions and Distribution in the Internet

7.1 BIND

BIND comes in 3 versions: Bind4 and Bind8, which are based on the same codebase, and the completely rewritten BIND9. The subversions like 9.2, 9.3 show added features and are always maintained concurrently for a certain time. The use of the latest version of Bind9, currently 9.3.2, is explicitly recommended by the manufacturer, as Bind4 is officially deprecated and development of Bind8 has stopped. Only security fixes are issued. In the past BIND has shown to be quite bug-prone (e.g. buffer overflow and denial of service attacks). This can be seen on the manufacturer's (ISC) webpage on [39]. Besides the security flaws which allow direct intrusion, two major fixes are to be highlighted in the context of the previously considered attacks. In 1997 with the versions *4.9.6*

and *8.1.1* the use of randomized TIDs was introduced to BIND. The behavior of starting parallel resolution processes for concurrent queries for the same address, exploitable for a *birthday attack*, was changed with BIND versions *4.9.11, 8.2.7 and 8.3.4* in 2002. With the DNS poisoning attacks in Spring 2005 it has been discovered, that *BIND4 and BIND8 servers shouldn't be used used as forwarders* for other DNS servers, as they do not apply filtering for bogus glue records in this case. This filtering was originally introduced against glue-attacks in 1997. There will be no fix for this behavior. An update to BIND9 is suggested by ISC as the use as forwarder is not part of the official functionality. BIND9 is officially intended to be used as forwarder, and works correctly.

7.2 MicrosoftDNS

The Microsoft Windows platform has shown to be quite vulnerable in the past. Thus application of the current security fixes is important. The web based Microsoft Update Service provides a convenient possibility to do this. Several vulnerabilities in the past allowed the intrusion and complete takeover of the system and with it the compromising of a DNS Server running on the server. Vulnerabilities specific to DNS Spoofing are presented in the following:

Windows Systems before WinNT SP4 are vulnerable to glue attacks, as they don't filter bogus glue records. Between the release of WinNT SP4 and the release of W2K SP4 the filtering of glue records was available, but needed to be explicitly enabled by a registry key [15]. Since W2K SP4 the filtering is enabled by default. Another weakness is the resolver component, which increments the TID or even uses a constant value of 1 in Windows versions before W2K SP4 Server and in XP Workstations before SP2, as shown by Larcher [40]. A current security exploit, detected by SANS [17] due to an ongoing exploit in Spring 2005, appears in combination of a MicrosoftDNS server with a BIND4 or BIND8 used as forwarder. As mentioned above, these BIND versions do not filter the replies while the Microsoft DNS relies on the filtering of the forwarder. Thus, the old glue attack, which was thought to be fixed back in 1997, became possible again. As there is no security patch yet for Windows systems, the use of forwarding has to be checked carefully. The number of possibly affected installations is unclear.

7.3 DJBDNS

DJBDNS has shown no security flaws since was released in 2000. It has not been changed since 2001. It was programmed with the aim of a lean, modular and secure DNS server. Furthermore, DJBDNS is a split DNS by design, uses port randomizing, has easier configuration files and shows better efficiency, as emphasized by Bauer [24]. The major disadvantage of DJBDNS is the lacking support of DNSSEC and resolution for IP6, because the author, D.J. Bernstein [36], strongly dislikes the current concepts for integration in the DNS .

7.4 Distribution in the Internet

The exact distribution of the used DNS server software is hard to determine, as many servers do not provide exact software / version information. Big ISPs / webhosters also develop own patches for their BIND versions. The market share can be calculated by served domains or by installations, which has an important impact on the result.

Domain server surveys are available from Bernstein [41] (2002), Moore [27] (2004), the ISC [42] (2005) and The Measurement Factory [43] (2005). In general it can be seen, that BIND, which is freely available from ISC [44] for many platforms, is the most used name server with over 70% of the installations. While the ratio of BIND versions in 2002 was 3% (v4) to 64% (v8) to 33% (v9) [41] this has changed to $< 1\%$ (v4) 25% (v8) to 73% (v9) in 2005 [43]. Exact data on subversions, to determine the number of vulnerable versions is not available. Nevertheless the surveys show, that 80% of the BIND servers allow external recursion, which should be unnecessary in the most cases but eases the different attacks.

The second biggest market share depends on the mode of counting. Taking the number of serviced domains into account, DJBDNS(TinyDNS) (free available from cryp.to [36], only for Unix) is second with 15% while MicrosoftDNS (coming with MS Windows Server OS) is third with 6%. Following are name servers for special purposes like MyDNS [45] with nearly 3%, PowerDNS [46] with 2% and SimpleDNS [47] with 1.2%. Taking the number of installations into account, MicrosoftDNS is the second with 21% while DJBDNS reaches only 2.5% and all other software is below 1% (data from Moore [27], 2004). This shows that the MicrosoftDNS is mainly used to serve single domains, while installations of DJBDNS(TinyDNS) and other alternative DNS software are often used to serve several domains, e.g. for ISPs.

8 Summary

The most promising DNS spoofing attacks for an external attacker are glue attacks, the exploitation of sequential IDs and birthday attacks. These vulnerabilities are fixed since several years, with exception of the reappearing glue attacks in certain configurations with forwarding.

The security mechanisms of DNS against spoofed packets, namely TID and source port, are quite weak compared to cryptographic methods. To guess the TID only an average of 32000 packets is needed. Nevertheless, for an external attacker it is quite hard to send the right spoofed packet before the answer of the legitimate server arrives. Furthermore, on a server without external recursion the correct source port is difficult to determine. For a server with port randomizing guessing is needed. This lowers the attack feasibility by nearly 2^{16}. The exploitation of PRNG weaknesses requires an a-priory analysis of a huge number of server queries and seems to be a more theoretical weakness.

In summary on good configured and maintained DNS server, especially if offering no external recursion and using port randomization, is unlikely to ex-

perience cache poisoning by external attacks which were discussed in this work. Nevertheless, a cryptographic authentication in the protocol is desirable, to make these external attacks nearly impossible. A cryptographic protection of the DNS would additionally address possible internal attacks, such as man-in-the-middle attacks.

References

1. Wikipedia: Domain name system (2006) http://en.wikipedia.org/wiki/Dns.
2. Albitz, P., Liu, C.: DNS and BIND. 4th edition edn. O'Reiley (2001) ISBN: 0-596-00158-4.
3. Allen, R., Larson, M., Liu, C.: DNS on Windows Server 2003. O'Reilly (2003) ISBN: 0-596-00562-8.
4. Salamon, A.: (Dns related rfcs) http://www.dns.net/dnsrd/rfc/.
5. phrack.org: The impact of rfc guidelines on dns spoofing attacks (2004) http://www.phrack.org/phrack/62/p62-0x03_Linenoise.txt.
6. Leyden, J.: Cybercrime 'more lucrative' than drugs (2005) http://www.channelregister.co.uk/2005/11/29/cybercrime/.
7. Murtagh, M.: The new hackers on the block (2005) http://www.biosmagazine.co.uk/op.php?id=314.
8. heise news: Botnetze an spammer vermietet (2005) http://www.heise.de/newsticker/meldung/65747.
9. LURHQ: Pay-per-click hijacking (2005) http://www.lurhq.com/ppc-hijack.html.
10. Liston, T.: Schädlingen auf der spur (follow the bouncing malware) (2004) http://www.heise.de/security/artikel/49687.
11. Center, S.I.S.: March 2005 dns poisoning summary (2005) http://isc.sans.org/presentations/dnspoisoning.php.
12. Schuba, C.: Addressing weaknesses in the domain name system protocol. Technical report, Department of Computer Sciences, Purdue University (1993) http://citeseer.ist.psu.edu/509869.html.
13. Inc., D.I.T.: A report about national dns spoofing in china on sept. 28th, 2002 (2002) http://www.dit-inc.us/hj-09-02.html.
14. Freude, A., Schäfers, J.O.: Funktionieren internet-filter/sperren? (2003) http://odem.org/informationsfreiheit/o-ton--filter.html.
15. Microsoft: How to prevent dns cache pollution (kb241352) (2005) http://support.microsoft.com/kb/241352/.
16. Microsoft: Description of the dns server secure cache against pollution setting (kb316786) (2005) http://support.microsoft.com/kb/316786/.
17. Center, S.I.S.: Dns cache poisoning update 7.4.2005 (2005) http://isc.sans.org/diary.php?storyid=502.
18. Sacramento, V.: Vulnerability in the sending requests control of bind versions 4 and 8 allows dns spoofing [cais, 19.11.2002] (2002) http://www.rnp.br/cais/alertas/2002/cais-ALR-19112002a.html.
19. US-CERT: Various dns service implementations generate multiple simultaneous queries for the same resource record (2002) http://www.kb.cert.org/vuls/id/457875.
20. Buchmann, J.: Einführung in die Kryptographie. Springer (2001)

21. Zalewski, M.: Strange attractors and tcp/ip sequence number analysis (2001) http://www.bindview.com/Services/Razor/Papers/2001/tcpseq.cfm.
22. Zalewski, M.: Strange attractors and tcp/ip sequence number analysis - one year later (2002) http://lcamtuf.coredump.cx/newtcp/.
23. Stewart, J.: Dns cache poisoning – the next generation (2002) http://www.lurhq.com/cachepoisoning.html.
24. Bauer, M.D.: Linux Server Security. O'Reilly (2005) ISBN: 0-596-00670-5.
25. Bernstein, D.: Bind, the buggy internet name daemon (2002) http://cr.yp.to/djbdns/blurb/unbind.html.
26. Liu, C.: Securing an internet name server (2001) http://www.linuxsecurity.com/resource_files/server_security/securing_an_internet_name_server.pdf.
27. Moore, D.: Dns server survey (2004) http://mydns.bboy.net/survey/.
28. MaraDNS: (Maradns – a security-aware dns server) http://www.maradns.org/.
29. NGSEC: (Ngsec antipharming tool) http://www.ngsec.com/ngproducts/antipharming/?lang=en.
30. Bachfeld, D.: Fangverbot. C't (22) (2005) 154–162
31. Eastlake, D.: Domain name system security extensions. IETF RFC 2535 (1999) http://www.ietf.org/rfc/rfc2535.txt.
32. Arends, R., Austein, R., Larson, M., Massey, D., Rose, S.: Dns security introduction and requirements. IETF RFC 4033 (2005) http://www.ietf.org/rfc/rfc4033.txt.
33. Arends, R., Austein, R., Larson, M., Massey, D., Rose, S.: Resource records for the dns security extensions. IETF RFC 4034 (2005) http://www.ietf.org/rfc/rfc4034.txt.
34. Arends, R., Austein, R., Larson, M., Massey, D., Rose, S.: Protocol modifications for the dns security extensions. IETF RFC 4035 (2005) http://www.ietf.org/rfc/rfc4035.txt.
35. DNSSEC.NET: (Dnssec – dns security extensions) http://www.dnssec.net/.
36. Bernstein, D.J.: (djbdns: Domain name system tools) http://cr.yp.to/djbdns.html.
37. Johanson, E.: The state of homograph attacks (2005) http://www.shmoo.com/idn/homograph.txt.
38. Song, D.: (dsniff) http://www.monkey.org/~dugsong/dsniff/.
39. ISC: (Bind vulnerabilities) http://www.isc.org/index.pl?/sw/bind/bind-security.php.
40. Larcher, R.: Predictability of windows dns resolver (2004) http://www.infosecwriters.com/text_resources/pdf/predictability_of_Windows_DNS_resolver.pdf.
41. Bernstein, D.: Dns software (2002) http://cr.yp.to/surveys/dns1.html.
42. ISC.ORG: (Distribution by domain server software jan 2005) http://www.isc.org/index.pl?/ops/ds/reports/2005-01/dist-servsoft.php.
43. Factory, T.M.: (Dns survey april 2005) http://dns.measurement-factory.com/surveys/200504-full-version-table.html.
44. Consortium, I.S.: (Isc – internet systems consortium homepage) http://www.isc.org/.
45. Moore, D.: (Mydns: Home) http://mydns.bboy.net/.
46. PowerDNS: (Powerdns) http://www.powerdns.com/.
47. Software, J.: (Simple dns plus - windows dns server) http://www.simpledns.com/.