# Trabalho Final da Disciplina de Segurança da Dados: Ataques de envenenamento de cache DNS(*DNS cache poisoning*)

**Pedro Rogério Vieira Dias**[1]
**Bacharelando em Ciência da Computação**

[1]Departamento de Ciência da Computação - Universidade de Brasília (UnB)
Brasília - DF - Brasil

pedrorvd@gmail.com

***Resumo.*** *Esse trabalho busca descrever o ataque de DNS cache poisoning, um ataque que insere registros de DNS falso nos servidores de nomes DNS e que permite ao atacante redirecionar usuários para aplicações hospedadas no servidor sob seu domínio.*

## 1. Introdução

O uso de nomes ao invés de números para referenciar endereços web é essencial para que o uso da internet pela sociedade seja uma experiência fácil e intuitiva. Todos estamos acostumados com os endereços web, que são fáceis de se lembrar e intuitivos. Porém a internet não funciona com nomes canônicos e sim com números de endereço IP para referenciar servidores e usuários na rede. O tradução entre o nome intuitivo e o endereço IP é feito por um sistema e protocolo da camada de aplicação chamado DNS, *Domain Name System*, cujos serviços são essenciais para a internet ser uma fonte de informações mais acessível. Praticamente todas as aplicações usadas na internet utilizam os serviços providos pelo DNS, e isso o torna uma atração para atacantes e pessoas de má fé.

Nesse trabalho o autor busca descrever um ataque que se baseia em uma vulnerabilidade do sistema DNS que permite ao atacante redirecionar clientes de um site, por exemplo um site de banco, para uma cópia deste no servidor sob seu domínio. Dessa maneira o atacante pode roubar informações e distribuir *malwares*, por exemplo. Essa vulnerabilidade e ataque recebe o nome de DNS *cache poisoning*, ou envenenamento de cache DNS.

Primeiramente o autor apresenta uma breve explicação do protocolo e sistema DNS, focando nos aspectos necessários para a compreensão das vulnerabilidades e ataque em questão. Ao final apresenta a descrição do ataque, sua ideia, as características das implementações DNS que permitem um ataque, as possíveis consequências de um ataque e termina com exemplos de como ele é procedido.

## 2. Protocolo e Sistema DNS

Nessa seção descrevo de maneira geral o funcionamento do DNS, os principais tipos de registros e o formato das mensagens, focando nos aspectos necessários para a compreensão da vulnerabilidade de DNS *cache poisoning*. Finalizamos apresentando um exemplo de requisição DNS.

O DNS, *Domain Name System*, é um sistema e protocolo da camada de aplicação cuja tarefa principal é traduzir o nome de domínio de um hospedeiro(*host*) para o seu

endereço IP. É um sistema pois sua estrutura é um grande banco de dados distribuído implementado em uma hierarquia de servidores de nome espalhados pelo mundo, e um protocolo pois permite que os hospedeiros consultem essa grande estrutura. Vários protocolos e aplicações da internet (HTTP. SMTP e FTP) utilizam os serviços providos pelo DNS, isso o torna um protocolo essencial para o correto funcionamento das principais aplicações da rede [Kurose and Ross 2006]. Além de traduzir um endereço de domínio o DNS realiza a tradução de apelidos de hospedeiros para nomes canônicos[1] e tradução de apelidos de servidor de emails para nomes canônicos. Esses serviços facilitam o uso da internet pela sociedade uma vez que um nome é muito mais fácil de se memorizar que um número de vários dígitos.

A estrutura hierárquica do DNS é composta de 3 níveis: os servidores de nome raiz; os servidores de nomes de domínio de alto nível(TLD)[2] e os servidores de nomes com autoridade. Cada um dos níveis é responsável por indicar para o cliente qual servidor de nível inferior ele deve consultar para obter a informação desejada. Os servidores raiz indicam os servidores TLD, que indicam os servidores com autoridade para um domínio. O servidor com autoridade é que realmente responde a requisição do cliente e traduz o nome de endereço para um número IP. Além desses existe o servidor de nomes local que não faz parte estritamente da hierarquia mas é essencial para a arquitetura DNS. É ele quem irá responder e resolver para os usuários as perguntas DNS.

### 2.1. Registros de Recurso

Os servidores DNS armazenam de maneira distribuída os registros de recurso(RR) que fornecem mapeamentos de nomes de hospedeiros para endereços IP. As mensagem de resposta do DNS carregam um ou mais desses registros. Um registro é representado por uma tupla: $(Name, Value, Type, TTL)$. O $TTL$ indica o tempo de vida útil do registro, o tempo que ele ficará armazenado nos caches dos servidores DNS. O fato dos servidores DNS armazenarem em cache os registros mais recentes requisitados ajuda muito no desempenho da rede, uma vez que diminuem o atraso e a quantidade de mensagens que varrem a internet. Existem vários tipos de RR, os principais são [Kurose and Ross 2006]:

- *Type* A: registro onde o *name* é um nome de hospedeiro e *value* é o respectivo endereço IP.
- *Type* NS: registro onde o *name* é um nome é um domínio e *value* é o nome de um servidor de nomes com autoridade que sabe como obter os endereços IP para hospedeiros do domínio. Esse tipo é usado para encaminhar consultas DNS para outros servidores.
- *Type* CNAME: registro onde o *name* é um apelido de hospedeiro e *value* é o respectivo nome canônico.
- *Type* MX: registro onde o *name* é um apelido de servidor de email e *value* é o respectivo nome canônico.

### 2.2. Campos da Mensagem

Os campos de uma mensagem DNS são ilustrados na figura 1, os mais relevantes para a compreensão do ataque descrito nessa artigo estão coloridos de amarelo. Abaixo faço alguns comentários sobre alguns campos importantes[Friedl's 2008]:

[1] Um hospedeiro com um nome complicado pode ter um ou mais apelidos. Por exemplo, um nome como *mercurio.andes.cic.unb.br*, que chamamos de nome canônico, pode ter um apelido *www.andes.cic.unb.br*.

[2] Domínios de alto nível são por exemplo domínios de países tais como .br, .uk, e .fr e outros como .edu, .org e .com

- **Endereço de origem / destino**: Esse campo indica o endereço IP das máquinas que enviam e recebem o pacote. É possível forjar o campo de origem, mas não tem fundamento forjar o campo de destino, uma vez que ele indica onde o pacote vai e isso mudaria o destino do seu pacote(no contexto do ataque em questão).
- **Porta de origem / destino**: Um servidor DNS escuta por requisições na porta UDP 53, assim o primeiro pacote de uma requisição será sempre destinado para essa porta. O número da porta origem não é padronizado e varia muito, em algumas implementações é fixo, em outras é escolhido aleatoriamente. Veremos mais a frente que uma implementação que utiliza números de porta de origem fixos são mais vulneráveis ao ataque de *cache poisoning*.
- **Query ID**: é um identificador único criado para o pacote de requisição DNS no lado cliente, o lado servidor deve responder a requisição com o mesmo valor de ID. Isso permite que o o servidor que fez a requisição associe-a a uma resposta. Isso é importante pois possibilita ao servidor ter várias requisições pendentes ao mesmo tempo, dando paralelismo as requisições. Veremos adiante que esse campo é importante para o ataque aqui descrito.
- **Seção de Perguntas**: contém informações sobre a consulta sendo feita. As informações são: um campo com o nome que está sendo consultado e um campo com o tipo de registro que está sendo consultado. A quantidade de pergunta é indicado no campo número de perguntas.
- **Seção de Respostas**: quando a mensagem é uma resposta, contém os registros RR para o nome que foi consultado. A quantidade de respostas é indicada pelo campo número de RRs de resposta.
- **Seção de Autoridade**: essa seção é utilizada quando um servidor não possui autoridade para responder por um domínio e indica na resposta os RR de tipo NS que indicam os nomes dos servidores com autoridade. A quantidade de RRs com autoridade é indicada no campo de número.
- **Seção Adicional**: contém outros registros úteis, por exemplo: quando uma resposta inclui um registro NS que indica o próximo servidor a ser consulado, um RR tipo A para o servidor indicado é enviado na seção adicional. Isso faz com que o cliente que fez a requisição não precise enviar outra requisição perguntando pelo IP do próximo servidor a ser consultado.

### 2.3. Exemplo de requisição

Os passos de uma requisição DNS são ilustrados na figura 2. Nesse exemplo o cliente deseja consultar o endereço para o domínio *www.exemplo.br*. Ele primeiramente pergunta para o seu servidor DNS local(passo 1); o servidor local pergunta para algum dos servidores DNS raiz(passo 2); que encaminha a requisição para o servidor TLD .br(passo3); os passos de requisições seguem até que o servidor local obtém a resposta do servidor de autoridade e encaminha para o cliente o registro com o endereço IP desejado.

Exitem dois tipos de consulta: a interativa e a recursiva. As consultas realizadas pelo servidor de nomes local para os servidores raiz, TLD e com autoridade são consultas iterativas. As consultas feitas por um cliente para o seu servidor de nomes local é chamada de recursiva, pois o servidor local é quem resolve a requisição feita pelo cliente.

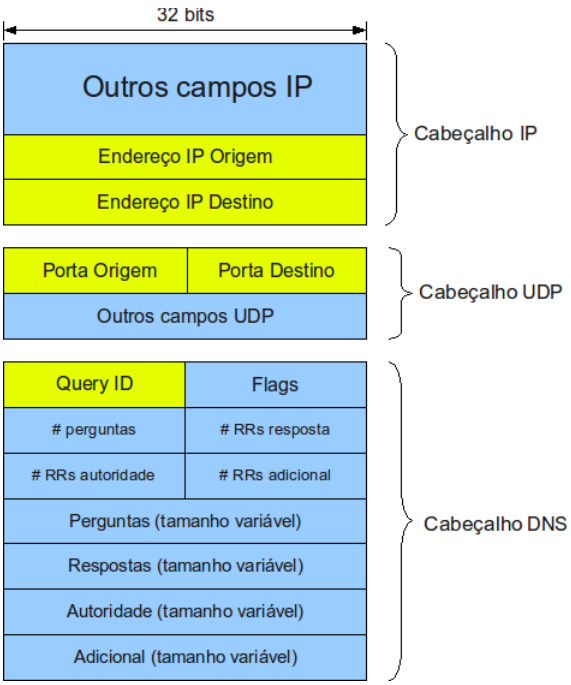


**Figura 1. Cabeçalho de uma mensagem DNS**

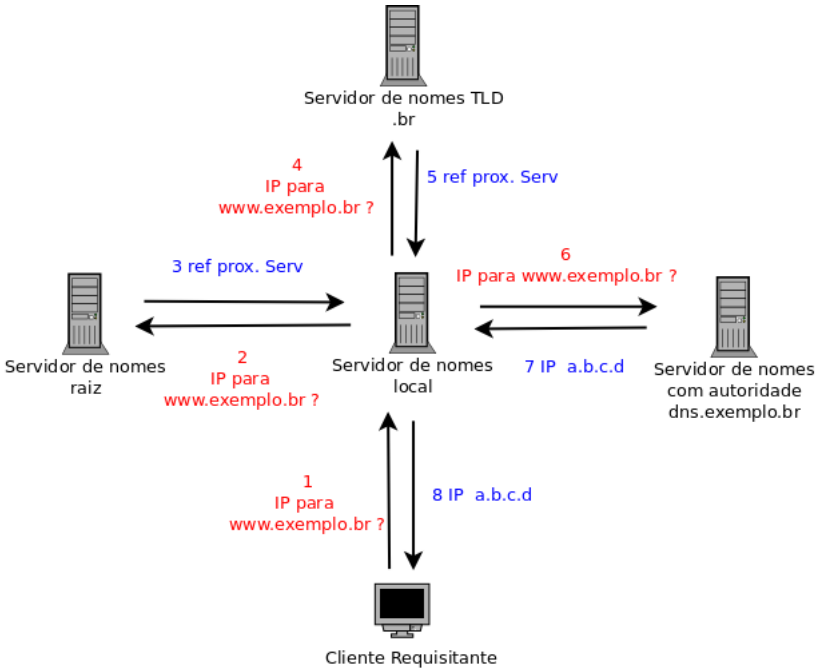


**Figura 2. Consulta DNS**

## 2.4. Cache DNS

Como dito acima uma consulta DNS exige uma grande quantidade de trocas de mensagens, por conta disso os servidores guardam os registro mais recentemente requisitados. Esses registros são armazenado nos caches por um tempo determinado pelo TTL do registro, esse TTL é definido pelo administrador do servidor de nomes com autoridade para o domínio do registro. Os registros do tipo NS também são armazenados em cache. Na

próxima seção veremos como uma atacante pode envenenar os caches de um servidor de nomes.

## 3. DNS Cache Poisoning

Entendendo o funcionamento do DNS podemos apresentar o ataque de DNS *cache poisoning*. Primeiramente apresento o que o ataque faz, a ideia geral do ataque e como um servidor de nomes aceita ou rejeita uma possível resposta a uma requisição feita, depois apresento as premissas para que o ataque funcione e com base nelas relato as vulnerabilidades que um servidor de nomes pode apresentar. No final descrevo as possíveis consequências de um ataque e finalizo descrevendo detalhadamente dois tipos de ataque de envenenamento de cache. O primeiro envenena um registro do tipo A, ou seja, ele frauda o endereço de algum host. O segundo é muito mais poderoso, ele envenena um registro do tipo NS, ou seja, ele pode fraudar todo um domínio de endereços pelo qual o servidor de autoridade que o registro NS apontava era responsável.

DNS *cache poisoning* ou envenenamento de cache de DNS é uma situação, provocada por algum atacante com mas intenções ou por algum imprevisto, em que algum registro que não foi enviado por um servidor de nomes com autoridade para esse registro é armazenado propositalmente ou acidentalmente no cache de registros de algum servidor de nomes. Quando essa situação acontece e o servidor de nomes está habilitado para fazer cache de registros de requisição(praticamente todos o fazem), o registro envenenado e não autêntico é repassado para os clientes desse servidor de nomes. Isso permite a um atacante redirecionar usuários de aplicações de Internet para um servidor sobre o seu domínio, pode por exemplo falsificar a interface de uma aplicativo de banco online ou loja online e roubar números de cartões de crédito e senhas bancárias, chamamos esse tipo de ataque de ataque de *Pharming* [Olzak 2006]. Chamamos de servidor envenenado o servidor que sofreu essa situação. Nesse trabalho focamos no caso em que o envenenamento ocorreu por conta de um ataque.

### 3.1. Ideia do ataque

O ideia geral do ataque é enviar uma resposta falsa fraudada(envenenada) para uma requisição pendente que o servidor de nomes a ser envenenado está tentando resolver, normalmente uma requisição feita pelo próprio atacante. A resposta falsa deve chegar e ser aceita pelo servidor de nomes antes que a resposta original e autêntica. Não é tão simples quanto parece enviar uma resposta falsa que seja aceita pelo servidor de nomes, uma vez que ele aceita somente respostas para requisições pendentes. Para o servidor aceitar uma resposta, falsa ou autêntica, ela deve obedecer as seguintes restrições de parâmetro, que chamamos de parâmetros chaves, caso contrario a mensagem será descartada:

- Chegar na mesma porta UDP pela qual a requisição foi enviada, caso contrario a pilha de protocolos não irá entregar o pacote para o processo servidor de nomes.
- O ID de requisição(*Query ID*) da resposta deve ser o mesmo da requisição pendente.
- O nome dos servidores com autoridade presentes nos RRs de autoridade da mensagem pertencem ao mesmo domínio do nome para o qual um endereço é requisitado. Isso impede por exemplo que um servidor de nomes com autoridade *www.exemplo.br* responda por um endereço como *www.bancoDoPedro.br*.

Se um atacante for capaz de forjar uma mensagem de resposta com essas restrições antes que a resposta autêntica chegue, ele pode envenenar o servidor de nomes e causar vários tipos de danos para os clientes desse servidor. Note que os pacotes não esperados serão simplesmente descartados, assim o atacante deve fazer várias tentativas de forjar uma resposta fazendo um *flooding* de respostas com variações dos parâmetros chaves, comentados acima. Quanto mais difícil para o atacante acertar os parâmetros chaves no limite de tempo disponível menos um servidor de nomes é vulnerável ao ataque.

Não são todos os servidores que são vulneráveis a esse tipo de ataque atualmente, mas a literatura mostra que vários já o foram, a exemplo: o BIND [Klein 2007], o Microsoft DNS Server [Bezroutchko 2007]. Faremos a frente uma análise das vulnerabilidades que um servidor de nomes pode apresentar para o ataque de *DNS cache poisoning*.

Para fazer um ataque o atacante deve encontra algum servidor vulnerável que ele deseje ter controle e fazer o *flooding* de respostas envenenadas no momento em que o servidor está resolvendo uma requisição de endereço. Paralelamente, ele pode também fazer um ataque de negação de serviço(DoS) no servidor com autoridade para a resposta que ele deseje fraudar, dessa maneira o atacante ganha mais tempo para acertar a combinação de parâmetros chaves correta. A sua ideia é enganar os clientes do servidor atacado, uma vez obtido o sucesso com o envenenamento os usuários do servidor de nomes envenenado estão a merce do atacante.

### 3.2. Premissas para que o ataque funcione

Para o atacante obter sucesso no *cache poisoning* as seguinte premissas devem ser satisfeitas [Friedl's 2008]:

- O registro RR do nome para o qual o atacante quer forjar não pode estar no cache do servidor de nomes sendo atacado, uma vez que todas as requisições para esse registro não ativarão o processo de resolução desse nome pelo servidor vítima. Assim o atacante não tem uma requisição pendente para envenenar. O atacante deve, dessa maneira, esperar o tempo TTL que o registro ficará no cache do servidor para tentar um ataque.
- O atacante deve acertar os valores dos parâmetros de ID de requisição e porta de origem, para que a resposta envenenada seja aceita.
- O atacante deve ser mais rápido que o servidor de nomes com autoridade para o RR a ser forjado. O atacante deve acertar os parâmetros antes que a resposta autêntica chegue no servidor de nomes sendo atacado.

### 3.3. Vulnerabilidades e chances do atacante

Nessa seção descrevo as características vulneráveis que uma implementação de um servidor de nomes DNS pode apresentar e como elas ajudam nas chances de sucesso do atacante no envenenamento de cache.

As vulnerabilidades estão relacionas principalmente na escolha do parâmetros chaves, o id de requisição e o número de porta origem, pelo servidor que faz a requisição, assumindo que o atacante tem conhecimento dos campos da pergunta que foi feita para a resposta que ele deseja fraudar. Se a escolha dos parâmetros for determinística o atacante pode facilmente deduzi-los e ter sucesso no envenenamento. Dessa maneira a escolha desses parâmetros deve ser aleatória, ou o mais próximo de aleatória possível. Um servidor

que usa números de porta origem e de IDs de requisição aleatórios mas possui um gerador de números pseudo-aleatórios previsível estão vulneráveis da mesma maneira. Esse tipo de vulnerabilidade foi observada por exemplo no servidor de nomes BIND [Klein 2007].

Uma outra vulnerabilidade é um servidor de nomes local que trabalhe em modo recursivo, aceitar requisições de qualquer tipo de cliente. Um servidor trabalha em modo recursivo quando é ele quem resolve qualquer requisição de endereço feita por um cliente. Isso permite que um atacante faça requisições nesse servidor para força-lo a resolver essa requisição e dessa maneira iniciar o processo de consulta DNS que permite ao atacante injetar o registro envenenado no cache do servidor [Davies 2008]. É interessante que o servidor de nomes local aceite somente requisições feitas por clientes que realmente precisem do serviço do servidor, isso pode ser feito pelo administrador do servidor de nomes que fixa os números de IP que podem fazer requisições, o que reduz a possibilidade de ataque.

Em antigas implementações o ID de requisição da mensagem DNS utilizado era simplesmente incrementado a cada nova requisição feita [Friedl's 2008]. Com esse tipo de escolha o atacante pode facilmente adivinhar o ID de requisição da próxima mensagem a ser utilizada. Uma maneira simples de fazer isso é: o atacante faz uma requisição ao servidor vítima para procurar por um endereço para um nome de domínio que ele controle; o servidor vítima irá procurar recursivamente pelo registro para o nome requisitado e no final do processo fará a pergunta para o servidor de nomes com autoridade sobe controle do atacante; o atacante agora simplesmente observa o campo de ID de requisição da mensagem enviada pelo servidor vítima, ele pode responder ou não a sua própria requisição. Se um servidor de nomes utiliza número de portas origem fixo o mesmo processo pode ser utilizado para obter o número de porta da requisição.

Considerando que o servidor de nomes utilize os número de porta origem e o ID de requisição aleatórios o suficiente, a ponto de impedir que o atacante utilize algum método de previsão desses parâmetros, o espaço de possibilidade de parâmetros que o atacante deve percorrer é: quantidade de IDs possíveis vezes quantidade de portas possíveis. A quantidade de IDs é $2^{16}$, uma vez que o campo *query ID* tem 16 bits. A quantidade de portas disponíveis é também na teoria de $2^{16}$ e na prática, na maioria dos sistemas operacionais, é de $2^{12}$, devido as restrições de portas que podem ser utilizadas [Klein 2007]. Como resultado temos um espaço de busca com tamanho de aproximadamente:

$$2^{16} \times 2^{12} = 2^{28} = 268.435.456$$

O uso de portas e IDs aleatórios reduz bastante a chance de sucesso de um ataque de envenenamento, porém não elimina a possibilidade de sucesso de ataque e pode ser visto como uma prevenção temporária para o problema, uma vez que esse espaço de busca pode ser facilmente percorrido por um atacante com alta capacidade computacional e de banda de rede. É essencial que um servidor de nomes local utilize tanto IDs aleatórios como portas aleatórias para prevenir esse tipo de ataque a seus clientes. O leitor pode utilizar uma ferramenta web disponibilizada pela DNS-OARC(*Domain Name Service Operantions Analysis and Reasearch Center*), que analisa automaticamente a aleatoriedade dos IDs e dos números de portas utilizados nas requisições do seu servidor de nomes local fornecido pelo seu ISP. A ferramenta está disponível no site da DNS-OARC

[DNS-OARC 2008] e permite uma análise visual das escolhas dos parâmetros chave.

A atual arquitetura do protocolo DNS está sujeita a esse tipo de ataque, a literatura aponta que a solução para um futuro próximo é a utilização do DNSSEC(*Domain Name System Security Extensions*)[Seifried 2010, Registro.br 2009, Jeremy Hitchcock and Vixie 2009]. O DNSSEC é uma extensão do DNS que permite a clientes verificar a origem(autenticação) e integridade de respostas a requisições DNS. Para isso o protocolo utiliza um sistema de assinaturas digitais implementadas com criptografia assimétrica e certificados digitais para distribuição de chaves públicas. Assim um servidor DNSSEC antes de responder a uma requisição assina com a sua chave privada a mensagem de resposta e o cliente ao receber uma mensagem verifica essa assinatura com a chave pública do servidor, impedindo que um atacante forje uma mensagem de resposta. É claro que o DNSSEC possui suas premissas de operação e que por ser um protocolo novo provavelmente está sujeito a descobertas de vulnerabilidades. Não é foco desse trabalho descrever o DNSSEC, mas é importante mencioná-lo como uma solução promissora para o ataque de envenenamento de cache.

### 3.4. Possíveis consequências do *Cache Poinsoning*

Existem uma série de consequências graves que um ataque de *Cache Poinsoning* pode causar, algumas delas são [Olzak 2006]:

- Roubo de identidade: quando um atacante tem o usuário sobe o domínio do seu servidor ele pode tentar enganar o usuário a deixar informações pessoais. Uma maneira de fazer isso é clonar um site que exija algum tipo de informação pessoal do usuário tais como sites de cadastro em conferências, sites de bancos online e sites de compras. Direcionando os usuários, com a informação do registro do cache envenenado, para o seu servidor da ao atacante acesso a informações como número de CPF, RG e até de cartão de crédito. [3]
- Distribuição de Malware: um atacante pode direcionar usuários a sites clonados que distribuem software malicioso que podem roubar informações da máquinas dos usuários ou causar algum dano ao seu sistema.
- Disseminação de informações falsas: um atacante pode distribuir informações falsas sobre alguma organização. Por exemplo divulgando preços e promoções falsas.
- *Man-in-the-middle Attack*: um usuário pode ser forçado a iniciar uma seção com o atacante, que inicia um seção com o servidor que o usuário deseja acessar, assim o atacante repassa as informações entre as vítimas. Dessa maneira o atacante tem acesso a todas as informações trafegando entre as vítimas.

### 3.5. Exemplo de envenenamento de cache com RR tipo A

Nessa seção descrevo, com um exemplo, como um atacante pode envenenar um servidor de nomes local com um registro de recurso do tipo A, ou seja, ele frauda o endereço de algum host. No exemplo o atacante deseja envenenar um servidor de nomes de algum ISP com um endereço IP falso para um site de banco online (`www.bancoDoPedro.com.br`), e dessa maneira redirecionar todos os clientes do ISP para a sua copia maliciosa do site do banco. O nome canônico

[3] Um ataque desse tipo ao banco brasileiro Bradesco foi reportado em abril de 2009 no portal de noticias online G1 da Globo [Rohr 2009]

do servidor DNS de autoridade responsável por `www.bancoDoPedro.com.br` é `dns.bancoDoPedro.com.br`. Os seguinte passos são realizados:

1. Primeiramente o atacante envia uma requisição para o servidor de nomes vítima requisitando pelo endereço para o nome `www.bancoDoPedro.com.br`. O servidor vítima irá resolver recursivamente esse nome, fazendo perguntas pra o servidor raiz, TLD e finalmente de autoridade para o nome requisitado.
2. O atacante sabendo que a vítima, na cadeia de requisições DNS, irá perguntar pelo endereço do banco para o servidor DNS com autoridade para ele `dns.bancoDoPedro.com.br`, começa a inundar(*flooding*) o servidor vítima com mensagens de resposta com registros falsos para o nome `www.bancoDoPedro.com.br` com o IP do servidor sobe seu domínio. Ele tenta uma série de combinações de parâmetros chaves que tentam bater com os que foram utilizados pelo servidor vítima e uma hora acerta.
3. A resposta falsa vinda do atacante chega antes da resposta autêntica vinda do servidor `dns.bancoDoPedro.com.br`. A resposta autêntica é descartada pela vítima, já que a requisição já foi respondida, e a resposta falsa é guardada no cache.
4. O servidor de nomes vítima repassa o registro falso para os seus usuários.

Esse ataque é bastante viável se o servidor de nomes vítima apresentar as vulnerabilidades descritas na seção 3.3. Como o servidor de nomes aceita como resposta para a requisição a primeira resposta bem formada, basta para o atacante acertar os parâmetros em uma das várias mensagens enviadas no *flooding* antes que a resposta autêntica chegue. É uma corrida que só o atacante sabe que esta acontecendo.

### 3.6. Exemplo de envenenamento de cache com RR tipo NS

Nessa seção descrevo, com um exemplo, um ataque mais maduro e mais danoso, em que o atacante envenena um servidor de nomes local com um registro de autoridade do tipo NS com informações falsas sobre qual é o endereço IP do servidor de nomes com autoridade para um domínio. Dessa maneira um atacante pode personificar um servidor de nomes com autoridade e fraudar consultas de nomes para todo um domínio. Esse ataque foi relatado por Dan Kaminsky em 2008, e deixou clara a eficácia e praticidade do DNS *cache poisoning* [US-CERT 2008].

No exemplo o atacante deseja envenenar um servidor de nomes de algum ISP com um registro NS falso com o seu endereço de IP para um servidor de nomes com autoridade de nome `dns.bancoDoPedro.com.br`. Os seguinte passos são realizados:

1. Primeiramente o atacante faz uma requisição para o servidor de nomes vítima requisitando um endereço para um nome aleatório(ex. `www3718783.bancoDoPedro.com.br`) que pertença ao domínio alvo a ser dominado (`bancoDoPedro.com.br`). Um nome aleatório provavelmente não estará no cache do servidor alvo. Dessa maneira o servidor vítima irá resolver recursivamente esse nome, fazendo perguntas pra o servidor raiz, TLD e finalmente de autoridade para o nome requisitado.
2. O atacante sabendo que a vítima, na cadeia de requisições DNS, irá perguntar pelo endereço do banco para o servidor DNS TLD para ele `.com.br`, começa a inundar(*flooding*) o servidor vítima com mensagens de resposta com registros falsos

do tipo NS com campo *name* com nome de domínio `bancoDoPedro.com.br` e com campo *value* com o nome canônico do servidor de nomes com autoridade para o domínio `dns.bancoDoPedro.com.br`, porém com um registro adicional do tipo A que indica que o endereço IP para `dns.bancoDoPedro.com.br` é o endereço IP do servidor DNS sobre domínio do atacante. Ele tenta uma série de combinações de parâmetros chaves que tentam bater com os que foram utilizados pelo servidor vítima e uma hora acerta.

3. A resposta falsa vinda do atacante chega antes da resposta autêntica vinda do servidor TLD `com.br`. A resposta autêntica é descartada pela vítima, já que a requisição já foi respondida, e a resposta falsa é guardada no cache.
4. O servidor de nomes vítima irá agora consultar o servidor DNS do atacante para todos os nomes requisitados pelos seus cliente que pertençam ao domínio `bancoDoPedro.com.br`, ou seja, o atacante controla todo o domínio vítima.

## 4. Conclusão

Esse trabalho mostrou que um dos serviços mais utilizados na rede global, o DNS, que está presente no núcleo da Internet e é essencial para que ela funcione está suscetível ao ataque de DNS *cache poisoning*. Abordou como esse ataque pode ser feito, as premissas para que obtenha sucesso e quais precauções os administradores de servidores de nomes devem tomar para se prevenirem desse tipo de ataque. Descreveu alguns tipos de consequências e ataques graves que podem acontecer quando o servidor DNS do seu ISP é envenenado. Mencionou que atualmente existe o DNSSEC que é visto como a solução promissora para impedir esse tipo de ataque.

## Referências


Bezroutchko, A. (2007). Predictable dns transaction ids in microsoft dns server. Disponível em `http://www.scanit.be/advisory-2007-11-14.html` Acessado em: 31 de agosto de 2010.

Davies, K. (2008). Dns cache poisoning vulnerability. Disponível em `http://www.iana.org/about/presentations/davies-cairo-vulnerability-081103.pdf`.

DNS-OARC (2008). Web-based dns randomness test. Disponível em `https://www.dns-oarc.net/oarc/services/dnsentropy` Acessado em: 31 de agosto de 2010.

Friedl's, S. (2008). An illustrated guide to the kaminsky dns vulnerability. Disponível em `http://unixwiz.net/techtips/iguide-kaminsky-dns-vuln.html` Acessado em: 31 de agosto de 2010.

Jeremy Hitchcock, Olaf M. Kolkman, R. M. v. R. and Vixie, P. (2009). Dnssec advantage: Reasons for deploying dnssec. Disponível em `http://www.dnssec.net/why-deploy-dnssec` Acessado em: 31 de agosto de 2010.

Klein, A. (2007). Bind 9 dns cache poisoning. Disponível em `http://www.trusteer.com/list-context/publications/bind-9-dns-cache-poisoning` Acessado em: 31 de agosto de 2010.

Kurose, J. F. and Ross, K. W. (2006). *Redes de computadores e a Internet: uma abordagem top-down*. Pearson Addison Wesley.

Olzak, T. (2006). Dns cache poisoning: Definition and prevention. Disponível em http://adventuresinsecurity.com/Papers/DNS_Cache_Poisoning.pdf.

Registro.br (2009). Anúncio do site registro .br sobre dnssec.

Rohr, A. (2009). Noticia publicada no portal g1: Ataque leva clientes do virtua a site clonado de banco. Disponível em http://g1.globo.com/Noticias/Tecnologia/0,,MUL1088103-6174,00-ATAQUE%20LEVA%20CLIENTES%20DO%20VIRTUA%20A%20SITE%20CLONADO%20DE%20BANCO.html Acessado em: 31 de agosto de 2010.

Seifried, K. (2010). Artigo respostas confiáveis, da revista linux magazine de agosto 2010.

US-CERT (2008). Multiple dns implementations vulnerable to cache poisoning. Disponível em http://www.kb.cert.org/vuls/id/800113 Acessado em: 31 de agosto de 2010.