UMA ESTRATÉGIA DE DEFESA CONTRA ATAQUES DE DNS SPOOFING

Vagner Sacramento[1] vagner@natalnet.br
Thais Batista[1] thais@dimap.ufrn.br
Guido Lemos S. Filho[1] guido@dimap.ufrn.br
Claudio Monteiro[2] ccm@ulbra-to.br

[1]Grupo NatalNet - Departamento de Informática e Matemática Aplicada - DIMAP
Universidade Federal do Rio Grande do Norte - UFRN
Campus Universitário - Lagoa Nova - 59072-970, Natal – RN

[2]Núcleo de Pesquisa em Tecnologia da Informação
GPARC - Grupo de Pesquisa Aplicada em Redes de Computadores
Centro Universitário Luterano de Palmas - CEULP/ULBRA - Palmas-Tocantins

ABSTRACT

The Domain Name System (DNS) is one of the most important protocols for the operation of the Internet. Many network applications depend on the correct operation of DNS to provide reliable services. However, this protocol is vulnerable to attacks of IP falsification in the resolution of domain names. DNS SPOOFING, as this attack is known, consists of sending a forged response to a resolution of domain name requisition. This kind of attack can generate countless consequences, such as: denial of service, web pages redirecting, redirecting of all e-mails of a domain, among others. In this article, we present results that prove the application of the local and remote DNS SPOOFING attack and present a security mechanism that detects and inhibits attacks of this kind. We implemented an application that accomplishes local and remote DNS SPOOFING attacks in the bind of the versions 4. ,8.* and 9.*. Additionally, we designed and implemented a defense mechanism against attacks of this kind. The defense mechanism was incorporated to the code of the BIND application - the most used implementation of DNS on the INTERNET.*

1. INTRODUÇÃO

O Sistema de Nomes de Domínio – DNS - é responsável por traduzir os diversos nomes de domínios existentes para os seus respectivos endereços IP, atendendo requisições de clientes DNS (*resolvers*) na porta cinqüenta e três (53) (MOCKAPETRIS, 1987). O DNS gerencia uma base de dados distribuída dando suporte a uma variedade de aplicações de rede tais como, servidores de correio eletrônico, servidores Web (www), login remoto (rlogin ou telnet), ftp e outros (CHEUNG & LEVITT, 2000). Estas aplicações de rede enviam requisições ao DNS para traduzir o nome de uma determinada máquina (*hosts*) para o respectivo endereço IP.

Analisando a especificação original (MOCKAPETRIS, 1987) que define a implementação do protocolo DNS (*Domain Name System*), identificamos alguns pontos de falha que podem comprometer a segurança das aplicações que são implementadas com base nesta especificação.

Um dos possíveis ataques que pode ser aplicado no DNS é a falsificação de IP na resolução de nomes de domínio. O *DNS SPOOFING*, como é mais conhecido, consiste em enviar uma resposta falsa a uma requisição de nome de domínio. A especificação original do DNS não apresenta soluções eficazes para prevenção de ataques de DNS SPOOFING.

A falsificação de endereços IP no processo de resolução de nomes pode ser realizada a partir de máquinas que estejam no mesmo segmento de rede que o servidor DNS vítima, ou pode ser realizada a partir de máquinas de redes diferentes da rede que se encontra o servidor DNS vítima. Denominamos o primeiro caso de Ataque Local e, o segundo, de Ataque Remoto.

Como o DNS é amplamente utilizado e extremamente necessário na Internet, *firewalls* e outros mecanismos de defesa não podem bloquear o acesso a este serviço e, portanto, aceitam e permitem a passagem de pacotes que tenham a porta de destino igual a 53.

Cientes desta facilidade, *hackers* podem aplicar ataques de *DNS SPOOFING* direcionados ao DNS. O *DNS SPOOFING* é uma técnica de ataque que explora as facilidades de predição dos IDs[1] utilizados nas diferentes requisições dos servidores DNS. Historicamente esta técnica de ataque foi introduzida por (VIXIE, 1995), que relatou as possibilidades de predição de ID na resolução de nomes dos servidores DNS e as inúmeras conseqüências que podem ser exploradas a partir da concretização deste ataque.

A aplicação BIND (*Berkeley Internet Name Domain*) é uma das mais utilizadas na Internet para implementar o serviço DNS. Vários problemas de segurança relacionados a esta aplicação são discutidos em (BELLOVIN, 1989, 1995), (GAVRON, 1993), (SCHUBA & SPAFFORD, 1994) e (CERT advisory CA-98.05, 1998).

Existem registros (CORE, 1997), (PHARCK MAGAZINE, 1998) que relatam a aplicação de ataques de *DNS SPOOFING* em servidores DNS que executam a aplicação *BIND* com versões mais antigas que 8.2.*. Analisamos e testamos as técnicas de ataques publicadas, e constatamos que elas se limitam a aplicação do ataque *DNS SPOOFING* local (SCUT, 1999), ou seja, a máquina de onde parte o ataque e o servidor DNS estão localizados no mesmo segmento de rede *Ethernet*. Foram relatados ataques ao *bind* nas versões (4.8.*, 4.9.*, 8.1.*, 8.2.* e 9.*). A estratégia utilizada dificilmente consegue aplicar este ataque remotamente no *bind* das versões (4.8.*, 4.9.3, 4.9.5, 8.0 e 8.1) (RAFTER, 1997).

A principal deficiência das técnicas de ataques encontradas é a incapacidade de identificar qual é o endereço IP de origem a ser utilizado no pacote de resposta a ser enviado para o servidor DNS vítima. Todas elas só enviam pacotes de respostas para o

[1] ID – Identificador utilizado para associar as requisições enviadas por servidores DNS às respostas recebidas de outros servidores.

servidor DNS vítima utilizando o endereço IP do servidor DNS que tem autoridade sob o domínio que está sendo vítima do ataque. A partir da análise do funcionamento do DNS conseguimos resolver esta limitação apresentada por outras técnicas de ataque, aumentando as possibilidades na concretização da aplicação do ataque local e remoto.

Recentemente foi divulgado pelo CERT[1] (CA-2001-01, 2001) um *bug* no *bind 8.2.2-p7* ou inferior que permite ataques de manipulação de *string*, possibilitando um determinado atacante fazer com que o executável do *bind* execute comandos arbitrários com permissão do usuário que está executando esta aplicação. Posteriormente o ISC[2] enviou um notificado recomendando todos administradores de rede atualizarem o *bind* para a versão 8.2.3 ou 9.*, mas o problema relatado pelo CERT não está relacionado com a aplicação de um ataque de *DNS SPOOFING*.

Não encontramos nenhuma referência que relata a ocorrência de ataques de *DNS SPOOFING* remoto nas versões do *bind* 8.2.* e 9.*, porém, conseguimos implementar e comprovar este ataque (SACRAMENTO & BATISTA & LEMOS & MONTEIRO, 2001) no *bind* com estas versões. Conseguimos também sucesso em ataques a versões mais recentes (*bind* 8.2.4 e 9.1.3) utilizada na maioria das redes que compõem a Internet.

Aplicando o DNS SPOOFING o *hacker* poderá comprometer outros serviços, gerando inúmeras conseqüências para o administrador da rede e usuários finais. Ataques direcionados ao serviço DNS podem causar impedimento de serviço (quando um cliente não pode localizar o endereço de rede de um servidor), falhas nas entidades de autenticação (quando os nomes dos *hosts* são usados nos mecanismos de autenticação) e redirecionamento da origem de determinadas informações (quando o servidor DNS informa um endereço IP que não diz respeito ao nome de domínio questionado por um cliente) (DAVIDOWICZ, 1999).

Neste trabalho relatamos as possibilidades de ataques de DNS SPOOFING nas aplicações que implementam o serviço DNS e apresentamos mecanismos de defesa capazes de detectar e bloquear ataques deste gênero.

O texto que segue está assim estruturado. A seção 2 apresenta brevemente a estrutura e o funcionamento do serviço de nomes, a aplicação da falsificação de IP em servidores de nomes e exemplifica algumas conseqüências desta falsificação. A seção 3 apresenta brevemente os resultados da implementação do ataque *DNS SPOOFING* local e remoto. A seção 4 apresenta os mecanismos de defesa que elaboramos para detectar e inibir ataques de *DNS SPOOFING* tanto local quanto remoto. Para validar e testar as soluções projetadas inserimos algumas regras dentro do código da aplicação BIND. A seção 5 descreve os trabalhos correlatos. A seção 6 contém as conclusões desse trabalho.

[1] CERT – Computer Emergency Response Team.
[2] ISC (Internet Software Consortiun) - Equipe responsável pelo desenvolvimento do BIND.

2. SISTEMA DE NOMES DE DOMÍNIOS – DNS

2.1 Estrutura Hierárquica do DNS

O DNS gerencia uma base de dados hierárquica e distribuída indexada por nomes. Um nome de domínio ou uma zona (por exemplo, uol.*com.br*) possui uma estrutura que reflete a sua colocação dentro da hierarquia de nomes. A Figura 1 mostra parte da estrutura hierárquica utilizada por um servidor DNS.

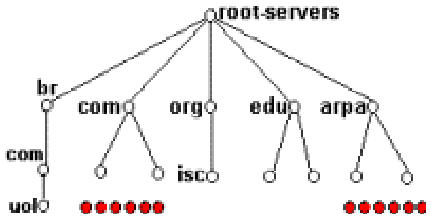

Figura 1-Estrutura Hierárquica de nomes utilizada pelo DNS

2.2 Funcionamento do Protocolo DNS

O processo de busca de dados no DNS é chamado de resolução de nomes ou simplesmente resolução. Existem dois modos de resolução de nomes no DNS: *interativo* e *recursivo*. No modo interativo o servidor DNS não assume a responsabilidade de resolver a requisição recebida. Ele envia uma resposta contendo a resolução do nome questionado caso ele tenha a informação solicitada armazenada em *cache* ou ele envia a indicação de outros servidores DNS que estão aptos a enviar uma resposta mais exata. No modo recursivo o servidor DNS assume a responsabilidade de resolver a requisição recebida. Este servidor terá que encontrar uma resposta para a requisição solicitada e enviá-la ao requisitante (MOCKAPETRIS, 1987).

Cada servidor é inicializado com o endereço IP de alguns servidores que são autoridades do domínio *root-servers.net*. Os servidores *root* conhecem os servidores autoridades de todos os domínios de segundo nível (br, org, com, edu). Os servidores de segundo nível conhecem o endereço IP dos servidores com autoridade nos domínios de terceiro nível (yahoo.com.br, microsoft.com, isc.org), e assim por diante. Deste modo, seguindo a estrutura da árvore hierárquica de resolução de nomes, a resolução de um determinado nome de domínio pode chegar a devida resposta depois de cada indicação de servidores de outro domínio (CHEUNG & LEVITT, 2000).

Para que o processo de resolução de nomes seja uma tarefa rápida, os servidores DNS armazenam o resultado das requisições resolvidas em *cache*. Cada resposta recebida por um servidor DNS é armazenada em *cache* por um determinado tempo em segundos, dependendo do tempo de vida atribuído a mensagem (*time-to-live*-TTL). Quando termina a validade, o nome é removido do *cache* (MOCKAPETRIS, 1987). Caso o mesmo nome de domínio seja questionado duas vezes a um servidor DNS, este poderá responder

imediatamente a segunda requisição, uma vez que já possui a informação armazenada em *cache*.

2.3 "*SPOOFING*" de IP em Servidores DNS

A falsificação de IP na resolução de nomes de domínios -*DNS SPOOFING* - ocorre quando é enviada uma resposta falsa a uma requisição de resolução de nome para o servidor DNS, fazendo com que ele armazene em *cache* um endereço IP falso para um determinado nome de domínio(MEN & MICE, 1999).

Quando um servidor DNS faz uma pergunta a outro servidor, questionando o endereço IP de um determinado nome de domínio, ele fica aguardando uma resposta do mesmo. Caso ele receba uma resposta, ele verifica se determinadas informações estão corretas e, em caso positivo, armazena a resposta em seu *cache*. Uma das informações verificadas é o campo ID do cabeçalho do pacote DNS.

As informações verificadas na resposta recebida pelo servidor DNS são (SACRAMENTO & MONTEIRO, 2000):

- Se o campo ID e o campo "Nome a ser resolvido" do pacote DNS são iguais aos enviados na pergunta;
- Se a porta de origem é igual à porta de destino do pacote UDP enviado na pergunta;
- Se o endereço IP de origem é igual ao endereço IP de destino enviado na pergunta.

Supondo que esta resposta tenha sido enviada por um terceiro que está se passando pelo servidor DNS questionado, as informações da resposta recebida como resolução do nome de domínio podem estar incorretas, fazendo com que o servidor DNS armazene em *cache* um endereço IP falso para o domínio em questão. Portanto, o *spoofing* de IP em servidores DNS polui[1] o *cache* do DNS vítima do ataque.

A Figura 2 mostra o cenário utilizado para aplicar a falsificação de IP em uma resolução de nomes com o ID do cabeçalho DNS igual a 2005.

2.4. Conseqüências do Ataque *DNS SPOOFING*

Quando o ataque *DNS SPOOFING* ocorre com êxito, o *cache* do servidor DNS permanece poluído por um determinado tempo, armazenando uma informação incorreta sobre um determinado nome de domínio. Várias conseqüências podem ser exploradas a partir da aplicação deste ataque. Algumas delas são:
1ª) A alteração da origem da informação requisitada na abertura de uma página *web*. Por exemplo: Quando o usuário digitar no *browser* de navegação o endereço da página que ele deseja abrir (suponhamos que seja um *site* de um banco) em vez de aparecer a página requisitada, irá aparecer a página hospedada no servidor *web* informado pelo atacante, através do endereço IP utilizado como resposta na resolução de nome efetuada no ataque. Ou seja, ao invés de ser exibido o site requisitado pelo usuário para consultar o seu saldo, poderá aparecer um outro site muito similar ao do banco requisitado pelo cliente, e, quando o usuário informar a senha, esta será recebida pelo atacante. Neste caso, as conseqüências podem ser desastrosas.
2ª) O *DNS SPOOFING* pode ser aplicado para driblar mecanismos de defesa baseados em *wrappers*[2] que utilizam o nome dos *hosts* para verificar a autenticidade entre eles;
3ª) Todas aplicações (e-mail, Web, rlogin, ftp,**...**) que necessitam do serviço DNS para tradução de nomes poderão ser atacadas.

3. *DNS SPOOFING* LOCAL E REMOTO

Na implementação de ataques de DNS SPOOFING utilizamos como estudo de caso as seguintes aplicações que implementam o DNS: BIND 4.8.*, 4.9.*, 8.1.*, 8.2.* e 9.* e o Microsoft DNS SERVER 4.0 (DNS SERVER, 1999).

Estudamos o funcionamento de várias versões da implementação da aplicação *bind*. O propósito da avaliação das várias versões existentes do *bind* foi entender a evolução do algoritmo de geração de ID, com o intuito de provar a viabilidade da aplicação do

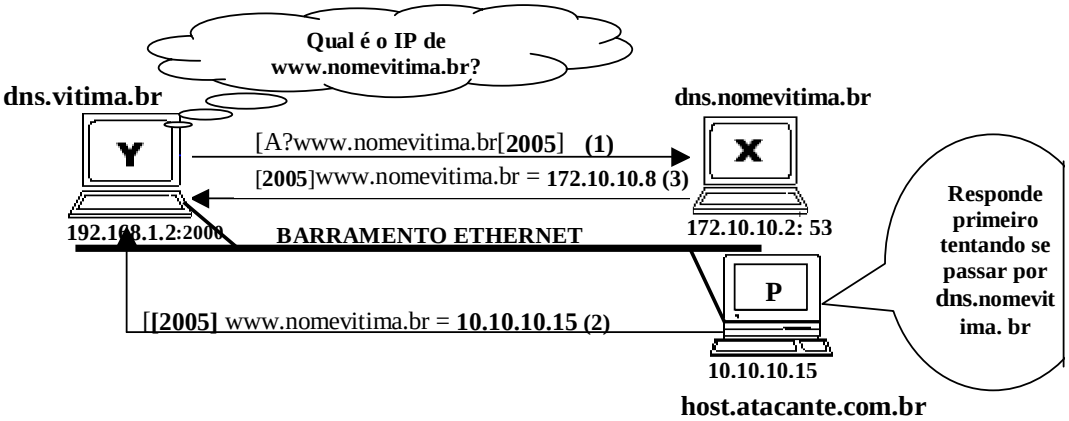

Figura 2 - Falsificação de IP em servidores de nomes

[1] Poluir o cache do servidor DNS significa fazer com que este servidor armazene em cache uma informação falsa sobre a resolução de um determinado nome de domínio.

[2] *wrappers*–Mecanismos de software usados para filtrar a utilização de alguns serviços: telnet, ftp, talk,etc .

ataque *DNS SPOOFING* remotamente em todas elas.

A partir das estratégias elaboradas e relatadas em (SACRAMENTO & BATISTA & LEMOS & MONTEIRO, 2001), conseguimos implementar ataques deste gênero localmente ou remotamente nas aplicações que implementam o protocolo DNS executando as versões mais estáveis atualmente (BIND 8.2.4, 9.1.3 e DNS SERVER 4.0).

Localmente conseguimos aplicar o ataque com 100% de sucesso, em todas as aplicações citadas.

Os resultados dos experimentos realizados no ataque remoto são mostrados na seção 4.1.2 e na figura 6.

4. MECANISMOS DE DEFESA

A aplicação do *DNS SPOOFING* pode gerar inúmeras conseqüências não deixando nenhuma pista que possibilite identificar a sua origem. Ou seja, é praticamente impossível detectar a identidade do atacante, pois este, em geral, utiliza o *IP SPOOFING* para falsificar o endereço de origem do pacote IP na resposta de resolução de nome enviada para o servidor DNS vítima.

Diante das dificuldades de prover segurança contra ataques deste gênero, projetamos e implementamos uma solução para detectar e bloquear ataques de *DNS SPOOFING* tanto localmente, quanto remotamente. Para validar a solução projetada utilizamos como estudo de caso a aplicação BIND. Alteramos o código desta aplicação possibilitando a detecção e prevenção de ataques de *DNS SPOOFING*. No entanto, as soluções elaboradas e projetadas para detecção de ataques deste gênero podem ser adequadas a qualquer outra aplicação que implementa o serviço de DNS.

A solução que elaboramos e implementamos foi estender o código da implementação do DNS. O objetivo dessa estratégia é prover uma solução que possa ser adotada dentro do próprio funcionamento do DNS, sem prejudicar o seu desempenho no processamento das mensagens e que possa, ao mesmo tempo, inibir e detectar as possibilidades da aplicação do ataque *DNS SPOOFING* local e remoto.

4.1 Análise realizada pelo DNS no processamento das respostas

Como já foi dito, o *DNS SPOOFING* visa antecipar uma resposta de resolução de nome falsa para o servidor DNS vítima, ou seja, a resposta de resolução de nome enviada pelo atacante tem que chegar antes que a resposta verdadeira enviada pelo servidor DNS questionado. Sendo assim, mesmo que a resposta de resolução de nome enviada pelo atacante chegue primeiro, logo em seguida o servidor DNS vítima receberá a resposta de resolução de nome verdadeira. Como este servidor já recebeu uma primeira resposta com as informações esperadas, ele irá desprezar as demais.

Realizamos um estudo detalhado do código do bind-*8.2.4* - o mais estável no momento, com o objetivo de encontrar o local exato em que ele despreza uma segunda resposta de resolução do mesmo nome de domínio. Nosso intuito era saber qual é o tipo de análise que ele realiza, pois sabemos que em nenhuma outra circunstância, a não ser uma tentativa de ataque, um servidor DNS receberá simultaneamente duas respostas de resolução do mesmo nome de domínio contendo as mesmas informações de autenticidade esperadas (ID, nome de domínio, Portas UDP, IP de origem, ...), com respostas diferentes sobre a resolução do nome questionado (*RDATA*[1] diferentes).

Identificamos que o DNS tem uma tabela (conhecida por ***nshead{}***) que armazena todas requisições de resolução de nomes enviadas para outros servidores DNS. Sempre que ele recebe uma resposta de resolução de nome, ele verifica na tabela de requisições enviadas se existe um registro correspondente à resposta recebida. O servidor DNS tenta encontrar na tabela de requisições enviadas, algum registro que tenha o mesmo ID da resposta recebida. A função (***qfind()***) do DNS realiza essa verificação e retorna **NULL** quando não existe nenhuma requisição de resolução de nome correspondente a resposta recebida. Ou seja, quando este método retorna **NULL**, a resposta recebida deve ser desprezada. É exatamente isso que acontece com a resposta de resolução de nome verdadeira, quando a resposta falsa de resolução de nome enviada pelo atacante é recebida primeiro pelo servidor DNS vítima.

A partir da afirmação de que, em nenhuma hipótese, um servidor DNS em circunstâncias normais receberá simultaneamente duas respostas de resolução do mesmo nome de domínio contendo as mesmas informações de autenticidade esperadas (ID, nome de domínio, Portas UDP, IP de origem, ...), com *RDATA* (resposta da resolução do nome questionado) diferentes, elaboramos uma estratégia de defesa considerando a análise realizada pelo DNS para validar uma resposta (verificar se o ID é válido). Segunda a estratégia que estamos propondo o DNS deve realizar outras ações, ao invés de desprezar as respostas que caracterizam uma tentativa de ataque. Quando é recebida uma segunda resposta válida para uma dada requisição caracterizando uma tentativa de ataque, as seguintes ações serão executadas:

- Registrar em log detalhes do acontecido;
- Remover do *cache* o registro que contém o nome utilizado no ataque ao servidor DNS;
- Enviar *e-mail* para o administrador da rede.

A análise realizada pelo DNS para verificar se existe uma pergunta correspondente a resposta recebida é mostrada na Figura 3.

```
if ((qp = qfindid(hp->id)) == NULL ) {
ns_debug(ns_log_default, 1, "DUP? drop (id %d)
nameserIncr(from.sin_addr, nssRcvdDupR);
return; }
```

Figura 3 – Análise realizada pelo *bind* para validar uma resposta de resolução de nome.

[1] RDATA – campo que armazena a resposta da resolução do nome. Maiores informações RFC 1035.

4.2 Soluções propostas para detecção de ataques de DNS SPOOFING no processamento das respostas realizada pelo DNS

O primeiro passo realizado para dificultar ataques de *DNS SPOOFING* foi identificar o local em que o *bind* está permitindo o envio de várias requisições para resolução de um mesmo nome de domínio. Após identificarmos a análise realizada pelo *bind* para envio de requisições, alteramos este comportamento fazendo com que ele só realize o repasse de uma requisição para resolução de um determinado nome de domínio.

Para detectar ataques deste gênero, estamos realizando uma análise mais detalhada de todas respostas recebidas. Para realizarmos as análises necessárias précisamos acrescentar outras funcionalidades ao comportamento do *bind*. Como dito anteriormente, o *bind* possui uma tabela que armazena uma identificação de todas requisições enviadas para validar as respostas recebidas. Após receber uma resposta com as informações de autenticidade esperada, ele remove a requisição correspondente da tabela de requisições. A partir deste momento qualquer resposta que possa identificar uma tentativa de ataque, será "desprezada" devido à requisição correspondente ter sido removida de ***nshead{ }***.

Para identificar uma tentativa de ataque, é necessário manter uma referência da requisição equivalente à resposta recebida por um determinado intervalo de tempo. Para tanto, criamos uma tabela com este propósito que armazena durante um intervalo de tempo uma identificação de todas as requisições enviadas.

Esta tabela ficou definida com o nome **tb_request{ }** a sua estrutura é mostrada na figura 4:

```
struct tb_request{
  char *name;
  struct in_addr  ns_addr;
  struct tb_request *next;
};
```

Figura 4 – Estrutura da tabela tb_request

Todas as requisições enviadas são armazenadas na tabela **tb_request{}** para que seja possível autenticar as respostas recebidas. Atualmente, quando o *bind* original recebe uma resposta, ele utiliza a função ***qfind()*** mostrada na figura 2 para autenticá-la como uma resposta válida. Ao invés de fazermos a simples análise realizada por esta função (***qfind()***), iremos utilizar uma outra função definida por ***find_request()***. Quando o bind recebe uma resposta, esta função (***find_request()***) verifica em ***tb_request{}*** se existe uma requisição correspondente à resposta recebida. Caso exista, esta resposta é armazenada no *cache* do servidor DNS e adicionada à tabela de resposta auxiliar (conhecida por ***tb_resp{}***).

A tabela **tb_resp{}** armazena temporariamente as informações de todas as respostas recebidas. Antes de adicionar uma resposta recebida em ***tb_resp{ }*** é verificado se já existe nesta tabela um registro contendo as mesmas informações de autenticidade (ID, NOME, IP DE ORIGEM) da resposta recebida. Caso seja encontrado um registro armazenado em ***tb_resp{ }*** contendo as mesmas informações de autenticidade da resposta recebida, é identificada a presença de duas respostas de resolução de nome para o mesmo domínio, contendo as mesmas informações de autenticidade. Diante desta circunstância serão analisadas as informações enviadas como resolução do nome de domínio recebido nas duas respostas identificadas. Se estas informações forem diferentes, é registrada uma tentativa de ataque devido ao fato do servidor DNS ter recebido duas respostas contendo as mesmas informações de autenticidade esperadas (ID, Nome de domínio, IP-Source, IP-Dest) com repostas de resolução do nome diferentes. A tabela ***tb_resp{ }*** foi definida com a estrutura mostrada na figura 5.

```
struct tb_resp {
  char *reply;
  char *name;
  struct in_addr  ns_addr;
};
```

Figura 5 – Estrutura da tabela tb_resp

Na tentativa de detectar e bloquear ataques de *DNS SPOOFING*, cada resposta recebida é armazenada em ***tb_resp{ }*** por um determinado intervalo de tempo. Só será identificada uma tentativa de ataque se o atacante conseguir enviar uma resposta para o servidor DNS vítima, contendo as informações de autenticidade esperadas por este servidor, ou seja, somente quando o atacante conseguir êxito na aplicação do ataque é que os alertas serão gerados. Desta forma, estamos evitando que circunstâncias não esperadas no funcionamento do DNS gerem alarmes falsos.

Após receber uma resposta, esta será armazenada em **tb_resp{ },** caso não exista nenhuma outra resposta similar (utilizando o mesmo ID, NOME, IP-SOURCE) armazenada nesta tabela. Esta resposta permanece sob análise durante um tempo que varia entre 4000 e 7999 ms. Se durante este intervalo de tempo, não chegar nenhuma outra resposta com as mesmas informações de autenticidade esperadas, caracterizando uma tentativa de ataque, o registro em questão será removido de ***tb_resp{ }*** e de ***tb_request{}***. O tempo estimado para análise das respostas recebidas baseai nas estatísticas mostradas na figura 8.

A detecção do ataque tornou-se possível porque a requisição equivalente à resposta recebida não foi removida da tabela de requisições enviadas quando a primeira resposta foi recebida.

Após identificada uma tentativa de ataque, são gerados os alertas necessários:

- Gravar nos logs os detalhes do acontecido;
- Enviar e-mail para o administrador da rede;
- Remover do *cache* do servidor DNS somente o registro que contenha o nome utilizado para aplicar o ataque.

4.3 Resultados dos mecanismos de defesa incorporados ao DNS

Após realizar as modificações descritas na seção 4 no código do *bind* 8.2.4 recompilamos o mesmo e realizamos alguns experimentos para validar as alterações realizadas.

Utilizando o bind 8.2.4 com as novas funcionalidades de segurança obtivemos 100% de êxito na detecção do ataque local. Nos experimentos realizados aplicamos 675 ataques contra domínios diferentes.

A comparação da eficiência do ataque local no bind 8.2.4 modificado e nas outras implementações do protocolo DNS é mostrada na tabela 1.

675 Tentativas de ataque	BIND 8.2.4 Modificado	BIND 8.2.4 Original	BIND 9.1.3 Original	DNS SERVER 4.0 Microsoft
Resultados De Sucesso	0 %	100%	100%	100%

Tabela 1- Resultados da eficiência do ataque local

Os resultados dos mecanismos de defesa obtidos para o ataque remoto são mostrados nos gráficos da figura 6. A figura 6**a** mostra os resultados da aplicação do ataque ao BIND 8.2.4 que inclui as novas funcionalidades de defesa. As figuras 6**b** e 6**c** mostram os resultados da aplicação do ataque ao BIND 8.2.4 e 9.1.3 sem os mecanismos de defesa propostos. A figura 6 mostra o número de sucesso obtido em uma faixa de intervalo a cada 10 tentativas de ataques. Para diminuir o tamanho do gráfico, aumentamos esta faixa de intervalo quando os valores ficam muito dispersos. Ex.: Na figura 6**b** obtivemos: 911 sucessos de ataques no intervalo de 1 à 10 tentativas, 483 sucessos no intervalo de 11 à 20 tentativas, 218 sucessos no intervalo de 21 à 30 tentativas, 108 sucessos no intervalo de 31 à 40 tentativas, 74 sucessos no intervalo de 41 à 50 tentativas e 78 sucessos de ataques no intervalo de 51 à 190 tentativas.

Atacamos remotamente o servidor DNS do laboratório NatalNet executando a aplicação BIND com as versões: 8.2.4 e 9.1.3. Todos resultados dos ataques mostrados na figura 6 foram realizados durante um período de 90 horas.

Para este experimento definimos uma seqüência de ataque como o envio de 1 a 1764 tentativas, sendo que cada tentativa utiliza um domínio diferente. Nos testes realizados foram enviadas várias seqüências de ataques ao DNS. Em cada seqüência são enviadas várias tentativas de ataques, até que seja obtido sucesso em uma delas, ou seja alcançado o número máximo de tentativas possíveis. Após uma destas condições se tornar verdadeira é iniciada uma nova seqüência de ataque.

Os resultados do gráfico da figura 6**a** foram obtidos em 31752 tentativas de ataques aplicadas ao bind 8.2.4 com os mecanismos de defesa. Dentre estas tentativas foram obtidos três (3) resultados de sucesso. O primeiro deles foi obtido na seqüência de número 9 na tentativa de número 1498, ou seja, o primeiro resultado de sucesso obtido foi após 15610 tentativas de ataques. Este número foi calculado da seguinte forma:

- Em 8 seqüências não foi obtido nenhum sucesso de ataque atingindo o número máximo de tentativas em cada uma delas.
- Na seqüência de numero 9 na tentativa 1498 foi obtido o primeiro sucesso.

Desta forma temos o seguinte cálculo: 1764 x 8 + 1498 = 15610.

O segundo resultado de sucesso obtido foi na décima seqüência na tentativa de número 1105 e o último resultado de sucesso obtido ocorreu na décima segunda seqüência na tentativa de número 740. Desta forma, o gráfico da figura 6**a** mostra os seguintes resultados: 1 sucesso de ataque no intervalo de 731 à 740, 1 sucesso de ataque no intervalo de 1101 à 1110 e 1 sucesso de ataque no intervalo de 1491 à 1500. Os resultados deste gráfico mostram que 0,013 % das tentativas de ataques não foram detectadas.

Os resultados do gráfico da figura 6**b** mostram os resultados obtidos em 31761 tentativas de ataques aplicadas ao bind 8.2.4 sem os mecanismos de defesa. Dentre estas tentativas 1872 ataques foram bem sucedidos. O primeiro deles foi obtido na primeira seqüência de ataque na tentativa de número 4, ou seja, o primeiro resultado de sucesso obtido foi após 4 tentativas de ataque. Os resultados deste gráfico mostram que 5,894 % das tentativas de ataque foram bem sucedidas.

Os resultados do gráfico da figura 6**c** mostram os resultados obtidos em 32572 tentativas de ataque aplicadas ao bind 9.1.3 sem os mecanismos de defesa. Nestas tentativas 88 ataques foram bem sucedidos. O primeiro deles foi obtido na primeira

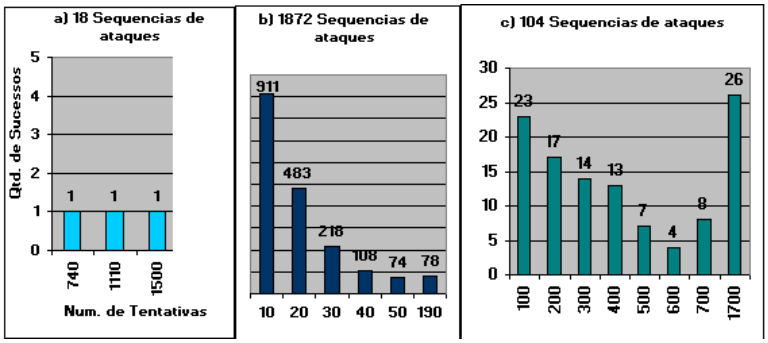

Figura 6 – Resultados dos mecanismos de defesas

seqüência de ataque na tentativa de número 64, ou seja, o primeiro resultado de sucesso obtido foi após 64 tentativas de ataque. Os resultados deste gráfico mostram que 0,270 % das tentativas de ataques foram bem sucedidas.

Comparando os resultados obtidos temos os seguintes valores: De 31752 tentativas de ataques aplicadas ao bind 8.2.4 com os mecanismos de defesas obtivemos 3 sucessos. De 31761 tentativas de ataques ao bind 8.2.4 original obtivemos 1872 sucessos. De 32752 tentativas de ataques aplicados ao bind 9.1.3 obtivemos 88 sucessos.

O ataque ao *bind* com as novas funcionalidades de defesa não será detectado caso não sejam recebidas duas respostas para a resolução do mesmo nome de domínio, contendo as mesmas informações válidas esperadas pelo servidor DNS vítima, com **RDATAs** diferentes. Como o DNS utiliza o protocolo UDP para enviar as suas requisições, não temos certeza de que a requisição de resolução de nome enviada pelo servidor DNS vítima vai chegar ao seu destino e, mesmo que chegue ao destino, não se tem a certeza de que a resposta enviada pelo servidor DNS questionado retorne ao servidor DNS vítima. O UDP é um protocolo da camada de transporte não confiável que não oferece mecanismos de recuperação dos datagramas perdidos (SOARES & LEMOS & COLCHER, 1995).

Quando, por algum motivo, a resposta enviada pelo servidor DNS verdadeiro não for recebida pelo servidor DNS vítima, o ataque não é detectado.

Na tentativa de amenizar a perda de pacotes UDP, o BIND faz um controle de recebimento das respostas. Se nenhuma resposta for recebida num intervalo de 4 segundos, ele retransmite uma requisição utilizando o mesmo ID. A figura 7 mostra o tempo utilizado pelo BIND para retransmitir uma requisição.

```
#define RETRYBASE  4 /* base time between retries (4 Seconds)*/
#define MAXRETRY  3 /* max number of retries per addr */
```

Figura 7 – Tempo de retransmissão de requisições

Na tentativa de não ser detectado, o atacante pode tentar aplicar o ataque ao servidor DNS vítima e ao mesmo tempo aplicar um ataque de "*flooding*[1]" ao suposto servidor DNS que será questionado. O atacante pode tentar imobilizar o DNS verdadeiro fazendo com que ele não envie uma resposta para o servidor DNS vítima impossibilitando a identificação do ataque, pois não será recebida uma segunda resposta. A grande dificuldade do atacante é saber qual é o servidor DNS que ele deverá aplicar o ataque de "*flooding*". Como os servidores DNS resolvem nomes de forma interativa, o atacante não sabe para qual servidor o DNS vítima enviou uma requisição. O que o atacante pode tentar fazer, é aplicar um *flooding* distribuído direcionado a vários servidores DNS que provavelmente receberam uma requisição de resolução de nome enviada pelo servidor DNS vítima. Ele terá que aplicar um ataque de *flooding* a cada nova tentativa de ataque.

Uma situação muito inesperada poderá ocorrer quando o cliente DNS enviar uma requisição de resolução de nome, no exato momento em que o servidor DNS vítima recebeu a resposta falsa enviada pelo atacante, mas ainda não recebeu a resposta de resolução de nome verdadeira. A diferença de tempo (Resp1-Resp2) entre a resposta de resolução de nome falsa enviada pelo atacante e a resposta verdadeira enviada pelo servidor DNS questionado é, em média, 250 ms de acordo com as estatísticas mostradas na figura 8. Ou seja, após a resposta do atacante ser recebida pelo servidor DNS vítima, a resposta verdadeira provavelmente será recebida nos próximos 250 ms. Se, nesse intervalo de tempo, algum cliente DNS enviar uma requisição para o servidor DNS vítima, questionando o mesmo nome de domínio utilizado no ataque, ele irá receber uma resposta contendo informações falsas. A figura 8 mostra a quantidade de requisições resolvidas a cada

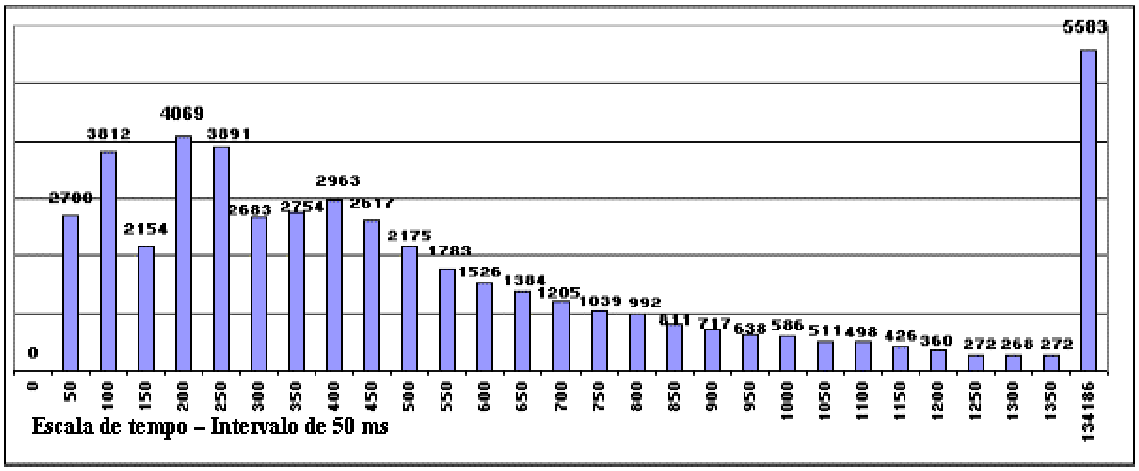

Figura 8 - Status do processamento inicial do BIND modificado

[1] *Flooding (inundação)* – este ataque consiste basicamente no envio de um grande número de mensagens, para um determinado servidor, com endereço IP de origem forjado, com intuito de fazer com que a fila de recepção de mensagens em andamento fique lotada.

intervalo de 50 ms. As informações apresentadas foram calculadas a partir das estatísticas realizadas no servidor de nomes da rede ufrn.br na resolução de cinqüenta mil (50000) requisições.

4.4 Análise de desempenho

As informações apresentadas na figura 11 mostram o desempenho do DNS modificado e do DNS não modificado no processamento de 1000 requisições de resolução de nomes. Os testes foram realizados com o BIND 8.2.4.

Passos para execução do experimento foram:

1. Iniciar o servidor DNS modificado;
2. Executar o programa cliente que envia ao servidor DNS modificado 1000 requisições de resolução de nome seqüencialmente;
3. Registrar a diferença de tempo entre a requisição enviada e a resposta recebida;
4. Finalizar o DNS modificado;
5. Repetir os passos acima utilizando o servidor de nomes não modificado.

Para todos experimentos realizados foi utilizada uma máquina com a seguinte configuração:

- COMPAQ, Processador Intel Celeron 466 MHZ, com 128 KB de memória *cache*, *bogomips* 927.33, 96 MB de RAM, Placa de rede ATM de 155 Mbps, com RedHat 7.1.

O status inicial da CPU e memória desta máquina executando o DNS modificado e não modificado é mostrado na figura 9 e 10. Estas informações foram adquiridas através do programa TOP[1].

```
12:57pm  up  1:40,  4 users,  load average: 0,10, 0,04, 0,01
38 processes: 37 sleeping, 1 running, 0 zombie, 0 stopped
CPU states:  0,1% user,  0,5% system,  0,0% nice, 99,2% idle
Mem: 94364K av, 89100K used,5264K free, 0K shrd, 2636K buff
Swap: 72252K av, 14516K used, 57736K free 71412K cached
```

PID	USER	PRI	SIZE	RSS	SHARE	%CPU	%MEM	TIME	COMMAND
1431	named	10	1752	1752	988	0,0	1,8	0:00	named

Figura 9 - Status do processamento inicial do BIND não modificado

```
12:54pm  up  1:37, 4 users,  load average: 0,00, 0,01, 0,00
38 Processes: 36 sleeping, 1 running, 0 zombie, 0 stopped
CPU states:  0,3% user,  0,1% system,  0,0% nice, 99,4% idle
Mem: 94364K av, 67632K used,   26732K free,  0K shrd, 2100K buff
Swap: 72252K av, 15424K used, 56828K free 49988K cached
```

PID	USER	PRI	SIZE	RSS	SHARE	%CPU	%MEM	TIME	COMMAND
1279	named	10	3106	3106	988	0,0	3,2	0:00	named

Figura 10 - Status do processamento inicial do BIND modificado

Percebemos que uma das diferenças do status inicial do DNS mostrado nas figuras 9 e 10 é o consumo de memória. O BIND modificado consome mais memória devido à necessidade de criar algumas tabelas auxiliares.

A figura 11 mostra as estatísticas relacionadas ao tempo de resposta para resolução de 1000 nomes de domínios diferentes, configurando uma situação real desempenhada por um servidor DNS. Para registrar estas informações implementamos um cliente DNS similar ao *nslookup* que calcula a diferença de tempo em ms (milisegundos) entre a requisição enviada e a resposta recebida. Os diferentes domínios utilizados são compostos pelas extensões (.br, .com, .org, .au, .gu, .co, .uk, .gr, ...).

A média de tempo de resposta para o DNS modificado foi 0.638 segundos ou 638 ms por requisição, e para o servidor DNS original foi de 607 ms. Ambas estatísticas foram realizadas com a resolução dos mesmos nomes sob as mesmas circunstâncias.

5. TRABALHOS CORRELATOS

Nesta seção descreveremos alguns mecanismos de defesa para ataques de *DNS SPOOFING*, e os comparamos com a solução que propomos e apresentamos na seção 4.

5.1 DNS Wrapper

O projeto apresentado em (CHEUNG & LEVITT, 2000) relata a elaboração de um sistema de nomes conhecido como *DNS wrapper*.

O DNS *wrapper* é uma aplicação modelada e especificada em VDM (BELLOVIN, 1989) que foi incorporada ao código do *bind* com objetivo de examinar as mensagens de DNS que estão sendo enviadas e recebidas. Esta aplicação consiste em duas partes principais: *Wrapper-q* para processar as requisições e *Wrapper-r* para processar as respostas.

Como estudo de caso, este projeto utilizou o *bind* 4.9.5 que possui um algoritmo de geração de ID seqüencial. Na tentativa de prevenir ataques de *DNS SPOOFING*, o *DNS WRAPPER* implementa uma geração aleatória dos IDs a serem enviados nas requisições de resolução de nomes. Desta forma, o *DNS WRAPPER* não leva em conta a possibilidade de ataques locais que capturam o ID corrente utilizado pelo servidor DNS, e não considera a existência de técnicas de ataques específicas, que realizam a predição de IDs aleatórios. Além da estratégia de aleatoriedade dos IDs ser ineficiente, ela já é implementada no *bind* das categorias 8.2.* e 9.*,

	Servidor DNS Original			Servidor DNS modificado		
Requisições	MIN	MAX	MÉDIA	MIN	MAX	MÉDIA
1000	57 ms	7731 ms	607 ms	55 ms	8182 ms	638 ms

Figura 11 - Tempo de resposta para resolução de nomes

[1] TOP – Mostra o consumo de processamento das aplicações.

não sendo necessário utilizar o *DNS WRAPPER* . No trabalho realizado em (SACRAMENTO & BATISTA & LEMOS & MONTEIRO, 2001) foram citadas algumas técnicas que comprovam a aplicação de ataques de *DNS SPOOFING* em todas versões existente do *bind*, independente do tipo de algoritmo de geração de ID que esteja sendo utilizado (seqüencial ou aleatório).

Ao contrário do DNS WRAPPER propomos uma solução de defesa contra ataques de *DNS SPOOFING* que não se baseia no tipo de algoritmo utilizado por uma determinada aplicação que implementa o serviço de DNS. A solução projetada e implementada apresenta bons resultados na detecção, além de que a estratégia de defesa utilizada pode ser incorporada em qualquer aplicação que implementa o DNS.

5.2 DNSSec

Uma solução que está sendo projetada pelo IETF[1] desde 1994 (CHAR, 1994) e ainda não está sendo amplamente usada é o DNSSec (DAVIDOWICZ, 1999). DNSSEC é o nome dado às extensões de segurança que estão sendo propostas para o protocolo DNS definida pela RFC 2535. Atualmente, existem alguns grupos de pesquisa realizando experimentos com as novas características oferecidas pelo DNSSec. Alguns grupos que estão atuando na fase de experimentos, validando as definições transcritas na RFC 2535 são: (DNSEXT WG, 1996) (DNSOP WG, 1996). Detalhes de como utilizar o DNSSec com o bind 9 podem ser encontrados em (NEMETH, 2000) e (NLnet Labs, 2000).

A RFC 2535 estabelece os detalhes das alterações propostas, para a implantação de um DNS seguro. A característica principal deste novo modelo do DNS é prover autenticação da origem dos dados das zonas e verificar a integridade destes dados usando criptografia de chave pública (DAVIDOWICZ, 1999).

Alguns problemas existentes na utilização do DNSSec apresentados por (NEMETH, 2000) são:

- Deficiente estrutura de atualização de chaves públicas;
  - Uma estrutura de distribuição de chaves pública é necessária não só para o DNSSec, como também para outras aplicações como o IPSec;
  - O que deve ser feito se uma chave privada é violada ?
    - Deve-se gerar rapidamente um novo par de chaves (privada/pública), porém, haverá incompatibilidade com a chave antiga que está armazenada no *cache* de outros servidores.
- Performance;
  - Este trabalho relata as conseqüências da utilização de servidores DNS assinando requisições e respostas DNS. Tanto as requisições quanto às respostas são grandes. Por padrão, o DNS utiliza o UDP como protocolo de transporte. Se uma resposta é maior que 512 bytes, ela será truncada e enviada para o cliente, este por sua vez, reenviará uma requisição do mesmo nome utilizando TCP. O problema de utilizar TCP, é que este protocolo gera mais tráfego para gerenciar a transmissão dos dados pela rede. Se imaginarmos os servidores *ROOT-SERVERS* assinando todas respostas que enviam, (em torno de três bilhões por dia) a maior parte do tráfico DNS utilizaria TCP.
  - Validar assinaturas digitais também consome tempo de CPU e memória; O custo de processamento real gerado pela encriptação/decriptação de pacotes DNS depende do algoritmo utilizado. Com DSA-512 é possível assinar aproximadamente 135 dominios/Segundos em um PC de 500 Mhz com FreeBSD e utilizando o RSA-1024 temos uma taxa de aproximadamente 17 domínios/segundos e com o DSA-768 temos em média 62 domínios/segundos.

O laboratório NLnet está atualmente executando uma série de experimentos utilizando o DNSSec e os resultados destes experimentos podem ser encontrados em (Nlnet Labs, 2000).

Como podemos perceber, o DNSSec dispõe de uma série de recursos para prover segurança na comunicação entre servidores DNS. Porém, estes recursos ainda não estão sendo utilizados pela comunidade Internet devido à algumas limitações comentadas anteriormente. O custo de processamento vai ser um dos maiores obstáculos para a implantação do DNSSec. Para termos idéia do custo de processamento de um servidor DNS muito solicitado, podemos levar em conta a média de requisições recebidas por um servidor pertencente aos *ROOT-SERVERS*. Cada servidor recebe em média 272 milhões requisições por dia, em torno de 1.800.000 por minuto (ISC F-Roor-server, 1998). Se todas estas requisições e suas respectivas respostas fossem criptografadas o tempo de processamento seria consideravelmente aumentado o que torna impraticável o uso do DNS.

Desta forma, percebemos que o DNSSec é uma solução que está na fase de planejamento, testes, experimentos e que no futuro poderá ser utilizado. No entanto, ele não é uma opção que possa ser adotada para remediar os problemas de segurança atualmente encontrados na comunicação entre servidores DNS.

Para prover mecanismos de segurança que possam ser incorporados nas aplicações atuais que implementam o serviço de DNS, projetamos e implementamos uma solução, que visa detectar e inibir ataques de *DNS SPOOFING*.

6. CONCLUSÃO

Neste trabalho comentamos brevemente a aplicação do ataque *DNS SPOOFING* local e remoto no BIND das versões atuais (8.2.4, 9.1.3). Após realizar os experimentos das seqüências de ataques citadas na seção 4.3, concluímos que os servidores DNS de qualquer versão, configurados sob os sistemas operacionais ***Unix-Like*** e Microsoft Windows NT são vulneráveis a ataques deste gênero.

[1] IETF – Internet Engineering Task Force

Como medida de segurança projetamos e implementamos uma solução para detectar e bloquear ataques de *DNS SPOOFING*. A solução que elaboramos e implementamos foi incorporada dentro do código da aplicação *bind*. O objetivo dessa estratégia é prover uma solução que possa ser adotada dentro do próprio funcionamento do DNS, sem prejudicar o seu desempenho no processamento das mensagens e que possa, ao mesmo tempo, detectar e bloquear as possibilidades da aplicação do ataque *DNS SPOOFING* remoto e local.

De acordo com os resultados mostrados na figura 6 obtivemos bons resultados na detecção de ataques de *DNS SPOOFING* local e remoto, e comprovamos que existem facilidades nas aplicações atuais do DNS que permitem que ataques deste gênero sejam aplicados com alto percentual de sucesso.

Os testes foram realizados com os servidores DNS do Laboratório NatalNet e com os servidores DNS do laboratório de pós-graduação do DIMAP da UFRN.

REFERÊNCIAS BIBLIOGRÁFICAS

CHEUNG & LEVITT 2000 — CHEUNG, Steven e LEVITT, Karl N. **A Formal-Specification Based Approach for** Protecting **the Domain Name System.** Department of Computer Science University of California. Davis, CA 95616. www.cs.ucdavis.edu, 2000-06.

MOCKAPETRIS, 1987 — RFC 1035 MOCKAPETRIS, P. **Domain Names - Implementation and Specifications.** STD 13, RFC 1035. November 1987. (URL: ftp://ftp.isi.edu/in-notes/rfc1035.txt)

SACRAMENTO & MONTEIRO, 2000 — SACRAMENTO, Vagner e MONTEIRO, Cláudio. **Falsificação de endereços IP: Um Perigo que Ronda os Serviços de Resolução de Nomes na Internet**. Simpósio Sobre Segurança em Informática. ITA- Instituto Tecnológico de Aeronáutica. São José dos Campos – SP 24 a 26 de Outubro de 2000.

VIXIE, 1995 — VIXIE, P. **DNS and BIND Security Issues.** Proc. Of the 5th UNIX *Security Simposium*, June 5-7, 1995, pp.209-216.

CORE, 1997 — CORE. **SECURE NETWORKS Inc - CORE Seguridad de la Information.** April 22, 1997.

SCUT, 1999 — SCUT. **Skilled Security Group**. 1999. URL(http://teso.scene.at)

CA-98-05, 1998 — CA-98-05. CERT Coordination Center. **Multiple Vulnerabilities in BIND**. CERT Advisory CA-98-05, April 8, 1998.

CA-2001-01, 2001 — CA-2001-01. **Advisory CA-2001-01. Topic: Multiple Vulnerabilities in BIND**. CERT, Janeiro 2001. (URL: http://www.cert.org/advisories/CA-2001-02.html)

ACME, 1997 — ACME. LARSON, M. e LIU, C. **Using BIND: Don't get spoofed again**. SunWorld, November 1997. (URL: http://www.sunworld.com/swol-11-1997/swol-11-bind.html)

CHAR, 1994 — CHAR. IETF DNSSEC WG. **DNS Security (dnssec) Charter.** IETF, 1994.(URL:http://www.ietf.cnri.reston.va.us/proceedings/94dec/charters /dnssec-charter.html)

MEN & MICE, 1999 — MEN e MICE. **What is DNS Spoofing**. 1999. (URL:http://www.menandmice.com/infobase/mennmys/vefsidur.nsf /index/6.2.1.1)

RAFTER, 1997 — RAFTER, M. **Kashpureff, AlterNIC Founder, Busted for Hijacking DNS Traffic.** Internet World, November 10, 1997. (URL: http://www.internetworld.com/print /1997/11/10/news/ 19971110-busted.html)

SOARES & LEMOS & COLCHER, 1995 — SOARES, Luiz Fernando G. LEMOS, Guido. COLCHER, Sergio. **Redes de computadores: das LANs , MANs e WANs às redes ATM.** Rio de Janeiro: Campus, 1995.

NEMETH, 2000 — NEMETH, Evi. **Securing the DNS.** Proc. in USENIX. November 2000. (URL: http://www.usenix.org/publications/login/2000-11/index.html)

LIU, 1999 — LIU, Cricket. **Securing an Internet Name Server.** ACME & Wire LLC, 1999.

DAVIDOWICZ, 1999 — DAVIDOWICZ, Diane. **Domain Name System (DNS) Security**. 1999 (URL: http:// http://compsec101.antibozo.net/papers/dnssec/dnssec.html)

GALVIN, 1996 — GALVIN, James. **Public Key Distribution with Secure DNS**. 6th Usenix Security Symposium, 1996. (URL: http://www.usenix.org./publications/library/proceedings/sec96/galvin.html)

NLNET, 2000 — NLnet Labs. **DNS Resources**. Amsterdam, 01 jan, 2000. (URL: http://www.nlnetlabs.nl/dnssec).

DNSEXT, 1996 — DNSEXT. **Work Group IETF DNS Extensions Workgroup**. 1996. (URL: http://www.ietf.org/ html.charters/dnsext-charter.html)

DNSOP, 1996 — DNSOP. **Work Group IETF Domain Name Server Operations**. 1996. (URL: http://www.ietf.org /html.charters/dnsop-charter.html)

ISC F-Roor-server, 1998 — ISC F-Root-server. **Internet Software Consortiun (ISC)**. 1998. (URL: http://www.isc.org/services/public/F-root-server.html)

BELLOVIN, 1989 — S. M. BELLOVIN. **Security Problems in the TCP/IP Protocol Suite**. Computer Communicatins Review, Vol. 19, Nº 2, April 1989, pp.32-48.

BELLOVIN, 1995 — S. M. BELLOVIN. **Using the Domain Name System for System Break-ins**. Proc. Of the 5th UNIX Security Symposium, June 5-7, 1995, pp.199-208.

GAVRON, 1993 — E, Gavron. **A Security Problem and Proposed Correction with Widely Deployed DNS Software**. RFC 1535, October 1993.

SCHUBA & SPAFFORD, 1994 — C. L. Shuba, and E.H. Spafford. **Addressing Weaknesses in the Domain Name System Protocol**. Technical Report, Department of Computer Sciences, Purdue University, 1994.

DNS SERVER, 1999 — **MICROSOFT DNS SERVER** – Part of Service Pack 6:http://www.microsoft.com/ntserver/nts/downloads/recommended/SP6/allSP6.asp. 1999.

SACRAMENTO & BATISTA & LEMOS & MONTEIRO, 2001 — SACRAMENTO & BATISTA & LEMOS & MONTEIRO; **RELATÓRIO TÉCNICO – PROVENDO SEGURANÇA NA RESOLUÇÃO DE NOMES DO DNS;** Publicado no Departamento de Informática e Matemática Aplicada – DIMAP - UFRN, 2001.