DNS-Spoofing und Sicherheit

Seminar Rechner- und Netzwerksicherheit WS 06/07

Matthäus Wander
<matthaeus.wander@stud.uni-duisburg-essen.de>

10. Februar 2007

Zusammenfassung

Das Domain Name System (DNS) ist einer der am häufigsten verwendeten Dienste im Internet. Die Bedeutsamkeit ergibt sich dadurch, dass ein großer Teil der Internetdienste einen funktionierenden DNS-Betrieb voraussetzt. Dieses Dokument entstand im Rahmen des von Dr. Werner Otten betreuten Seminars „Rechner- und Netzwerksicherheit“ an der Universität Duisburg-Essen im WS 06/07. Es beschäftigt sich mit der DNS-spezifischen Sicherheit, wobei allgemeine Betrachtungen wie die Notwendigkeit von Redundanz und fehlerfreier Software weitestgehend ausgeklammert sind. Zunächst erläutert Kapitel 1 in knapper Form die wichtigsten DNS-Grundlagen. Kapitel 2 stellt das Cache Poisoning und Spoofing vor. Kapitel 3 befasst sich mit einigen Sicherheitsmechanismen, um die bekannten Standard-Angriffsszenarien einzudämmen oder abzuwehren. Kapitel 4 erweitert den Sicherheitshorizont und zeigt Gefahren auf, die durch das DNS entstehen können. Kapitel 5 fasst die wichtigsten Aussagen zusammen und zieht ein Fazit. Um im Rahmen zu bleiben, verzichte ich auf eine Betrachtung der Root-Server-Verlässlichkeit, ungeachtet der Aktualität des Themas durch den jüngsten DDoS-Angriff am 6. Februar 2007.

Inhaltsverzeichnis

1 Domain Name System

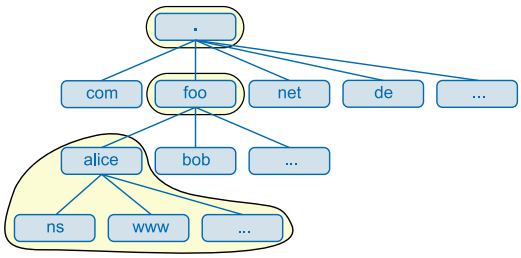
1.1 Grundlagen

Das *Domain Name System* (DNS) ist eine verteilte Datenbank und einer der wichtigsten Dienste im Internet. Es wurde 1983 von Paul Mockapetris entworfen, die heute gültige Spezifikation in den RFCs 1034 und 1035 ist aus dem Jahre 1987. Hinzu kommen bis zum heutigen Tage einige Erweiterungen, die in verschiedenen weiteren RFCs festgelegt sind.

Die Datensätze heißen *Resource Records* (RR). Die Datentypen der RRs heißen *Record Types* – ihre Menge ist ebenfalls spezifiziert, Verbunddatentypen sind nicht vorgesehen. Der wohl wichtigste Record Type ist der A RR (*Address RR*), mit dem ein Hostname zu einer 32 Bit großen IP-Adresse aufgelöst wird. Dies entspricht dem ursprünglichen Zweck der Namensauflösung, womit die bis dahin übliche *hosts*-Datei abgelöst wurde. Zusätzlich zum Record Type

hat jeder Resource Record eine Klassenangabe, die jedoch nahezu ausschließlich *Internet* (IN) lautet.

Aus Performance-Gründen spielt das Caching von abgefragten RRs eine wichtige Rolle. Für jeden RR ist eine *Time-to-Live* (TTL) definiert. Dies ist die maximale Zeit, für die ein RR gecached werden darf. Da sich die meisten RRs relativ selten ändern, wird häufig eine TTL von 86400s (24 Stunden) gewählt. Wurde ein nicht existierender Resource Record abgefragt, lautet die Antwort *Name Error*, auch als NXDOMAIN (*Non-existent Domain*) bezeichnet. NXDOMAIN-Antworten können ebenfalls gecached werden, dies bezeichnet man als *Negative Caching*.



Während die meisten Internetdienste überwiegend dezentral strukturiert sind, ist das DNS in einer hierarchischen Struktur angeordnet. Ein Knoten im Namensraum bildet eine *Domain*. RRs einer Domain werden in *Zonen* organisiert. Für jede Zone ist mindestens ein *Nameserver* (NS) als *Autorität* ausgezeichnet. Zwecks Lastverteilung und Ausfallsicherheit sind meistens zwei oder mehr redundante NS für die selbe Zone autoritativ. Eine Zone muss dabei nicht zwangsläufig die komplette Domain umfassen. Subdomains können an untergeordnete NS *delegiert* werden, wodurch die Hierarchie entsteht. Die delegierte Subdomain bildet eine weitere Zone mit einer dafür verantwortlichen Autorität.

Beispiel: Hostname → IP-Adresse

```
www.uni-due.de.    86400 IN CNAME sp607.power.uni-essen.de.
sp607.power.uni-essen.de.    86400 IN A 132.252.184.87
```

1.2 Transport

Als Transportprotokoll werden vom DNS sowohl UDP als auch TCP genutzt, für den Server ist die Portnummer 53 reserviert. Bei einem UDP-Datagramm sind die Nutzdaten gemäß DNS-Spezifikation bis 512 Bytes beschränkt, da nur bis zu dieser Länge sichergestellt ist, dass keine IP-Fragmentierung notwendig ist. IP-Fragmentierung bzw. Zusammensetzung benötigt zusätzliche Ressourcen und ist nicht auf jedem System aktiv. Da TCP einen Datenstrom auch ohne IP-Fragmentierung über mehrere Pakete hinweg senden kann, sind DNS-Transaktionen über TCP in ihrer Länge unbeschränkt.

Eine DNS-Antwort über UDP wird ggf. vom Server mit dem Flag TC (*Truncation*) als abgeschnitten markiert, es liegt dann beim Client die Transaktion über TCP erneut durchzuführen. Da die meisten DNS-Transaktionen nur geringe Datenmengen beinhalten, findet in der Praxis nahezu ausschließlich UDP Verwendung, da TCP durch den Handshake beim Verbindungsaufbau und -abbau einen Overhead darstellen würde. Eine Ausnahme bilden Zonentransfers, die stets über TCP abgewickelt werden.

In einigen Netzen erlaubt die Firewall lediglich DNS über UDP, was in einigen Randfällen zu Problemen bei der Namensauflösung führt.

1.3 Reverse Lookup

Neben dem Forward Lookup, bei dem ein Hostname zu einer Adresse aufgelöst wird, gibt es mit dem *Reverse Lookup* die Möglichkeit eine Adresse zu einem Host aufzulösen. Anwendung findet dies in der meisten Server-Software bei einer eingehenden Verbindung. Der Reverse Lookup ist jedoch nicht, wie man intuitiv vermuten könnte, die Abfrage des selben Resource Records wie beim Forward Lookup. Bei einem Reverse Lookup wird ein PTR RR unterhalb der Domain `in-addr.arpa.`[1] abgefragt. Der PTR RR kann auf einen beliebigen syntaktisch korrekten Domainnamen zeigen, auch auf nicht existente Domains. Die Kontrolle darüber hat die Organisation, der das IP-Netz zugewiesen wurde.

Die Domainzuweisung eines PTR RRs kann erst dann als korrekt angenommen werden, wenn die Gegenprüfung per Forward Lookup gelingt.

Beispiel: Reverse Lookup

```
5.1.1.10.in-addr.arpa.    86400 IN PTR    not-existent.foo.
```

1.4 Rekursive und iterative Anfragen

Anfragen an einen Nameserver können entweder rekursiv oder iterativ gestellt werden. Bei beiden Anfragen konsultiert der angefragte NS zunächst seine autoritativen Zonendaten und seinen Cache. Falls der Resource Record bei einer iterativen Anfrage nicht bekannt ist, antwortet der NS mit einem Verweis auf den nächstbesten Server – häufig die NS der Top-Level-Domain, falls im Cache vorhanden, ansonsten die Root-Server.

Ist der Resource Record bei einer rekursiven Anfrage nicht im Cache vorhanden, so kontaktiert der NS selbst den nächstbekannten Server und durchläuft die Hierarchie hinunter bis zum autoritativen NS. Der gefundene RR oder NXDOMAIN wird an den Anfragesteller als Antwort geschickt und gleichzeitig in den eigenen Cache übernommen.

1.5 Glue Records

Ein *Glue Record* ist ein A RR, der in einer nicht-autoritativen Zone vorhanden ist. Der Glue Record ist dann notwendig, wenn eine Subdomain an einen NS delegiert wird, dessen Domainname selbst Teil eben dieser Subdomain ist. Ohne Glue Record wäre die Adresse des autoritativen Subdomain-Nameservers nicht bestimmbar.

Beispiel: Delegation der Subdomain sub.alice.foo. mit Glue Record

```
sub.alice.foo.      86400 IN NS    ns.sub.alice.foo.
ns.sub.alice.foo.   86400 IN A     10.5.1.10
```

[1] Das Suffix `arpa.` ist ein Rudiment des früheren ARPANET. Es bekam auf Vorschlag des Internet Architecture Board (IAB) Mai 2000 die neue Bedeutung „Address and Routing Parameter Area“. [1]

2 DNS Cache Poisoning und Spoofing

2.1 Cache Poisoning

Durch die Glue Records eröffnet sich ein Angriffsszenario, bei dem eine Angreiferin *Mallory* gefälschte Resource Records in den Cache der Benutzerin *Alice* einspeisen kann. Dies bezeichnet man als *Cache Poisoning* oder *Cache Pollution*. Angenommen, Alice fragt bei Mallorys NS (Adresse 10.1.2.3) nach dem A RR des Domainnamens `www.pharming.mallory.foo` an. Mallory gibt eine Delegation an `www.deine-bank.foo` vor und übermittelt einen gefälschten Glue Record für diesen Domainnamen, der zu Mallorys eigener Hostadresse auflöst.

Beispiel: Gefälschter Glue Record

```
pharming.mallory.foo.   86400 IN NS   www.deine-bank.foo.
www.deine-bank.foo.     86400 IN A    10.1.2.3
```

Alice würde bei gutgläubiger Annahme dieser Antwort einen gefälschten A RR für den Host `www.deine-bank.foo` cachen – Alice' Cache wäre *vergiftet*. Hinzu kommt, dass Mallory die Time-to-Live des gefälschten RRs selbst festlegen kann. Spätestens nach Ablauf der TTL wäre der RR und damit der Nachweis der Fälschung aus Alice' Cache entfernt worden.

Bei einer anderen Variante des Cache Poisonings fügt Mallory in die Antwort gefälschte RRs ein, nach denen Alice überhaupt nicht angefragt hat.

Beispiel: Gefälschter A RR

```
www.pharming.mallory.foo.   86400 IN A   10.1.2.3
www.deine-bank.foo.         86400 IN A   10.1.2.3
```

Beispiel: Gefälschte Delegation der com-Domain

```
com.                    86400 IN NS   www.pharming.mallory.foo.
www.pharming.mallory.foo.   86400 IN A   10.1.2.3
```

All diese Varianten haben gemein, dass Mallory sog. Out-of-bailiwick Records (engl. *bailiwick* – Amtsbezirk) in ihre Antwort einfügt, da die RRs außerhalb der Autorität von Mallory liegen.

Chronologie

In BIND 4 und 8, der Referenzimplementation eines DNS-Servers, sind die beiden genannten Poisoning-Varianten seit 1997 behoben [2], indem BIND die Autorität eines RR prüft, ehe der RR in den Cache übernommen wird. Im selben Jahr nutzte E. Kashpureff Poisoning gegen ungepatchte BIND-Server, um aus Verdruss über die damalige Domain-Vergabepraxis die Domain `www.internic.net` auf die Website seines Unternehmens AlterNIC umzuleiten. Kashpureff einigte sich später außergerichtlich mit Network Solutions, dem betroffenen Teilhaber von InterNIC, und entschuldigte sich öffentlich für den Vorfall. Neben dem dadurch abgewandten zivilrechtlichen Prozess wurde er jedoch von US-Behörden strafrechtlich belangt und zu einer Freiheitsstrafe von zwei Jahren auf Bewährung sowie 100 US-Dollar Geldstrafe verurteilt.

November 1998 untersuchte Men & Mice [3] im Zuge der „Domain Health Survey“ 4 000 im Internet zugängliche NS und stellte fest, dass jeder dritte noch anfällig für Poisoning war.

Der Windows 2000 DNS-Server war zwar in der Lage vergiftete RRs auszusondern, jedoch war er standardmäßig bis zum Service Pack 3 im Jahre 2002 nicht dafür konfiguriert [4]. Nach Angaben von Microsoft geschah dies aus Performancegründen.

September 2003 wurde in BIND 8 ein Bug behoben, durch den Negative Cache Poisoning möglich war [5]. Ein Fälschen solcher Antworten von tatsächlich vorhandenen Domains gleicht effektiv einem Denial of Service.

Die mittlerweile überholten Versionen 4 und 8 des BIND-Servers sind bis heute eine Gefährdungsquelle gegenüber Cache Poisoning, falls sie als Forwarder konfiguriert sind. Zwar filtern sie gefälschte RRs vor der Übernahme in den Cache, leiten jedoch bei der ersten Anfrage die ungefilterte Antwort weiter. Der Fehler wird offenbar nicht mehr behoben, der Hersteller warnt vor dem Einsatz als Forwarder und empfiehlt stattdessen ein Update auf BIND 9 [6]. 2005 untersuchte D. Kaminsky die Forwarding-Struktur im Internet zugänglicher NS und stellte fest, dass 217 000 potentiell gefährdet und 13 000 definitiv für diese Schwäche anfällig sind, da sie BIND 4 oder 8 als Forwarder nutzen [7].

2.2 Spoofing

Unter Verwendung von UDP eröffnet sich beim DNS das klassische UDP-Spoofing-Szenario.[2] Angenommen, die Angreiferin Mallory ist in der Lage bei Alice eine DNS-Anfrage zu Bob auszulösen, ohne jedoch den Netzwerkverkehr zwischen Alice und Bob abhören zu können.[3] Mallory möchte nun Bobs Antwort spoofen, um gefälschte RRs in Alice' Cache unterzubringen.

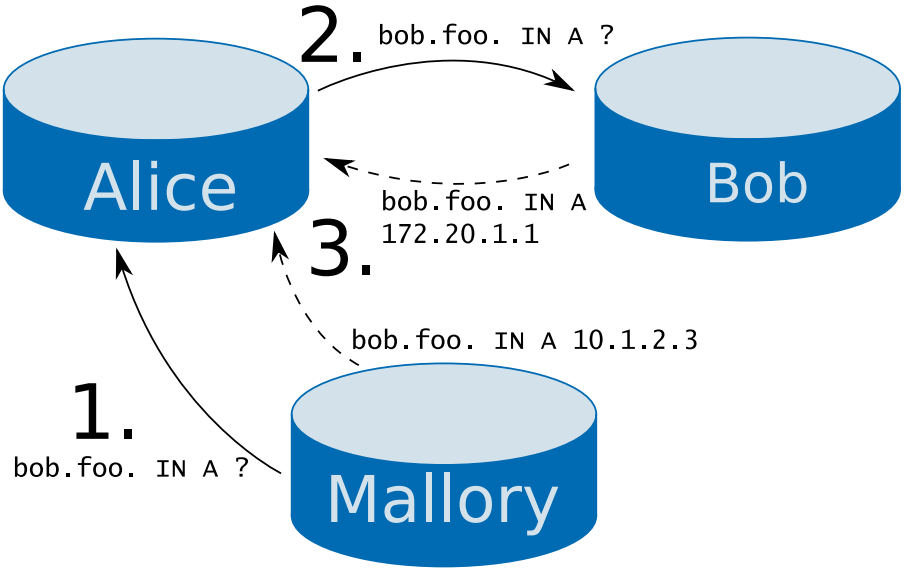


Abbildung 1: DNS-Spoofing

Um eine gefälschte DNS-Antwort zu konstruieren, ist folgendes Wissen über Alice' Anfrage notwendig: Quell-IP-Adresse (entspricht Alice' Adresse), Ziel-IP-Adresse (entspricht Bobs öf-

[2] vgl. Seminararbeiten über Spoofing von S. Tomanek und H. Bier
[3] vgl. Seminararbeit über Sniffing von S. Tarassenko

fentlicher NS-Adresse), Zielport (stets 53), Quellport (zunächst unbekannt) und Transaction ID (zunächst unbekannt).

Quellport

BIND 9 wählt den Quellport zufällig, verwendet denselben Port jedoch immer wieder. Mallory kann Alice' Port also bestimmen, indem sie eine Anfrage zu ihrem eigenen NS triggert. BIND bietet außerdem die Option den Quellport fest auf 53 zu setzen – zwar erleichtert dies eine restriktive Paketfilter-Konfiguration, gleichzeitig erleichtert es aber auch das Spoofing.

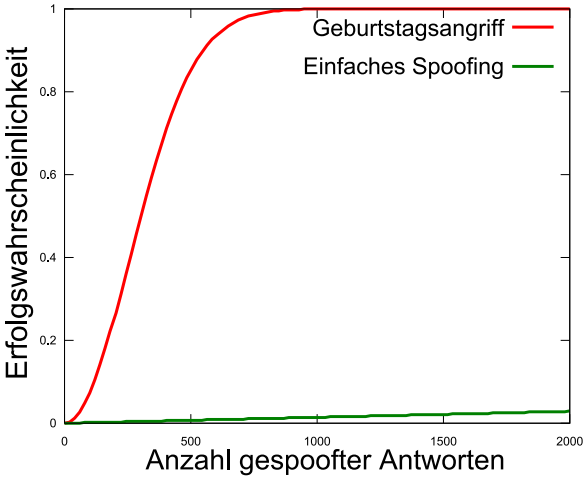
Die Nameserver-Software `djbdns` wählt für jede Anfrage einen neuen zufälligen Quellport.

Transaction ID

Die 16 Bit große *Transaction ID* (TID) dient dazu die Antworten den Anfragen zuordnen zu können, für den Fall dass zu einem Zeitpunkt mehrere Transaktionen gleichzeitig durchgeführt werden. Für diesen Zweck ist eine sequentielle ID-Vergabe ausreichend, was in frühen Implementierungen üblich war. Seit die Transaction ID als Spoofing-Schutz erkannt wurde, wird sie bei jeder Anfrage zufällig generiert.

J. Stewart untersuchte die TID-Erzeugungsalgorithmen von BIND 8, BIND 9 und `djbdns` auf Regelmäßigkeiten [8]. Zunächst erstellte er einen Datenbestand von jeweils 100 000 hintereinanderliegenden TIDs. Mittels der *Phase Space Analysis* auf dem TID-Datenbestand untersuchte er nun, wie gut eine TID prognostiziert werden kann, wenn drei zuvorgehende TIDs bekannt sind. Bei BIND 8 ergab sich eine Prognosegenauigkeit von 100 %. BIND 9 und `djbdns` nutzen `/dev/random` zur Zufallszahlenerzeugung, getestet wurde unter Linux 2.4.19. Bei ihnen ließ sich Stewart die 5 000 besten Prognosen für die nächste TID erzeugen. Im Durchschnitt betrug die Wahrscheinlichkeit, dass sich unter den 5 000 Prognosen die richtige TID befand, bei BIND 9 20 % und bei `djbdns` 30 %. Damit ist diese Angriffsmethode in der Theorie nachgewiesen, ein praktischer Machbarkeitsbeweis steht allerdings noch aus.

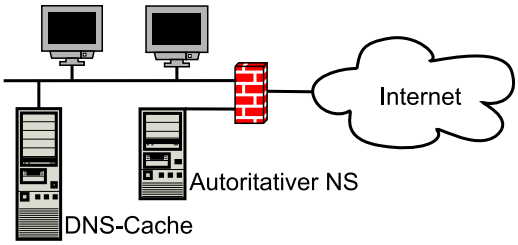
BIND 8 ist zudem bis heute für einen Geburtstagsangriff anfällig [9] [8]. Beim einfachen Spoofing sendet man eine Anfrage und n gespoofte Antworten. Beim Geburtstagsangriff sendet man n identische Anfragen und n gespoofte Antworten. Anstatt ein mal fragt BIND 8 daraufhin beim autoritativen NS n mal denselben Resource Record ab. Da jede Anfrage eine andere TID enthält, erhöht sich aufgrund des Geburtstagsparadoxons die Erfolgswahrscheinlichkeit des Spoofings enorm. Die Nameserver BIND 9 und `djbdns` sind nicht für den Geburtstagsangriff anfällig, da sie bereits gesendete Anfragen nicht erneut senden.



3 Sicherheitsmaßnahmen

3.1 Split DNS

Split DNS bezeichnet kein technisches Verfahren, sondern eine organisatorische Maßnahme, um zwei unterschiedliche Nameserver-Funktionen in unterschiedlichen Teilen des Netzwerks zu betreiben.



Autoritativer Nameserver

Der autoritative Nameserver liefert lediglich die eigenen DNS-Zonen aus. Er beantwortet alle Anfragen ausschließlich iterativ. Er baut keinen Cache auf und ist somit resistent gegenüber Cache Pollution. Damit die autoritativen DNS-Zonen uneingeschränkt aus dem Internet abrufbar sind, sollte er in der DMZ platziert werden.[1]

Rekursiver Cache

Der DNS-Cache hat durch die rekursive Arbeitsweise und den Cache einen etwas höheren Ressourcen-Anspruch. Er ist potentiell gefährdet gegenüber gefälschten Resource Records, sei es durch Cache Poisoning oder durch UDP-Spoofing. Die etwas höhere Komplexität der Rekursion bietet eine größere Angriffsfläche für Denial-of-Service-Angriffe. So sind zum Beispiel nur Server mit eingeschalteter Rekursion für den am 25. Januar 2007 veröffentlichten DoS-Bug in BIND 9 anfällig [10]. Da der Cache nur den lokalen Benutzern zugänglich sein muss, sollte er im internen LAN platziert werden. Er benötigt dennoch ausgehenden Internetzugang, um iterative Anfragen an autoritative Nameserver im Internet zu stellen. Er kann idealerweise von einer Firewall geschützt werden, die Port Address Translation und Stateful Inspection von UDP-Paketen unterstützt.[1]

Einen im Internet öffentlich zugänglichen DNS-Cache mit eingeschalteter Rekursion bezeichnet man als *Open Resolver*.

Konfigurationsbeispiel BIND 9

BIND 9 ist bislang in der Standardkonfiguration als Open Resolver eingestellt. Dies soll sich erst mit Version 9.4 ändern, die gerade als Release Candidate kurz vor der Veröffentlichung steht.

In folgendem Konfigurationsbeispiel wird beim autoritativen Nameserver der generelle Zugriff gesperrt und der Zugriff bei jeder einzelnen autoritativen Zone erlaubt. Damit ist die Rekursion deaktiviert.

```
options { allow-query { none; }; };
```

[1] vgl. Seminararbeiten über Firewalls von F. Ban und M. Karnuth

```
zone "alice.foo" {
        type master;
        file "/path/to/zonefile";
        allow-query { any; };
};
```

Der Zugriff auf den DNS-Cache wird auf den internen Netzbereich beschränkt.

```
options { allow-query { 192.168.1.0/24; }; };
```

An dieser Stelle sei beispielhaft die Nameserver-Software `djbdns` genannt, die durch die getrennten Softwaremodule `tinydns` und `dnscache` die Umsetzung von Split DNS zwingend erfordert.

3.2 TSIG

Das Mai 2000 in RFC 2845 spezifizierte Verfahren *Transaction Signature* (TSIG) stellt die Datenintegrität und gegenseitige Authentisierung der Kommunikationspartner einer DNS-Transaktion sicher. Vertraulichkeit wird nicht umgesetzt, da die Entwickler des Verfahrens keine Notwendigkeit für die Verschlüsselung von ohnehin öffentlich abrufbaren DNS-Daten sahen. In die eigentliche DNS-Nachricht wird ein TSIG RR eingefügt, der kein Teil der Zone ist, sondern nur während dieser DNS-Transaktion existiert daher auch nicht gecached wird. Der Grund für diesen *Meta-RR* liegt darin, die Parameter des Signierverfahrens zu übertragen, ohne die Kompatibilität des DNS-Protokolls zu beeinträchtigen.

TSIG fußt auf dem *Keyed-Hash Message Authentication Code* (HMAC) und nutzt derzeit MD5 als Hash-Funktion. Bei HMAC wird mittels einer Formel[4] von einem symmetrischer Schlüssel und einer zu übertragenden Nachricht der Hash-Wert berechnet. Gemäß Spezifikation werden die zwei Feldern „MAC“ und „MAC Size“ des TSIG RR von der Hash-Berechnung ausgenommen, da an ihre Stelle anschließend der Hash-Wert und seine Länge geschrieben werden. Der Empfänger kann mit Kenntnis des Schlüssels denselben Hash-Wert berechnen und somit die Integrität der Nachricht und die Authentizität des Senders überprüfen. Zum Schutz vor Replay-Angriffen ist im TSIG RR ein Timestamp-Feld untergebracht, welches ebenfalls signiert wird.

Die Schlüsselverteilung ist nicht spezifiziert und muss manuell über einen sicheren Kanal erfolgen. TSIG eignet sich daher in erster Linie für die Absicherung von DNS-Transaktionen im Zuständigkeitsbereich einer einzelnen Autorität, zum Beispiel um Namensauflösungen zwischen Mailserver und DNS-Cache abzusichern, oder um Man-in-the-Middle-Angriffe bei Zonentransfers vom Master-Nameserver zu den Slaves zu verhindern.

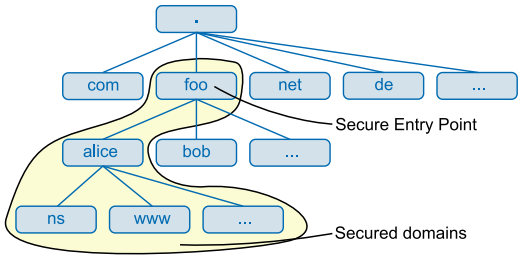
Mit TKEY wurde Oktober 2000 in RFC 2930 eine Methode spezifiziert, um Schlüssel direkt über DNS auszutauschen. Da TKEY hierfür jedoch eine bereits abgesicherte Verbindung benötigt, muss der initiale Schlüssel weiterhin manuell übertragen werden.

[4] vgl. die dem Seminar vorangehende Kryptographie-Veranstaltung

3.3 DNSSEC

Die *Domain Name System Security Extensions* (DNSSEC) verfolgen ähnlich wie TSIG das Ziel der Datenintegrität und Authentisierung, aber keiner Vertraulichkeit. Die Idee hinter DNSSEC wurde bereits 1993 vorgeschlagen, 1997 bis 1999 wurde die erste Spezifikation fertiggestellt. Da sich das Keymanagement als unpraktikabel erwies, wurde die Spezifikation jedoch verworfen und neu entwickelt, die aktuell gültigen RFCs 4033 bis 4035 sind von 2005. DNSSEC benötigt die Protokollerweiterung EDNS, die wie TSIG auf einen Meta-RR aufbaut, hier OPT genannt.

DNSSEC hat im Gegensatz zu TSIG den Anspruch sämtliche DNS-Kommunikation abzusichern, auch über mehrere Zuständigkeitsbereiche hinweg. Dazu basiert es auf einem asymmetrischen Kryptosystem, es finden derzeit die Signierverfahren RSA und DSA Verwendung.



Bei DNSSEC wird jeder RR einer Zone mit einem privaten Schlüssel signiert und die Signatur jeweils als RRSIG abgelegt. Der öffentliche Schlüssel wird in einem DNSKEY RR abgelegt. Innerhalb einer Zone können auch mehrere Schlüssel verwendet werden – üblich ist die Aufteilung in *Zone Signing Key* (ZSK), der die RRs signiert, und *Key Signing Key* (KSK), der den ZSK signiert und als Eintrittsschlüssel in die Zone dient. Der Hash-Wert des KSKs wird an die übergeordnete Instanz in der DNS-Hierarchie übermittelt und dort in der Zone zusätzlich zu Delegation und Glue Record als DS RR (*Delegation Signer*) abgelegt. Der DS RR, der wie alle anderen RRs ebenfalls signiert wird, verkettet die DNS-Hierarchie vertikal zu einer *Chain of Trust*, wobei die oberste Zone als *Secure Entry Point* dient. Mit Kenntnis des obersten öffentlichen Schlüssels, dem *Trusted Key*, können nun sämtliche darunterliegenden Schlüssel und Subzonen verifiziert werden und bilden eine *Island of Security*. Ideal wäre eine Deckung einer einzelnen Island of Security mit dem gesamten DNS-Namensraum, da in diesem Fall sämtliche DNS-Kommunikation abgesichert wäre.

NXDOMAIN-Antworten

Da nur RRs signiert werden können, sind NXDOMAIN-Antworten nicht verifizierbar. Als Abhilfe wurde der NSEC RR geschaffen, der einen Ring aller vorhandenen RRs bildet. Wird zum Beispiel ein nicht existierender RR „test.alice.foo“ abgefragt, so wird in die Antwort der signierte RR „mail.alice.foo. NSEC www.alice.foo.“ eingefügt. Damit wird angezeigt, dass dem Namen „mail“ in alphabetischer Reihenfolge der Name „www“ folgt, es also keinen RR mit Namen „test“ gibt. Dadurch wird allerdings eine Kopie der gesamten Zonendaten ermöglicht, da man alle NSEC RRs nacheinander abfragen kann. Dieses als *Zone Walking* bekannte Verfahren wird von einigen nicht geduldet, daher arbeitet die *Internet Engineering Task Force* (IETF) derzeit an einer anderen Lösung für NXDOMAIN-Antworten.

Kritik

DNSSEC hat in seiner gegenwärtigen Form einige namhafte Kritiker, darunter den PowerDNS-Entwickler B. Hubert, der in DNSSEC einen zu großen Aufwand für einen zu geringen Nutzen

der Gesamtsicherheit des Internets sieht [11]. Neben dem einmaligen Konfigurationsaufwand erfordert DNSSEC insbesondere eine regelmäßige Wartung, da die Schlüssel eine begrenzte Lebenszeit haben. Der `djbdns`-Entwickler D. Bernstein kritisiert an DNSSEC das Fehlen eines einheitlichen Konzepts zum Schlüsselaustausch [12]. Zur Zeit sind weder der sichere Kanal zur Übermittlung eines DS RRs an die übergeordnete Instanz festgelegt, noch der sichere Kanal zur Publizierung eines Secure-Entry-Point-Schlüssels.

Dennoch testet `se` als bislang einzige TLD seit September 2005 DNSSEC. `de` wartet vor der Einführung aus Datenschutzgründen auf eine Lösung des Zone Walkings.

4 Weitere Gefahren durch das DNS

Erweitert man die Betrachtung der DNS-Sicherheit, so ergeben sich einige interessante Szenarien, wo nicht das Domain Name System selbst angegriffen wird, sondern wo das DNS als Mittel zu einem sicherheitstechnisch bedenklichen Zweck eingesetzt wird. Diese sollen im Folgenden beleuchtet werden.

4.1 DNS Amplification Attack

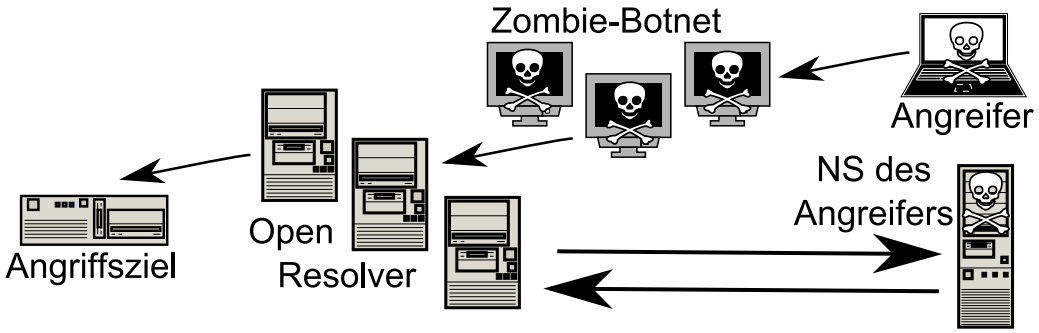


Abbildung 2: DNS Amplification Attack

Die *DNS Amplification Attack* ist eine Variante des DDoS-Angriffs. Angenommen Angreiferin Mallory hat Zugriff auf die Infrastruktur eines Botnets und hat die Absicht die Internetverbindung von Alice zu überfluten.[5] Zunächst richtet Mallory bei einem Nameserver unter ihrer Kontrolle (häufig ein kompromittierter Nameserver eines Dritten) Resource Records so an, damit der NS bei einer Anfrage Antwortpakete mit maximaler Länge erzeugt. Bei dem Angriff sendet das Botnet wiederholt gültige DNS-Anfragen an eine Liste von möglichst vielen Open Resolvern, wobei als Quelladresse Alice' Adresse gespoofed wird. Für die Open Resolver sieht es wie eine gewöhnliche DNS-Anfrage von Alice aus, die den RR von Mallorys NS abfragt. Die Open Resolver holen sich diesen großen RR bei der ersten Anfrage selbständig von Mallorys NS und schicken ihn bei jeder weiteren gespooften Anfrage aus dem Cache heraus an Alice. Auf Mallorys NS entsteht durch das Caching der Open Resolver nur eine geringe Belastung.

[5] vgl. Seminararbeit über Denial of Service von D. Willuhn

Die Open Resolver dienen als Reflektoren. Sie vervielfachen den Datenstrom des Botnets, wobei der Faktor zwischen Anfrage und Antwort etwa 1:10 betragen kann. Mit der Protokollerweiterung EDNS, wo die 512-Bytes-Längenbeschränkung von DNS-Nachrichten per UDP weiter hochgesetzt wurde, ist ein Verstärkungsfaktor von 1:70 möglich, falls die Open Resolver EDNS unterstützen.

Amplifications Attacks sind ein grundsätzliches Problem aller für IP-Spoofing anfälligen Dienste, deren Antwort größer als die Anfrage ist. Die DNS-Variante dieser Angriffsform ist jedoch durch die hohe Zahl der Open Resolver für den Angreifer besonders attraktiv.

4.2 IDN Homograph Spoofing Attack

Mit dem Verfahren *Internationalizing Domain Names in Applications* (IDNA, RFCs 3490 bis 3492) wurde die Möglichkeit geschaffen, Unicode in Domainnamen zu benutzen. Dazu muss weder das DNS-Protokoll noch die Serverinfrastruktur angepasst werden. Die Client-Software bildet einen Unicode-Domainnamen (*Internationalized Domain Name* – IDN) mittels der Kodierverfahren *Nameprep* und *Punycode* auf einen eindeutigen ASCII-kompatiblen Domainnamen ab.

Beispiel: Kodierung eines IDNs
www.STRäßé.de → www.strässé.de → www.xn--strss-ira1b.de

Ein *Homograph* ist in der Linguistik eine Zeichenfolge, die in derselben Schreibweise unterschiedliche Bedeutungen hat. Übertragen auf IDNA bezieht sich der Begriff auf Zeichenfolgen, die identisch aussehen, aber unterschiedlich kodiert werden.[6] Im weiteren Sinne dehnt man den Begriff auch auf sehr ähnlich aussehende Zeichenfolgen aus.

Ein lange bekannter Täuschungsversuch ist das Austauschen von I und L in unterschiedlicher Groß- und Kleinschreibung, oder von O und 0, die in manchen Schriftarten sehr ähnlich aussehen. Durch den großen Zeichensatz von Unicode erlangt dieses Problem eine neue Dimension [13]. 2005 ersetzte E. Johanson beim Domainnamen `paypal.com` das erste lateinische A durch das identisch aussehende kyrillische Pendant und registrierte den sich daraus ergebenden Domainnamen mit der ASCII-kompatiblen Kodierung `xn--pypal-4ve.com`. Unter dem Domainnamen betreibt er eine Website ohne Schadfunktion als Machbarkeitsbeweis.

Um den Angriff einzuschränken, können die Domain-Vergabestellen, wenn sie IDNs zulassen, die verfügbare Zeichenmenge einschränken. Second-Level-Domains mit kyrillischen Zeichen können z. B. unterhalb von `de` nicht registriert werden. Allerdings sind einige der erlaubten Zeichen zum ASCII-Zeichensatz sehr ähnlich, so könnte man bei einem flüchtigen Blick auf `www.deıne-bank.foo` übersehen, dass das kleine I durch die aus dem türkischen Alphabet bekannte punktlose Variante ersetzt wurde.

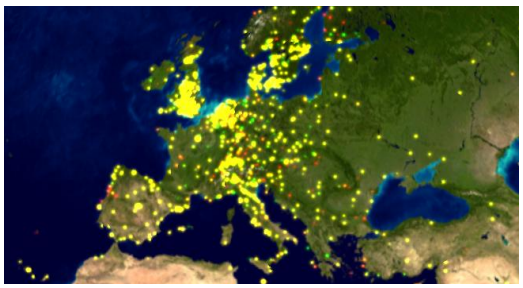
Darüber hinaus könnten Client-Anwendungen den Benutzer vor unterschiedlichen Schriften innerhalb eines Domainnamens z. B. durch farbliche Hinterlegung warnen.

[6] Eigentlich ist der Begriff in diesem Zusammenhang nicht korrekt, da Kodierung (Syntax) mit Bedeutung (Semantik) gleichgesetzt wird.

4.3 DNS Cache Snooping

Beim *Cache Snooping* oder *Cache Probing* fragt man einen Open Resolver ab, ob dieser einen bestimmten Resource Record im Cache hält [14]. Anhand des Ergebnisses kann man Rückschlüsse auf das Verhalten der Benutzer dieses Nameservers schließen, zum Beispiel ob eine bestimmte Website aufgerufen wurde.

Beim Snooping setzt man eine iterative Anfrage an den Open Resolver nach einem bestimmten RR ab. Der iterativ angefragte Nameserver wird also die Antwort lediglich aus seinem Cache heraus konstruieren. Ein Cache Hit liegt vor, wenn in der Antwort der angefragte RR enthalten ist, ein Cache Miss liegt entsprechend dann vor, wenn der RR nicht enthalten ist. Fragt man einen im DNS nicht existierenden RR ab, liegt dann ein Cache Hit vor, wenn der Open Resolver aufgrund von Negative Caching mit NXDOMAIN antwortet, ein Cache Miss, falls kein Namensfehler auftritt.



2005 suchte D. Kaminsky [15] mittels Cache Snooping nach potentiellen Hinweis auf den Kopierschutz XCP von Sony BMG. Diese auch als Sony Rootkit bekannte Software installiert sich automatisch unter Windows beim Einlegen der CD und nimmt Kontakt zu einem Webserver auf.[7] Da die Website auch von Benutzern ohne XCP besucht wird, berücksichtigte Kaminsky nur die öffentlichen DNS-Caches, die auch die WWW-Domain der XCP-Entwickler im Cache hatten, da auf dieser Informationen zur Entfernung des Rootkits zu finden waren. Dies ist keine sichere Erkennungsmethode, da zum einen auch Interessierte ohne XCP beide Websites besucht haben könnten, und zum anderen auch Benutzer ahnunglos von XCP infiziert sein könnten, ohne Informationen zur Deinstallation aufgerufen zu haben. Dennoch demonstriert dieses Beispiel eindrucksvoll, wie DNS Cache Snooping zusammen mit IP-basierter Geolokalisierung ein Indiz für die Rootkit-Verteilung liefert.

4.4 DNS Tunneling

In manchen LANs ist der Internetzugang für Benutzerin Alice nur verfügbar, wenn sie dafür freigeschaltet wurde und sich über eine VPN-Software oder auf einer HTTPS-Seite einloggt. Gleichzeitig ist jedoch im LAN ein DNS-Cache zugänglich. Tunnelt Alice Nutzdaten in DNS-Paketen, so kann sie die Internetverbindung ohne Login nutzen.

Alice benötigt eine Tunneling-Software, die die Nutzdaten in DNS-Anfragen an eine ihrer Domains kapselt. Diese Domain hat Alice an einen präparierten Nameserver delegiert, auf dem die Entkapselung stattfindet.

Um Daten in Upstream-Richtung zu übertragen, kann Alice eine DNS-Anfrage an einen TXT RR unterhalb ihrer präparierter Domain senden. Die Nutzdaten kodiert sie im Subdomainnamen. Daten in Downstream-Richtung sind in der Antwort als Textstring des TXT RRs kodiert. Sowohl beim Domainnamen als auch beim Antwortstring gibt es zahlreiche Einschränkungen in der Syntax – so ist die Menge der erlaubten Zeichen gering. Um beliebige Binärdaten

[7] vgl. Seminarbeit über Viren und Trojaner von M. Engel

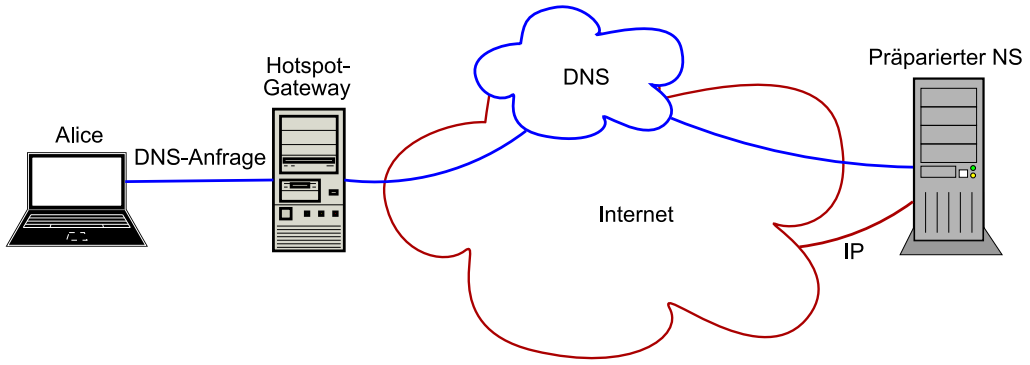


Abbildung 3: DNS-Tunneling

übertragen zu können, können diese z. B. mittels Base32 (up) und Base64 (down) in DNS-taugliche Form überführt werden. Außerdem sind die Datenmengen pro DNS-Transaktion sehr gering – ohne TCP oder EDNS ca. 100 Bytes up und 200-400 Bytes down. Die Nutzdaten müssen also fragmentiert und in mehreren DNS-Transaktionen übermittelt werden. Da Alice keine direkte Internetverbindung hat, kann zudem der präparierte Nameserver nicht von sich aus Daten zu Alice senden. Stattdessen muss Alice in Intervallen beim Nameserver anfragen (*pollen*), ob Daten in Downstream-Richtung zur Übertragung bereit stehen. [14]

DNS-Tunneling wurde in der Praxis tatsächlich implementiert und erprobt. Mit Ozyman-DNS kann eine OpenSSH-Verbindung über DNS getunnelt werden. NSTX überträgt IP-Pakete über DNS mittels des Linux-Kernelmoduls TUN/TAP. Beide Tools haben experimentellen Charakter, d. h. sie sind äußerst fehleranfällig. Die Nutzdatenrate beträgt ca. 1-6 KB/s.

5 Fazit

Die Rekursion eines DNS-Caches sollte ausschließlich auf das interne LAN beschränkt werden, sodass dieser keinen Open Resolver darstellt. Dadurch wird die Gefahr für DNS-Spoofing reduziert, zudem verhindert dies die ungewollte Nutzung des Servers als DDoS-Reflektor oder als Datenspeicher für Dritte. Rechtfertigt eine höhere DNS-Sicherheit weiteren Konfigurationsaufwand, so empfiehlt sich die Umsetzung von Split DNS.

Da BIND in der Vergangenheit zahlreiche sicherheitskritische Bugs aufwies und auch in jüngerer Zeit im Abstand einiger Monate neue Bugs bekannt wurden, ist es erwägenswert eine andere Nameserversoftware einzusetzen. Zu nennen seien da etwa `djbdns` und MaraDNS, bei denen die Entwickler besonderes Augenmerk auf die Sicherheit gelegt haben wollen. Weitere bekannte Nameserver-Implementationen sind z. B. PowerDNS und NSD.

Benutzt man Forwarder für rekursive Anfragen, so müssen diese in die Sicherheitsbetrachtung miteinbezogen werden. Ist die Sicherheit der Forwarder fraglich, sollte die Namensauflösung ohne Forwarder stattfinden. Insbesondere der teilweise noch gängige BIND 8 sollte nicht als Forwarder eingesetzt werden.

Offen bleibt, ob sich DNSSEC in Zukunft durchsetzen wird. Der gegenwärtige Stand ermöglicht bislang keinen großflächigen Produktivbetrieb.

Literaturverzeichnis

[1] John Klensin. IAB Statement on Infrastructure Domain and Subdomains. http://www.iab.org/documents/docs/iab-arpa-stmt.txt, Mai 2000.

[2] CERT. CA-1997-22. http://www.cert.org/advisories/CA-1997-22.html, August 1997.

[3] Men & Mice. Domain Health Survey. http://www2.menandmice.com/6000/6000_domain_health.html, November 1998.

[4] CERT. Incident Note IN-2001-11. http://www.cert.org/incident_notes/IN-2001-11.html, August 2001.

[5] CERT. Vulnerability Note VU#734644. http://www.kb.cert.org/vuls/id/734644, Dezember 2003.

[6] ISC. BIND4/BIND8 Unsuitable for Forwarder Use. http://www.isc.org/sw/bind/.

[7] Dan Kaminsky. White-Hat Hacking across the Domain Name System. *Communications of the ACM*, 49(6), 2006.

[8] Joe Stewart. DNS Cache Poisoning – The Next Generation. http://www.lurhq.com/dnscache.pdf, 2003.

[9] Vagner Sacramento. CAIS ALR-19112002a. http://www.rnp.br/cais/alertas/2002/cais-ALR-19112002a.html, November 2002.

[10] Heise Newsticker. Nameserver BIND anfällig für DoS-Attacken. http://www.heise.de/newsticker/meldung/84282, Januar 2007.

[11] Bert Hubert. The role of DNS and DNSSEC in information security. http://ds9a.nl/secure-dns.html, 2003.

[12] Dan Bernstein. DNS forgery. http://cr.yp.to/djbdns/forgery.html, 2004.

[13] Evgeniy Gabrilovich und Alex Gontmakher. The Homograph Attack. http://www.cs.technion.ac.il/~gabr/papers/homograph_full.pdf, 2002.

[14] Dan Kaminsky. Black Ops of DNS. http://www.ccc.de/congress/2004/fahrplan/event/121.de.html, 2004.

[15] Dan Kaminsky. Separation Anxiety. http://www.doxpara.com/?q=node/1131, 2005.