FACULDADE DE TECNOLOGIA IBRATEC DE JOÃO PESSOA
CURSO DE ESPECIALIZAÇÃO EM SEGURANÇA DA INFORMAÇÃO

SÍLVIO LUCAS DA SILVA

Segurança em DNS: Investigando o DNSSEC através de experimento prático

João Pessoa – PB

Junho de 2009

SÍLVIO LUCAS DA SILVA

Segurança em DNS: Investigando o DNSSEC através de experimento prático

Monografia apresentada ao curso de Pós Graduação em Segurança da Informação da Faculdade de Tecnologia IBRATEC de João Pessoa, como requisito parcial para conclusão do curso de Especialização em Segurança da Informação.

Orientador: Prof. Márcio Luiz Machado Nogueira, M.S.c.

João Pessoa – PB

Junho de 2009

SÍLVIO LUCAS DA SILVA

Segurança em DNS: Investigando o DNSSEC através de experimento prático

Monografia apresentada ao curso de Pós Graduação em Segurança da Informação da Faculdade IDEZ, como requisito parcial para conclusão do curso de Especialização em Segurança da Informação.

Orientador: Prof. Márcio Luiz Machado Nogueira, M.S.c.

Aprovada em de de 2009

Prof. Márcio Luiz Machado Nogueira, M.S.c.

Orientador

Profa. Marileuza Fernandes Corrreia de Lima, M.S.c

Examinadora

DEDICATÓRIA

À minha esposa que permaneceu diversos momentos longe da minha convivência em prol da elaboração deste trabalho.

Ao imenso esforço dos meus pais; esforço este que propiciou-me chegar onde eu cheguei.

AGRADECIMENTOS

Um agradecimento especial ao professor Márcio Nogueira pela sua paciência e imensa contribuição para a elaboração deste trabalho.

Agradeço imensamente também a Massimo Rimondini, professor da Universidade de Roma, criador do NetKit, pela sua imensa disposição em tirar as dúvidas pertinentes neste excelente emulador.

“ Na prática, a eficácia de uma contramedida normalmente depende de como ela é usada; o melhor cofre do mundo é inútil se ninguém se lembrar de fechar a porta. ”

National Research Council, 1991.

SILVA, Sílvio Lucas da. **Segurança em DNS: Investigando o DNSSEC através de experimento prático**. - 69p. Monografia (Especialização em Segurança da Informação) – Faculdade de Tecnologia IBRATEC de João Pessoa.

RESUMO

O *Domain Name System* – DNS é um dos principais serviços que compõem a internet. Através de um banco de dados distribuído, ele é o responsável por converter os endereços IPs em nomes e vice-versa. O DNS, como qualquer outro serviço, possui vulnerabilidades e falhas que estão sendo nos últimos tempos o alvo principal de pessoas mal-intencionadas na internet. O DNSSEC surge como uma alternativa interessante para prover segurança entre as transações DNS, utilizando-se para isso de técnicas de criptografia e assinaturas digitais. Este trabalho tem como objetivos realizar uma revisão de literatura dos problemas do DNS e verificar o funcionamento do DNSSEC, tanto em segurança como em facilidade, utilizando-se para isso de experimento prático.

Palavras-chave: criptografia, DNS, DNSSEC, emulação.

SILVA, Sílvio Lucas da. **Segurança em DNS: Investigando o DNSSEC através de experimento prático**. - 69p. Monografia (Especialização em Segurança da Informação) – Faculdade de Tecnologia IBRATEC de João Pessoa.

ABSTRACT

The Domain Name System - DNS is one of the key services that comprise the Internet. Through a distributed database, it's responsible for converting the IP addresses into names and vice versa. DNS, like any other service, has flaws and vulnerabilities that are recently the focus of ill-intentioned people on the internet. The DNSSEC is an interesting alternative to provide security between the DNS transactions, using it for the techniques of cryptography and digital signatures. This work aims to conduct a literature review of the problems of DNS and verify the operation of DNSSEC, both in safety and in ease, using for this practical experiment.

Keywords: encryption, DNS, DNSSEC, emulation.

LISTA DE ILUSTRAÇÕES

LISTA DE QUADROS

LISTA DE SIGLAS

DNS	Domain Name System
IP	Internet Protocol
ARPANET	Advanced Research Projects Agency Network
LAN	Local Area Network
ICANN	Internet Corporation for Assigned Names and Numbers
CAIS	Centro de Atendimento de Incidentes de Segurança
ISC	Internet Systems Consortium
BIND	Berkeley Internet Name Domain
DNSSEC	DNS Security Extensions
RFC	Request For Comments
DoS	Denial of Service
DDoS	Distributed Denial of Service
OSI	Open System Interconnection
ISO	International Organization for Standardization
UDP	User Datagram Protocol
ICMP	Internet Control Message Protocol
GTLD	Generic Top Domain Level Domain
TLD	Top Domain Level Domain
RR	Resource Record
TCP	Transmission Control Protocol
MITM	Main In The Middle
DHCP	Dynamic Host Configuration Protocol
TSIG	Transaction Signatures
IETF	Internet Engineering Task Force
SEP	Secure Entry Point
RSA	Inciais dos nomes de Ron **R**ivest, Adi **S**hamir e Len **A**dleman
SHA	Secure Hash Algorithm
TTL	Time To Live
DLV	DNSSEC Lookaside Validation
KSK	Key Signing Key
ZSK	Zone Signing Key
RAM	Random Access Memory
DIG	Domain Information Groper

SUMÁRIO

1. INTRODUÇÃO

O Sistema de Nomes de Domínio (DNS) é um serviço primordial para o funcionamento da internet como conhecemos. Ele é o responsável por converter os endereços IPs em nomes e vice-versa. Para uma pessoa que acessa a internet torna-se muito mais simples lembrar do mnemônico http://www.google.com.br, ao invés de 209.85.193.104 (endereço IP que identifica o servidor que hospeda o site de buscas *Google* na internet).

Kurose (2008) afirma que o DNS é um banco de dados distribuído implementado em um hierarquia de servidores de nome (servidores DNS), e um protocolo da camada de aplicação (protocolo DNS) que permite que *hosts* consultem o banco de dados distribuído.

Tanenbaum (2003) demonstra que durante a década de 80 a Agência de Projetos de Pesquisa Avançada em Redes – ARPANET (agência de pesquisa do governo norte americano criadora da rede que originou a internet) começou a crescer rapidamente com a inserção de novas redes (particularmente LANs). Com isso, a tarefa de localizar *hosts* tornou-se dispendiosa, e assim da necessidade de organizar as máquinas em domínios e mapear nomes de *hosts* em endereços IPs, foi criado o DNS.

Por essa mesma estrutura de funcionamento do DNS e devido a sua importância para o funcionamento das redes e da internet, verificamos que o DNS está sendo o grande alvo de pessoas mal intencionadas. Observe texto extraído do site da Corporação para Atribuição de Nomes e Números na Internet – ICANN, alertando sobre vulnerabilidades no serviço de DNS:

> Por causa da distribuição do DNS, nenhuma organização é capaz de implementar uma solução para esse ponto fraco. Isso exige a cooperação de todos os operadores de servidores de nomes e produtores de software para o DNS. No entanto, a ICANN julga ser muito importante despertar a consciência para a necessidade de atualizar a infra-estrutura da Internet a fim de enfrentar a ameaça. A organização tem tomado medidas urgentes para assessorar os operadores de domínios de primeiro nível nesse problema. Ela também preparou uma lista de FAQs e uma ferramenta on-line para teste de domínios, com o propósito de chamar a atenção para o problema e estimular os operadores de rede a corrigirem ou atualizarem seus servidores (ICANN, 2008).

Com isso, vemos que há uma grande preocupação com a segurança do DNS na comunidade de uma forma geral (seja ela acadêmica ou comercial), principalmente com novas vulnerabilidades e falhas de segurança relatadas recentemente. O Centro de Atendimento a Incidentes de Segurança (CAIS), relatou em seu boletim de alertas de segurança o alerta intitulado “Vulnerabilidades no ISC BIND e outras implementações de

DNS”. A divulgação deste provocou um grande estardalhaço na rede mundial de computadores; vulnerabilidade esta que acometia versões de servidores DNS fornecidas em sistemas operacionais utilizados por milhões de pessoas em todo o mundo. Observe trecho do alerta emitido:

> O CAIS está repassando o alerta do US-CERT, intitulado "VU#800113 - Multiple DNS implementations vulnerable to cache poisoning", que trata de deficiências no protocolo DNS (Domain Name System) e em implementações de DNS que permitem ataques de envenenamento de cache DNS (CAIS, 2008).

Com tantos problemas correlacionados e ataque sofridos, o próprio CAIS recomenda o uso do *DNS Security Extensions* (DNSSEC) como a única alternativa como “solução definitiva”:

> No momento a única solução definitiva para este tipo de vulnerabilidade, que é recorrente no protocolo DNS, é a implantação de DNS Security Extensions (DNSSEC). Para mais informações sobre DNSSEC no contexto brasileiro por favor consulte o site do Registro .br, disponível na seção "Mais informações" (CAIS, 2008).

O DNSSEC vem como uma grande solução aos problemas de segurança encontrados no DNS. Com a utilização de técnicas de criptografia, sob o pretexto de promover a autenticidade e integridade de um domínio, torna-se quase impossível o forjamento de um site.

Baseado nestas informações, este trabalho tem como objetivo geral averiguar quais as melhorias propostas pelo DNSSEC.

Os objetivos específicos deste trabalho são:

- Realizar uma revisão na literatura sobre os principais problemas de segurança no DNS;
- Analisar a proposta de melhoria do DNSSEC e
- Avaliar em experimento a usabilidade, facilidade e segurança dessa proposta em um cenário prático.

Este trabalho está estruturado da seguinte maneira:

O capítulo 02 relata o histórico do DNS e seu funcionamento, além dos principais

riscos e vulnerabilidades no DNS tradicional, como DNS *Spoofing*, ataques de negação de serviço (*DoS*) e ataques de negação de serviço distribuídos (DDoS), estes dois últimos também contra servidores DNS. Este capítulo abordará também algumas soluções já existentes para alguns dos problemas estudados, como por exemplo, *transaction signatures* – TSIG.

O capítulo 03 trata do DNSSEC, dando detalhes sobre seu funcionamento e sua proposta.

O último capitulo deste trabalho trará detalhes sobre o experimento de uma árvore de consultas DNS, como o intuito de avaliarmos a usabilidade, facilidade e desempenho do DNSSEC.

Quanto a metodologia deste trabalho, realizou-se como primeiro passo a pesquisa bibliográfica sobre o assunto estudado com o intuito de obtermos maiores informações sobre o DNS e o DNSSEC. Porém, os materiais encontrados quase sempre foram materiais redundantes (muitas vezes desencontrados) e pouco esclarecedores sobre o assunto. Além disso, a maioria referiam-se de forma (muito) teórica, sem pouco ou nenhum apontamento que permitisse-nos criar um cenário de simulação. Mesmo consultando documentos técnicos que foram os responsáveis por determinar padrões do DNS e DNSSEC (através de *Request For Comments* – *RFCs*), as informações contidas nestes documentos não foram suficientes para o nosso objetivo.

Porém, no trabalho de DE CAMPOS e JUSTO (2009) encontrou-se informações aproximadas do nosso objetivo. Estes autores apresentaram o funcionamento do DNS e do DNSSEC através de simulação na própria internet com servidores DNS reais, o que tornaria-se algo inviável, devido a ausência de recursos computacionais e financeiros. Mesmo assim, este trabalho foi primordial para a elaboração desta obra.

Com o objetivo criarmos uma maneira de testarmos o DNSSEC, utilizamos a ferramenta NetKit para emular um cenário semelhante ao de uma árvore DNS. Nesta emulação criamos seis computadores, cada um representando um componentes específico do DNS.

Os computadores emulados não necessitavam de acesso à internet. Esse era um fator primordial nesta simulação: com isso, poderíamos testar em um ambiente fechado (porém semelhante ao real) diversas situações e evitar que possíveis modificações/alterações em um servidor emulado causasse problemas a terceiros.

A partir deste cenário, foi possível realizarmos os testes necessários para a

conclusão deste trabalho.

As especificações do equipamento utilizado na emulação são as seguintes:

- Notebook Toshiba Sattelite M55-S351;
  - Processador Pentium M 1,73 Ghz;
  - 1,5 Gbytes de RAM;
  - HD 120 Gbytes;
  - Sistema operacional Opensuse 11.1
  - Kernel versão 2.6.27.21-0.1.2
  - NetKit versão 2.6
  - Wireshark versão 1.0.4;
  - tcpdump versão 3.9.8;
  - dnssec-tools versão 1.5

A metodologia é composta dos seguintes procedimentos:

- Realizar consultas DNS;
- Capturar o respectivo tráfego;
- Realizar consultas DNSSEC;
- Capturar novamente o tráfego gerado;
- Comparar os resultados obtidos entre as consultas DNS e DNSSEC.

Além dos procedimentos descritos anteriormente, a implementação em si do cenário de testes fazia parte do experimento para encontrar as respostas indagadas.

As ferramentas utilizadas neste trabalho (além do próprio NetKit) são ferramentas *opensource* e disponíveis na internet para download gratuitamente.

2. FUNDAMENTAÇÃO TEÓRICA

2.1. Surgimento do DNS

Na década de 1970, a ARPANET resumia-se a algumas centenas de computadores. Dada a proporção de *hosts* (computadores e dispositivos de rede) presentes na ARPANET, a tarefa de gerenciar o mapeamento de nomes para endereços era algo relativamente fácil. Na ARPANET existia um único host responsável por converter endereços e nomes: o host SRI-NIC. O host SRI-NIC era mantido pelo *SRI's Network Information Center* (conhecido por “NIC”). Neste host existia um simples arquivo, chamado de HOSTS.TXT, arquivo este onde continha o mapeamento de nomes/endereços de toda ARPANET. Quando algum administrador de rede da ARPANET necessitava alterar ou adicionar algum novo host, estes enviavam correios eletrônicos (*e-mails)* para o *Network Information Center* e posteriormente a equipe do NIC atualizava o arquivo HOSTS.TXT (uma ou duas vezes por semana). Por sua vez, os administradores de rede da ARPANET baixavam periodicamente a versão atualizada do arquivo HOSTS.TXT para os seus computadores.

Porém com o aumento dos *hosts*, a tarefa de gerenciar este procedimento tornou-se emblemática: o arquivo HOSTS.TXT crescia descontroladamente. Além disso, cada novo computador inserido na ARPANET implicava em nova transferência do arquivo HOSTS.TXT para todos os computadores da ARPANET. Com isso, a performance da rede degradou-se rapidamente e concluiu-se que esta não era a melhor maneira de gerenciar a resolução de nomes.

Paul Mockapetris, da *USC's Information Sciences Institute*, foi o responsável por desenvolver a arquitetura de um novo sistema para substituir a antiga forma de resolução de nomes. Em 1984 ele criou o Domain Name System como o conhecemos, e Mockapetris descreveu os detalhes deste novo sistema nas RFCs 882 e 883. Posteriormente foram lançadas as RFCs 1034 e 1035 em substituição as anteriores.

Várias outras RFCs foram lançadas a respeito do DNS, inclusive RFCs que descrevem vulnerabilidades e falhas de segurança que acometem o *Domain Name System.*

2.2. O protocolo DNS

O protocolo DNS segundo o modelo *Open System Interconnection – OSI de* camadas proposto pela *International Organization for Standardization* – ISO, o DNS é um protocolo da camada 7 – camada de aplicação.

Embora os protocolos associados ao modelo OSI raramente sejam usados nos dias de hoje, o modelo em si é de fato bastante geral e ainda válido, e as características descritas em cada camada são muito importantes (TANENBAUM, 2003):

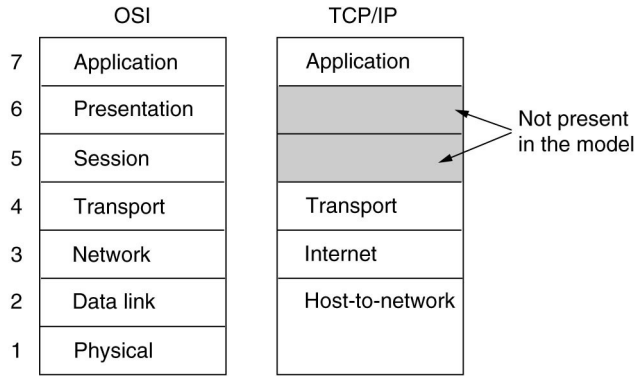


Figura 01 – O modelo OSI e o modelo de arquitetura TCP/IP
Fonte: **TANENBAUM**, 2003

O protocolo DNS utiliza-se do protocolo *User Datagram Protocol – UDP* da camada de transporte (camada 4 no modelo OSI). O protocolo UDP é um protocolo não orientado a conexão, ou seja, não guarda estado de suas conexões.

É interessante conhecermos maiores detalhes sobre a estrutura do protocolo DNS, como forma de entendimento do seu funcionamento. A figura a seguir exemplifica com detalhes o cabeçalho DNS, com os *flags* na forma detalhada:

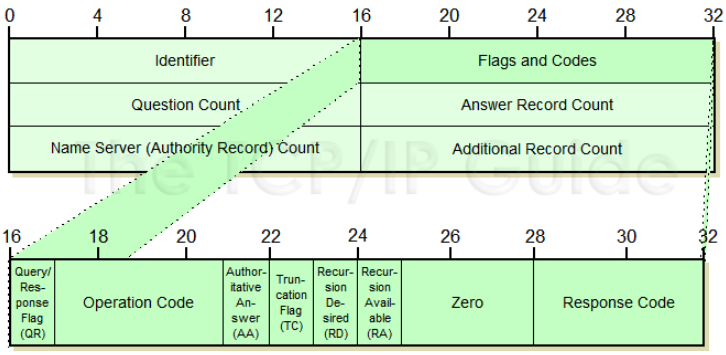


Figura 02 – Cabeçalho DNS
Fonte: **KOZIEROK**, 2005

O quadro 01 descreve a funcionalidade de todos os campos do cabeçalho DNS:

Quadro 01 – Descrição dos campos do protocolo DNS

<table>
<tr><th>Nome do Campo</th><th>Tamanho (bytes)</th><th>Descrição</th></tr>
<tr><td>ID</td><td>2</td><td>Identifier: A 16-bit identification field generated by the device that creates the DNS query. It is copied by the server into the response, so it can be used by that device to match that query to the corresponding reply received from a DNS server. This is used in a manner similar to how the Identifier field is used in many of the ICMP message types.</td></tr>
<tr><td>QR</td><td>1/8 (1 bit)</td><td>Query/Response Flag: Differentiates between queries and responses. Set to 0 when the query is generated; changed to 1 when that query is changed to a response by a replying server.</td></tr>
<tr><td>Opcode</td><td>½ (4 bits)</td><td>Operation Code: Specifies the type of query the message is carrying. This field is set by the creator of the query and copied unchanged into the response:
<table>
<tr><th>Opcode Value</th><th>Query Name</th><th>Description</th></tr>
<tr><td>0</td><td>QUERY</td><td>A standard query.</td></tr>
<tr><td>1</td><td>IQUERY</td><td>An inverse query; now obsolete. RFC 1035 defines the inverse query as an optional method for performing inverse DNS lookups, that is, finding a name from an IP address. Due to implementation difficulties, the method was never widely deployed, however, in favor of reverse mapping using the IN-ADDR.ARPA domain. Use of this Opcode value was formally obsoleted in RFC 3425, November 2002.</td></tr>
<tr><td>2</td><td>STATUS</td><td>A server status request.</td></tr>
<tr><td>3</td><td>(reserved)</td><td>Reserved, not used.</td></tr>
<tr><td>4</td><td>NOTIFY</td><td>A special message type added by RFC 1996. It is used by a primary (master, authoritative) server to tell secondary servers that data for a zone has changed and prompt them to request a zone transfer. See the discussion of DNS server enhancements for more details.</td></tr>
<tr><td>5</td><td>UPDATE</td><td>A special message type added by RFC 2136 to implement “dynamic DNS”. It allows resource records to be added, deleted or updated selectively. See the discussion of DNS server enhancements for more details.</td></tr>
</table></td></tr>
<tr><td>AA</td><td>1/8 (1 bit)</td><td>Authoritative Answer Flag: This bit is set to 1 in a response to indicate that the server that created the response is authoritative for the zone in which the domain name specified in the Question section is located. If it is 0, the response is non-authoritative.</td></tr>
<tr><td>TC</td><td>1/8 (1 bit)</td><td>Truncation Flag: When set to 1, indicates that the message was truncated due to its length being longer than the maximum permitted for the type of transport mechanism used. TCP doesn't have a length limit for messages, while UDP messages are limited to 512 bytes, so this bit being sent usually is an indication that the message was sent using UDP and was too long to fit. The client may need to establish a TCP session to get the full message. On the other hand, if the portion truncated was part of the Additional section, it may choose not to bother.</td></tr>
<tr><td>RD</td><td>1/8 (1 bit)</td><td>Recursion Desired: When set in a query, requests that the server receiving the query attempt to answer the query recursively, if the server supports recursive resolution. The value of this bit is not changed in the response.</td></tr>
<tr><td>RA</td><td>1/8 (1 bit)</td><td>Recursion Available: Set to 1 or cleared to 0 in a response to indicate whether the server creating the response supports recursive queries. This can then be</td></tr>
</table>

<table>
<tr><td></td><td></td><td>noted by the device that sent the query for future use.</td></tr>
<tr><td>Z</td><td>(3/8) (3 bits)</td><td>Zero: Three reserved bits set to zero.</td></tr>
<tr><td>RCode</td><td>½ (4 bits)</td><td>Response Code: Set to zero in queries, then changed by the replying server in a response to convey the results of processing the query. This field is used to indicate if the query was answered successfully, or if some sort of error occurred:
<table>
<tr><th>RCode Value</th><th>Response Code</th><th>Description</th></tr>
<tr><td>0</td><td>No Error</td><td>No error occurred.</td></tr>
<tr><td>1</td><td>Format Error</td><td>The server was unable to respond to the query due to a problem with how it was constructed.</td></tr>
<tr><td>2</td><td>Server Failure</td><td>The server was unable to respond to the query due to a problem with the server itself.</td></tr>
<tr><td>3</td><td>Name Error</td><td>The name specified in the query does not exist in the domain. This code can be used by an authoritative server for a zone (since it knows all the objects and subdomains in a domain) or by a caching server that implements negative caching.</td></tr>
<tr><td>4</td><td>Not Implemented</td><td>The type of query received is not supported by the server.</td></tr>
<tr><td>5</td><td>Refused</td><td>The server refused to process the query, generally for policy reasons and not technical ones. For example, certain types of operations, such as zone transfers, are restricted. The server will honor a zone transfer request only from certain devices.</td></tr>
<tr><td>6</td><td>YX Domain</td><td>A name exists when it should not.</td></tr>
<tr><td>7</td><td>YX RR Set</td><td>A resource record set exists that should not.</td></tr>
<tr><td>8</td><td>NX RR Set</td><td>A resource record set that should exist does not.</td></tr>
<tr><td>9</td><td>Not Auth</td><td>The server receiving the query is not authoritative for the zone specified.</td></tr>
<tr><td>10</td><td>Not Zone</td><td>A name specified in the message is not within the zone specified in the message.</td></tr>
</table></td></tr>
<tr><td>QDCount</td><td>2</td><td>Question Count: Specifies the number of questions in the Question section of the message.</td></tr>
<tr><td>ANCount</td><td>2</td><td>Answer Record Count: Specifies the number of resource records in the Answer section of the message.</td></tr>
<tr><td>NSCount</td><td>2</td><td>Authority Record Count: Specifies the number of resource records in the Authority section of the message. (“NS” stands for “name server”, of course. J)</td></tr>
<tr><td>ARCount</td><td>2</td><td>Additional Record Count: Specifies the number of resource records in the Additional section of the message.</td></tr>
</table>

Fonte: **KOZIEROK**, 2005

O conhecimento destes detalhes do protocolo DNS é de suma importância. Muitas vulnerabilidades do DNS baseiam-se nestes detalhes e no seu mecanismo de funcionamento. Por exemplo, o ataque do tipo *spoofing* de DNS explora a maneira como o campo ID é gerado. Daremos maiores detalhes sobre este tipo de ataque nas próximas seções deste trabalho.

2.3. Estrutura de funcionamento do DNS

Como citado anteriormente, o DNS é um banco de dados distribuído. A estrutura do DNS é conhecida como “árvore invertida”, como exemplifica a figura 03:

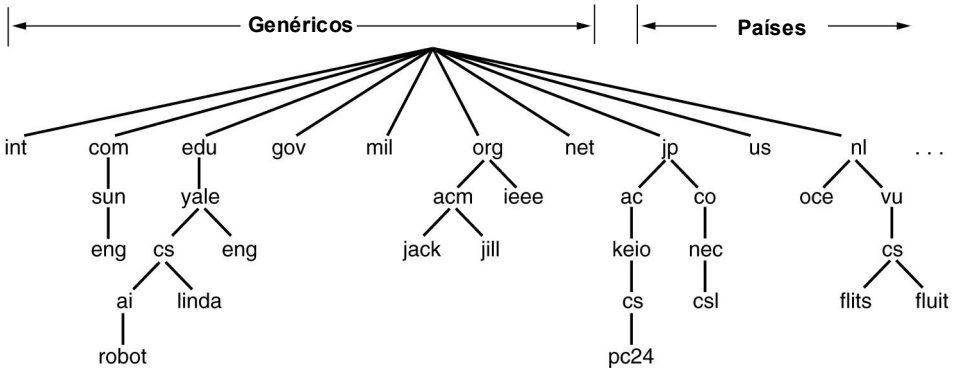


Figura 03 – Uma parte do espaço de nomes de domínios da Internet
Fonte: **TANENBAUM**, 2003

Podemos perceber que a estrutura do DNS é semelhante a estrutura de diretórios dos sistemas UNIX. O ponto (“.”) representa o nó raiz da hierarquia – o topo da árvore. Abaixo do nó raiz, encontramos os *Generic Top Domains Levels* (gTLDs). Os gTLDs são uma das categorias do *Top Level Domains (TLDs)* mantidos pelo IANA para uso na internet. Exemplos de gTLDs são os domínios .com, .edu, .org, .gov, dentre outros. A primeira RFC sobre TLDs (a RFC 920) foi lançada em outubro de 1984.

Os *Country Code Top Level Domains* – ccTLDs são domínios reservados para um país ou território. Cada país ou território possui uma entidade para gerenciar seu ccTLD (ICANN, 2008). O Brasil, por exemplo, possui a extensão .br para os domínios registrados em nosso país.

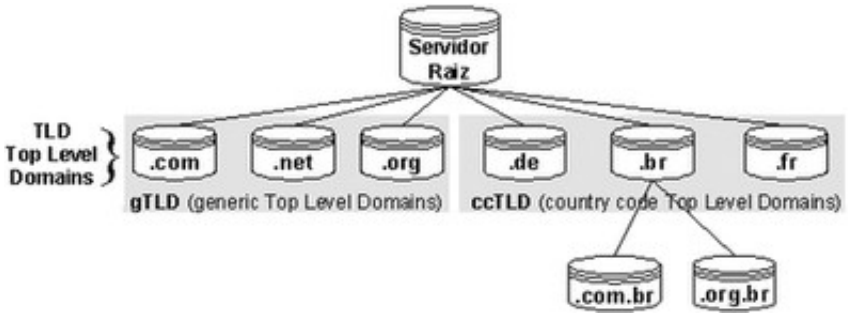


Figura 04 – Estrutura dos *Top Levels Domains* (TLDs).
Fonte: **VAZ**, 2009.

Cada nó também é a raiz de uma nova subárvore da árvore total. Cada uma das subárvores representam um domínio no *Domain Name System*. Cada domínio pode ser subdividido em divisões adicionais, chamados de subdomínios DNS, da mesma forma como os subdiretórios de um sistema de arquivos.

O entendimento da estrutura da árvore DNS remete-nos ao mecanismo de funcionamento das consultas DNS. O funcionamento dos servidores recursivos, que realizam a consulta de forma hierárquica, utilizam a estrutura em árvore invertida para realizar consultas até o servidor raiz caso necessário que por sua vez, na sua forma mais intrínseca remete-nos aos *resource records*.

2.4. Registros de Recursos (*Resource Records*)

A configuração de uma zona ou domínio fica armazenada em um arquivo específico no servidor DNS. Dentro deste arquivo existem *resource records* – RRs. Os *resource records* são entradas nos arquivos de zona do servidor DNS que refletem, por exemplo, a resolução de um endereço IP em nome, dentre outras informações.

Basicamente, a resposta recebida por um cliente DNS (resolver) durante uma consulta resume-se ao *resource record* encontrado no servidor consultado. Tanenbaum (2003, p. 620) conclui que “a principal função do DNS é mapear nomes de domínios em registros de recursos” .

Observe alguns exemplos de *resource records*:

Type	Meaning	Value
SOA	Start of Authority	Parameters for this zone
A	IP address of a host	32-Bit integer
MX	Mail exchange	Priority, domain willing to accept e-mail
NS	Name Server	Name of a server for this domain
CNAME	Canonical name	Domain name
PTR	Pointer	Alias for an IP address
HINFO	Host description	CPU and OS in ASCII
TXT	Text	Uninterpreted ASCII text

Quadro 02 – Principais Registros de Recursos (RRs)
Fonte: **TANEMBAUM**, 2003

2.5. Partes essenciais ao funcionamento do DNS

Com o intuito de simplificar o complexo mecanismo de funcionamento do DNS, as tarefas essenciais para o seu bom funcionamento foram subdivididas em diversos componentes distintos. Cada componente possui uma função específica, sendo esta imprescindível para o DNS como um todo.

O bom entendimento destes componentes de forma individualizada e em seu contexto, e também da maneira como estes interagem entre si, contribuem para compreender melhor o funcionamento do DNS e conhecer suas possíveis fragilidades. Além destes fatores, torna-se mais fácil compreender como as soluções de segurança propostas ao DNS abordam os pontos considerados críticos. Nos tópicos seguintes descreve-se de forma detalhada cada uma destas partes envolvidas.

2.5.1. Resolvers (clientes DNS)

Resolvers são clientes que acessam os servidores de nomes. Programas que são executados em um *host* e precisam de informações DNS usam o *resolver* para solicitar informações a um servidor DNS. Outras funções do *resolver* em um *host* são interpretar respostas (que podem ser *resource records* – *RRs* ou um erro) e devolver a informação para os programas que a requisitaram.

Os *hosts* também possuem um *cache* DNS como forma de evitar que consultas a servidores DNS realizadas anteriormente sejam refeitas desnecessariamente.

Um bom exemplo do uso dos *resolvers* ocorre no momento em que uma pessoa tenta acessar um site através do seu navegador de internet. Neste momento, o navegador solicita ao *resolver* que faça uma consulta a um servidor DNS (LIU e ALBITZ, 2006, tradução nossa).

2.5.2. Servidores DNS autoritativos e recursivos

Ao receber requisições de resolução de nome, um servidor DNS autoritativo responde um endereço caso possua, uma referência caso conheça o caminho da resolução ou uma negação caso não conheça (DE CAMPOS, JUSTO, 2009).

Um servidor recursivo, ao receber requisições de resolução de nomes, faz requisições para os servidores autoritativos e conforme a resposta recebida dos mesmos

continua a realizar requisições para outros servidores autoritativos até obter a resposta satisfatória (idem).

Um fator importante a ser citado é o fato dos servidores DNS recursivos possuírem cópias das suas últimas consultas realizadas, cópias estas chamadas de *DNS cache*. Portanto, não se faz necessário refazer a mesma consulta de DNS, até um período pré-estabelecido na configuração do servidor consultado.

Os servidores autoritativos são divididos em servidores *master* e servidores *slave*. Um servidor autoritativo *master* é o servidor principal do domínio. Ele responde as consultas DNS do qual ele possui autoridade. O servidor *slave* é o servidor que responde as consultas DNS caso o servidor *master* falhe (problemas no equipamento, problemas conexão com a internet do servidor *master*, etc.). O servidor *slave* é um servidor que armazena uma cópia dos domínios dos quais o *master* é responsável.

Qualquer atualização nos arquivos de zona do servidor master, o servidor *slave* também é atualizado automaticamente através de um procedimento intitulado transferência de zona. Na transferência de zona, os arquivos que contém os RRs são copiados para o servidor *slave*, exatamente como estão no servidor *master*. Após esta transferência, diz-se que os servidores DNS estão sincronizados entre si.

A figura 05 ilustra a transferência de zona:

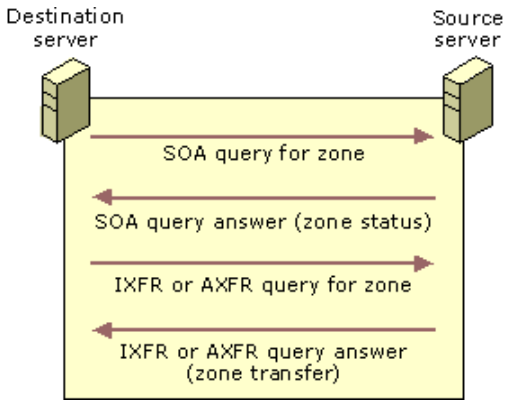


Figura 05 – Transferência de Zona
Fonte: **MICROSOFT**, 2005

A transferência de zona pode ocorrer de duas maneiras: completa (AXFR) ou incremental (IXFR). No primeiro caso a transferência das informações contidas no servidor principal ocorre de forma completa, enquanto no segundo a transferência é interativa, ou seja, apenas as informações que sofreram alterações são transferidas. Na transferência de zona, o servidor DNS utiliza o protocolo *Transmission Control Protocol* –

TCP (protocolo da camada de transporte), como forma de garantir a entrega dos dados da zona ao servidor slave.

2.5.3. Servidores DNS raiz (*root servers*)

Os servidores DNS raiz são os servidores que estão no topo da árvore DNS. Quando a consulta recursiva chega a este servidor, ele é responsável por redirecionar esta consulta para o servidor de um TLD inferior, ou então relatar que a informação procurada não existe.

Estes servidores são cruciais para o bom funcionamento da internet, já que muitas consultas DNS “terminam” nestes servidores.

Existem doze *root servers* espalhados pelo mundo, representados pelas doze primeiras letras do alfabeto (letra “A” até “M”). Porém, para cada um destes servidores raiz existem vários outros duplicados (funcionando em *cluster*, em redundância, etc.).

A figura a seguir mostra de forma didática a localização dos doze *root servers* distribuídos pelo globo terrestre. Observe na figura que há um *root server* em nosso país:

Figura 06 – Distribuição Geográfica dos servidores DNS raiz (*root servers*)

Fonte: **ROOT-SERVERS.ORG**, 2009

Concluindo o entendimento das ferramentas necessárias ao funcionamento do DNS, verifique na figura a seguir a junção de todos os componentes e uma breve explanação:

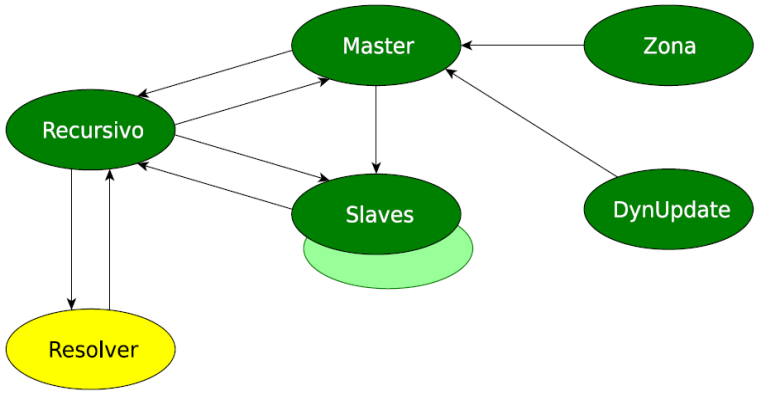


Figura 07 – Componentes necessários ao funcionamento do DNS

Fonte: **DE CAMPOS**, **JUSTO**, 2009

a) O *resolver* faz consultas no servidor recursivo e recebe as respostas solicitadas;
b) O servidor recursivo faz consultas nos servidores *masters* e *slaves*;
c) O servidor *master* armazena o arquivo de zona (este pode ser atualizado através de atualizações dinâmicas ou manualmente – configurado pelo administrador da rede);
d) O servidor *slave* recebe a zona do *master,* seguindo o esquema de transferência de zona.

2.6. Riscos e Vulnerabilidades Associadas ao DNS

Os servidores DNS são susceptíveis a uma grande variedade de tipos de ataques e falhas de segurança. Grande parte dos servidores DNS instalados no mundo utilizam o software *Berkeley Internet Name Domain,* conhecido na internet simplesmente pelo acrônimo “BIND”. O BIND é um servidor para o protocolo DNS criado pela Universidade de Berkeley, em meados dos anos 80. Atualmente está em sua nona versão, e apartir desta, suporta o DNSSEC.

Para fins didáticos, todas as informações contidas neste documento são relacionadas ao servidor DNS BIND.

Além das falhas de segurança que serão relatadas neste capítulo (falhas de segurança já conhecidas), ainda perdura o problema de servidores mal configurados ou

desatualizados: no primeiro caso este problema é causado por administradores de rede ou segurança que descuidam no momento da configuração de um servidor (ou seja, aplica-se a qualquer servidor, independente do serviço que ele provê – não é exclusividade do DNS) e com isso favorecem pessoas mal intencionadas a utilizar-se desta falha em benefício de si próprio. Os *crackers* são um bom exemplo de pessoas mal intencionadas da internet. Tanenbaum (2003, p.768) relata que *crackers* tem como objetivo “testar o sistema alheio de alguém; roubar dados”.

No segundo caso, quando uma nova falha é descoberta em um software (seja esta falha ela de segurança ou não), na maioria das vezes a mantenedora do software disponibiliza (geralmente na internet), uma atualização/correção para que o problema seja sanado. Quando o administrador não atualiza o seu software, este torna-se vulnerável para as mesmas pessoas mal intencionadas citadas anteriormente.

A seguir descrevemos os tipos mais comuns de ataques ao DNS.

2.6.1. Ataques “Homen no Meio” (Man in The Middle – MITM)

Segundo TURNBULL (2005, p. 464, tradução nossa), ataques *main in the middle* são “ataques que permitem um atacante interceptar seu tráfego DNS ou forjar seu servidor DNS”. Ainda segundo TURNBULL (2005, p. 464, tradução nossa), a finalidade deste ataque é “fazer o *spoofing* dos dados de entrada ou saída ou assumir a identidade do seu servidor DNS”.

É bastante interessante a opinião de GREEN (2005, p.06, tradução nossa) sobre ataques do tipo MITM:

> “(..) a vítima não necessita estar executando um sistema operacional ou versão de software cliente vulnerável ou ainda não precisa manter comunicação com servidor malicioso comprometido. Um ataque MITM é possível sempre que duas partes estão se comunicando umas com as outras.”

Este ataque caracteriza-se pela interceptação do tráfego DNS de tal maneira que o atacante interpõe-se entre a vítima e o servidor de nomes. A figura 07 exemplifica este tipo de ataque:

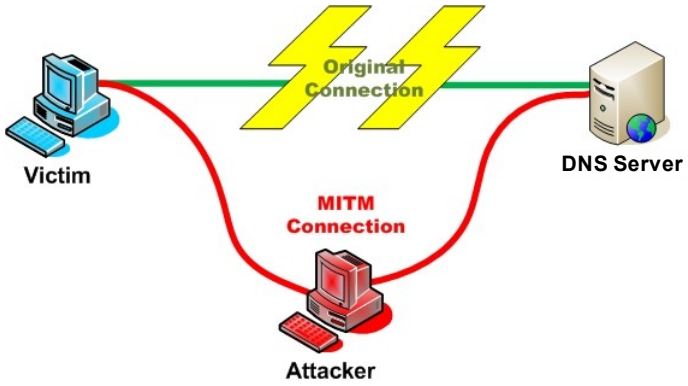


Figura 08 – Ataque do tipo MITM (modificada pelo autor)
Fonte: **OWASP,** 2009 - Modificada pelo Autor

Conclui-se que neste tipo de ataque o atacante consegue ter acesso a informações importantes da infra-estrutura e do funcionamento da rede, bem como permite forjar respostas à vítima sem o conhecimento da mesma (*spoofing*).

Uma segunda configuração interessante deste ataque é caso o atacante situe-se entre o servidor DNS primário e o servidor DNS secundário. Com isso, ele consegue obter informações detalhadas de todas as zonas das quais estes servidores DNS são responsáveis.

2.6.2. *Spoofing* de DNS

Este tipo de ataque caracteriza-se pela interceptação do tráfego DNS e através desta interceptação atacante “substitui” o tráfego legítimo por pacotes DNS forjados ao seu bel-prazer e com isso consegue, por exemplo, desviar o tráfego da vítima para um servidor fake, servidor este que pode conter um site falsificado de uma instituição, por exemplo. Este desvio é conseguido através da resolução de nome: ao invés do cliente receber a resposta que o *site* http://www.exemplo.com.br equivale ao endereço de Internet Protocol – IP 10.10.10.1, recebe como resposta falsa que este mesmo site é o IP 192.168.1.1, ou seja, um site falso – *fake*.

Interceptar o tráfego DNS não é uma tarefa difícil: o tráfego DNS não é criptografado, tampouco assinado.

O cabeçalho do protocolo DNS, segundo a RFC 1035, possui um campo intitulado *ID*, de 16 bits. Durante uma consulta DNS, este campo identifica a consulta (*query*) de tal forma que a resposta da consulta (*response)* também possui o mesmo identificador (ou seja, a consulta e a resposta possuem o mesmo identificador).

Observe as figuras a seguir explanam melhor esta situação:

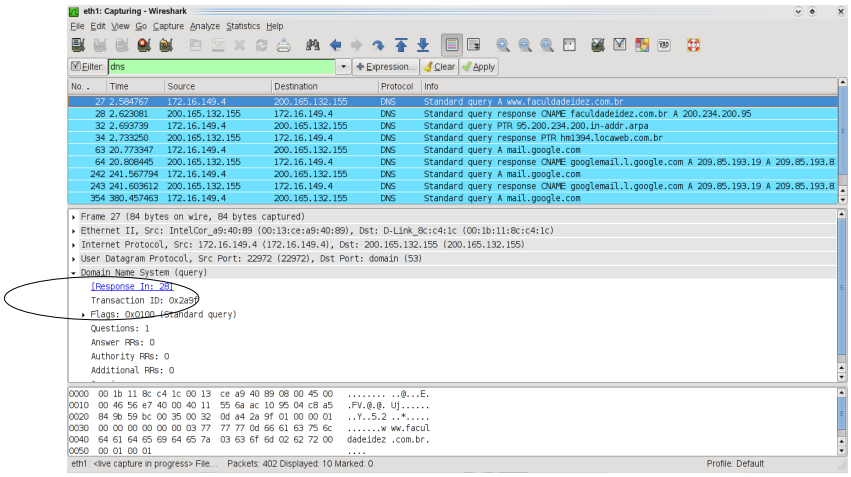


Figura 09 – Captura de tráfego DNS

Fonte: Software Wireshark

Nesta consulta DNS, verificamos que o identificador da consulta é 0x2a9f. Observe agora a resposta a essa consulta:

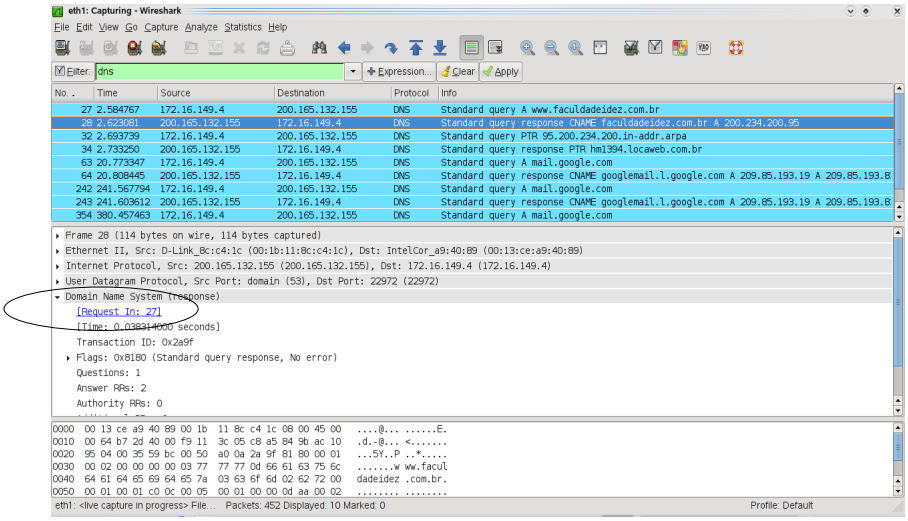


Figura 10 – Captura de tráfego DNS

Fonte: Software Wireshark

Como percebemos, o identificador da transação é o mesmo. O problema neste ataque consiste em forjamos o campo ID da próxima consulta DNS, já que este campo é incrementado de um, o que permite ao atacante inferir o seu valor subseqüente, e com isso, substituir o tráfego DNS legítimo. Este fator é determinante para a concretização do ataque *spoofing* de DNS.

Para conseguir concretizar seu objetivo, o atacante geralmente desfere um ataque de negação de serviço, fazendo com que o servidor DNS verdadeiro “saia do ar” e com isso o servidor *fake* assume o seu lugar.

Felizmente, as versões mais recentes de servidores DNS possuem mecanismos que randomizam os valores do campo ID para diversas consultas DNS (ou seja, o valor para este campo em uma consulta DNS não é gerado na forma seqüencial para a próxima consulta, e sim aleatória). Com isso, diminui-se consideravelmente o risco para este tipo de ataque.

2.6.3. Envenenamento de Cache (*Cache Poisoning*)

Também conhecido por *DNS pollution*, o ataque do tipo DNS *cache poisoning* consiste em fornecer falsas respostas ao um servidor de cache DNS, antes que o verdadeiro servidor autoritativo responda a consulta realizada.

A principal diferença entre o ataque do tipo *spoofing* de DNS e *cache poisoning* é que no segundo caso o atacante envia uma resposta falsa a um **servidor recursivo**, e não diretamente ao usuário, como no primeiro. Com isso, o servidor recursivo passa a armazenar resoluções DNS falsas em seu *cache* e portanto, resolve endereços para um servidor *fake*. Com isso, o usuário que solicitou a consulta dificilmente saberá que está acessando um site forjado.

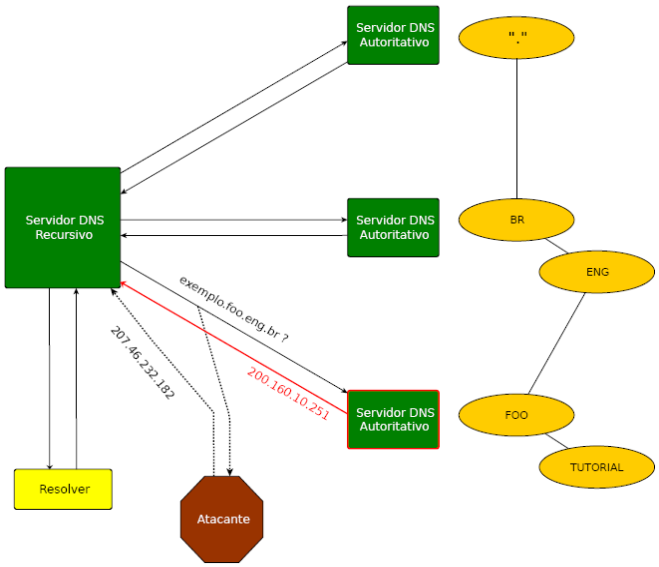


Figura 11 – Ataque *DNS cache poisoning*

Fonte: **DE CAMPOS, JUSTO,** 2009

Na figura acima o atacante respondeu a consulta feita pelo servidor recursivo antes da resposta do servidor autoritativo. Com isso, o cache do servidor recursivo ficou “poluído”, com uma falsa resolução de nome.

2.6.4. Ataques de Negação de Serviço (Denial of Service – DoS) e ataques de negação distribuída (Distributed Denial of Service – DDos)

O ataque do tipo negação de serviço tem como principal objetivo deixar um servidor ou serviço indisponível. Esse ataque consiste em “inundar” o servidor com inúmeras requisições de tal maneira que o servidor não consiga responder as consultas legítimas dos usuários; por causa disso, o servidor fica inoperante e inacessível.

Um outro objetivo do ataque *DoS* é fazer com que o servidor autêntico (por exemplo, um servidor DNS recursivo) fique inoperante e seja substituído por um servidor falso, fazendo com que este assuma o lugar do servidor real e responda as requisições destinadas a ele.

Um ataque *DoS* parte de um computador específico, enquanto um ataque de negação distribuída – *DDoS* pode ter como origem vários computadores, até mesmo distantes geograficamente entre si, o que dificulta muito mais o seu rastreamento e combate.

2.6.5. DNS *Amplification*

O ataque do tipo DNS *amplification* foi comentado largamente na internet no início de 2009. Um site pornográfico utilizou esta técnica sob pretexto de retirar um concorrente do ar. Atente a fragmento da reportagem retirada do site IdgNow:

> Uma ‘rixa’ entre dois sites de pornografia mostrou como pode ser explorada uma falha desconhecida no Domain Name System (DNS) da internet.
>
> O ataque é conhecido como DNS *Amplification* (extensão do DNS) e tem sido usado esporadicamente desde dezembro do ano passado. Ele tomou dimensão, contudo, apenas quando o provedor de internet ISPrime começou a seguir um ataque de ‘distributed denial of service’ (DDoS). Um site pornô tentava derrubar o rival (IDGNOW,2009).

O efeito amplificação em um ataque de DNS recursivo é baseado no fato que pequenas consultas podem gerar grandes pacotes UDP na resposta. O atacante faz o *spoofing* do endereço de origem, substituindo o endereço IP original da consulta DNS

pelo endereço IP de sua vítima. Com isso, a resposta é direcionada para o alvo escolhido.

Na especificação inicial do DNS, os pacotes UDP eram limitados a 512 bytes. Como conseqüência, uma simples consulta de 60 bytes pode gerar uma uma resposta de 512 bytes – uma amplificação de 8,5 vezes (VAUGHN e EVRON apud CIAC 1999, GNUPG 2002, tradução nossa).

Porém com o advento de novas extensões ao DNS, por exemplo, a extensão EDNS, uma simples consulta pode resultar em uma resposta de 4000 bytes – uma resposta 60 vezes maior.

Por conseqüência, um ataque desta natureza também pode ser interpretado com um ataque do tipo negação de serviço, devido aos problemas que este ataque resulta a vítima.

2.6.6. Transferência de Zona (*zone transfers*) e Atualizações Dinâmicas (*dynamic updates)*

Conforme comentado no item 2.5.2 deste trabalho, a transferência de zona ocorre no momento em que um servidor autoritativo *master* sofre uma atualização/alteração em seus arquivos de configuração de zona e replica esta alterações para o servidor autoritativo *slave.*

Durante a transferência de zona há a exposição de todas a informações referentes aquela zona, principalmente os *resource records*. Como o tráfego da transferência (assim como todo o tráfego DNS) não é um tráfego criptografado, qualquer pessoa mal intencionada pode interceptá-lo e inspecioná-lo, obtendo com isso informações valiosas daquela zona.

Já as atualizações dinâmicas do DNS ocorrem quando o protocolo Dynamic Host Configuration Protocol (DHCP) insere ou remove RRs de acordo com as modificações ocorridas no *leasing* do DHCP. O DHCP é o protocolo responsável por distribuir endereços IPs automaticamente em uma rede. À medida que um computador recebe ou perde um endereço IP, o DHCP atualiza automaticamente o DNS, inserindo ou removendo *resources records* nos arquivos de zona.

Um usuário mal intencionado pode utilizar as atualizações dinâmicas para desferir ataques de negação de serviço (DoS), remoções (intencionais) de RRs e IP spoofing (VIXIE, et al. 1997, tradução nossa).

2.7. Soluções de Segurança para o DNS

2.7.1. OpenDNS

O OpenDNS é uma alternativa para os usuários que utilizam servidores DNS recursivos. O OpenDNS é um serviço gratuito disponibilizado que oferece como vantagens a proteção contra filtragem de conteúdo (w*eb filtering)*, controle de acesso dos pais (*parentals controls*), e grande disponibilidade e velocidade do serviço, dentre outras características. Um usuário doméstico que deseja utilizar o OpenDNS necessita apenas alterar a configuração do seu computador para utilizar como servidores DNS recursivos os fornecidos pelos OpenDNS: 208.67.222.222 e 208.67.220.220.

Uma das principais vantagens do OpenDNS é a comunidade intitulada *"The PhishTank Community"*. Esta comunidade consiste em mais de de 50.000 especialistas em *phishing* espalhados por todo o mundo que trabalham para manter o acesso dos usuários do OpenDNS seguro. Eles trabalham localizando e bloqueando sites fraudulentos que utilizam a técnica de *phishing*. (OPENDNS,2009, tradução nossa).

De acordo com o *Anti-Phishing Work Group* – APWG (2007, p. 01, tradução nossa), *phishing* é "uma de forma de furto *on-line* de identidade que emprega tanto técnicas de engenharia social como técnicas de subterfúgio para furtar dados da identidade pessoal e credenciais de contas financeiras de consumidores".

2.7.2. Transaction Signatures – TSIG

Segundo TURNBULL (2005, p. 500, tradução nossa), *transaction signatures* "provêem um mecanismo para verificar a identidade dos servidores DNS com os quais você está se comunicando".

O TSIG tem como foco principal assegurar a autenticidade nas transferências de zona e atualizações dinâmicas do DNS (*dynamic updates*). O TSIG utiliza-se de chaves secretas compartilhadas entre as entidades envolvidas, como forma de fornecer uma relação de confiança.

Contudo, o mecanismo de chaves públicas indicado na RFC 2535 (*Domain Name System Security Extensions*) pode ser tornar impraticável devido ao fato deste mecanismo requerer uma criptografia de chave pública cara e complexa do ponto de vista computacional, além de uma lógica de autenticação complexa. (VIXIE, et al. 2000,

tradução nossa).

Portanto, TSIG tem funcionalidade semelhante as extensões do DNS (DNSSEC), porém possui aplicação restrita aos dois casos citados neste item.

2.7.3. DNSSEC

O DNSSEC é uma extensão de segurança para o DNS que tem como foco principal oferecer autenticidade e integridade nas consultas DNS. O DNSSEC originou-se de um grupo de trabalho do IETF (grupo *dnsext*).

O DNSSEC apresenta-se como a melhor alternativa para os problemas mais acometidos pelo protocolo DNS. À medida que mais servidores DNS adotam esta tecnologia, as transações DNS são tornando-se mais seguras e confiáveis. O DNSSEC é o foco do próximo capítulo deste trabalho.

3. O DNSSEC

Devido aos vários problemas citados anteriormente com o modelo tradicional do *Domain Name System*, a Força Tarefa de Engenharia da Internet – IETF decidiu criar um grupo para pesquisas relacionadas a segurança do DNS (grupo chamado de *dnsext*). A partir destas pesquisas foram criadas as Extensões de Segurança do DNS – DNSSEC.

A RFC 3833 analisa as ameaças ao DNS. Esta RFC descreve como metas principais do DNSSEC manter a integridade dos dados e a autenticação da origem dos dados (ATKINS E AUSTEIN, 2004, tradução nossa).

A integridade, autenticidade e a confidencialidade são os três principais pilares da segurança da informação. A integridade é a garantia que os dados recebidos são exatamente idênticos como foram enviados (não foram modificados, inseridas novas informações ou excluídas); a confidencialidade trata da proteção da informação para que ela não esteja disponível a quem não seja de direito, e a autenticidade dá garantias que a entidade se comunicando é quem realmente afirma ser (STALLINGS, 2008).

O DNSSEC resolve a maioria dos problemas de segurança relacionados ao DNS. Ataques que exploram a integridade e autenticidade das transações DNS são solucionadas através da criptografia de chaves públicas e assinaturas utilizadas no DNSSEC. Além disso, ele é compatível com o modelo de DNS existente e deixa para os administradores a opção de utilizá-lo ou não.

3.1. O uso da criptografia no DNSSEC

O DNSSEC é baseado na criptografia e assinaturas digitais por chaves públicas e privadas, como forma de prover a integridade e a autenticidade durante suas transações.

As técnicas criptográficas são meios que permitem que um emissor disfarce as informações (dados) de modo que um intruso não consiga decifrar nenhuma informação, mesmo que estes dados sejam interceptados. Apenas o legítimo destinatário deve conseguir recuperar as informações disfarçadas (KUROSE, 2006).

Segundo Stallings (2008, p.181) “a criptografia assimétrica é uma forma de criptossistema em que a criptografia e a decriptografia são realizadas usando chaves diferentes – uma chave pública e uma chave privada”.

Stallings complementa sabiamente a definição acima explicando sobre o mecanismo de funcionamento da criptografia assimétrica (2008, p. 181):

> a criptografia assimétrica transforma o texto claro em texto cifrado usando uma de duas chaves e um algoritmo de criptografia. Usando a outra chave associada e um algoritmo de decriptografia, o texto claro é recuperado a partir do texto cifrado.

A assinatura digital deriva dos conceitos citados anteriormente na criptografia de chaves públicas. Utilizando este princípio, a assinatura digital permite ao destinatário a verificação da integridade e autenticidade de uma mensagem. Isso permite ao destinatário verificar se a informação foi alterada, seja na origem, durante a sua transmissão ou mesmo no destinatário. Além disso, é possível verificar se a informação foi forjada por outrem, em nome do remetente legítimo.

A figura a seguir ilustra bem o funcionamento da assinatura digital:

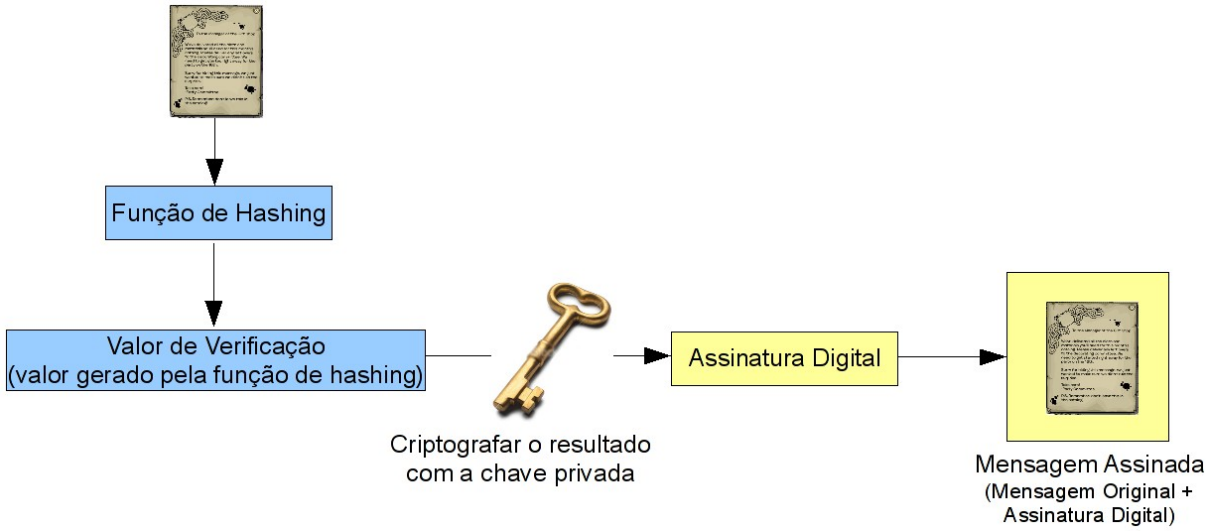


Figura 12 – Mecanismo da Assinatura Digital

Em um primeiro momento, a mensagem é submetida a uma *função de hashing*. Segundo STALLINGS (2008, p.228), uma função de *hashing* é “uma função que relaciona uma mensagem de qualquer tamanho a um valor de hash de tamanho fixo, que serve como um autenticador”.

A seguir, este valor gerado é criptografado com a chave privada do usuário e o resultado desta operação (a assinatura digital) é anexado a mensagem original.

Observe que em nenhum momento a intenção da assinatura digital é manter a confidencialidade da informação, e sim a integridade e a autenticidade.

Quando a mensagem é recebida, ocorre processo inverso: a assinatura digital é

retirada da mensagem, decriptografada com a chave pública do emissor, e assim é obtido novamente o valor de verificação (*hash*) gerado no emissor.

Após o passo anterior, a mensagem recebida é submetida novamente a função de *hashing*, e assim é gerado um outro valor de verificação que deve ser igual ao valor de verificação recebido do emissor. Caso contrário, esta mensagem não é confiável.

3.2. Domínios disponíveis para o funcionamento como DNSSEC

No dia 15 de janeiro de 2009, o órgão responsável pelo registro de domínios em nosso país, o registro.br, liberou a disponibilidade do uso do DNSSEC para os domínios .com.br e .org.br (os mais utilizados) . Antes desta data, apenas alguns domínios tinham a disponibilidade do DNSSEC, com destaque para os domínios .b.br (instituições financeiras) e .jus.br (órgãos da justiça) - estes dois com obrigatoriedade da adoção do DNSSEC para seu uso.

3.3. A aplicação do DNSSEC nos problemas de segurança

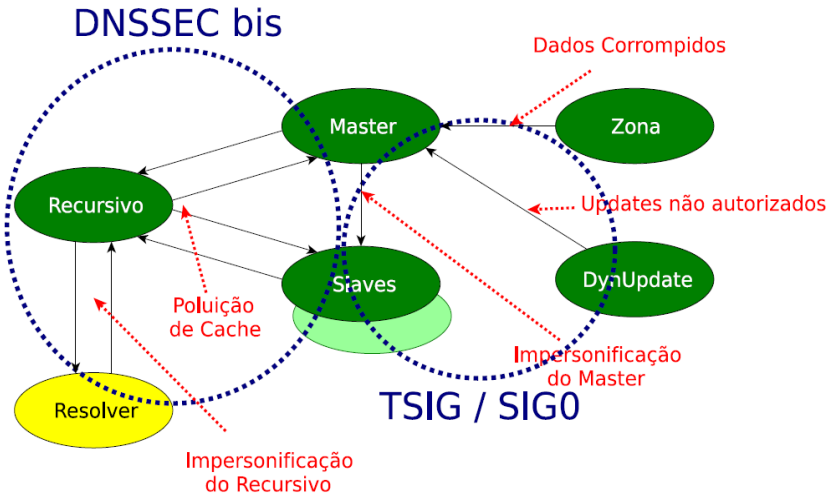


Figura 13 – Aplicação do DNSSEC

Fonte: **DE CAMPOS**, **JUSTO**, 2009

Concluímos na figura anterior que o DNSSEC provê segurança sob os seguintes aspectos:

- Na comunicação entre os servidores *master/slave* e os servidores recursivos;

- Na comunicação entre o resolver e os servidores recursivos.

A segurança entre os servidores *master* e *slave* e a segurança das atualizações dinâmicas são implementadas através das t*ransaction signatures – TSIG,* assunto comentado anteriormente em outra parte deste trabalho.

3.4. Resource Records no DNSSEC

O DNSSEC adiciona mais alguns *resource records* aos tradicionais já existentes:

3.4.2. DNSKEY

Para cada zona segura, há uma chave pública e uma chave privada associada a ela. O DNSKEY é o responsável por disponibilizar a chave pública da zona na forma de *resource record*. Salientamos que a chave privada deve ser guardada de forma secreta e não deve ser disponibilizada sob nenhuma hipótese.

Um resolver pode, então, usar a chave pública para validar assinaturas que abrangem os DNS *resource records sets* – RRsets na zona, e, portanto, para autenticar-lhes (ARENDS, et all, 2005, tradução nossa).

Segundo HUSTON (2006, p. 04, tradução nossa), "um r*esource record set* (RRset) é uma coleção de RRs em uma zona DNS que compartilham um nome em comum, uma classe e tipo".

A seção RDATA do DNSKEY é descrita a seguir:

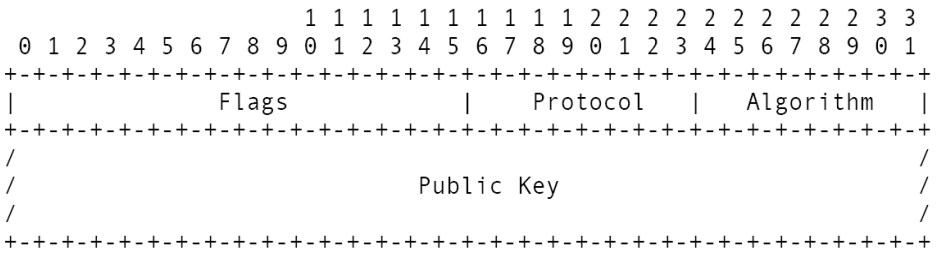


Figura 14 – seção RDATA do DNSKEY

Fonte: **ARENDS**, et al., 2005

Observe que a seção RDATA do DNSKEY está subdivido em quatro campos: o

campo flag consiste em dois octetos, o campo protocolo um octeto, o campo algoritmo um octeto e o campo correspondente a chave pública quatro octetos (lembrando que um octeto equivale a oito bits).

O RR DNSKEY a seguir armazena uma chave pública da zona DNS para o domínio fictício example.com:

```
example.com. 86400 IN DNSKEY 256 3 5 ( AQPSKmynfzW4kyBv015MUG2DeIQ3
                                       Cbl+BBZH4b/0PY1kxkmvHjcZc8no
                                       kfzj31GajIQKY+5CptLr3buXA10h
                                       WqTkF7H6RfoRqXQeogmMHfpftf6z
                                       Mv1LyBUgia7za6ZEz0JB0ztyvhjL
                                       742iU/TpPSEDhm2SNKLijfUppn1U
                                       aNvv4w==  )
```

Figura 15 – Exemplo de RR do tipo DNSKEY
Fonte: **ARENDS**, et. al., 2005

O valor 86400 representa o campo TTL (time to live), em segundos. A seguir temos a classe (IN) e o tipo de resource record (neste caso, DNSKEY). O valor 256 indica que o bit zone key (bit 7) nos flags tem valor 1. Neste caso, o valor 5 indica qual algoritmo de chave pública foi utilizado para gerar a chave (neste caso, o valor 5 indica que trata-se do algoritmo RSA/SHA1). O texto restante é a chave pública, na codificação base64.

3.4.3. RRSIG

O DNSSEC usa a criptografia de chaves públicas para assinar e autenticar os DNS *resource record sets*. O *resource record* RRSIG é utilizado para armazenar as assinaturas digitais e são utilizadas na autenticação do DNSSEC. A assinatura é gerada através primeiramente da criação de um hash do RRset. Após isso, o hash é encriptado usando a chave privada do administrador da zona.

Por exemplo, para uma zona que contém cinco *resource records* (SOA, NS, A, MX e DNSKEY), existem, no mínimo, cinco RRsets, e cada um deles tem que possuir o seu próprio RRSIG.

Um validador pode usar os *resource records* RRSIG para autenticar os RRsets da zona (ARENDS, et al, 2005).

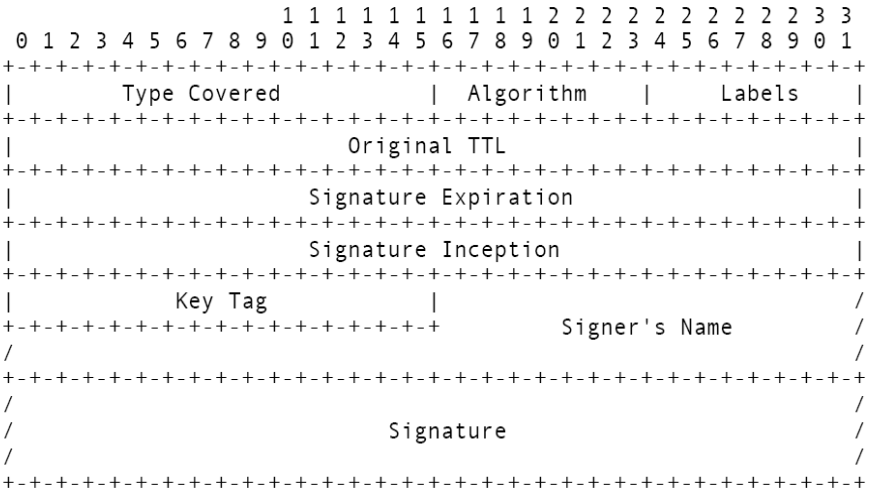


Figura 16 – seção RDATA do RRSIG

Fonte: **ARENDS**, et. al., 2005

A figura a seguir representa um *resource record* do tipo RRSIG, no qual o mesmo armazena a assinatura para o domínio *example.com (resource record* do tipo “A” - nome para endereço*)*:

```
host.example.com. 86400 IN RRSIG A 5 3 86400 20030322173103 (
                                20030220173103 2642 example.com.
                                oJB1W6WNGv+ldvQ3WDG0MQkg5IEhjRip8WTr
                                PYGv07h108dUKGMeDPKijVCHX3DDKdfb+v6o
                                B9wfuh3DTJXUAfI/M0zmO/zz8bW0Rznl803t
                                GNazPwQKkRN20XPXV6nwwfoXmJQbsLNrLfkG
                                J5D6fwFm8nN+6pBzeDQfsS3Ap3o= )
```

Figura 17 – Exemplo de RR do tipo RRSIG

Fonte: **ARENDS**, et. al., 2005

Seguindo da esquerda para a direita, temos os seguintes campos:

- O primeiro campo representa o nome do proprietário do domínio (host.example.com);
- O segundo campo representa o *time to live* (TTL);
- O próximo campo representa a classe (“IN”);
- O quarto campo representa o tipo do *resource record* (tipo RRSIG).
- O campo a seguir é o campo *type covered* (tipo “A”). Este campo identifica qual o tipo de RRset abrangido pelo registro RRSIG (neste caso, um *resource record* do

tipo “A” - um mapeamento de nome para IP);

- O campo representado pelo valor “5” trata-se do campo *algorithm number field*, e este identifica o algoritmo criptográfico utilizado para criar a assinatura. O valor 5 representa o algoritmo RSA/SHA1;
- O sétimo campo representado pelo valor 3 é o campo *label*. Este campo especifica o número de *labels* no original RRSIG;
- O valor 86400 é o TTL original para o RRset “A”;
- Os valores *20030322173103* e *20030220173103* representam a data de expiração e criação, respectivamente, indicando qual a data que o RRset foi assinado e qual será a data de expiração da assinatura. Por exemplo, este RRset foi assinado no dia 20 de fevereiro de 2003, às 17:31:03, e irá expirar no dia 22 de de março de 2003, às 17:31:03 (este é o intervalo padrão – um mês. Pode ser alterado).
- O número 2642 é o campo key tag. O valor deste campo contém a *key tag* do RR DNSKEY que valida esta assinatura;
- A seguir temos o signer's name (representado por *example.com*);
- O último campo é a assinatura em si.

3.4.4. NSEC – Next Secure

O registro NSEC tem como função apontar de forma canônica (em ordem alfabética) e consecutiva o próximo RRset de uma zona e assim criar uma cadeia de confiança. Com isso, **um *resolver* pode utilizar *resource records* NSEC autenticados para dizer que determinado RRset não está presente em uma zona assinada durante uma consulta** (ARENDS, 2005, tradução nossa).

Observe o fragmento a seguir, retirado apartir de um arquivo de zona em nossa simulação, exemplifica bem a idéia do RR NSEC:

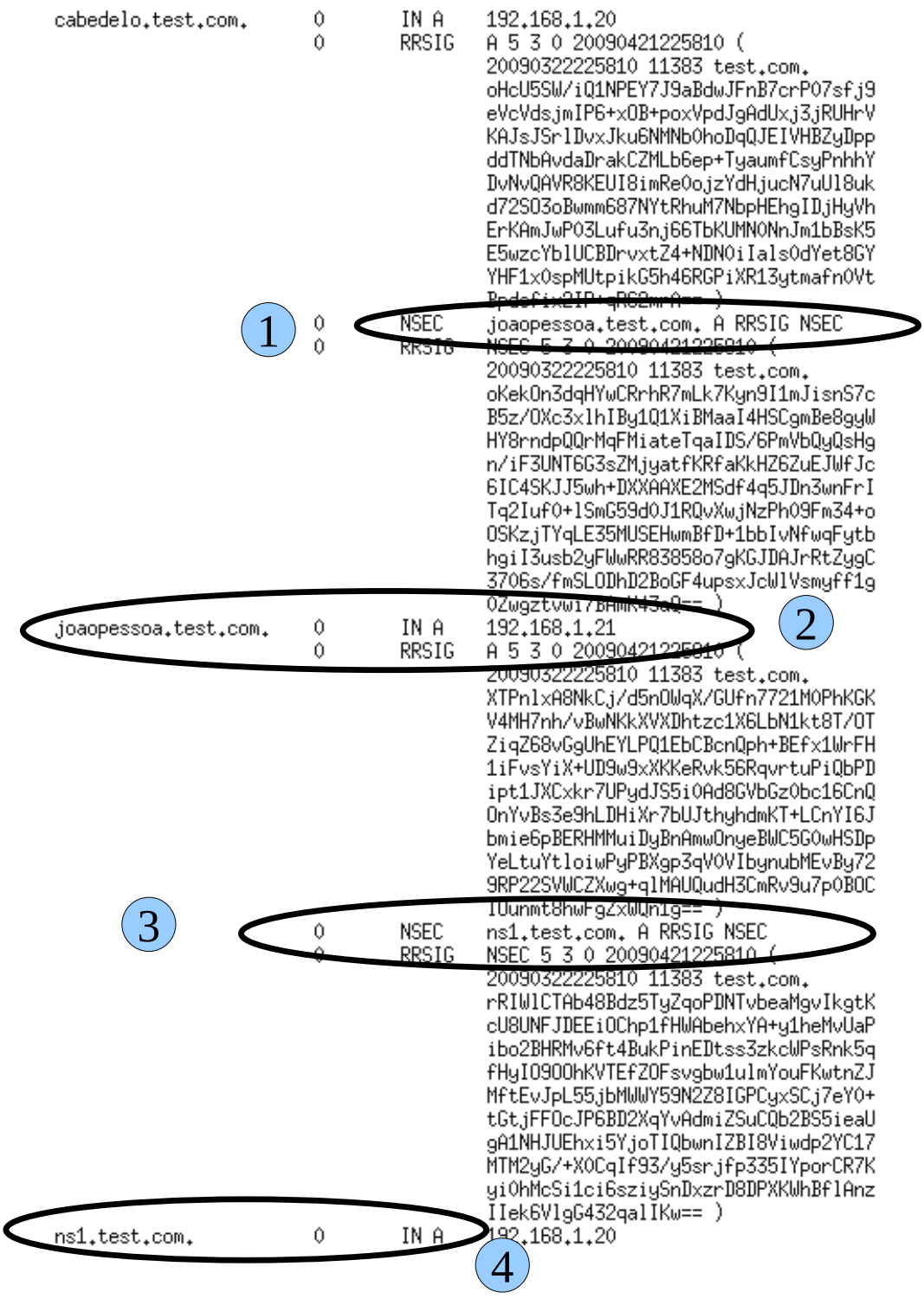


Figura 18 – Exemplo de RR do tipo NSEC

Verifique que a primeira ocorrência (1) do RR NSEC aponta para o nome **joaopessoa.test.com**. No RR do tipo “A” que contém a conversão do nome **joaopessoa.test.com** para o IP 192.168.1.21 (2), há logo a seguir um registro do tipo NSEC apontando para o nome **ns1.test.com** (3) e assim sucessivamente.

Analisando (3), o registro NSEC é composto dos seguintes itens:

- Os quatro primeiros campos são respectivamente o nome (test.com), *time to live* (em nosso caso, o valor do TTL está com valor zero de forma intencional), a classe e o tipo do *resource record* (NSEC) e o próximo nome autoritativo (em nosso caso, ns1.test.com);
- Logo após, vemos que os *resource records* associados com ao nome teste.com são A, RRSIG e NSEC.

O último registro NSEC aponta para o primeiro registro da zona. Por conseqüência, se uma consulta buscar por um nome de domínio que não existe na zona, um registro RRSIG que abrange todo o registro NSEC acompanha a resposta, autenticando a não-existência do nome de domínio ou do tipo de dados solicitado (BURITI, 2006, apud ALBITS e LIU, 2001).

A próxima ilustração descreve a seção RDATA do resource record NSEC:

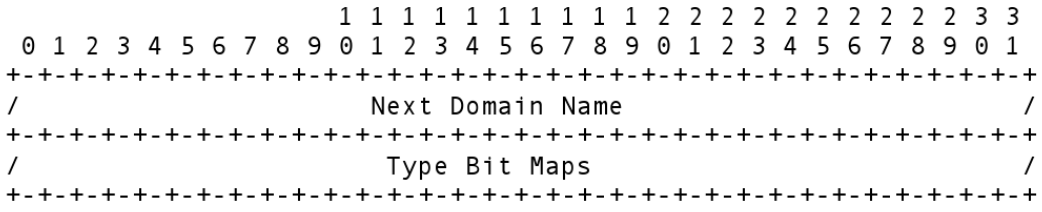


Figura 19 – Seção RDATA do NSEC

Fonte: **ARENDS**, et. al., 2005

3.4.5. DS – Delegation Signer

O *resource record* DS armazena o *hash* da chave pública de uma zona filha. Mesmo os *resource records* criados pelo DNSSEC comentados neste trabalho e citados até o momento, um atacante poderia fornecer RRs DNSKEY e o RRSIG que coincidissem com o presente na zona e com isso fazer com que a reposta parecesse ser autêntica (HUSTON, op. cit., tradução nossa).

Neste ponto abordaremos um fator importante ainda não comentado neste trabalho: a cadeia de confiança (*trust chain*).

Para entender a idéia da cadeia de confiança, novamente verifica-se o funcionamento do DNS. Com exceção da zona root (“.”), cada zona possui uma zona pai (*parent*). Este fato fica evidente na figura a seguir:

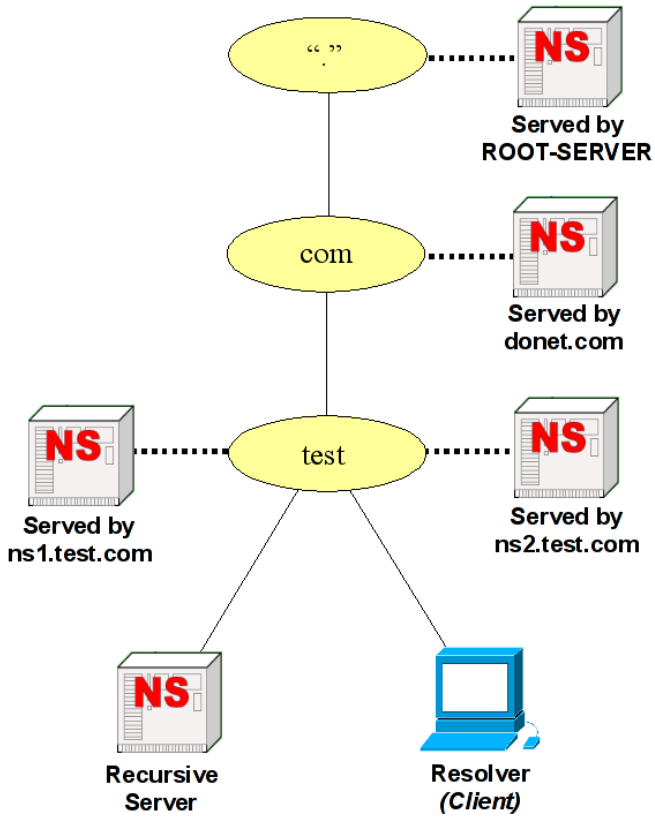


Figura 20 – Exemplo de Estrutura do DNS

Verificando nas “folhas” da árvore DNS, concluímos que o pai da zona *test* é a zona *com*, que por sua vez, tem como pai a zona raiz (“.”).

Cada registro DS armazenado no pai aponta para a zona filha de forma sucessiva. A cadeia de confiança é a responsável por prover a autenticidade das delegações de uma zona até um ponto de confiança. O ponto de confiança é onde há um *trusted key* - uma chave pública confiável fornecida pela empresa responsável por administrar os domínios de primeiro nível (TLDs). Em nosso país, o registro.br fornece uma *trusted key* para a zona “.br”.

Para criarmos uma cadeia de confiança, criamos o *record* DS na zona filha e adicionamos este RR no arquivo de zona do pai. A figura a seguir exemplifica bem esta idéia:

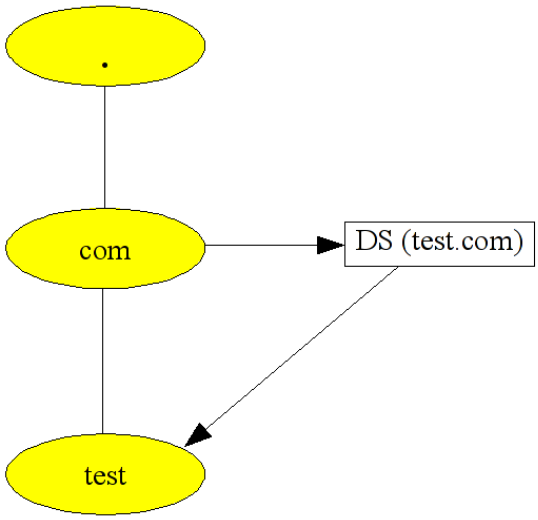


Figura 21 – Cadeia de Confiança

Em um primeiro momento, cria-se o *record* DS do domínio *test.com*, e a seguir insere este registro em seu pai, o domínio com. Após este processo, assina-se novamente a zona pai, para que a zona assinada permaneça íntegra: no momento em que inserimos o *record* DS da zona filha no arquivo de zona do pai, o seu conteúdo foi modificado, e com isso, a assinatura criada anteriormente não confere mais com o conteúdo do arquivo.

A próxima ilustração demonstra a seção RDATA do RR *Delegation Signer*:

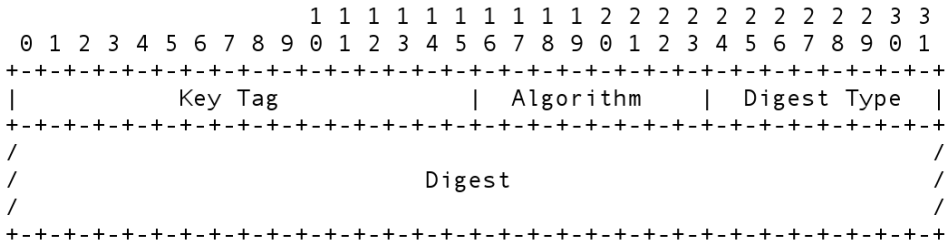


Figura 22 – Seção RDATA do DS

Fonte: **ARENDS**, et. al., 2005

A seção RDATA do *delegation signer* é composta pelos campos ***key tag, algorithm***, ***digest type*** e ***digest***. O campo digest refere-se ao RR DNSKEY, incluindo um *digest* da DNSKEY.

Para validar o DNSKEY da zona, o DS associado, registo RRSIG do DS, e o DNSKEY da zona pai é recuperado. O DS record é validado usando a DNSKEY para

decriptar o RRSIG(DS) record, e então checa se o resultado coincide com o DS record.

Um exemplo de registro do tipo DS pode ser observado na figura a seguir:

```
dskey.example.com. 86400 IN DS 60485 5 1 ( 2BB183AF5F22588179A53B0A
                                           98631FAD1A292118 )
```

Figura 23 – Exemplo de RR do tipo DS

Fonte: **ARENDS**, et. al., 2005

Os campos, da esquerda para a direita, representam o nome, o TTL, a classe (IN), e o tipo de *resource record* (*delegation signer*). O valor 60485 é a *key tag* para a DNSKEY correspondente de *dskey.example.com*. O valor 5 indica o algoritmo usado na DNSKEY de *dskey.example.com*, enquanto o valor 1 refere-se ao algoritmo utilizado para construir o *digest*. O restante refere-se ao *digest* em si.

Como alternativa ao processo da cadeia de confiança, foi criado o DLV – *DNSSEC Lookaside Validation*. Este é o alvo do nosso próximo assunto.

3.5. DLV – DNSSEC Lookaside Validation

Ciente do problema que seria proporcionado aos administradores de rede em ter que adicionar múltiplas *trusted keys* (diante das milhões de zonas existentes), o *Internet Systems Consortium* – ISC, teve a idéia de criar o *DNSSEC Lookaside Validation* - DLV. A idéia principal do DLV é evitar que um administrador tenha que adicionar múltiplas *trusted keys* em seu servidor recursivo. Em vez disso, basta o administrador adicionar em seu servidor recursivo a trusted key fornecida pelo ISC (disponível em http://dlv.isc.org).

DLV são uma maneira de garantir os dados publicados da zona até quando o domínio pai, ou os seus pais, de acordo com o caso, não estão assinados. Neste caso, é usado o DLV, o qual é muito parecido com *record* DS. Quando um resolver procura por informações, ele faz uma segunda consulta em uma zona especial para ver se há registros DLV presentes para ela. Se existe estes registros, eles provêem a confiança que o resolver pode usar para habilitar a validação DNSSEC para os registros da zona (ISC, 2009).

3.6. O uso das chaves no DNSSEC

No DNSSEC, há dois tipos de chaves que merecem ser citadas: *Zone Signing Key – ZSK* e *Key Signing Key – KSK*.

A chave ZSK é utilizada para assinar os registros (*RRsets*) da zona. Por conseqüência, esta chave pode (e deve) ser modificada constantemente.

No entanto, a chave KSK assina toda a zona, e esta possui uma freqüência menor de modificação. A chave KSK precisa ser publicada, enquanto a ZSK é apenas de uso interno a zona.

A idéia principal de termos duas chaves distintas é permitir substituir uma chave de uso freqüente (a chave ZSK) sem ser necessário modificar o DS do domínio pai (*hash* da KSK) (DE CAMPOS, JUSTO, 2009).

4. O EXPERIMENTO

Com o intuito de testarmos a eficiência do DNSSEC, utilizamos um emulador denominado NetKit. O NetKit é uma ferramenta capaz de emular cenários de redes de computadores, como exemplo o de uma rede local e diversas estações. O Netkit tem como principal atrativo o pouco uso de recursos (principalmente de memória) do computador que está hospedando o cenário emulado. Ele utiliza um *kernel* Linux modificado como base para os computadores emulados e além disso todos os comandos nas máquinas emuladas são feitas em modo texto (não há interface gráfica). Com isso, torna-se muito simples manter diversos computadores emulados simultaneamente com um uso de processamento e memória bastante aceitável, se comparados ao de máquinas virtuais.

Há uma diferença sutil entre um simuladores e emuladores. Um **simulador** visa reproduzir o desempenho de um verdadeiro sistema (tempo de latência, perda de pacotes, etc), enquanto um **emulador** visa reproduzir fielmente as funcionalidades de um sistema real (configurações, arquiteturas, protocolos) com atenção limitada para o desempenho (RIMONDINI, 2007, tradução nossa).

Em nosso estudo utilizaremos o NetKit para simular um cenário tradicional de consultas DNS, e em seguida as mesmas consultas serão realizadas no mesmo cenário com o DNSSEC em funcionamento.

4.1. A estrutura do experimento

A figura a seguir descreve o cenário utilizado para simulação:

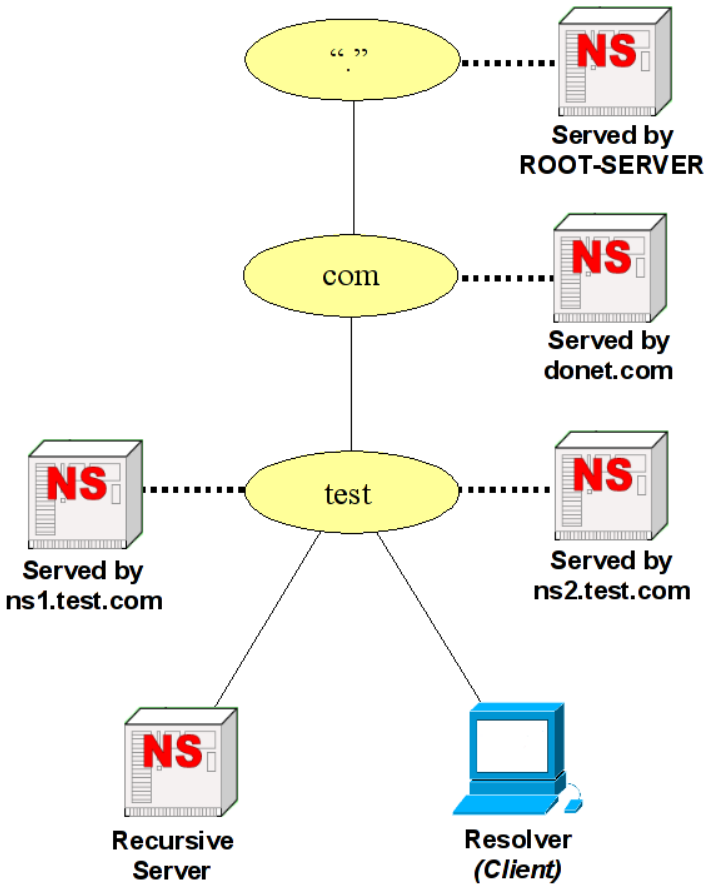


Figura 24 – Cenário da simulação

O domínio “.” é o domínio raiz de nossa simulação e é hospedado pelo servidor DNS intitulado **ROOT-SERVER**. O domínio .com é hospedado pelo servidor DNS **dotnet.com**.

Já o domínio test.com é hospedado pelo servidores DNS **ns1.test.com *(master)* e ns2.test.com *(slave)***.

A seguir temos o servidor recursivo e o resolver (este último fazendo a função de cliente DNS).

O quadro a seguir descreve os endereços IPs utilizados na simulação:

Quadro 03 – Endereços IPs utilizados na simulação

Nome do Computador	Endereço IP	Observações
ROOT-SERVER	192.168.1.254/24	Servidor DNS raiz (root server)
dotnet.com	192.168.1.253/24	Servidor DNS responsável pelo domínio .com

ns1.test.com	192.168.1.20/24	Servidor *master* responsável pelo domínio test.com
ns2.test.com	192.168.1.21/24	Servidor slave responsável pelo domínio test.com
recursive	192.168.1.10/24	Servidor recursivo
resolver	192.168.1.1/24	Computador cliente

4.2. Emulando consultas DNS

Para esta simulação, utilizamos o utilitário *ping* para gerar tráfego em nosso laboratório. No computador resolver, disparamos pacotes ICMP com destino ao domínio ns1.test.com (servidor autoritativo do domínio test.com) e capturamos os pacotes gerados durante a consulta através do software wireshark, software este sendo executado no computador recursive. Apesar de executarmos o wireshark no computador recursive, poderíamos utilizá-lo em qualquer um dos computadores emulados pois todos pertencem ao mesmo domínio de colisão. A figura a seguir descreve detalhadamente uma consulta DNS realizada em nosso cenário:

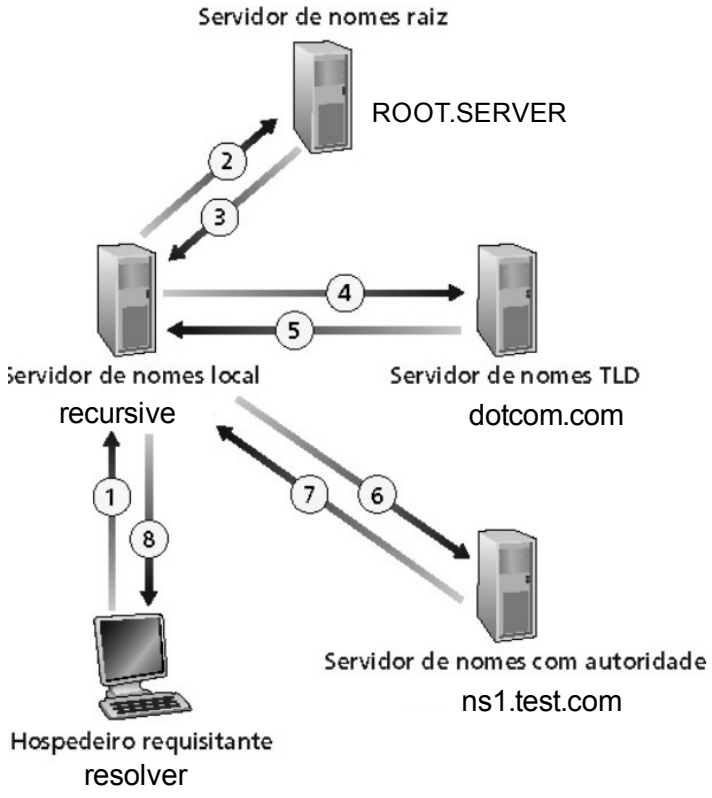


Figura 25 – Consulta DNS tradicional

Fonte: **KUROSE**, 2006 (modificada pelo autor)

Em linhas gerais, a consulta é feita do computador 192.168.1.1 (resolver) ao servidor recursive – IP 192.168.1.10 (consulta representada na figura por “1”) . Após isso, recursive pergunta ao ROOT-SERVER (192.168.1.254) quem é o servidor DNS autoritativo pelo domínio *com* (item 2), e recebe como resposta o nome dotcom.com, que corresponde ao endereço 192.168.1.253 (item 3). Logo após a consulta é feita ao mesmo servidor dotcom.com, indagando qual o servidor autoritativo do domínio test.com (item 4), que traz como resposta o servidor ns1.test.com (item 5). Após a consulta direta ao servidor ns1.test.com (itens 6 e 7), , a resposta é repassada do servidor recursive ao resolver, máquina esta que solicitou a consulta. Salientamos que abstraímos algumas etapas da consulta, como por exemplo, a conversão dos nomes dos servidores para endereços IP e vice-versa. A figura abaixo representa a captura do tráfego desta consulta:

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.1.1	192.168.1.10	DNS	Standard query A ns1.test.com
2	0.023655	192.168.1.10	192.168.1.254	DNS	Standard query A ns1.test.com
3	0.023665	192.168.1.10	192.168.1.254	DNS	Standard query NS <Root>
4	0.026196	192.168.1.254	192.168.1.10	DNS	Standard query response
5	0.027842	192.168.1.254	192.168.1.10	DNS	Standard query response NS ROOT-SERVER
6	0.033900	192.168.1.10	192.168.1.253	DNS	Standard query A ns1.test.com
7	0.035950	192.168.1.253	192.168.1.10	DNS	Standard query response
8	0.041928	192.168.1.10	192.168.1.21	DNS	Standard query A ns1.test.com
9	0.043352	192.168.1.21	192.168.1.10	DNS	Standard query response A 192.168.1.20
10	0.044041	192.168.1.10	192.168.1.1	DNS	Standard query response A 192.168.1.20
11	0.056779	192.168.1.1	192.168.1.10	DNS	Standard query PTR 20.1.168.192.in-addr.arpa
12	0.059836	192.168.1.10	192.168.1.254	DNS	Standard query PTR 20.1.168.192.in-addr.arpa
13	0.060649	192.168.1.254	192.168.1.10	DNS	Standard query response PTR ns1.test.com
14	0.061214	192.168.1.10	192.168.1.1	DNS	Standard query response PTR ns1.test.com

Figura 26 – Captura de uma consulta DNS tradicional

Fonte: Aplicativo Wireshark

Verificamos que para a execução de um simples *ping*, geramos 14 mensagens DNS. Porém, como forma de ilustrarmos melhor a consulta, utilizamos o software dnspktflow que tem como finalidade principal gerar gráficos de fluxos DNS capturados através do wireshark. O dnspktflow é integrante do pacote dnssec-tools, um conjunto de ferramentas que têm como objetivo principal facilitar a implementação do DNSSEC.

Observe a ilustração a seguir:

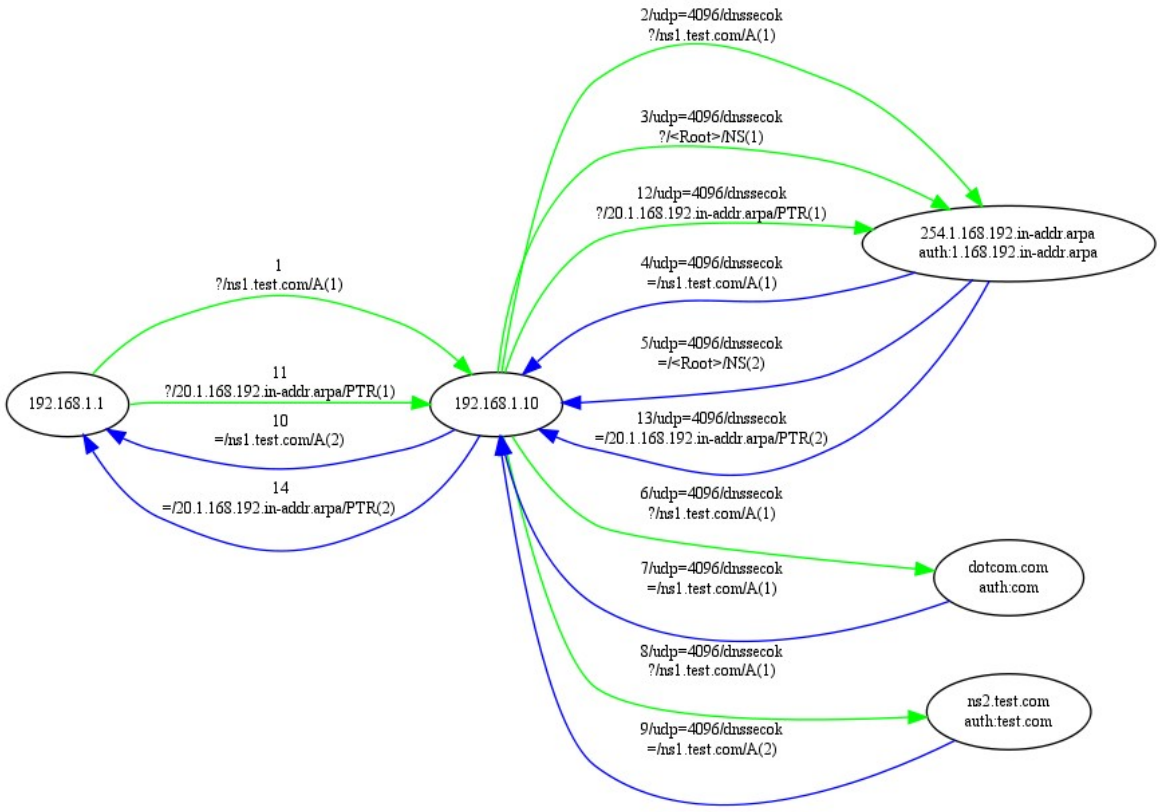


Figura 27 – Gráfico representativo de uma consulta DNS tradicional

Fonte: Aplicativo dnspktflow (dnssec-tools)

As linhas em verde representam as consultas (*queries*) enquanto as linhas em azul representam a resposta à consulta solicitada (*answers*). A seguir, realizamos novamente esta mesma consulta, como forma de verificarmos a quantidade de *frames* após a resolução DNS ter sido armazena em cache:

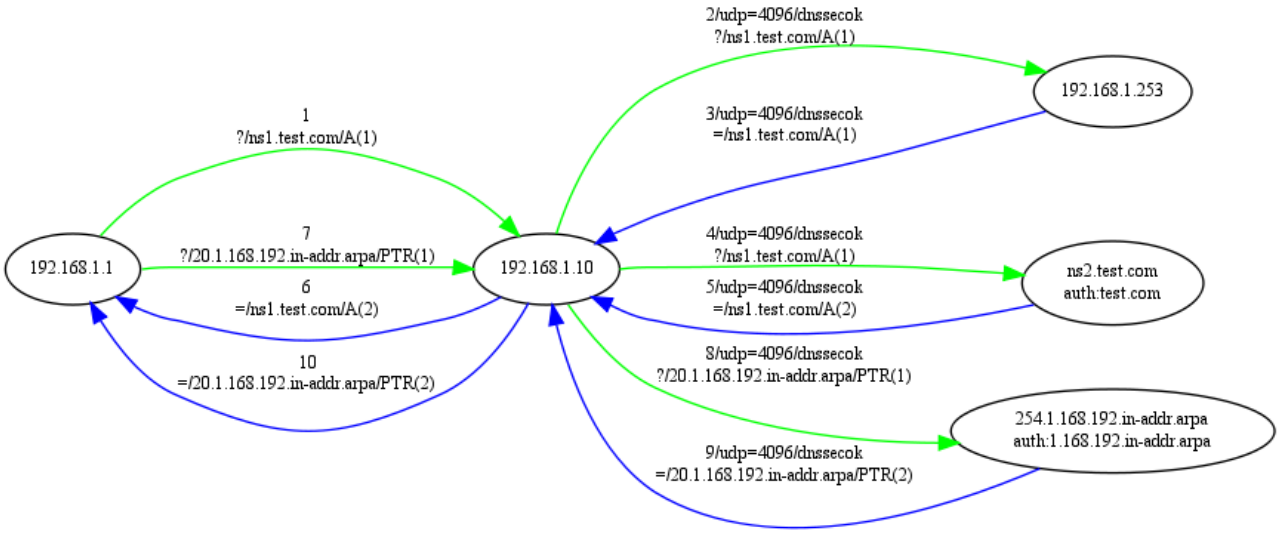


Figura 28 – Gráfico representativo de uma consulta DNS tradicional após o *cache*

Fonte: Aplicativo dnspktflow (dnssec-tools)

Concluímos que após a realização do cache, foram necessários apenas 10 *frames* para realizar a mesma consulta.

4.3. Emulando consultas DNSSEC

Para nossa simulação, configuramos o DNSSEC nos servidores autoritativos master e slave do domínio test.com e no servidor autoritativo do domínio com (neste caso, o servidor dotcom.com).

Além disso, configuramos o servidor recursive para verificar a autenticidade das consultas DNS: ancoramos-o com a chave pública (DNSKEY) do domínio com, como forma de criarmos uma cadeia de confiança. A esta chave que foi ancorada no servidor recursivo denominamos *trusted-key*.

Observe na figura a seguir o resultado desta consulta, simulação obtida de forma semelhante a consulta feita anteriormente:

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.1.1	192.168.1.10	DNS	Standard query A ns1.test.com
2	0.023364	192.168.1.10	192.168.1.254	DNS	Standard query A ns1.test.com
3	0.023377	192.168.1.10	192.168.1.254	DNS	Standard query NS <Root>
4	0.027034	192.168.1.254	192.168.1.10	DNS	Standard query response
5	0.027037	192.168.1.254	192.168.1.10	DNS	Standard query response NS ROOT-SERVER
6	0.030486	192.168.1.10	192.168.1.253	DNS	Standard query A ns1.test.com
7	0.033174	192.168.1.253	192.168.1.10	DNS	Standard query response
8	0.039363	192.168.1.10	192.168.1.20	DNS	Standard query A ns1.test.com
9	0.040253	192.168.1.20	192.168.1.10	DNS	Standard query response A 192.168.1.20 RRSIG
10	0.071460	192.168.1.10	192.168.1.21	DNS	Standard query DNSKEY test.com
11	0.078430	192.168.1.10	192.168.1.253	DNS	Standard query DNSKEY com
12	0.079353	192.168.1.253	192.168.1.10	DNS	Standard query response DNSKEY RRSIG
13	0.091105	192.168.1.10	192.168.1.1	DNS	Standard query response A 192.168.1.20
14	0.099400	192.168.1.1	192.168.1.10	DNS	Standard query PTR 20.1.168.192.in-addr.arpa
15	0.103136	192.168.1.10	192.168.1.254	DNS	Standard query PTR 20.1.168.192.in-addr.arpa
16	0.103978	192.168.1.254	192.168.1.10	DNS	Standard query response PTR ns1.test.com
17	0.107150	192.168.1.10	192.168.1.1	DNS	Standard query response PTR ns1.test.com

Figura 29 – Captura de uma consulta DNSSEC
Fonte: Aplicativo wireshark

Em linhas gerais, a mudança perceptível foi na solicitação feita do servidor recursive (192.168.1.10) ao servidor dotcom.com (192.168.1.253) da sua DNSKEY. O servidor recursivo solicita a DNSKEY da zonas filhas até o momento em que ele encontra um DNSKEY igual a presente em sua configuração (a *trusted key*). Este fato fica bastante evidente no momento em que verifica-se que o servidor recursivo solicita a DNSKEY de test.com e logo após solicita a DNSKEY do domínio com. Após isso, o servidor responsável pela zona *com* envia a sua DNSKEY e a RRSIG. Com a DNSKEY o servidor recursivo é capaz de checar se a assinatura (RRSIG) do *record* DS (*Delegation Signer*) é

válida. Caso seja, ele continua a realizar requisições. Da mesma forma o servidor recursive faz a verificação dos outros RRs (*resource records* do tipo nome para endereço IP – *A record*). Através destes mecanismos, o servidor *recursive* é capaz de dizer se este servidor DNS é válido ou não.

Verifique agora a mesma captura, só que de forma gráfica, forma esta gerada pelo software dnspktflow:

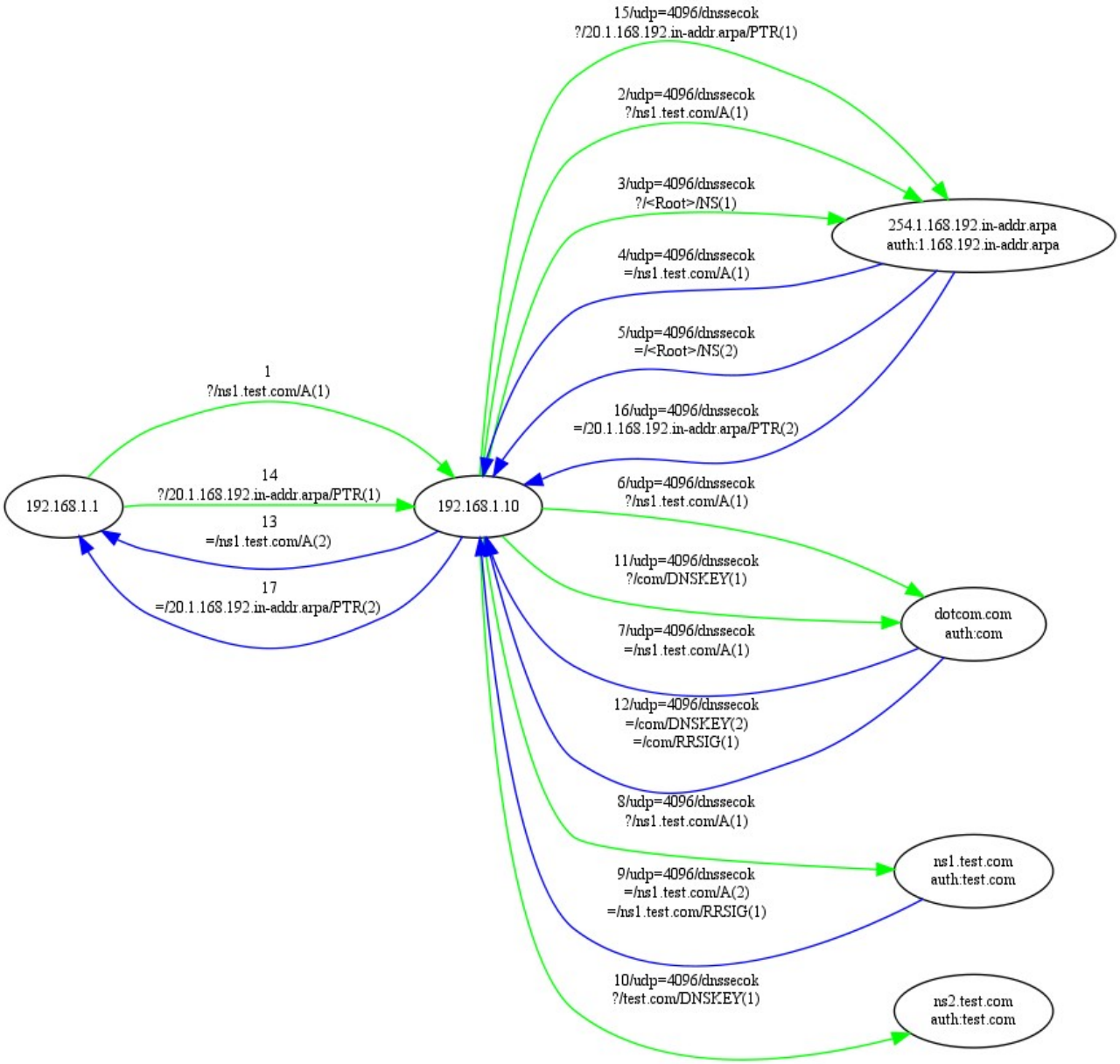
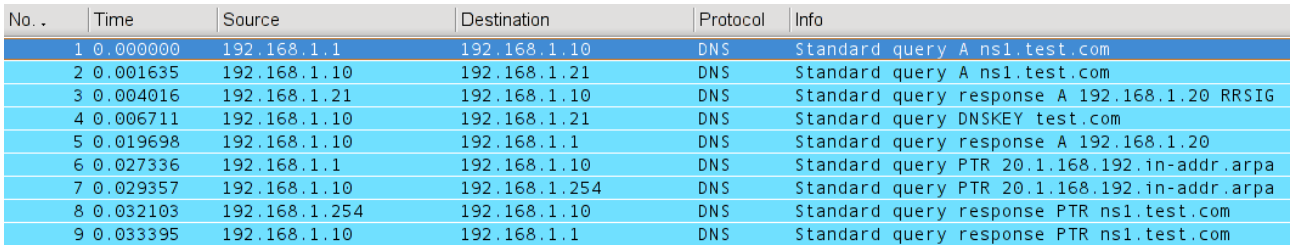


Figura 30 – Gráfico representativo de uma consulta DNSSEC

Fonte: Aplicativo dnspktflow (dnssec-tools)

Vejamos agora a mesma consulta DNSSEC “cacheada” pelo servidor recursive:



No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.1.1	192.168.1.10	DNS	Standard query A ns1.test.com
2	0.001635	192.168.1.10	192.168.1.21	DNS	Standard query A ns1.test.com
3	0.004016	192.168.1.21	192.168.1.10	DNS	Standard query response A 192.168.1.20 RRSIG
4	0.006711	192.168.1.10	192.168.1.21	DNS	Standard query DNSKEY test.com
5	0.019698	192.168.1.10	192.168.1.1	DNS	Standard query response A 192.168.1.20
6	0.027336	192.168.1.1	192.168.1.10	DNS	Standard query PTR 20.1.168.192.in-addr.arpa
7	0.029357	192.168.1.10	192.168.1.254	DNS	Standard query PTR 20.1.168.192.in-addr.arpa
8	0.032103	192.168.1.254	192.168.1.10	DNS	Standard query response PTR ns1.test.com
9	0.033395	192.168.1.10	192.168.1.1	DNS	Standard query response PTR ns1.test.com

Figura 31 – Captura de uma consulta DNSSEC

Fonte: Aplicativo wireshark

Percebe-que a quantidade de frames diminuiu consideravelmente em relação a primeira consulta DNSSEC. Verificamos que o servidor recursive solicita apenas a DNSKEY, como forma de verificar a autenticidade do *resource record* do tipo “A” recebido anteriormente. Portanto, o servidor recursive não verifica novamente a cadeia de confiança (*trust chain*) devido a esta ter sido verificada no momento da primeira consulta.

O quadro a seguir descreve de forma sucinta as principais diferenças observadas entre as capturas de tráfego DNS e DNSSEC:

Quadro 04 – Diferenças observadas entre as capturas de tráfego DNS e DNSSEC

Descrição	**Consultas DNS (sem a realização de cache)**	**Consultas DNSSEC (sem a realização de cache)**
Tamanho da consulta (aproximadamente, em bytes)	Entre 28 e 41 bytes	Aproximadamente 35 bytes
Tamanho da resposta (aproximadamente, em bytes)	Entre 65 e 105 100 bytes	Entre 990 e 1200 bytes (envio da DNSKEY e/ou RRSIG)
Tempo transcorrido desde a primeira *query* DNS até o recebimento do primeiro pacote ICMP	0,061 segundos	0,107 segundos
Quantidade de frames desde a primeira query DNS até o recebimento do primeiro pacote ICMP	14 frames	17 frames

Fonte: Aplicativo wireshark

4.4. Verificando a eficácia do DNSSEC

4.4.1. Autenticidade das Consultas DNS

Neste momento verificaremos a autenticidade das consultas DNS. Para isso, utilizaremos o utilitário *Domain Information Groper* – DIG. O DIG é uma ferramenta bastante eficaz para testarmos as configurações de um servidor DNS, através das várias opções de consultas (*queries*) que ele permite. O DIG faz parte do pacote do servidor DNS BIND e funciona tanto para o DNS tradicional bem como para o DNSSEC.

A consulta a seguir será feita ao servidor de nomes do domínio test.com, sendo esta realizada no servidor ***resolver***, com o intuito de verificar o seu suporte ao DNSSEC e a autenticidade das informações:

```
dig test.com +multiline +sigchase
```

A opção *multiline* formata a saída do comando DIG de tal maneira que fica mais fácil a sua interpretação e leitura, enquanto a opção *sigchase* traça a cadeia de assinaturas.

A saída a seguir foi fracionada intencionalmente para facilitar a formatação e a leitura. Observe o resultado encontrado:

```
;; NO ANSWERS: no more
We want to prove the non-existance of a type of rdata 1 or of the
zone:
We have a NSEC for this zone :OK
OK the NSEC said that the type doesn't exist
prove_nx: OK type does not exist
;; An NSEC prove the non-existence of a answers, Now we want validate
this NSEC
;; NO ANSWERS: no more
```

```
Launch a query to find a RRset of type DNSKEY for zone: test.com.

;; DNSKEYset that signs the RRset to chase:
test.com.         0 IN DNSKEY 257 3 5 (
                  AwEAAbPDLGRs8txT5xiQptpWTRhdZNuJx/7lwgFKnJdg
                  2VKQu7Hm03jmgqnTrn5FgFNiV7eeTIJj1E/EU3sj5Fpz
                  2LCsJxpDr63uhfoxsIt7576iB3HPu0FcF/1cOy1+31K0
                  GPLEsVr0HNEEIO/U/eLY7Gnf+jvY+BEavBg35lwYDe5R
                  JCfpLRKyDyGFv4NPv6Sib4k7HQlT+EcDWRepL0MGeYZT
                  BbHviNF8fCqhsTXE3oz0GyAYQeRnnp2FyZtxlrTgEOP6
                  kov9hsPzKrAVeOoV4hwMMdO7DdMm8R8IfNkokuDz60nC
                  /hL4si6XmFQZ0LwdaXYRTPcj+YLR51FZZOGH/cs=
                  ) ; key id = 11383


;; RRSIG of the DNSKEYset that signs the RRset to chase:
test.com.         0 IN RRSIG DNSKEY 5 2 0 20090421225810 (
                  20090322225810 11383 test.com.
                  ngPq0096wzOg3Sxr9L1eLA3ww2/jSYhpk+t1bfJJdqRJ
                  WzMIPe1X2NR2/a6C9dWkGQYdKOHO7ECAhaC7QkXo+/SE
                  1dsPSi3CGnJyqBgmVYikVNngwGius86DmOYetvZwmsmG
                  t52t51zS+X1QMzzTYLw0WDUypiUVwXzT/MpVkNih2CrO
                  vMw7YziNv2Wyyyub2URvfH6uccp1V098ra9RX3ERc4yV
                  2YSZ09hFUyy6EIZzcrA2riDkluB6QJRiCG3IzPxJu1Vu
                  jkiVtsxLuKnoQ5mqux/bbnbNYNIfM4OqRs413AAqUDqr
                  WrfFlWEqSUfBxCDVdWmRX5ysPMJ3+gDWyg== )



Launch a query to find a RRset of type DS for zone: test.com.

;; DSset of the DNSKEYset
test.com.         59930 IN DS 11383 5 1 (
                  20DD5A0127A4C34C36189B8D707CFB27DEAC6A91 )
            59930 IN DS 11383 5 2 (
                  087C71E8B974A8844CBB692E6FB0999AFE668D8CC7BB
                  933BEA64374DEA86AB44 )


;; RRSIG of the DSset of the DNSKEYset
test.com.         59930 IN RRSIG DS 5 2 60000 20090421230814 (
                  20090322230814 61576 com.
                  mwqR6WI8NILJVlAQD3BLN1r+GEaH8ATr01JhhMDhmhqz
                  QRYdBLw/P2VILg7DaU1ij9L5iukw22V1mWInr+Qgewvt
                  5bg2rYIhiw8JMFRFmBE+faRvOsNz55tvl8HFAUSGWk7E
                  A8MC3hH1Ux62WeIoUhc71+dYepXT9nRhTkKa1XpN/rCp
                  feBsAIC3BI/p7KPw1eKvvKk9tBvYrpjl7QqXCvZoLSSS
                  S7AKn6I0E+PRqxRMBEWCYZULN7XxU4M6AOeTULjahcsN
                  XeDSIsz6MDgFHBS7H/TUGp4xCeB5TUuu73jJKrOFromo
                  Qy3VmdklTgnK6IXe0sjjFSDulOJb7s5yww== )
```

```
;; WE HAVE MATERIAL, WE NOW DO VALIDATION
;; VERIFYING NSEC RRset for test.com. with DNSKEY:11383: success
;; OK We found DNSKEY (or more) to validate the RRset
;; Now, we are going to validate this DNSKEY by the DS
;; OK a DS valids a DNSKEY in the RRset
;; Now verify that this DNSKEY validates the DNSKEY RRset
;; VERIFYING DNSKEY RRset for test.com. with DNSKEY:11383: success
;; OK this DNSKEY (validated by the DS) validates the RRset of the
DNSKEYs, thus the DNSKEY validates the RRset
;; Now, we want to validate the DS :  recursive call


Launch a query to find a RRset of type DNSKEY for zone: com.

;; DNSKEYset that signs the RRset to chase:
com.        59930 IN DNSKEY   257 3 5 (
                  AwEAAc/pZLb2FNqsZoiPIAn0keNXDBLwJpTM/EnNQ0xu
                  tHS+2BVpV+jwQu2o7L60PlcHOpbwNEGOYohAB8qod5nb
                  mizOmxfyZJ1Oao/RuzYXbBaIUl90ZvdKlB2g5m1XkG+W
                  umLttkyJbDnE7jfbr1cI3Dj15Du9DQuKKmVDq/6m01rU
                  EbNfpvnidnSiKsZsp0KuUnv45rPG7xzf6TxhCQTvpgxv
                  ZMOFi3PRhg0LU1xr6/m7BHPf3rChjznOITbn22fAR4R2
                  FmFhfAoivQbKfQmzPqXjrRJwpbyVNRxSD0fUKFAF3bzv
                  0wtQeYihPkuqRBLGOiZLIFDtXWiNtvqjRULTg5E=
                  ) ; key id = 61576


;; RRSIG of the DNSKEYset that signs the RRset to chase:
com.        59930 IN RRSIG DNSKEY 5 1 60000 20090421230814 (
                  20090322230814 61576 com.
                  G8GLpJoja7ybrfIVJSenvR1r2kLCMRF6BPfiFkClfuH5
                  ZQBGV+B2nNbxhbjY0pqWwhfluFCXckqL3VA3nSDVj0TR
                  /so2OyWPVQS2CkaHm3rk0j26nAdjE/IIFoLqD3zJ8wR4
                  bGZ/heMcIehUFHFLk/1ki+v7gsDA+ynAp4bKCJthMUKa
                  iNpY4cneCBygkQZ4yvbdU24g9nDdaqYvf8h4R3tN8hFn
                  FfFSaI5UJucWDfIGcaFt2BvBryGIzNB4xjz+e3n7dv//
                  QpSiZ+2blK5Nx2Tgoon2fSP19FJzc7JYGN3IMYtUMClp
                  A1hKVGX8rimIU6uuPzFfTwiarn38wSMR0A== )



Launch a query to find a RRset of type DS for zone: com.
;; NO ANSWERS: no more

;; WARNING There is no DS for the zone: com.

;; WE HAVE MATERIAL, WE NOW DO VALIDATION
;; VERIFYING DS RRset for test.com. with DNSKEY:61576: success
;; OK We found DNSKEY (or more) to validate the RRset
;; Ok, find a Trusted Key in the DNSKEY RRset: 61576
;; VERIFYING DNSKEY RRset for com. with DNSKEY:61576: success

;; Ok this DNSKEY is a Trusted Key, DNSSEC validation is ok: SUCCESS
```

Em um primeiro momento o aplicativo DIG testa o registro NSEC, fazendo uma consulta de um nome não existente, lembrando que o RR NSEC é capaz de trazer esta resposta – negativa caso um nome não exista em um domínio.

A seguir o DIG solicita a chave pública (DNSKEY) do domínio test.com e logo após a assinatura da DNSKEY (RRSIG). De posse destes dois *resources records*, o DIG é capaz de verificar se a assinatura é autêntica.

Com o *record* DS, o DIG verifica a cadeia de confiança, e continua a realizar o processo descrito anteriormente (solicitar a DNSKEY e o RRSIG da DNKEY) até o ponto em que a t*rusted key* armazenada no servidor recursivo seja idêntica a encontrada no domínio consultado.

4.4.2 Modificando o arquivo da zona de forma proposital

Agora modificaremos o arquivo assinado da zona test.com no servidor master. O nome joaopessoa.test.com atualmente aponta para o IP 192.168.1.21. Com a alteração, o IP deste nome será 192.168.1.31.

Antes disso, executamos a consulta com o comando dig, da mesma forma da consulta anterior, alterando apenas para consultar o nome joaopessoa.test.com. O resultado da consulta foi satisfatório e afirmou que os dados presentes são autênticos. Veremos agora com a alteração manual no arquivo db.test.com.signed:

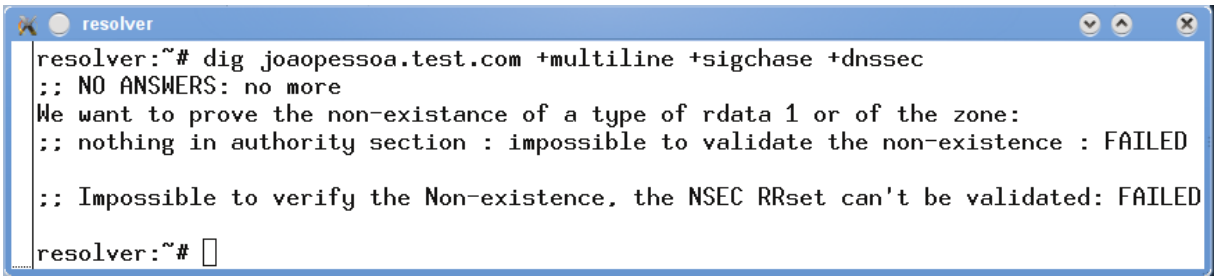


Figura 28 – Saída do comando DIG após modificação do arquivo do domínio test.com

Apenas para fins de compreensão, após a alteração realizada no arquivo da zona, foi necessário reiniciar o BIND no servidor master; não apenas para que o serviço fosse recarregado, mas sim com o objetivo de que a alteração realizada tivesse efeito. Sem o reinício, por mais que alterássemos o arquivo da zona, o nome joaopessoa.test.com sempre seria resolvido para o endereço IP antigo (192.168.1.21).

O interessante é quando tentamos disparar pacotes ICMP (com o utilitário PING) contra joaopessoa.test.com. Com o conhecimento adquirido neste estudo, teoricamente, este nome deveria corresponder ao endereço IP 192.168.1.31. Porém a saída apresentada não correspondeu a esperada. Observe na figura a seguir o resultado:

Figura 29 – Saída do comando PING após modificação do arquivo do domínio test.com

O nosso servidor recursivo não foi capaz de resolver o nome, justamente devido ao fato de termos alterado o endereço IP de joaopessoa.test.com. Com isso fica claro que a mínima alteração no arquivo da zona resulta em uma resposta negativa para o nome solicitado.

A figura abaixo foi gerada através do aplicativo dnspktflow, figura esta que representa esta última consulta mal-sucedida devido a modificação intencional do arquivo da zona, com detalhe (em vermelho) para a resposta proveniente do servidor recursive fornecida ao resolver:

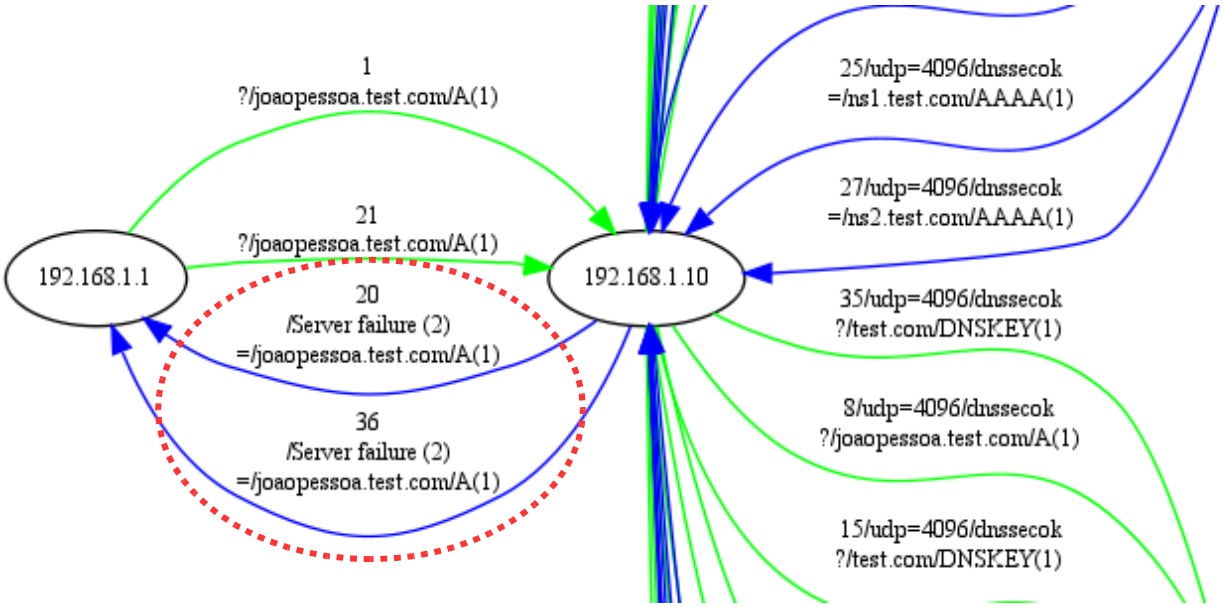


Figura 30 – Gráfico de uma consulta DNSSEC após alteração do arquivo da zona (propositalmente)

Fonte: Aplicativo dnspktflow (dnssec-tools)

4.4.3 Modificando a hora do servidor para burlar a expiração das chaves

As chaves ZSK e KSK são as chaves responsáveis por assinar os registros da zona e assinar toda uma zona, respectivamente. Porém, estas chaves possuem uma data de expiração que, caso não seja informada no momento da criação, por padrão, é de trinta dias. Portanto, como forma de verificarmos este mecanismo, esperamos o prazo de expiração das chaves acontecer, e logo após esta expiração, realizamos o experimento de modificamos intencionalmente (atrasarmos) a hora nos computadores emulados até uma data que estivesse dentro do prazo de validade das chaves, e com isso, verificarmos se o DNSSEC consideraria estas chaves como válidas (apesar delas terem expirado e apenas termos modificado a hora dos computadores).

A figura 31 representa uma consulta DNSSEC realizada logo após a realização do procedimento descrito:

No.	Time	Source	Destination	Protocol	Info
1	0.000000	192.168.1.1	192.168.1.10	DNS	Standard query A ns1.test.com
2	0.023655	192.168.1.10	192.168.1.254	DNS	Standard query A ns1.test.com
3	0.023665	192.168.1.10	192.168.1.254	DNS	Standard query NS <Root>
4	0.026196	192.168.1.254	192.168.1.10	DNS	Standard query response
5	0.027842	192.168.1.254	192.168.1.10	DNS	Standard query response NS ROOT-SERVER
6	0.033900	192.168.1.10	192.168.1.253	DNS	Standard query A ns1.test.com
7	0.035950	192.168.1.253	192.168.1.10	DNS	Standard query response
8	0.041928	192.168.1.10	192.168.1.21	DNS	Standard query A ns1.test.com
9	0.043352	192.168.1.21	192.168.1.10	DNS	Standard query response A 192.168.1.20
10	0.044041	192.168.1.10	192.168.1.1	DNS	Standard query response A 192.168.1.20
11	0.056779	192.168.1.1	192.168.1.10	DNS	Standard query PTR 20.1.168.192.in-addr.arpa
12	0.059836	192.168.1.10	192.168.1.254	DNS	Standard query PTR 20.1.168.192.in-addr.arpa
13	0.060649	192.168.1.254	192.168.1.10	DNS	Standard query response PTR ns1.test.com
14	0.061214	192.168.1.10	192.168.1.1	DNS	Standard query response PTR ns1.test.com

Figura 31 – Captura de uma consulta DNSSEC após a alteração da data do servidor

Fonte: Aplicativo Wireshark

Como fica evidente, a consulta procedeu-se normalmente e a chave foi validada. Com esse último experimento é fácil observarmos que um *hacke*r pode fazer com que um domínio/zona fique indisponível simplesmente alterando a hora do servidor (atrasando ou adiantando a data). No primeiro momento, parece algo simplório, porém para um administrador de redes relapso, este fato pode passar despercebido e causar uma indisponibilidade no serviço.

5. CONCLUSÕES

O DNSSEC possui uma proposta bastante interessante para solucionar alguns dos problemas do *Domain Name System*. Baseado na criptografia de chaves públicas e assinaturas digitais, o DNSSEC vêm tornando-se um padrão na internet e quase uma obrigação adotá-lo.

Porém, por este mesmo fato dele utilizar técnicas de criptografia, o mesmo torna-se susceptível a falhas de segurança decorrentes dos algoritmos utilizados por ele (por exemplo, o SHA-1). Os algoritmos de *hash* podem sofrer colisões (ou seja, mensagens distintas podem gerar *hashes* iguais), fato este que recai sobre técnicas de ataques conhecidas por “ataques de aniversário”. Diversos autores comentam que é praticamente impossível (computacionalmente) encontrar colisões no SHA-1; contudo, diversas coisas consideradas impossíveis computacionalmente estão tornando-se possíveis devido aos avanços da tecnologia.

Um outro fator inerente ao DNSSEC é a dificuldade de configurá-lo em uma zona/domínio: gerar chaves, assinar zonas, adicionar registros na zona pai, *trusted-keys*, dentre outras tarefas tornam a implementação do DNSSEC uma tarefa não muito simples. Como era de esperar-se, algo que não seja tão simples de implementarmos recai sobre os conhecidos erros de configuração, que por sua vez, podem ao invés de solucionar problemas, podem na verdade criar novos. Percebemos esta dificuldade ao criarmos o ambiente de simulação.

Questões de performance devem ser observadas e abrem espaços para novas pesquisas. O procedimento de criptografar/decriptografar, verificar chaves e assinaturas é algo que possivelmente gera impactos no desempenho de uma rede e também nos servidores DNS. Partindo do pressuposto que operações envolvidas na criptografia de uma forma geral são funções e cálculos matemáticos, estas operações produzem impactos de processamento nos servidores que *a priori* não foram observadas suas conseqüências. Este é um tópico para pesquisas futuras e foge do escopo deste trabalho.

Muitos autores na internet estão céticos quanto a eficácia do DNSSEC. Questões como a necessidade todos os servidores DNS na internet necessitarem implementar o DNSSEC e sua ineficácia a ataques de negação de serviço são fatores bastante questionados nos fóruns de discussão, bem como entre os pesquisadores.

Até que as questões citadas nesta conclusão, questões estas observadas no desenrolar deste trabalho sejam dirimidas, o DNSSEC é uma das poucas alternativas

disponíveis para assegurarmos um nível aceitável de segurança nos servidores DNS e, portanto, devemos adotá-lo como forma de proteger-nos contra a crescente onda de ataques a este serviço, que sem dúvida nenhuma, é primordial na internet.

6. REFERÊNCIAS BIBLIOGRÁFICAS

TANENBAUM, Andrew. **Redes de Computadores**. 4. ed. Rio de Janeiro: Campus Elsevier, 2003.

KUROSE, James F.; **ROSS**, Keith W. **Redes de Computadores e a Internet – Uma Nova Abordagem**. 3 ed. Rio de Janeiro: Pearson Addison Wesley, 2006.

***INTERNET CORPORATION FOR ASSIGNED NAMES AND NUMBERS* – ICANN**. **ICANN destaca vulnerabilidade do Sistema de Nomes de Domínio e fornece ferramentas**: ICANN chama a atenção para importante problema de segurança na Internet. Disponível em <http://www.icann.org.br/announcements/announcement-06aug08.htm>. Acesso em: 30 Nov. 2008.

CENTRO DE ATENDIMENTO A INCIDENTES DE SEGURANÇA – CAIS. **Vulnerabilidade no ISC BIND e outras implementações de DNS**. Disponível em <http://www.rnp.br/cais/alertas/2008/uscert-vu8000113.html>. Acesso em: 05 Dez. 2008.

KOZIEROK, Charles M. **The TCP/IP Guide**. Disponível em <http://www.tcpipguide.com/free/t_DNSMessageHeaderandQuestionSectionFormat.htm >. Acesso em: 19 Fev. 2009.

VAZ MONTEIRO, Ricardo. **Tudo sobre registro de domínio**: DNS e DNSSEC. Disponível em <http://ricardovazmonteiro.blogspot.com/2007/10/dns-e-dnssec.html>. Acesso em: 07 Jan. 09.

ALBITZ, Paul; **LIU**, Cricket. **DNS e BIND**, 4ª ed., Rio de Janeiro: Campus, 2001.

DE CAMPOS, David Robert Camargo; **JUSTO**, Rafael Dantas. **Tutorial DNSSEC**. Versão 1.4.4. Disponível em <ftp://ftp.registro.br/pub/doc/tutorial-dnssec.pdf>. Acesso em 11 Fev. 2009.

ROOT-SERVERS.ORG. **Root Server Technical Operation Assn**. Disponível em <http://www.root-servers.org>. Acesso em: 19 Jan. 09

TURNBULL, James. **Hardening Linux**. [s.l]: Apress, 2005.

GREEN, Ian. **DNS Spoofing by The Man In The Middle** – GSEC Practical Assignment. Versão 1.4c. Disponível em <http://www.sans.org/reading_room/whitepapers/dns/dns_spoofing_by_the_man_in_the_middle_1567?show=1567.php&cat=dns>. Acesso em: 20 Jan. 09.

OPEN WEB APPLICATION SECURITY PROJECT – OWASP. **Main in The Middle Attack.** Disponível em <http://www.owasp.org/index.php/Man-in-the-middle_attack>. Acesso em 19 Fev. 09.

IDGNOW. **Briga entre sites mostra novo ataque que explora bug em servidores DNS**. Disponível em <http://idgnow.uol.com.br/seguranca/2009/02/05/briga-entre-sites-mostra-novo-ataque-a-servidores-dns-da-internet>. Acesso em: 14 Fev. 09.

VAUGHN, Randal; **EVRON**, Gadi. **DNS Amplification Attacks** – Preliminary. Disponível em <http://www.isotf.org/news/DNS-Amplification-Attacks.pdf>. Acesso em: 16 Fev. 09.

VIXIE, et al. **RFC 2136** - **Dynamic Updates in the Domain Name System (DNS UPDATE**). Disponível em <http://www.faqs.org/rfcs/rfc2136.html>. Acesso em: 24 Fev. 09.

OPENDNS – **About OpenDNS**. Disponível em <http://www.opendns.com/img/opendns_info.pdf>. Acesso em: 25 Fev. 09

APWG (Anti-Phishing Work Group) – **Phishing Activity Trends**: Report For The Month of May, 2007. Disponível em <http://www.antiphishing.org/reports/apwg_report_april_2007.pdf>. Acesso em: 25 Fev. 09.

VIXIE, et al. **RFC 2845** - **Secret Key Transaction Authentication for DNS (TSIG)**. Disponível em http://www.faqs.org/rfcs/rfc2845.html. Acesso em 24 Fev. 2009.

ATKINS, D.; **AUSTEIN**, R. **RFC 3833 – Threat Analysis of The Domain Name System (DNS)**. Disponível em ftp://ftp.rfc-editor.org/in-notes/rfc3833.txt. Acesso em 24 Fev. 2009.

STALLINGS, William. **Criptografia e Segurança de Redes**: Princípios e Práticas. 4 ed. São Paulo: Pearson Prentice Hall, 2008.

ARENDS, R. et all. **RFC 4034 – Protocol Modifications for the DNS Security Extensions**. Disponível em http://www.ietf.org/rfc/rfc4034.txt. Acesso em 24 Fev. 2009.

BURITI, George Luiz Cardoso. **Extensões de Segurança para o DNS, 2006**. (Monografia – Curso de Especialização em Segurança da Informação). Centro Federal de Educação Tecnológica da Paraíba – CEFETPB, Paraíba.

HUSTON, Geoff. **DNSSEC – The Theory**. Disponível em http://www.potaroo.net/papers/isoc/2006-08/dnssec.pdf . Acesso em 28 Fev. 2009.

ARENDS, R. Et all. **RFC 4035 – Resource Records for the DNS Security Extensions**. Disponível em http://www.ietf.org/rfc/rfc4034.txt. Acesso em 24 Fev. 2009.

RIMONDINI, Massimo Et All. **Netkit**: The poor man's system for experimenting computer networking. Versão 2.2. Universidade de Roma, 2007. Disponível em http://www.netkit.org/netkit-labs/netkit_introduction/netkit-introduction.pdf. Acesso em 05 Nov. 2008.

WIRESHARK: Go Deep. Versão 1.0.4. Disponível em http://www.wireshark.org/download.html. Acesso em 09 Out. 2009

DNSSEC-TOOLS: The dnssec-tools projetct. Versão 1.5. Disponível em http://www.dnssec-tools.org/wiki/index.php/Installing_DNSSEC-Tools. Acesso em 20 Abr. 2009