INSTITUTO ENSINAR BRASIL
FACULDADES DOCTUM DE CARATINGA

JEFTE DE LIMA FERREIRA

IMPLEMENTAÇÃO DE SERVIDOR DE DNS RECURSIVO COM *SOFTWARE UNBOUND* E ANÁLISE DE DESEMPENHO COMPARATIVA: ESTUDO DE CASO EM UM PROVEDOR DE *INTERNET*

CARATINGA
2018

JEFTE DE LIMA FERREIRA

FACULDADES DOCTUM DE CARATINGA

IMPLEMENTAÇÃO DE SERVIDOR DE DNS RECURSIVO COM *SOFTWARE UNBOUND* E ANÁLISE DE DESEMPENHO COMPARATIVA: ESTUDO DE CASO EM UM PROVEDOR DE *INTERNET*

Trabalho de conclusão de curso apresentado ao curso de Ciência da Computação das Faculdades Doctum de Caratinga, como requisito parcial para obtenção do título de Bacharel em Ciência da Computação.

Área de Concentração: Redes de computadores

Orientador: Prof. Hudson Silva de Souza.

CARATINGA

2018

rede de ensino DOCTUM	FACULDADES DOCTUM DE CARATINGA	FORMULÁRIO 9
	TRABALHO DE CONCLUSÃO DE CURSO	
TERMO DE APROVAÇÃO		

TERMO DE APROVAÇÃO

O Trabalho de Conclusão de Curso intitulado: IMPLEMENTAÇÃO DE SERVIDOR DE DNS RECURSIVO COM SOFTWARE UNBOUND E ANÁLISE DE DESEMPENHO COMPARATIVA: ESTUDO DE CASO EM UM PROVEDOR DE INTERNET, elaborado pelo(s) aluno(s) JEFTE DE LIMA FERREIRA foi aprovado por todos os membros da Banca Examinadora e aceito pelo curso de CIÊNCIA DA COMPUTAÇÃO das FACULDADES DOCTUM DE CARATINGA, como requisito parcial da obtenção do título de

BACHAREL EM CIÊNCIA DA COMPUTAÇÃO.

Caratinga 12/12/2018

HUDSON SILVA
Prof. Orientador

MAICON RIBEIRO
Prof. Avaliador 1

GLAUBER COSTA
Prof. Examinador 2

AGRADECIMENTOS

Agradeço primeiramente aos meus familiares que, diretamente e indiretamente contribuíram para a realização deste trabalho, em especial aos meus pais que sempre acreditam que era possível chegar até aqui.

A todos os professores que contribuíram para o meu crescimento, através de seus ensinamentos, sendo pessoas nas quais obtive uma grande admiração, agradeço de uma forma especial o meu orientador professor Hudson Silva de Souza pelo apoio e empenho no qual contribuíram com a realização deste trabalho.

Agradeço a todos os amigos que fiz durante a graduação, em especial a todos que fizeram parte da turma de Ciência da Computação.

Agradeço aos responsáveis da empresa Prodata Informática, que possibilitaram a realização do estudo de caso.

A todos citados, meu sincero obrigado por fazerem parte deste trabalho.

“A maior recompensa para o trabalho do homem não é o que ele ganha com isso, mas o que ele se torna com isso.”

- John Ruskin.

LISTA DE SIGLAS

ADX – Portal de acesso à alunos e professores da rede de ensino DOCTUM

APT – *Advanced Packaging Tool*

ARPANET – *Advanced Research Projects Agency Network*

BIND – *Berkley Internet Name Domain*

BSD - *Berkeley Software Distribution*

CPF – Cadastro de Pessoa Física

DIG - *Domain Information Groper*

DNS – *Domain Name System*

DNSSEC – *Domain Name System Secutiry Extentions*

DoS – *Denial of Service*

DS – *Delegation Signer*

ECMP - *Equal-Cost Multi-Path Routing*

FOSS – *Free and Open-source Software*

FTP – *File Transfer Protocol*

GNU - *GNU is Not Unix*

GPL - *General Public License*

GTER - Grupo de Trabalho de Engenharia e Operação de Redes

HTTP – *Hypertex Transfer Protocol*

ICANN - *Internet Corporation for Assigned Names and Numbers*

ICMP - *Internet Control Message Protocol*

InterNIC - *Internet Network Information Center*

IP – *Internet Protocol*

IPv4 – Protocolo IP versão 4

IPv6 – Protocolo IP versão 6

MIT - *Massachusetts Institute of Technology*.

OSI – *Open Source Initiative*

OSI - *Open System Interconnection*

RFC - *Request for Comments*

RG – Registro Geral

SMTP – *Simple Mail Transfer Protocol*

SNMP - *Simple Network Management Protocol*

SSH - *Secure Shell*

TCP/IP - *Transmission Control Protocol / Internet Protocol*

TLD – *Top Level Domain*

URL – *Uniform Resource Locator*

VIM - *Vi Improved*

LISTA DE FIGURAS

LISTA DE GRÁFICOS

RESUMO

A comunicação, divulgação de marcas e o comércio eletrônico faz com que as empresas utilizem a *internet* de forma exaustiva, divulgando seus produtos e serviços através de *websites* e *e-mails*, tornando-se uma forma de alcançar públicos variados. Com o avanço tecnológico, a quantidade de dispositivos conectados à *internet* tende a crescer cada vez mais, sendo assim, a quantidade de dados trafegados pela *internet* aumenta a cada dia e a velocidade passa a ter cada vez mais importância para experiência dos usuários. Mediante a essa evolução o objetivo deste trabalho foi realizar um estudo de caso na empresa Prodata Informática e Cadastro Ltda no segmento de provedoria de *internet*, buscando obter um serviço de maior qualidade através da implementação de um novo servidor de DNS recursivo, utilizando o sistema operacional *GNU/Linux* Debian e o *software* de DNS *Unbound.* Com a proposta de avaliar o desempenho deste servidor, foram realizados testes comparativos de desempenho entre o servidor implementado e o servidor que a empresa já utilizava, através da ferramenta *Namebench*, podendo assim, obter uma análise detalhada do desempenho dos servidores, o que determinou a escolha do melhor serviço de DNS proporcionando maior qualidade e melhor tempo de resposta no serviço ofertado pela empresa.

Palavras-chave: *Opensource*. DNS. *Benchmark*. *Unbound*. BIND.

ABSTRACT

Communication, brand disclosure and e-commerce make companies use the internet in an exhaustive way, promoting their products and services through websites and e-mails, making it a way to reach different audiences. With the technological advance, the number of devices connected to the internet tends to grow more and more, so, the amount of data trafficked by the internet increases every day and speed is becoming increasingly important for users' experience. Through this evolution, the objective of this work was to carry out a case study in the company Prodata Informática and Cadastro Ltda in the Internet Provider segment, seeking to obtain a higher quality service through the implementation of a new recursive DNS server, using the operating system GNU / Linux Debian and the DNS Unbound software. With the proposal to evaluate the performance of this server, comparative performance tests were performed between the implemented server and the server that the company already used, using the Namebench tool, so that a detailed analysis of the servers' performance was obtained. choice of the best DNS service providing higher quality and better response time in the service offered by the company.

Keywords: *Opensource*. *DNS*. *Benchmark*. *Unbound*. BIND

SUMÁRIO

1 INTRODUÇÃO

Para Forouzan (2010, p.16) a *internet* revoluciona diversos aspectos no dia a dia, afetando a forma pela qual são realizados os negócios e a maneira como é utilizado o tempo de lazer. Enumerando as maneiras em que a *internet* é utilizada a ultimamente, é possível destacar as vezes em que são enviados *e-mails* a colegas de trabalho, quando são realizadas transações bancárias, divulgadas notícias ou até mesmo quando é possível ver a programação dos cinemas.

Segundo a empresa responsável pelos registros web no Brasil, o registro.br[1], houve cerca de 3.989.232 domínios de *internet* registrados em novembro de 2018. A *Internet World Stats*[2] estima que houve cerca de 4,2 bilhões de usuários na *internet* em 30 de junho 2018, cerca de 55,1% da população mundial.

O DNS (*Domain Name System*) é um sistema versátil, fascinante e uma peça fundamental das comunicações pela *internet*. Desempenhando um papel significativo na experiência do usuário na *internet*, geralmente, sendo o primeiro passo para estabelecer uma comunicação entre o remetente e o destinatário. Se a resolução de DNS estiver lenta ou não estiver funcionando, estes problemas são perceptíveis para os usuários (YU,2012).

A medição e análise de desempenho do serviço de DNS é significativa para a continuidade dos serviços comerciais e críticos de uma organização. A análise de capacidade do serviço de DNS através de técnicas sofisticadas auxilia na implantação de infraestrutura adequada, podendo garantir a continuidade dos serviços de uma empresa (AHMAD; SARWAR,2014).

Mediante a essa evolução e a importância do serviço de DNS para o funcionamento da *internet*, o objetivo deste estudo foi propor uma melhoria no serviço de DNS recursivo da empresa Prodata Informática e Cadastro Ltda, no segmento de provedoria de *internet*, de modo a promover uma maior qualidade nos serviços ofertados pela empresa. Sendo assim, realizado a implementação de um novo servidor de DNS recursivo, utilizando o *software Unbound e* um comparativo de desempenho em relação

[1] https://registro.br/estatisticas.html
[2] https://www.internetworldstats.com/stats.htm

ao servidor que a empresa já utilizava. Através da ferramenta *Namebench,* com a finalidade de obter uma análise detalhada de cada servidor, foram realizados testes comparativos de desempenho.

Para a produção deste trabalho foram realizados estudos sobre o protocolo DNS, com abordagem sobre seu funcionamento, a hierarquia de domínios, a importância do protocolo para o funcionamento da *internet*, os tipos de servidores de DNS e as vulnerabilidades do protocolo. Além de estudos relacionados às técnicas e finalidade dos testes de *benchmark*, apresentados no segundo capítulo. Kistowski (2015) cita que os *benchmarks* padronizados se tornaram aceitos para a comparação de hardwares e softwares, sendo utilizados regularmente para avaliar problemas em múltiplos campos da ciência da computação.

O terceiro capítulo tem a finalidade de abordar o objetivo do estudo de caso e o ambiente no qual foi realizado a pesquisa. Além de apresentar as etapas de implementação, análise de segurança e a execução dos testes.

Os resultados obtidos através dos testes de *benchmark* estão apresentados no quarto capítulo, onde é possível observar através de gráficos os valores de tempo médio de respostas que cada servidor obteve nos testes. Sendo possível obter uma análise detalhada e comparativa dos servidores.

O quinto capítulo teve a finalidade de expor a conclusão obtida após o estudo de caso, sendo possível destacar os benefícios que a pesquisa proporcionou e determinar a escolha do melhor serviço de DNS para a empresa, promovendo uma maior qualidade de navegação para os clientes.

2 REFERENCIAL TEÓRICO

Este capítulo tem a finalidade de tratar os assuntos referentes ao trabalho desenvolvido, onde serão apresentados seções com a abordagem sobre funcionamento do protocolo DNS, servidores, vulnerabilidades, *benchmarking* e os softwares *opensource* utilizados.

2.1 Servidores

Segundo Mitchell (2018) servidores são computadores projetados para processar requisições e fornecer serviços a outros dispositivos pela rede local ou até mesmo a *internet*. Os servidores *web* são os que melhor definem a palavra servidor, onde páginas *web* podem ser acessadas através da *internet* por um cliente através de um navegador. Porém, há diversos servidores que estão presentes em diversas ocasiões.

Os servidores podem ter várias funções e executar mais de um serviço, dentre os mais comuns estão os de compartilhamento de arquivos, de impressão, web, de acesso remoto, servidores de aplicações, de *e-mail*, banco de dados, de monitoramento entre outros (WILEY; SONS ,2011).

A palavra servidor muitas vezes é referenciada por máquinas com grande poder de processamento. Pode-se dizer que servidor na maioria das vezes é o *software* que executa uma tarefa específica, sendo apto a fornecer serviço aos clientes e que em diversas das situações necessitam de um *hardware* poderoso, sendo capaz de manter serviços ininterruptamente e em qualidade satisfatória para milhões de usuários (MITCHELL,2018).

Para Wiley e Sons (2011) antes de selecionar os componentes para fazer parte de um servidor, é necessário entender qual a funcionalidade do mesmo e em seguida analisar a quantidade de clientes que farão o uso deste serviço, para que seja possível determinar qual *hardware* deve ser utilizado a pensar em uma capacidade de expansão futura.

A próxima seção abordará assuntos referentes ao endereçamento IP (*Internet Protocol*) responsável pela comunicação entre os dispositivos de rede.

2.2 *Internet Protocol*

De acordo com Torres (2014) para que seja possível entregar um pacote de dados entre dispositivos conectados à mesma rede deve-se saber o endereço do destinatário para que ocorra a troca de informações. Cada máquina possui dois endereços de rede: um endereço físico, denominado por MAC (*Media Access Control*) que é gravado na sua placa de rede e um endereço lógico, o endereço IP (*Internet Protocol*) que é atrelado e configurado por *software*.

Ainda, Torres (2014) afirma que para que uma rede seja capaz de enviar informações entre os dispositivos, necessita-se que cada um destes tenham um endereço único e exclusivo para que não haja conflito ou um congestionamento na entrega dos pacotes. Um endereço IP funciona de forma organizada e padronizada, sendo possível a configuração na versão 4 que é utilizado pela maioria dos computadores e a versão 6 sendo menos utilizada.

> Na *internet*, cada host e cada roteador tem um endereço IP que codifica seu número de rede e seu número de host. A combinação é exclusiva: em princípio, duas máquinas na *internet* nunca têm o mesmo endereço IP (TANENBAUM,2011, p.337).

Segundo Korose e Ross (2014) o endereço IPV4 tem o tamanho de 32 *bits*. Sendo possível totalizar 2^{32} endereços IP possíveis de serem gerados, cerca de 4 bilhões destes. São escritos em forma decimal separado por pontos, onde cada *byte* do endereço é descrito da sua forma decimal separada por outro *byte* do endereço. Por exemplo, 192.168.10.155, onde o 192 corresponde ao número decimal dos primeiros 8 *bits*.

Com a quantidade de dispositivos que acessam a *internet* cresce a cada dia, o IPv6 (*Internet Protocol* Versão 6) dá a possibilidade de atrelar um endereço único para cada dispositivo conectado à rede, contendo um endereçamento de 128 *bits*, sendo possível gerar cerca de 340 undecilhões de IPs. Os endereços IPv6 são constituídos por grupos de 16 *bits* separados por dois pontos ‘:’ cada número de 16 *bits* é representado por quatro algarismos hexadecimais (TORRES, 2014).

A seção seguinte trará uma abordagem sobre o protocolo DNS, explicitando o funcionamento da hierarquia de domínios e interação entre os servidores de DNS.

2.3 DNS

Algumas pessoas possuem registros únicos seja o CPF (Cadastro de Pessoal Física) ou RG (Registro Geral), o que os tornam únicos perante os outros. Mesmo sabendo que cada pessoa possui um registro único ainda se prefere a identificação diária através de nomes, por exemplo: João, Pedro ou Maria.

Com os computadores não é diferente. Cada computador pode ser identificado pelo seu nome de hospedeiro(*hostname*), por exemplo: '*adx.doctum.edu.br'*, '*localhost'*, '*pc-joão'* ou pelo seu identificador único de rede que é o seu endereço IP, por exemplo '*38.225.49.50'*. Os dispositivos de redes foram projetados para se comunicar através do seu identificador único que é o IP, pelo fato de ser menos custoso o processo para analisar cada caractere do seu nome, tornando assim mais rápido a identificação. (TANENBAUM,2011).

O portal dos alunos da Rede Doctum denominado ADX, é um sistema *online* onde os alunos e professores para obterem acesso devem possuir um usuário único e senha além de acessarem pelo navegador *web* o seu endereço '*https://adx.doctum.edu.br*'.

Seja um cenário onde o portal do aluno ADX fosse somente possível acessar através do seu identificador único que seria o seu IP, por exemplo: 'http://*38.225.49.50'*. O aluno requisitante além de gravar o seu o usuário e senha terá que memorizar o endereço do portal em forma de números. Caso a empresa responsável pelo portal do ADX trocasse esse IP por algum motivo os alunos teriam que gravar um novo IP do portal.

De acordo com Tanenbaum (2011) no passado com a ARPANET (*Advanced Research Projects Agency Network*) existia somente um arquivo chamado *host.txt* que listava todos os nomes de computadores e IPs. Este arquivo era acessado por todos *hosts* no local onde era mantido. Com o crescimento das redes de computadores foi identificado que desta forma a pesquisa poderia não funcionar corretamente, o arquivo ficaria muito grande tornando difícil a consulta, a organização e aumentando o tempo de resposta. Pensando nestas questões em 1983 foi criado o DNS (*Domain Name System*).

> O DNS é um banco de dados distribuído executado em uma hierarquia de servidores de DNS, e um protocolo de camada de aplicação que permite que hospedeiros consultem o banco de dados distribuído. Os servidores DNS são muitas vezes máquinas UNIX que executam o *software* BIND (*Berkeley Internet Name Domain*) [BIND, 2012]. O protocolo DNS utiliza UDP e usa a porta 53(KUROSE; ROSS, 2014, p. 97).

Segundo Kurose e Ross (2014) o DNS costuma ser utilizado por outros serviços da camada de aplicação, os mais utilizados como HTTP (*Hypertext Transfer Protocol*), SMTP (*Simple Mail Transfer Protocol*) e FTP (*File Transfer Protocol*) para traduzir os nomes de *hosts* para endereços IPs. Considere-se o exemplo do que acontece quando utilizamos o navegador (ou seja, um cliente HTTP) para requisitar no URL (*Uniform Resource Locator*) *adx.doctum.edu.br*. Para que a máquina cliente possa enviar requisições para o portal ADX ela precisa obter o seu endereço IP.

A seção seguinte busca descrever os servidores de DNS que compõem a hierarquia de domínios.

2.3.1 Tipo de servidores de DNS

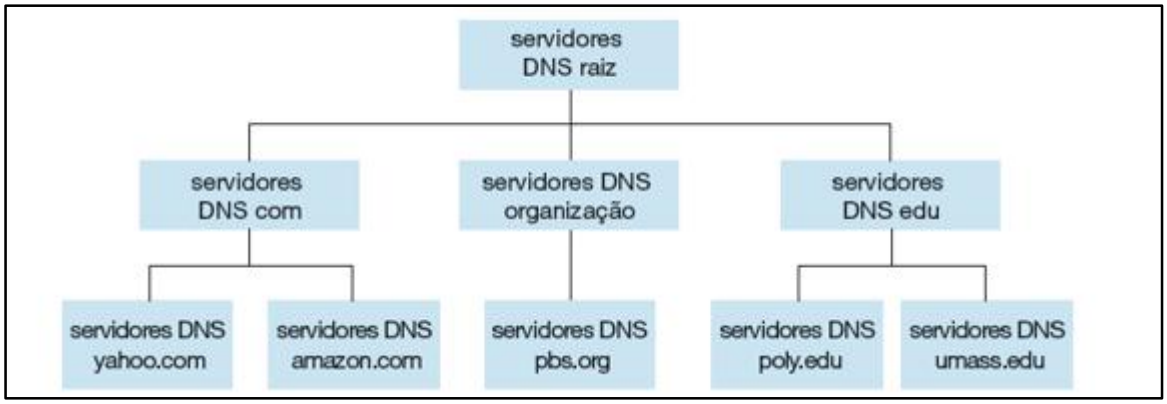
Segundo Tanenbaum (2011) em teoria, um único servidor de nomes poderia conter o banco de dados DNS completo e ser capaz de responder a todas consultas. Entretanto, na prática, esse servidor ficaria sobrecarregado ao ponto de ser inutilizável. Caso houvesse uma interrupção deste servidor, toda a *internet* ficaria *offline*.

Para Kurose e Ross (2014) o DNS faz o uso de um número grande de servidores, organizados e distribuídos hierarquicamente pelo mundo todo. Nenhum servidor DNS detém todas as informações de todos os hospedeiros da *internet*, estas informações são distribuídas e mapeadas de forma inteligente entre os servidores existentes.

> Além de facilitar a delegação de autoridade, a hierarquia de uma grande organização apresenta operação autônoma. Por exemplo, quando um funcionário precisa de informações como o número de telefone de um novo empregado, ele começa perguntando aos funcionários administrativos locais (que podem contatar funcionários administrativos em outras divisões). O ponto é que, embora a autoridade sempre passe para baixo na hierarquia corporativa, a informação pode fluir pela hierarquia de um cargo para outro (COMER, 2015. p.316).

Ainda Kurose e Ross (2014) citam que os mapeamentos são distribuídos em três classes de servidores como exemplifica a Figura 1. A camada superior onde se encontra os servidores raízes que estão no topo da hierarquia, os TLDs (*Top Level Domain*) e por fim os Autoritativos.

Figura 1 - Exemplo de hierarquia de domínios



Fonte: Redes de Computadores e a *Internet*. 6ª Edição, pág 99

Os tópicos a seguir buscam abordar a funcionalidade de cada tipo de servidor:

- Os *root servers* são responsáveis pela zona iniciada com '.' (*ponto*) onde tudo começa na hierarquia de identificação de domínios. Atualmente no mundo há 13 servidores de DNS raízes, classificados de A à M. Por questões de segurança e confiabilidade nos serviços, existem diversas réplicas destes servidores no mundo. (KUROSE; ROSS,2014).

Através do *site www.root-servers.org* é possível encontrar os locais das réplicas existentes, sendo possível observar pela Figura 2 onde as cores de amarelo e laranja representam os servidores em IPv4 e verde os em IPv6. Os números correspondem a quantidade de réplicas existente naquela localidade, já as letras L e K demonstram que há somente um servidor naquela região, como por exemplo: *Papeete, Apia e Majuro* que são pequenas ilhas.

Figura 2 - Localização dos *root servers* e suas réplicas

Fonte: www.root-servers.org

Ainda de acordo com o *site root-servers.org*, em novembro de 2018 os *root servers* estão sendo operados por 12 organizações independentes, totalizando 919 instâncias.

- Servidores de DNS TLD (*Top Level Domain*) são servidores responsáveis por manter informações sobre os domínios de alto nível, como, por exemplo *'.com', '.org', '.net', '.edu', '.gov'* e por todos os domínios referentes as siglas de países, *'.br', '.uk', '.fr', '.cz'*. A empresa *Verisign Global Registry Services* é a responsável por manter o alto nível *'.com'*, um dos mais utilizados no mundo todo (KUROSE; ROSS,2014).
- Servidores de DNS Autoritativos, qualquer empresa que possua algum hospedeiro que deseja ser acessado publicamente através da *internet*, como exemplo, servidores *web* ou *e-mail*, deve se registrar a um órgão responsável por gerenciar aquela localidade, criando assim o seu próprio domínio e passando as suas configurações de acesso necessárias referente aos seus serviços. No Brasil temos o *'Registro.br'*, responsável por manter a zona *'.br'* (KUROSE; ROSS,2014).

A próxima seção tem a finalidade abordar sobre os servidores de DNS recursivos, que são responsáveis por enviar requisições aos *root servers* e armazenando as informações em *cache*.

2.3.2 DNS Recursivos

De acordo com a ISC (2018) as bibliotecas de *resolvers* disponibilizadas pela maioria dos sistemas operacionais não são capazes de executar o processo completo de resolução de DNS, não sendo possível realizar a comunicação direta com os servidores autoritativos. Esse sistema depende de um servidor de nomes local para executar a resolução de nomes. Sendo mais conhecido como servidor de nomes recursivo, responsável por realizar consultas de forma recursiva para clientes locais.

As próximas seções trarão informações sobre os serviços públicos de servidores recursivos. Abordando dois dos principais serviços gratuitos, como *GoogleDNS* e *OpenDNS.* Servidores utilizados como uma alternativa por provedores de *internet* como *resolvedores* de nomes ou até mesmo clientes finais.

2.3.2.1 *OpenDNS*

O *OpenDNS* é um serviço gratuito de resolução de DNS recursiva, seu grande proprietário é a *Cisco System*, lançado em julho de 2006 por David Ulevitch, com sede em São Francisco na Califórnia, a empresa arrecada parte de sua receita enviando os usuários a uma página de busca própria quando os nomes de domínios não são validos através de propagandas, os servidores estão disponíveis para uso nos endereços IPv4 e IPv6 sendo possível obter estes números no *site* oficial da empresa (OPENDNS,2018). Apesar do nome O*penDNS,* não se trata de um *software* de código-fonte aberto, a empresa arrecada capital através de propagandas mencionadas anteriormente.

De acordo com o *site dnsperf.com* o *OpenDNS* na categoria *Public DNS Resolvers* no tipo de avaliação *performance* encontra-se em segundo lugar do *rank*, perdendo somente para o serviço da *Cloudflare*.

A próxima seção apresentará informações sobre o serviço de DNS oferecido pela *Google*.

2.3.2.2 *GoogleDNS*

De acordo com Turgut (2015) o *Google* afirma oferecer serviço de DNS gratuito e rápido a seus clientes. Apesar de sua popularidade, há poucas informações a respeito do serviço, podendo ser identificado como uma ‘caixa-preta’. O DNS público do *Google* é um serviço gratuito podendo ser utilizado como uma alternativa aos provedores de *internet* locais. Usando endereços globais como 8.8.8.8 e 8.8.4.4 para receber consultas dos clientes. Os DNS globais do Google estão espalhados pelo mundo e o trafego é roteado pela rota mais curta, visto da perspectiva do cliente.

O *Google* utiliza três métodos principais para atenuar a latência do DNS. Usando servidores poderosos, usando *edns-cliente-subnet*[3] e uma alta coerência de *cache*. O DNS do *Google* possui dois níveis de *cache*, sendo *cache* de nível 1 um pequeno *cache* por máquina contendo os mais populares domínios. Se uma consulta não for satisfeita pelo cache de nível 1, ela será encaminhada para outras máquinas do *pool*, nas quais o *cache* é dividido por nomes. Cada consulta para o mesmo nome é manipulada pela mesma máquina (TURGUT,2015).

A seção seguinte apresentará como a interação entre os servidores resolvem as consultas de DNS.

2.3.3 Interação entre servidores de DNS

A Figura 3 exemplifica um cenário onde um aluno deseja acessar o portal do ADX, através do seu endereço: *adx.doctum.edu.br* e em seu computador está configurado o servidor de DNS do *Google*.

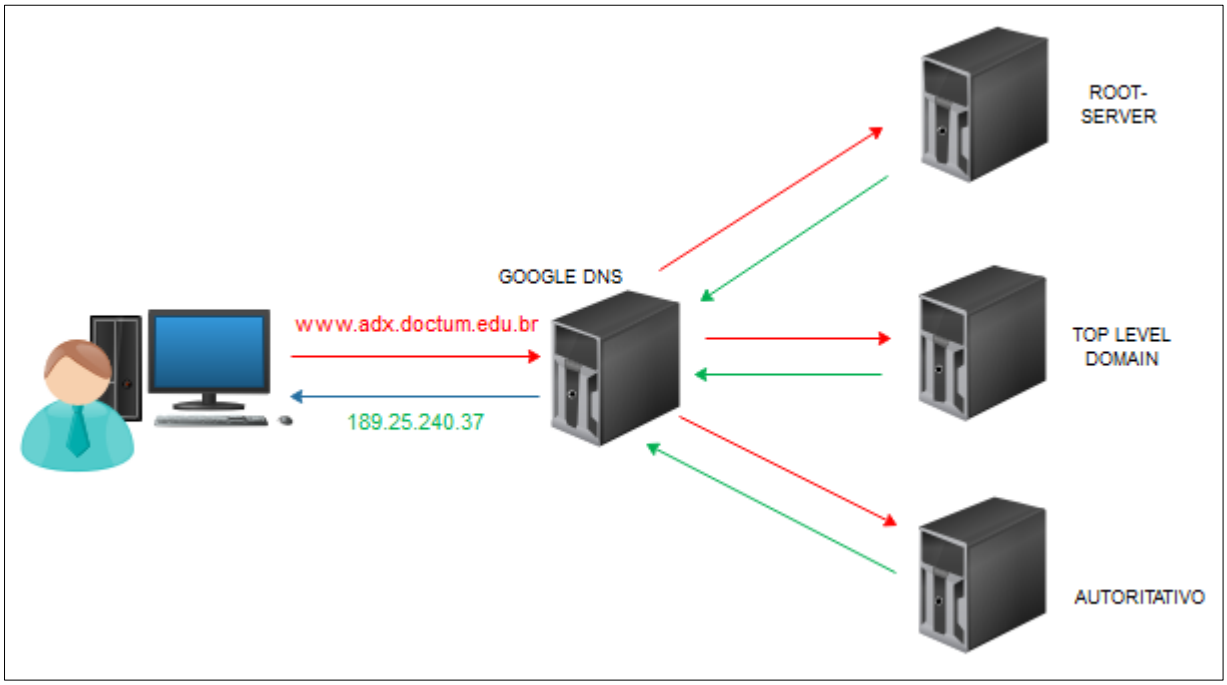
> Os usuários da *internet* não conhecem o endereço IP de nenhum serviço, mas conhecem o seu nome. O PC desses usuários, por sua vez, precisa do endereço IP para acessar o serviço. Dessa forma, uma tradução nome-IP deverá ocorrer (BATISTA,2013, p.68).

Neste caso o computador do aluno envia uma mensagem de consulta ao servidor configurado em sua máquina, contendo o nome de hospedeiro que deseja ser acessado,

[3] Trata-se de uma extensão do DNS que possibilita que um servidor de DNS recursivo especifique a sub-rede para o *host* em qual nome está realizando uma consulta. Possibilitando um aceleramento na entrega dos dados, proporcionando um melhor uso do balanceamento de carga baseado em DNS (RFC-7871).

em seguida o servidor *Google* transmite a mensagem para um servidor DNS raiz que percebe o sufixo ‘.br’ e retorna ao servidor *Google* uma lista de endereços IP dos servidores TLD responsáveis pela zona ‘.br’. Em seguida o servidor *Google* retransmite a mensagem de consulta a um dos servidores TLD que percebem o sufixo ‘edu’ e responde com o endereço do servidor DNS autoritativo, onde está hospedado o portal ADX. Por fim, o servidor *Google* envia uma mensagem de consulta direta ao adx.doctum.edu.br, que é respondida com o seu endereço IP e retransmitida a resposta final ao cliente requisitante (KUROSE; ROSS, 2014).

Figura 3 - Interação entre os servidores de DNS



Fonte: Próprio autor

Para obter o mapeamento completo de um endereço, neste exemplo foram utilizadas oito mensagens de DNS, quatro de consultas e quatro de respostas (KUROSE; ROSS, 2014).

A seção a seguir mostrará como o *cache* de DNS é importante para reduzir o tempo de respostas das consultas, trazendo uma melhor experiência para os usuários de *internet*.

2.3.4 DNS *Cache*

De acordo com KUROSE (2014) o *cache* DNS é sem dúvida uma característica muito importante do protocolo DNS, e que o DNS explora extensivamente o *cache* para melhorar o desempenho e reduzir o número de mensagens que são enviadas pela *internet*.

Ainda Kurose (2014) afirma que a ideia por trás do *cache* DNS é simples. Um servidor de DNS quando recebe em uma cadeia de consultas, respostas contendo o mapeamento de um nome de hospedeiro para um endereço IP, o mesmo faz o *cache* das informações da resposta em sua memória local. Por exemplo, anteriormente na Figura 3, toda vez que o servidor DNS *Google* recebe uma resposta de algum servidor superior, o mesmo poderá fazer o *cache* de qualquer informação contida na resposta. Se um par hospedeiro/endereço IP estiver em *cache* de um servidor DNS e outra consulta chegar em um espaço curto de tempo no mesmo servidor, o mesmo poderá fornecer o endereço IP da consulta requisitada mesmo não tendo autoridade para esse domínio, devido à informação estar salva em *cache*.

> O *caching* funciona bem no Domain Name System por que os vínculos entre nomes e endereço mudam com pouca frequência. No entanto, eles mudam. Se os servidores colocassem informações em *cache* na primeira vez que ela fosse solicitada e nunca a atualizassem, as entradas no *cache* ficariam passadas (ou seja, incorretas) (COMER, 2015. P 322).

Como todos os mapeamentos de hospedeiros e endereço IP não são permanentes e podem ocorrer mudanças dessas informações, após um período os servidores DNS descartam as informações armazenadas em *cache* (KUROSE; ROSS,2014).

Liu e Albitz (2006) ressaltam que os administradores da zona que contém os dados decidem o tempo de vida dos dados. Onde o tempo de vida é a quantidade de tempo que o servidor de DNS tem permissão para armazenar os dados em *cache*. Isso também é aplicado em dados negativamente armazenados em *cache*, ou seja, um servidor deve expirar uma resposta negativa após um certo período, caso dados novos tenham sidos adicionados nos servidores de DNS autoritativos.

A seção seguir abordará assuntos sobre a segurança do protocolo DNS, sendo uma extensão do serviço de DNS, o DNSSEC (*Domain Name System Security Extensions*).

2.3.5 DNSSEC

Quando um servidor de DNS recebe a resposta de uma consulta para um domínio, por exemplo, *'adx.doctum.edu.br'* espera-se que os dados estejam corretos. Porém, não há como provar a veracidade dos dados sem o uso do DNSSEC, pois, estes dados podem estar errados ou falsificados de várias maneiras. A resposta da consulta pode ter um fornecedor de uma zona envenenada, ou pode ter sido interceptada modificando a resposta. (AITCHISON,2011).

Para Kuroiwa (2012) o DNSSEC é uma extensão da tecnologia DNS, possibilitando maior segurança para os usuários na *internet*, corrigindo falhas do protocolo DNS, garantindo a autenticidade e a integridade da origem das informações. O DNSSEC utiliza o conceito de chave assimétrica[4]. Com a extensão DNSSEC os novos *Resource Records* são divididos em quatro etapas:

- DNSKEY – (incluída na própria zona) representa a chave pública.
- RRSIG – Representa a assinatura de um RRset especifico com uma chave DNSKEY (*somente registro com autoridade*) possui uma validade inicial (*inception*) e final (*expiration*).
- DS – *Delegation Signer*, O *Record* DS forma uma cadeia de confiança, garantindo a autenticidade das delegações de uma zona até um ponto de confiança (uma espécie de chave ancorada).
- NSEC(3) *Next Secure* permite autenticar uma resposta negativa, indica o próximo nome seguro na zona e os tipos de RRsets existentes para o nome, de forma circular (*o último aponta para o primeiro*).

A F5Network (2015) diz que o uso do DNSSEC garante que a resposta recebida ao solicitar uma resolução de nome é de fonte confiável.

Para habilitar o DNSSEC em um servidor de DNS recursivo é necessário ancorar uma chave pública que serve como início da cadeia de confiança. Para visualizar essa

[4] Modelo de criptografia na qual cada parte envolvida na comunicação usa duas chaves diferentes (assimétricas) e complementares, uma privada e outra pública. Neste caso, as chaves não são apenas senhas, mas arquivos digitais mais complexos. A chave pública pode ficar disponível para qualquer pessoa que queira se comunicar com outra de modo seguro, mas a chave privada deverá ficar em poder apenas de cada titular. É com a chave privada que o destinatário poderá decodificar uma mensagem que foi criptografada para ele com sua respectiva chave pública (OLIVEIRA, 2012. p 3-4).

chave é preciso acessar o *site* oficial do registro.br. A chave também podem ser encontrada no *site* oficial da IANA que é a organização mundial que supervisiona a atribuição global dos números na *internet* como portas, endereços IP, root servers e recursos referentes aos protocolos de *internet*. O registro.br foi em 2007 um dos pioneiros a adotarem o DNSSEC desde então utilizam as chaves KSK em seus servidores (KUROIWA,2012).

A seção seguinte trará informações referentes ao protocolo UDP (*User Datagram Protocol*), protocolo principal e responsável pelo funcionamento de diversas aplicações, sendo muito utilizado no protocolo de DNS.

2.3.6 *User Datagram Protocol*

Segundo Comer (2015) no conjunto de protocolos TCP/IP (*Transmission Control Protocol/Internet Protocol*), o UDP oferece um mecanismo para enviar datagramas de programas aplicativos de uns para os outros. As mensagens de UDP são compostas de número de porta de protocolo usadas para diferenciar vários aplicativos em execução em um único dispositivo. Ou seja, além das informações enviadas, cada mensagem UDP possui um número de portas de destino e origem, tornando possível para o *software* UDP no destino realizar a entrega de uma mensagem de entrada ao destinatário corretamente e para seu receptor uma resposta.

O UDP não garante a chegada das mensagens, nem que as mesmas cheguem na ordem correta em que foram enviadas, isto se deve ao fato que o UDP utiliza o protocolo *internet* subjacente para transportar uma mensagem, fornecendo uma semântica de entrega de *datagramas* sem conexão por melhor esforço que o TCP/IP. Com essas características do protocolo UDP, as mensagens podem ser duplicadas, perdidas, chegar fora de ordem ou até mesmo chegar antes que o destinatário possa ser capaz de processá-las (COMER,2015).

> Para aplicações que precisam ter controle preciso sobre o fluxo de pacotes, os erros ou a sincronização, o UDP fornece apenas aquilo que é determinado. Uma área em que ele é especialmente útil é nas situações cliente-servidor. Normalmente, o cliente envia uma solicitação curta para o servidor e espera uma resposta curta de volta. Se a solicitação ou a resposta se perderem, o cliente pode entrar em *timeout* e tentar novamente. (TANENBAUM,2011. p.341).

Para Forouzan (2010) o UDP é adequado para processos que necessitam de comunicações do tipo solicitação-resposta simples, com pequena preocupação com controles de fluxos e erros, exigindo uma menor interação entre emissor e receptor, sendo muito utilizado no gerenciamento de redes com o protocolo SNMP, também muito utilizado no protocolo de DNS.

A seção a seguir terá uma abordagem sobre as vulnerabilidades em servidores, em especial os problemas que podem ocorrer no serviço de DNS.

2.4 Vulnerabilidades

Segundo Lyra (2015) a vulnerabilidade é algo íntimo ao ponto frágil de um dispositivo que pode ser entendida como uma fragilidade. Considera-se um erro em algum procedimento, falha de algum componente ou uma configuração errada dos aplicativos, gerando assim um sistema não confiável.

De acordo com Martinelo e Bellezi (2014) as vulnerabilidades na maioria das vezes são de origens de falhas não intencionadas, podendo correr estas falhas em camadas:

- Físicas: Onde ocorre o acesso aos ativos por pessoas não autorizadas.
- *Hardware*: Onde algum componente físico venha a se danificar, ocasionando a indisponibilidade do sistema ou perda dos dados.
- Naturais: Onde desastres naturais comprometem a segurança do serviço e dados armazenados.
- Humanas: Onde o operador de forma errônea prejudica o funcionamento ou a perda das informações.
- *Software*: Quando ocorrem falhas na programação do sistema, ocasionando brechas exploradas de forma a prejudicar o seu funcionamento.

O serviço de DNS assim como diversos outros possuem vulnerabilidades que quando exploradas podem acarretar instabilidades, interoperabilidade ou até mesmo o sequestro do serviço.

As seções a seguir abordarão os principais problemas recorrentes que devem ser prevenidos de forma estratégica para que não ocorram ou sejam amenizados.

2.4.1 *Denial of Service*

Para Mitrokotsa (2003) os ataques DoS (*Denial of Service*) conhecidos como ataques de negação de serviço, são um dos problemas mais sérios existentes hoje na *internet*. A principal função do DoS é a perturbação dos serviços, fazendo com que o acesso seja limitado ou até mesmo inalcançável, este tipo de ataque é uma técnica simples, porém, muito eficaz, os atacantes enviam um fluxo enorme de pacotes de fontes diferentes aos serviços, realizando uma inundação na entrega, tornando os recursos dos serviços esgotados com o fluxo de rede, processamento, consumo de memória ou disco. Os ataques de DoS muitas vezes exploram as fraquezas dos sistemas que muitas vezes estão abertas para realizações destes ataques. Os ataques na maioria das vezes ocorrem com fluxos de origem não identificadas e do mundo todo, sendo assim, difícil a identificação dos pacotes que são de importância aos que são de ataque.

Assim como DoS pode ocasionar diversos problemas em um serviço, o DDoS (*Distributed Denial of Service*) pode ocasionar de forma pior, sendo uma evolução do DoS de forma distribuída. Muitas vezes difícil de ser controlada, pois, os ataques são de forma distribuída e coordenadas (MITROKOTSA,2003).

Hoepers (2016) ressalta que uma das técnicas de DDoS envolve uma exploração de servidores de DNS recursivos abertos, gerando uma grande quantidade de tráfego de respostas DNS para uma vítima com endereço IP que está sendo forjado. Um dos problemas explorados neste tipo de ataque é devido o sistema de DNS utilizar o protocolo UDP (*User Datagram Protocol*) como protocolo principal de comunicação. Devido protocolo UDP não requerer o estabelecimento de uma sessão entre o cliente e servidor e não possuir um método de autenticação isto facilita o forjamento da origem de uma consulta DNS.

A próxima seção trará uma abordagem a respeito das principais ameaças que o protocolo DNS pode sofrer, tais como envenenamento de *cache*, *phishing e spam*, sendo vulnerabilidade que pode ocasionar diversos incidentes na *internet*.

2.4.2 Ameaças

De acordo com F5Network (2015) o DNS funcionou muito bem desde a sua criação, mas como quase tudo que está relacionado à *internet*, foram encontradas maneiras de explorar as vulnerabilidades do protocolo. Uma vulnerabilidade conhecida é o envenenamento de *cache*. O envenenamento ocorre quando é requisitado um URL no navegador, o resolvedor realiza a tradução do nome para o IP e normalmente o DNS aceita a primeira resposta sem questionar, armazenando em *cache* essas informações por um certo período. Quando essas entradas são interceptadas e forjadas, as mesmas são armazenadas em *cache*, realizando assim um envenenamento.

Os atacantes produzem e enviam exaustivamente os pacotes de resposta forjados para os resolvedores, até que um pacote seja o mesmo daquele gerado pelo servidor de autoridade. Entretanto, uma contramedida para esses ataques é aumentar a complexidade dos pacotes serem adivinhados, por exemplo, o ID de transação aleatória e randomização de portas. Se a complexidade aumentar, a contramedida é mais forte contra ataques de envenenamento de cache (SUN et al,2009).

Para Son e Shmatikov (2010) muitos mecanismos de segurança da *internet* incluindo controle de acessos a *hosts*, defesa contra *spam* e *phishing* explicitamente depende da integridade do funcionamento do DNS, e que infelizmente a segurança não foi uma das considerações de design na construção do DNS.

Segundo Martinelo e Bellize (2014) Spam são e-mails enviados em massa que geram grande tráfego desnecessário nas redes. São muito utilizados para divulgação excessiva de produtos que na maioria das vezes são golpes aplicados pela *internet*, com a intenção de disseminar *malwares*.

Phishing são *e-mais* falsificados que se passam por mensagens de fontes confiáveis, como órgãos públicos, bancos, etc. Com a intensão de induzir o usuário ao erro, instalando *malwares* ou acessando páginas *web* falsificadas a fim de obter os dados pessoais da vítima como senhas e números de cartões de créditos. (MARTINELO; BELLIZE,2014).

> Envenenamento de DNS é um ataque que forja um endereço falso no servidor de DNS. Por exemplo: um registro de DNS www.meubanco.com.br configurado para o endereço 200.200.200.200 (servidor do atacante) ao invés do endereço correto 200.154.20.1. Assim, o atacante pode capturar senhas e números de cartões de crédito utilizando páginas clones do *site* original. (MARTINELO e BELLIZE, 2014. p.37).

A seção seguinte trará informações sobre o projeto *opensource*, com uma abordagem sobre a iniciativa do projeto, as licenças de códigos e os softwares utilizados neste trabalho.

2.5 *Opensource*

Segundo a IDABC (2010) o *software* de código-fonte aberto é um *software* como outro qualquer. Sendo diferenciado pela sua licença ou termos de uso que garante algumas liberdades, do contrário de softwares proprietários fechado que restringe direitos de liberdade. O *software opensource* garante o direito de acesso, uso, modificação do código-fonte, reutilização e redistribuição do *software*, sem cobrança financeira. Em alguns casos podem existir a obrigação de compartilhar melhorias com a comunidade, garantindo benefícios globais ao *software*.

A IDABC (2010) define que algumas garantias têm implicações poderosas tais como:

- Incentivar a reutilização.
- Permitir inovações, flexibilidade, integrações mais simples.
- Reduzir o preço do *software* a zero.
- Não ocultação de defeitos e vulnerabilidades devido a não monopolização.
- Não tendo fornecedor único, significando em suporte e escolhas de serviços.

Para a (FSF,2007) o *software* de código-fonte aberto é licenciado sob termos que permitem ao usuário praticar quatro liberdades, sendo elas:

- Usar o *software* sem restrição de acesso, seguindo os termos da licença aplicada
- Obter acesso e visualizar o código-fonte

- Prover melhorias e adicionar ao código-fonte, seguindo os termos da licença aplicada podendo incluir um termo obrigatório com o código modificado a comunidade.
- Distribuir o código-fonte.

Ainda a IDABC (2010) cita que a iniciativa de código-fonte aberto OSI (*Open Source Initiative*) é reconhecida mundialmente com a autoridade de certificar se uma licença é verdadeiramente de código-fonte aberto. Embora existam muitas licenças de código-fonte aberto, a maioria dos softwares usados empunham licenças comuns. Significando que uma sobrecarga legal e comercial com a finalidade de atender e gerenciar licenças de código-fonte aberto é reduzida.

A comunidade de computação distingue o termo 'livre' significando zero custo e livre a liberdade e garantias do *software*. A seção seguinte trará informações sobre a GPL (*General Public License*), sendo uma licença muito utilizada em *software* de código-fonte aberto. (IDABC,2010).

A próxima seção terá uma abordagem a respeito de uma das licenças mais conhecidas no cenário *opensource*, a GPL.

2.5.1 *General Public License*

A Licença Pública Geral do GNU (*GNU is Not Unix*) é a preferida para projetos autorizados pela FSF (*Free Software Foundation)* sendo aplicada em diversos projetos como editor GNU *Emacs*, compilador GNU C e o *kernel* Linux. Algumas licenças de softwares são projetadas para tirar a liberdade de compartilhar e mudar o *software*, do contrário a GPL tem o objetivo de garantir a liberdade de compartilhar e alterar o *software* de modo a garantir que o *software* seja livre para todos. As licenças públicas são projetadas para garantir que você tenha a liberdade de distribuir o *software* (FSF,2007).

Segundo Tsai (2008) o movimento do *software* livre deu início nos anos 70, quando Richard Stallman trabalhava como programador no laboratório de Inteligência Artificial do MIT (*Massachusetts Institute of Technology*). Richard resolveu um problema de atolamento de papel da impressora compartilhada no laboratório, com acesso ao código-fonte do *software* da impressora, modificou o *software* para que fosse realizado uma

notificação a todos os membros do laboratório quando a impressora emperrava. Quando o laboratório recebeu uma nova impressora, Richard tentou melhorar a impressora da mesma forma, porém, a empresa não liberava o código-fonte da impressora. Sendo assim Richard acreditava que o *software* proprietário era incompatível com a sua concepção, onde o código-fonte tem a obrigação moral de ser compartilhado, marcando assim a sua visão ao movimento do *software* livre.

> "Eu projetei a Licença Pública Geral GNU para um propósito muito simples: defender a liberdade de todo usuário de um programa livre". Assim, Richard Stallman, criador do movimento de *software* livre e fundador da Fundação de *Software* Livre (FSF) sem fins lucrativos, define a visão para a *GNU General Pulic Licens*e (GPL). A versão 2 da GPL (GPLv2) foi lançada em 1991 e é hoje uma das mais populares licenças de *software* livre e de código aberto (FOSS) do mundo. (TSAI, 2008. p.2).

A *Free Software Foundation* pode publicar versões revisadas ou novas versões da GPL de tempos em tempo. As novas versões serão similares ao espírito presente na versão, podendo diferir em detalhes para resolver novos problemas ou preocupações. Sendo assim cada versão recebe um número diferente. (FSF,2007).

A seção seguinte abordará outra de licença de *software* muito utilizada, assim como a GPL.

2.5.2 *Berkeley Software Distribution*

Para Hahn (2014) a licença BSD (*Berkeley Software Distribution*) é uma das mais reconhecidas e menos restritivas referente ao *software* de código-fonte aberto. Foi desenvolvida pela Universidade da Califórnia em Berkeley permitindo o uso gratuito do *software opensource* (OSS), incluindo a capacidade de modificar o *software* e redistribuir e usar o código-fonte, modificando ou não. A licença BSD permite que o *software* seja combinado com *software* proprietário ou modificado transformando em *software* proprietário. Permite que projetos derivados possam ser lançados sob uma licença diferente do BSD.

> Os principais motivos que levaram a licença BSD a ser tão difundida são a simplicidade de seu texto e o fato dela ter sido inicialmente adotada por um projeto amplamente disseminado, o que criou um ciclo virtuoso em que mais comunidades a adotaram, tornando-a ainda mais reconhecida (SABINO e KON, 2009. p.11).

As licenças livres como BSD não incluem a característica de *copyleft*, sendo um trocadilho com o *copyright*, cuja tradução se aproxima de ‘deixamos copiar’ ou ‘cópia permitida’ (CAMPOS,2006).

De acordo com Rosen (2004) a Universidade da Califórnia decidiu em usar a licença BSD para alavancar o tipo de liberdade acadêmica. Concluindo que alguns dos seus softwares teriam maior valor se fossem disponibilizados gratuitamente para todos, copiarem, modificarem e distribuir do que fossem mantidos em segredo ou vendê-los privadamente.

A próxima seção trará informações a respeito do *software* de DNS BIND, que carrega em sua essência a licença de *software* BSD.

2.5.3 BIND

Segundo a ISC (2018) o BIND (*Berkley Internet Name Domain*) é uma solução o*pensource* de DNS que foi desenvolvida pela Universidade de Berkeley na Califórnia na década de 80. O BIND é um dos softwares de DNS mais utilizado na *internet*, muito utilizado por provedores de *internet* por se tratar de uma solução distribuída gratuitamente sob a licença BSD (*Berkeley Software Distribution*), disponível para funcionamento em diversas plataformas de sistema operacionais.

De acordo com Rijswijk-Deij (2012) o BIND tem suporte para extensão DNSSEC integrado desde a versão 9 que foi lançada no ano 2000. Com decorrer da evolução do DNSSEC ao longo dos anos novos recursos foram incluídos no projeto BIND.

O BIND é um *software opensource* transparente. Caso seja necessária alguma funcionalidade que não esteja no BIND, é possível modificá-lo e contribuir com novos recursos para a comunidade, disponibilizando a contribuição desenvolvida. O BIND é a primeira e mais antiga solução de DNS, tendo uma ampla comunidade e engenheiros já familiarizados com o sistema. (ISC,2018).

O BIND é executado como um *daemon* no sistema operacional Linux, Unix e BSD e como um serviço nas plataformas Windows, passando por diversas iterações ao longo dos anos sendo utilizada de forma nativa em grandes sistemas operacionais como *FreeBSD* entre outros, está ativamente sendo desenvolvido e mantido pela ISC (*Internet*

Systems Consortium) empresa sem fins lucrativos com sede nos Estados Unidos (AITCHISON,2011).

A próxima seção abordará assuntos relacionados ao serviço de DNS *Unbound*.

2.5.4 *Unbound*

De acordo com a NLNETLABS (2018) o *Unbound* é um produto de resolução DNS com funções recursivas, validações e armazenamento em *cache*. Desenvolvido na linguagem C, criado e mantido pela NLnetLabs, baseado em ideias e algoritmos extraídos de um protótipo Java desenvolvidos pelos laboratórios *Verisign, Nominet, Kirei* e *Ep.net*. Seu lançamento inicial foi em 19 de fevereiro de 2011, com a proposta de ser simples e incorporando recursos modernos baseados em padrões abertos.

Unbound suporta diversos padrões modernos que limitam a quantidade de informações trocadas com os servidores autoritativos, melhorando a privacidade e tornando o DNS mais robusto, dentre elas a QNAME (*Query Name Minimisation*) RFC 7816. Uso agressivo do *cache* validado pelo DNSSEC RFC 8198 e suporte a zonas de autoridade RFC 7706, que são utilizadas para carregar uma cópia da zona raiz. (NLNETLABS,2018).

"O *Unbound* foi desenvolvido a partir do zero como um servidor de nomes recursivo de armazenamento em *cache* puro com suporte de validação DNSSEC" (RIJSWIJK-DEIJ, 2012, p14).

O *software* é distribuído gratuitamente no formato o*pensource* e está sob a licença BSD (*Berkeley Software Distribution*) encontra se na versão atual 1.7.0. É projetado como um conjunto de componentes modulares com recursos modernos, como validação de segurança aprimorada (DNSSEC), IPv6, originalmente escrito para o sistema operacional POSIX, está disponível para *FreeBSD, OpenBSD, NetBSD, OS X, Linux e Windows*. (NLNETLABS,2018).

> “Lançamos o *software* sob a licença BSD, que permite o uso em outros produtos sem maiores restrições”, disse Olaf Kolkman, diretor da NLnet Labs, uma fundação de pesquisa e desenvolvimento sem fins lucrativos da Holanda. “Esperamos que disponibilizar nosso *software* gratuitamente ajudará na implantação do DNSSEC, que se encaixa diretamente no regulamento da NLnet Labs" (VERISIGN,2008. p.1).

A seção seguinte trará uma abordagem a respeito do sistema operacional GNU/Linux Debian, realçando pontos importantes.

2.5.5 *Debian*

Hertzog e Mas (2015) citam que o Debian é uma distribuição do GNU/Linux, sendo um sistema operacional completo, contendo softwares livres, e gerenciamento baseado no *kernel* Linux. Criado por Ian Murdock em 1993 sob a liderança da FSF (*Free Software Foundation*). O sistema operacional baseia-se em pilares definidos pelo Manifesto Debian que possui duas características principais. Primeiro qualidade, onde o Debian seria desenvolvido com maior cuidado para ser digno do *kernel* Linux, segundo que não seria uma distribuição comercial, mas com poderes de competir com distribuições comerciais. Com esta dupla ambição isso seria somente alcançável se o Debian fosse aberto ao desenvolvimento, com a proposta de alcançar grandes desenvolvedores engajados no projeto.

Ainda Hertzog e Mas (2015) afirmam que o Debian segue todos os princípios do *software* livre, suas novas versões não são liberadas até estarem completamente prontas. Os desenvolvedores não são forçados a um cronograma definido, de modo a correr para atender datas limite. Algumas pessoas se queixam ao longo do tempo entre o lançamento estável do Debian, porém, isso se deve ao fato que o projeto se preocupa com a confiabilidade lendária do Debian. O Debian não compromete a qualidade, todas as falhas críticas são reconhecidas e resolvidas em qualquer nova versão.

O Debian possui um vasto repositório, sendo um dos maiores relacionado a *software* livre, sendo possível instalar softwares não livres, sendo um sistema operacional muito utilizado em servidores pela sua robustez e por ser uma distribuição com uma comunidade experiente e ativa. O Debian inspira diversas distribuições Linux como *Kali, Ubuntu e Mint*.(DEBIAN.ORG,2018).

As próximas seções abordarão assuntos referentes aos conceitos e técnicas de *benchmark* que são muito utilizadas para realizar comparativos de desempenho, a fim de auxiliar na escolha de uma melhor solução.

2.6 *Benchmark*

Para Kistowski (2015) *benchmark* pode ser definido como uma ferramenta para a avaliação competitiva e comparação de sistemas concorrentes ou componentes de acordo com características específicas, como desempenho, confiabilidade ou segurança.

Huppler (2009) afirma que um teste de *benchmark* deve atender critérios importantes, sendo eles:

- Relevância: Deve se medir o desempenho em alto nível dos sistemas quando executadas operações típicas.
- Justo e Portável: Deve ser simples a implementação do teste, sendo capaz de implementar em diversos sistemas e arquiteturas diferentes.
- Escalabilidade: O *benchmark* deve ser capaz de mensurar desde pequenos a grandes sistemas ou até mesmo funcionando em paralelo.
- Simplicidade: O *benchmark* deve ser compreensível e de fácil entendimento, caso contrário não terá credibilidade nos resultados.

Kistowski (2015) ressalta que todos os *benchmarkings* estão sujeitos aos mesmos critérios, sendo que cada categoria inclui questões adicionais específicas para os *benchmarkings* individuais, dependendo de seu objetivo. Com base nestes critérios, pode-se dizer que um *benchmark* realizado entre servidores de DNS locais ou grandes servidores como *Google* DNS devem ser simples a sua execução e de fácil entendimento, sendo assim um teste válido e compreensível.

Ahmad e Sarwar (2014) citam que, a medição do desempenho do DNS é um dos maiores desafios de hoje, já que a funcionalidade da *internet* inteira depende desse principal componente. Para uma organização em rede, qualquer problema inesperado com o DNS impactaria imediatamente a entrada e saída dos serviços.

A seção a seguir tratará sobre a ferramenta utilizada para realizar *benchmark* entre servidores de DNS.

2.6.1 *Namebench*

O *Namebench* é uma ferramenta de *benchmark* para serviços de DNS de código-fonte aberto desenvolvido inicialmente pelo *Google Inc*. Está sob a licença apache versão 2.0, criado pelo engenheiro Thomas Stromberg. O *Namebench* está disponível através do *Google Code Archive* para Mac OS X, Windows e Unix, possuindo uma interface gráfica, assim como uma *interface* de linha de comando (NAMEBENCH,2018). A ferramenta realiza diversos testes, usando o histórico do navegador, ou seja, os sites onde o computador já fez requisições, realiza testes usando uma lista dos sites mais acessados no mundo, conhecida como *Alexa Top Global Domains* criada pela *Alexa Internet Inc* companhia de *internet* que fornece análise e dados de tráfego na *internet*.

A finalidade da ferramenta é encontrar o servidor DNS mais rápido, realizando diversos testes. Começou como um projeto de 20% desenvolvido pelo *Google.* (NAMEBENCH,2018).

Um usuário de *internet* insatisfeito com o DNS fornecido pelo seu provedor pode escolher facilmente um serviço de terceiro sem nenhum custo e que ferramentas como *Namebench* podem auxiliar na escolha de um servidor com melhor desempenho (AGER et al, 2010).

A seção seguinte trará informações referentes a empresa Alexa e a lista *Top Alexa Rank.*

2.6.2 *Alexa*

A *Alexa Internet Inc* fundada em 1996 é uma grande empresa americana que trabalha com dados e análises de trafego pela *internet*, é uma subsidiária da *Amazon*. A empresa analisa dados e comportamentos de navegação pela *internet* fornecendo estas informações para grandes empresas como *insights* analíticos, uma das pioneiras no mundo com uma vasta experiência (ALEXA,2018).

A Alexa utiliza metodologias proprietárias que combinam a análise de quantidade de visitantes em um *site* diariamente e durante um período de 3 meses, sendo capaz de monitorar tendências e popularidades. Através dessas informações a empresa criou o

que é mais conhecido como *Alexa Traffic Rank* contendo um *rank* de todos os *sites* mais utilizados no mundo todo, hoje esta lista é liderada pelo google.com estando em primeiro lugar entre os sites mais acessados do mundo. (ALEXA,2018).

Thakur, Sangal e Bindra (2011) definem que as razoes para que as empresas utilizem o ATR (*Alexa Traffic Rank*) se deve ao fato que através das análises é possível observar que:

- *Sites* que tem um baixo nível no *Rank Alexa* tem boa perspectiva para investimento, financiamento ou até mesmo revenda.
- Um *site* que melhora a classificação no *Rank Alexa* desenvolve uma imagem melhor. Sites com pouca classificação no *Rank Alexa* perdem a credibilidade dos visitantes afetando negativamente as vendas.
- Uma ótima classificação no *Rank Alexa* significa uma maior receita de publicidade, em especiais sites que obtém muito trafego.
- Melhor imagem do *website* e credibilidade quando há o aumento no *Rank Alexa*.

Algumas ferramentas de *benchmark* de DNS utilizam uma lista contendo diversos domínios para realizar consultas e analisar o desempenho entre servidores de DNS. Um fato importante é que a *Alexa* considera e analisa somente domínios e não fornecem classificações para subpáginas dentro de um domínio (por exemplo, www.dominio.com.br/contato.html) também não fornece análise de subdomínios como, por exemplo, *subdominio.dominio.com.br*. (ALEXA,2018).

O próximo capítulo tem a finalidade de descrever os métodos utilizados para a execução da presente pesquisa, dividido em seções que apresentarão o objetivo do estudo, o ambiente do estudo, implementação do servidor, testes de funcionamento e a execução dos testes de *benchmarking*.

3 METODOLOGIA

Este capítulo busca descrever as etapas que foram realizadas na execução do trabalho, abordando o processo de implementação, testes de funcionamento, análise de segurança e realização do *benchmark*.

3.1 Objetivo de estudo

O objetivo principal deste trabalho foi propor uma nova solução para o serviço de DNS ofertado pela empresa Prodata Informática, no setor de provedoria de *internet*.

Para oferecer serviços de melhor qualidade a seus clientes, a empresa dispõe de um serviço de DNS recursivo local. Serviço implementado e mantido restritamente por uma empresa de terceiros. Esta solução utiliza o serviço de resolução de DNS BIND na versão *9.8.2*, sendo capaz de hospedar domínios, realizar consultas de DNS e armazenamento em *cache.*

Com a finalidade de melhorar a qualidade dos serviços e analisando o DNS ofertado pela empresa surgiu a proposta de implementação de um novo servidor, contendo outro *software* para realizar a função e outros parâmetros de configurações.

Através do teste disponibilizado pelo *site https://dnssec.vs.uni-due.de* foi possível identificar que o servidor que a empresa utilizava não dispõe da extensão de DNSSEC habilitada. Extensão muitas vezes ignorada por requerer um tempo de respostas maior dos servidores para a realizar consultas, porém, a segurança passa não ser prioridade.

Foi definido que o novo servidor implementado utilizaria o *software* de DNS *Unbound*, por tratar-se de uma solução *opensource* muito utilizada por grandes provedores de *internet,* bem avaliado e discutida em uma das maiores listas de *e-mail* de engenheiros de redes do Brasil, o GTER (Grupo de Trabalho de Engenharia e Operação de Redes).

Ao decorrer da análise foi definido que o servidor seria implementado seguindo os padrões recomendados na cartilha[5] disponibilizada pelo CERT.BR *(Recomendações para Evitar o Abuso de Servidores DNS Recursivos Abertos)*.

O Centro de Estudos, Resposta e Tratamento de Incidentes de Segurança no Brasil (CERT.BR), responsável por tratar incidentes de segurança em dispositivos que envolvem redes conectadas à *internet* do Brasil, atuando de forma crucial para notificar incidentes de segurança, trabalhos de conscientização sobre os problemas de segurança existentes, análise de tendências e correlação entre eventos na *internet* brasileira. É mantido pelo Comitê Gestor de *Internet* do Brasil o NIC.br que atende todas as redes brasileiras conectadas à *internet* (CERT.BR,2018).

Foi decidido que a ferramenta *Namebench* seria utilizada para a realizar os testes entre os servidores. Por tratar-se de uma ferramenta desenvolvida por engenheiros da *Google* em um projeto o*pensource,* trazendo uma maior credibilidade e seguindo a proposta de utilizar softwares gratuitos. Propondo ser uma ferramenta eficaz, que realiza os testes de forma simples e disponibiliza os resultados em formato CSV (*Comma-separated values*), o que facilita a análise e interpretação final dos resultados.

De acordo com a RFC-4180 (2005) o CSV é utilizado para trocar e converter dados entre vários programas de planilhas. Embora existam diversas especificações e implementações do formato CSV, não existe uma especificação formal, o que permite uma ampla variedade de interpretações de CSV.

Outro aspecto definido foi que os testes comparativos seriam utilizando a lista *Top Rank Alexa*, *Cache Hit* e C*ache Miss,* ambos os testes disponíveis nativamente pela ferramenta *Namebench*, que avaliam o tempo de resposta das consultas contidas em *cache*, recursividade e domínios mais acessados mundialmente.

A escolha do sistema operacional a ser utilizado na implementação do servidor *Unbound* foi o *GNU/Linux Debian*, devido as suas características de estabilidade e padronização da empresa, onde os administradores de redes têm uma maior familiaridade com o mesmo.

[5] *https://www.cert.br/docs/whitepapers/dns-recursivo-aberto*

O Debian é uma distribuição Linux que possui um vasto repositório de pacotes para utilização. Sistema operacional muito utilizado devido sua estabilidade e confiança, sendo uma das distribuições mais antigas em atividade.

Para Hertzog e Mas (2015) o que deixa o Debian ser um sistema operacional tão popular entre os administradores de sistema é a facilidade com que os softwares podem ser instalados e como são atualizados. Esta vantagem se dá em grande parte devido APT (*Advanced Packaging Tool*).

A próxima seção busca descrever o cenário no qual foi realizado o estudo de caso.

3.2 Ambiente de estudo

O presente estudo foi realizado em uma empresa com mais de 30 anos de experiência no ramo de tecnologia. Com sede situada em Caratinga-MG, a Prodata fundada em 1987 atua nos ramos de engenharia de *software*, assistência técnica, loja de produtos para informática e provedoria de *internet*, no qual atende 20 cidades. Sendo assim, o motivo pelo qual foi realizado o estudo de caso, por se tratar de uma empresa com uma vasta experiência no ramo de provedoria de *internet*.

A empresa possui dois setores no ramo de provedoria de *internet*, o suporte técnico e infraestrutura. O suporte técnico é responsável por manter os ativos de redes em pleno funcionamento, onde estão localizados os principais equipamentos e servidores que compõe a rede. Local onde foi adicionado o novo servidor.

Para que os testes comparativos fossem de forma igualitária, foi definido que o servidor implementado deveria estar no mesmo nó da rede que o servidor atual, podendo ser observado pela Figura 4 que as distâncias entre os servidores e a máquina que executou os testes são as mesmas.

Por questões de segurança o endereço IP dos servidores ao longo do trabalho foram exibidos somente o último octeto, sendo assim, para o *Unbound XXX.XXX.XXX.194* e *XXX.XXX.XXX.195* para o BIND.

O *hardware* que é utilizado no servidor BIND é composto por um processador *Intel(R) Core(TM) i7-7700 CPU 3.60GHz*, 8 *Gigabytes* de memória *ram*, disco SSD 240

Gigabytes. Utilizando o sistema operacional *CentOS release 6.5 (Final), Kernel 2.6.32-504.3.3.el6.x86_64 (SMP) x86_64*.

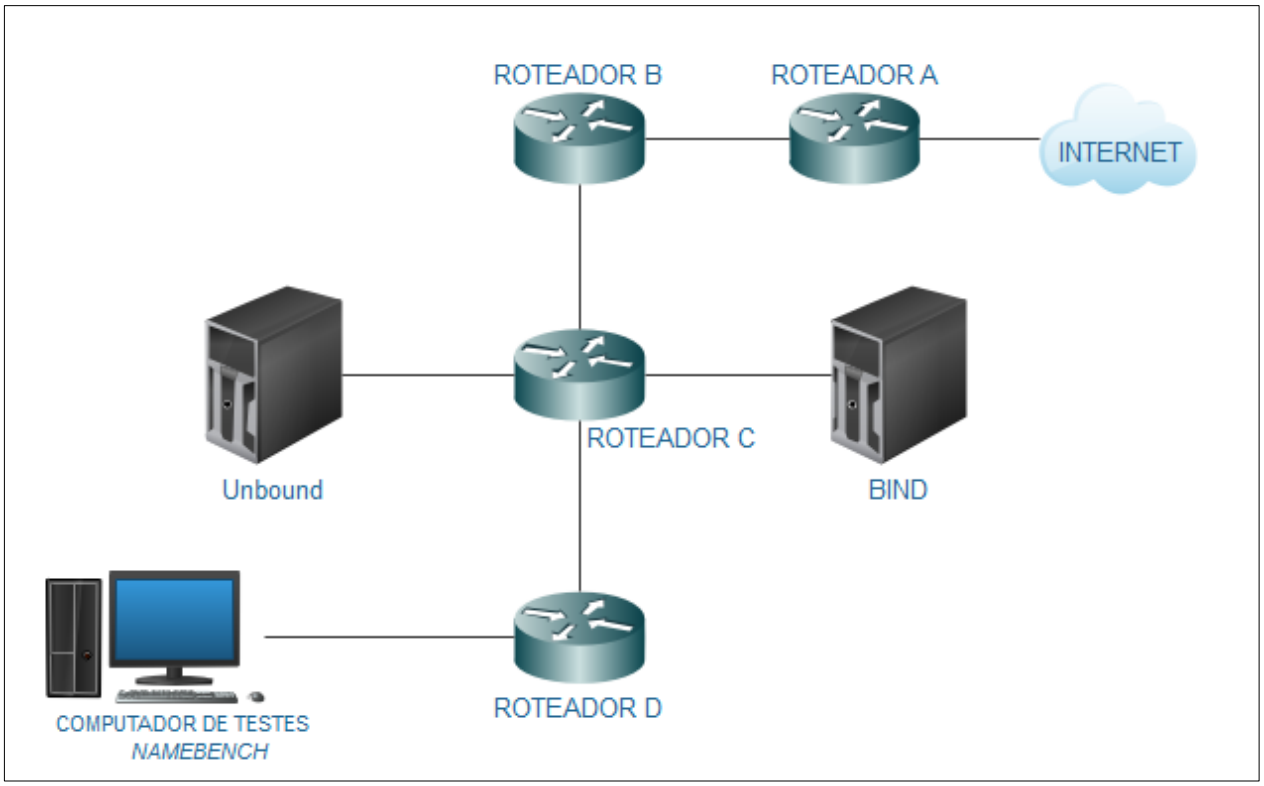
O *hardware* que foi utilizado no servidor *Unbound* é composto por um processador Intel(R) *Core(TM) i5-3330 CPU @ 3.00GHz*, 8 *Gigabytes* de memória *ram*, disco SATA 500 *Gigabytes*. Não sendo possível obter o mesmo processador que é utilizado no servidor BIND devido à logística e arquitetura dos componentes.

A ISC (2018) diz que os requisitos de *hardware* para servidores de DNS são tradicionalmente modestos. Em algumas instalações, os servidores velhos já aposentados tem um desempenho admirável como servidores de DNS. Os recursos como DNSSEC e IPv6 podem demandar mais recursos da CPU, portanto, os servidores que utilizam destes recursos devem considerar sistemas maiores para essas funções. Os requisitos de CPU para o *BIND9* podem variar desde clássicos i486 até máquinas de classes corporativas atendendo milhares de consultas por segundo. Já os requisitos de memória devem ser grandes o suficiente para realizar o *cache*, sendo uma boa prática ter memória suficiente para carregar todos os dados de zona e *cache* na memória.

Segundo Rijskijk-Deij (2012) o DNSSEC conta com criptografia de chave pública para realizar as assinaturas digitais, garantindo a autenticidade dos registros. A criptografia de chave pública é por natureza intensiva nos termos computacionais, requerendo um poder de processamento acima da média para funcionar corretamente. No passado, isso fez com que fosse introduzido aceleradores de criptografia em *hardware*. Entretanto, pesquisas mostraram que o *hardware* moderno (*construído depois de 2005*) seria o suficiente para operar um servidor de DNS recursivo com validação de DNSSEC. Significando que nenhuma aceleradora de *hardware* adicional é necessário e que produtos já desenvolvidos com arquiteturas modernas podem ser utilizado sem a necessidade de aceleradores.

O computador utilizado para execução dos testes foi definido que seria uma máquina da rede interna da empresa, contendo o *software Namebench* na versão 1.3.1 localizada apenas a um nó de distância entre servidores exemplificados pela Figura 4. Contendo o *hardware Intel(R) Core(TM) i3-3220 CPU 3.30Ghz, 8 Gigabytes de* memória *ram, Windows 7 64 bits*.

Figura 4 - Topologia de rede do cenário de pesquisa



Fonte: Próprio autor

Cada nó representado pela Figura 4 refere-se a um roteador[6], onde estão configurados endereços IP e rotas para o funcionamento da rede.

O próximo capítulo tratará sobre a implementação do servidor *Unbound*, abordando as etapas de instalação, teste de funcionamento e segurança.

3.3 Implementação

Para a implementação do servidor *Unbound* foi utilizado a distribuição GNU/Linux Debian na versão 9.5.0 *stable netinst* com o *kernel 4.9.0-4-amd64*. A versão *netinst* é uma versão simples do sistema operacional contendo apenas os pacotes necessários para o funcionamento, versão utilizada para implementar servidores que necessitam de

[6] Roteadores são pontes que operam na camada de rede do Modelo OSI (*Open System Interconnection*). Eles são responsáveis por tomar a decisão de qual caminho percorrer para interligar redes diferentes (ALENCAR, 2010. p.36).

poucos recursos. Sendo assim, foi necessário instalar dois pacotes utilitários, o *cURL* e o pacote *dnsutils*, ambos disponíveis no repositório oficial da distribuição Debian.

De acordo com Stenber (2018) o *cURL* é uma ferramenta de linha de comando e uma biblioteca para transferir dados com a sintaxe de URL, suportando *HTTP, HTTPS, FTP,* e diversos outros protocolos*. O libcurl* oferece uma infinidade de recursos poderosos.

O pacote *dnsutils* contém a ferramenta DIG (*Domain Information Groper*) que foi utilizada para verificar o funcionamento do servidor. O DIG é uma ferramenta para interrogar servidores de DNS, executando uma consulta de DNS e exibindo as respostas com informações do tempo de resposta, data da consulta e outras informações (ALBITZ; LIU,2006).

O APT é o gerenciador de pacote utilizado na distribuição GNU/Linux Debian e outras que derivam de sua origem. Utilizou-se esta ferramenta para instalar os pacotes contidos no repositório da distribuição.

O editor de textos utilizado neste trabalho foi o VIM (*Vi Improved*). O VIM é um editor de texto com diversas opções de configurações, desenvolvido para permitir a edição eficiente de textos. Sendo uma versão melhorada do editor VI que está presente na maioria dos sistemas UNIX. Sua licença é compatível com a GPL, ou seja, distribuído gratuitamente (VIM,2018).

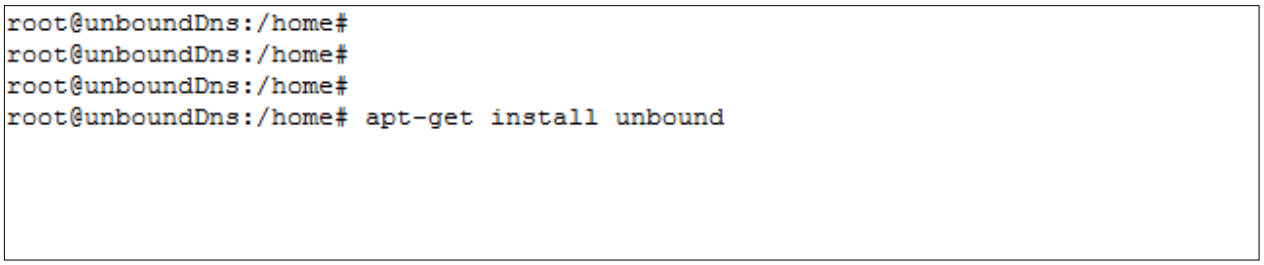
A seção seguinte apresentará as etapas que foram utilizadas para realizar a instalação do *software Unbound*.

3.3.1 Instalação do *Unbound*

Neste trabalho foi utilizado a versão 1.6.0 do DNS *Unbound,* sendo a versão estável e homologada pelo *GNU/Linux* Debian na versão *9.5*.

Para realizar a instalação do *software Unbound* foi necessário executar o comando *apt-get install unbound* exemplificado pela Figura 5.

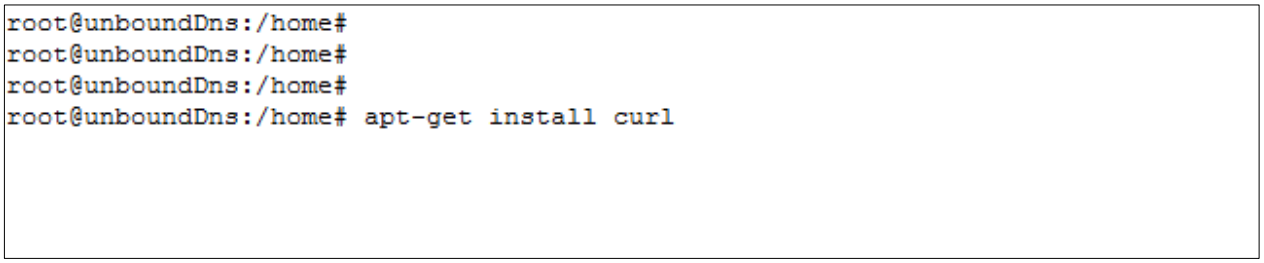
Figura 5 - Comando utilizado para instalação do *Unbound*



Fonte: Próprio autor.

A ferramenta *curl* foi utilizada para realizar o *download* via terminal das informações sobre os *root servers* e atribui-las ao arquivo *root.hints*. Para realizar a instalação do cURL foi necessário utilizar o comando *apt-get install curl*, podendo ser observado pela Figura 6.

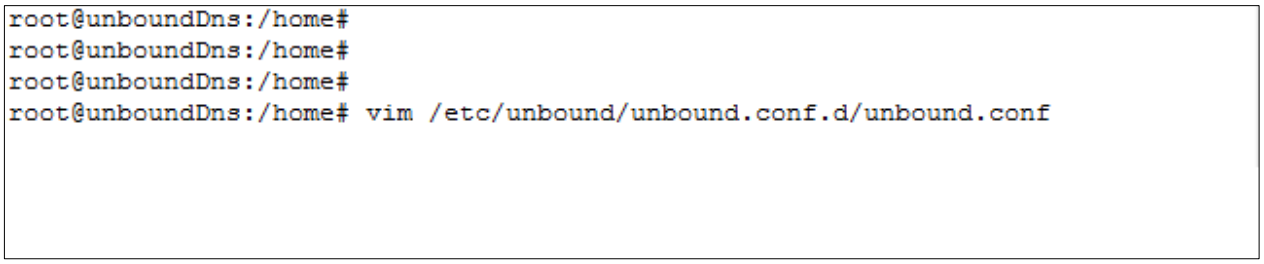
Figura 6 - Comando utilizado para instalar a ferramenta *cURL*



Fonte: Próprio autor.

Para criar o arquivo principal de configuração do *Unbound* foi necessário utilizar um editor de texto via linha de comando, podendo ser utilizado qualquer editor de texto para esta finalidade. Neste caso foi utilizado o VIM, para criar e já editar o arquivo através do comando *vim /etc/unbound/unbound.conf.d/unbound.conf* que pode ser observado através da Figura 7.

Figura 7 - Comando utilizado para criar e editar o arquivo *unbound.conf*



Fonte: Próprio autor.

O arquivo de configuração principal que foi utilizado neste trabalho contendo todos os parâmetros de configuração encontra-se disponível no Anexo 2, onde há a explicação de cada parâmetro utilizado.

Para que o servidor de DNS fosse capaz de realizar consultas nos *roots servers* foi necessário obter as informações com o endereço dos servidores. Essas informações foram obtidas através do *site www.internic.net/domain/named.cache e* adicionadas ao arquivo *root.hints*.

O *site* da *InterNIC* (*Internet Network Information Center*) é mantido pela ICANN (*Internet Corporation for Assigned Names and Numbers*), com a finalidade de divulgar informações sobre os serviços de registros dos nomes de domínios da *internet*. Onde em seu *site* oficial disponibiliza informações contendo o endereçamento IP dos *root servers* (ICANN,2018).

Para realizar o *download* das informações foi necessário executar o comando *curl –o /etc/unbound/root.hints https://www.internic.net/domain/named.cache* podendo ser observado pela Figura 8.

Figura 8 - *Download* das informações para o arquivo *root.hints*

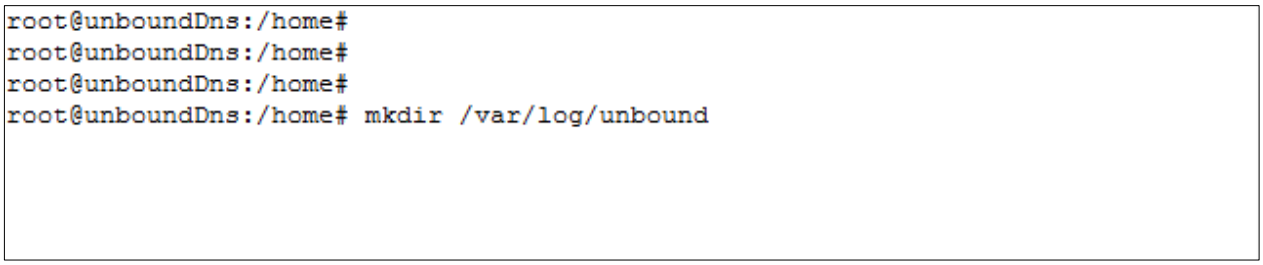
```
root@unboundDns:/home#
root@unboundDns:/home#
root@unboundDns:/home#
root@unboundDns:/home# curl -o /etc/unbound/root.hints https://www.internic.net/
domain/named.cache
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  3317  100  3317    0     0   1334      0  0:00:02  0:00:02 --:--:--  1334
root@unboundDns:/home#
```

Fonte: Próprio autor.

Para que o *software Unbound* fosse capaz de exibir informações referentes ao seu funcionamento, foi necessário criar o diretório e o arquivo de log.

Com o comando *mkdir /var/log/unbound* exemplificado pela Figura 9 foi possível criar o diretório para armazenar o arquivo de log.

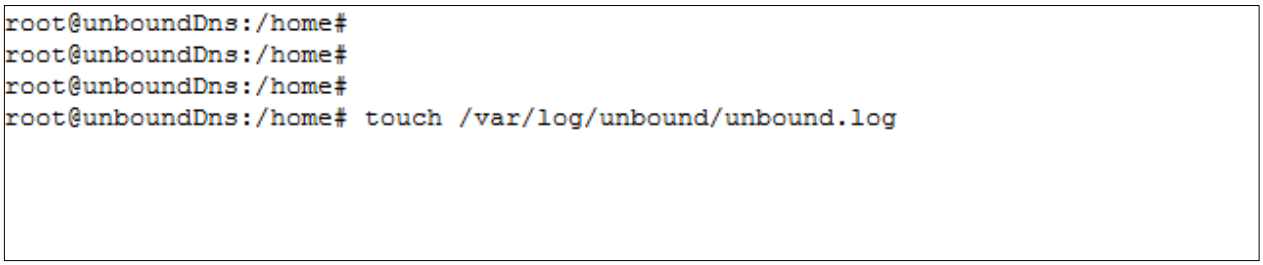
Figura 9 - Comando utilizado para criar o diretório do arquivo de *log*



Fonte: Próprio autor.

Após o diretório criado, foi necessário criar o arquivo de *log* através do comando *touch /var/log/unbound/unbound.log* mostrado pela Figura 10.

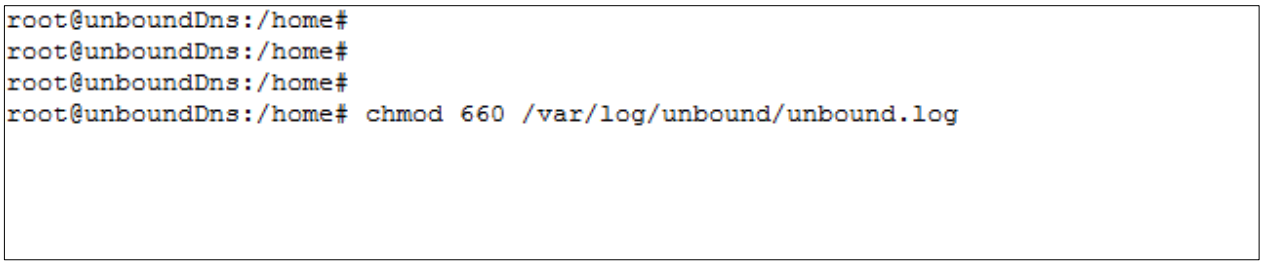
Figura 10 - Comando utilizado para criar o arquivo de *log*



Fonte: Próprio autor.

Após a criação do arquivo foi necessário dar permissões de leitura e escrita no arquivo de *log*, através do comando *chmod 660 /var/log/unbound/unbound.log*, podendo ser observado através da Figura 11.

Figura 11 - Comando utilizado para alterar permissões do arquivo de *log*

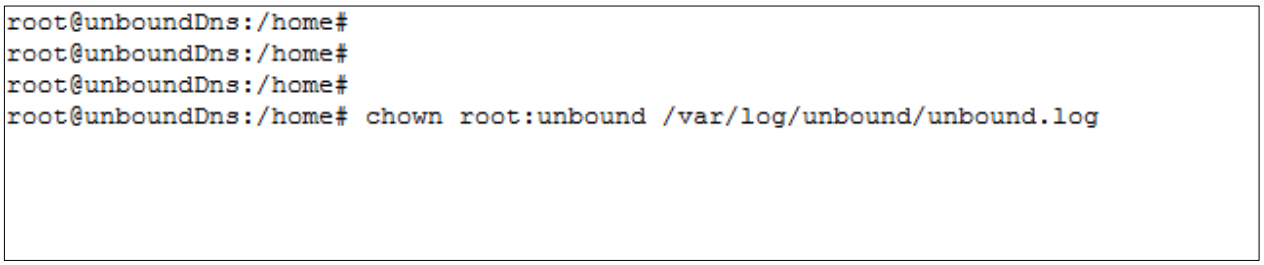


Fonte: Próprio autor

Outro passo realizado foi a alteração das propriedades do arquivo de *log*, determinando que o dono do arquivo seria o root e que os demais usuários do grupo *Unbound* seriam os responsáveis pela leitura e escrita do arquivo, através do comando

chown root:unbound /var/log/unbound/unbound.log, podendo ser observado pela Figura 12.

Figura 12 - Comando utilizado para alterar a propriedade do arquivo de *log*

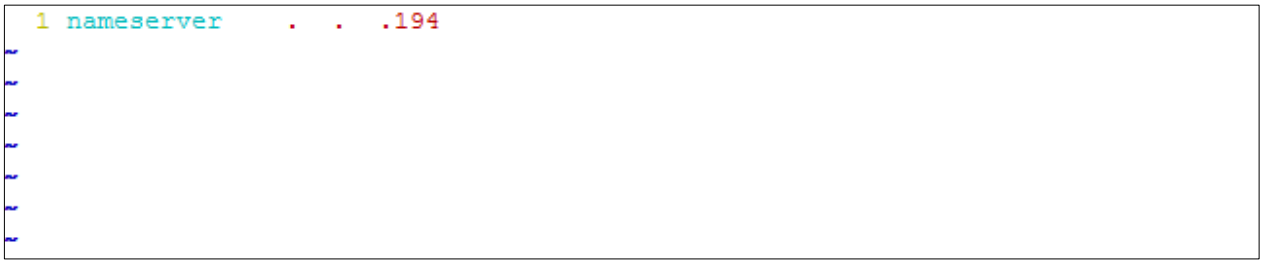


Fonte: Próprio autor

Para que o servidor *Unbound* fosse o próprio *resolvedor* de nomes, foi necessário alterar o arquivo de configuração padrão do GNU/Linux Debian onde estava configurado o endereço padrão de DNS.

O arquivo *resolv.conf* que estava localizado no diretório */etc/resolv.conf* é responsável por guardar o endereço do resolvedor de local. Para editar o arquivo neste exemplo foi utilizado novamente o editor de texto VIM através do comando *vim /etc/resolv.conf*, alterando o endereço referente ao servidor, localizado na única linha do arquivo, podendo ser observado pela Figura 13.

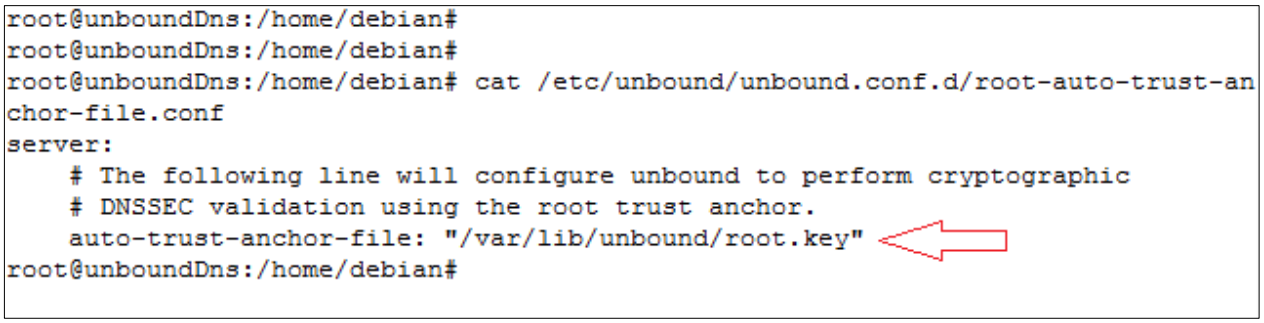
Figura 13 - Arquivo *resolv.conf* com o IP do próprio servidor



Fonte: Próprio autor

Para que a extensão de segurança DNSSEC funciona-se foi necessário verificar o arquivo *root-auto-trust-anchor-file.conf* através do comando exemplificado pela Figura 14, verificando se a linha *auto-trust-anchor-file* estava sem o comentário, ou seja, sem o caractere '#' a frente da linha.

Figura 14 - Retorno do comando *cat auto-trust-anchor-file*



Fonte: Próprio autor

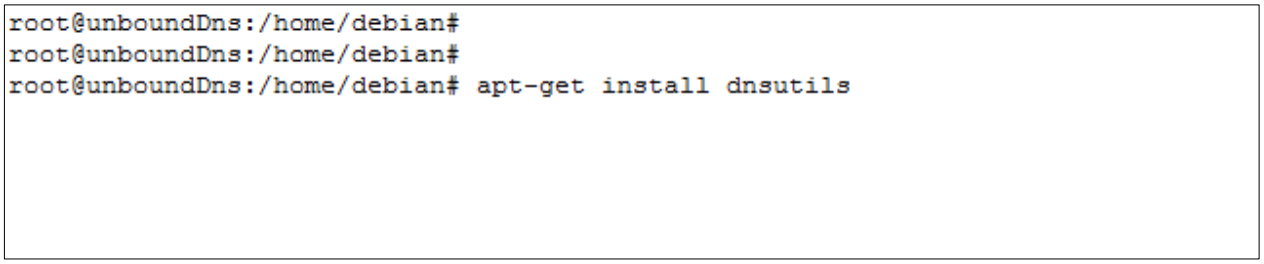
Por fim, foi necessário reiniciar o serviço para que as alterações fossem aplicadas, através do comando */etc/init.d/unbound restart*.

A seção a seguir busca descrever os testes realizados para verificar o funcionamento do *software Unbound*.

3.3.2 Teste de funcionamento

Para realizar um teste de funcionamento do servidor *Unbound* foi necessário instalar o pacote *dnsutils*, através do comando *apt-get install dnsutils* exemplificado pela Figura 15.

Figura 15 - Comando utilizado para instalar o pacote *dnsutils*



Fonte: Próprio autor

Para realizar um teste de funcionamento foi executado o comando *dig seguido do* domínio *www.adx.doctum.edu.br*. Com pode ser observado pela Figura 16, o comando retornou as informações referente a consulta contendo o tempo de resposta (*;;Query time: 97msec)*.

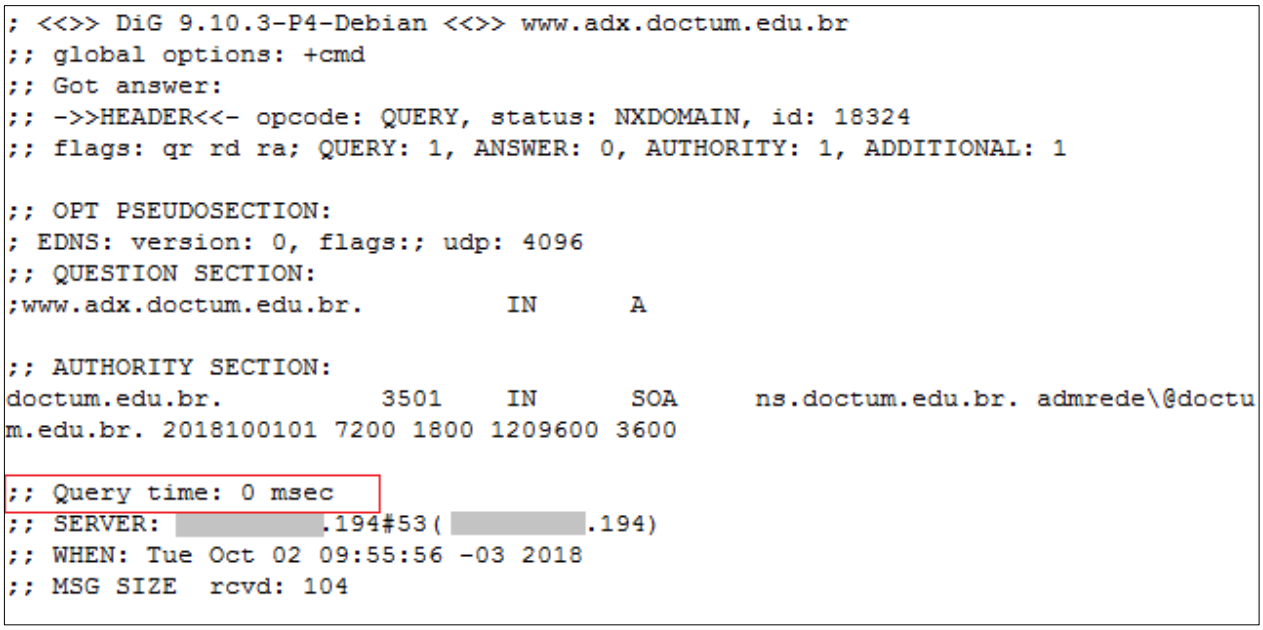
Figura 16 - Retorno do comando *dig* www.adx.doctum.edu.br

```
; <<>> DiG 9.10.3-P4-Debian <<>> www.adx.doctum.edu.br
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 52957
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.adx.doctum.edu.br.          IN      A

;; AUTHORITY SECTION:
doctum.edu.br.          3600    IN      SOA     ns.doctum.edu.br. admrede\@doctu
m.edu.br. 2018100101 7200 1800 1209600 3600

;; Query time: 97 msec
;; SERVER:           .194#53(          .194)
;; WHEN: Tue Oct 02 09:54:17 -03 2018
;; MSG SIZE  rcvd: 104
```

Fonte: Próprio autor

Para verificar o funcionamento do *cache*, o comando foi executado novamente, podendo ser observado pela Figura 17 que o tempo de resposta diminuiu drasticamente de 97ms (*milissegundos*) para 0ms, devido fato que o *Unbound* realizou o *cache* de todo o mapeamento da consulta anterior.

Figura 17 - Retorno do comando *dig* executado pela segunda vez



```
; <<>> DiG 9.10.3-P4-Debian <<>> www.adx.doctum.edu.br
;; global options: +cmd
;; Got answer:
;; ->>HEADER<<- opcode: QUERY, status: NXDOMAIN, id: 18324
;; flags: qr rd ra; QUERY: 1, ANSWER: 0, AUTHORITY: 1, ADDITIONAL: 1

;; OPT PSEUDOSECTION:
; EDNS: version: 0, flags:; udp: 4096
;; QUESTION SECTION:
;www.adx.doctum.edu.br.          IN      A

;; AUTHORITY SECTION:
doctum.edu.br.          3501    IN      SOA     ns.doctum.edu.br. admrede\@doctu
m.edu.br. 2018100101 7200 1800 1209600 3600

;; Query time: 0 msec
;; SERVER:            .194#53(            .194)
;; WHEN: Tue Oct 02 09:55:56 -03 2018
;; MSG SIZE  rcvd: 104
```

Fonte: Próprio autor

A seção seguinte apresentará os aspectos relacionados a segurança do servidor implementado, através da instalação de softwares e regras de *firewall*.

3.3.3 Aspectos de segurança

Por se tratar de um servidor que tem a exposição a *internet*, foi necessário analisar aspectos de segurança, sendo assim, os softwares e as configurações que serão apresentadas serviram de auxílio para a segurança do servidor.

Com intuito de evitar acessos indesejados ou tentativas de *brutal force* que exploram portas de serviços em servidores, foi utilizado o *software fail2ban*.

O *fail2ban* pode verificar arquivos de log e analisar endereços maliciosos proibindo-os de realizarem tentativas de força bruta, reduzindo a taxa de tentativas incorretas de autenticação em diversos serviços e protocolos como *SSH (Secure Shell), FTP* e etc. Podendo ser definido pelo administrador do sistema a quantidade de tentativas e o tempo em que os endereços maliciosos ficarão bloqueados (FAIL2BAN,2018).

Para instalar a ferramenta *fail2ban* foi necessário executar o comando *apt-get install fail2ban* exemplificado pela Figura 18.

Figura 18 - Comando utilizado para instalar o *fail2ban*

```
root@unboundDns:/home#
root@unboundDns:/home#
root@unboundDns:/home#
root@unboundDns:/home# apt-get install fail2ban
```

Fonte: Próprio autor

Através do arquivo *jail.conf* localizado no diretório */etc/fail2ban/* foi possível alterar os parâmetros de configurações, como tempo de banimento e quantidade de tentativas que serão necessárias para ocorrer o bloqueio dos endereços indesejados.

Para editar o arquivo *jail.conf* foi utilizado o editor de texto VIM através do comando *vim /etc/fail2ban/jail.conf*. Alterando os parâmetros de *bantime* = 600 (segundos) localizado na linha 59 e a linha 66 *maxretry* = 5 que é a quantidade de tentativas erradas para ocorrer o banimento, podendo ser observado pela Figura 19.

Figura 19 - Parâmetros de configuração do *fail2ban* que foram alterados

```
58 # "bantime" is the number of seconds that a host is banned.
59 bantime  = 600
60
61 # A host is banned if it has generated "maxretry" during the last "findtime"
62 # seconds.
63 findtime  = 600
64
65 # "maxretry" is the number of failures before a host get banned.
66 maxretry = 5
67
```

Fonte: Próprio autor

Outro aspecto importante foi realizar a mudança da porta padrão 22 do serviço de SSH para uma outra diferente, visando dificultar os ataques de *brutal force* e os *bots* que exploram a porta 22.

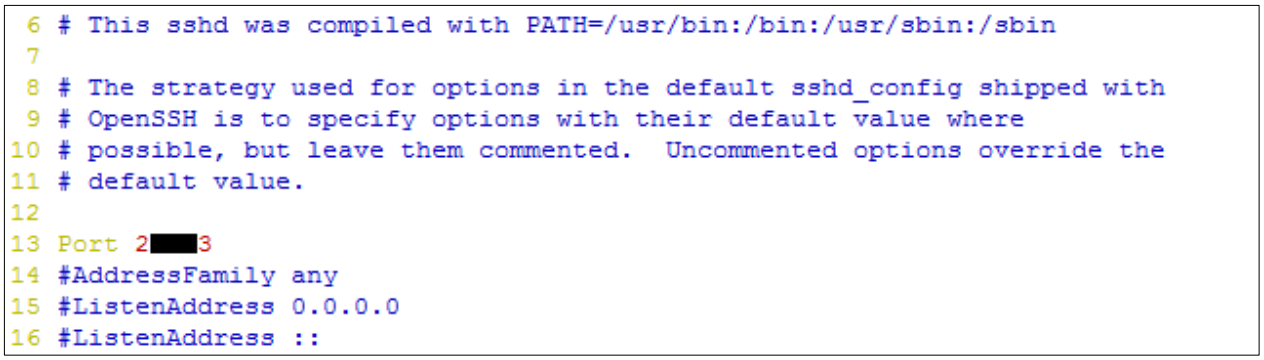
Segundo Martinelo e Bellezi (2014) os ataques de força bruta são softwares que utilizam várias combinações de usuário e senha com a finalidade de conseguir acesso indevido a sistemas ou para realizar a descriptografia de chaves e arquivos. Além do forçar o acesso indevido os ataques geram uma carga excessiva na vítima tendo que responder diversas tentativas de *logins*. Técnica utilizada para atacar servidores SSH mal

configurados. Os *bots* ou *botnets* são softwares robôs controlados remotamente pelo atacante possuindo a capacidade de propagação automática, uma *botnet* é uma rede composta por computadores infectados com *bot* utilizado para potencializar os ataques.

O SSH é de um protocolo para *login* remoto e seguro em servidores através do modo texto. Executado em cima do protocolo TCP/IP, podendo ser usado como base para vários serviços de rede segura fornecendo uma forte criptografia, autenticação e proteção de integridade. Utiliza o método de troca de chaves, algoritmo de chave pública, criptografia simétrica, algoritmo de autenticação de mensagens e algoritmo *hash* (YLONEN,2018).

Para alterar a porta do serviço de SSH foi necessário editar o arquivo de configuração *sshd_config* através do comando *vim /etc/ssh/sshd_config* e remover o comentário da linha 13 do arquivo e alterar o valor do parâmetro *Port* = 22 para um valor diferente do padrão, exemplificado pela Figura 20.

Figura 20 - Parâmetro do arquivo *sshd_config* alterado



Fonte: Próprio autor

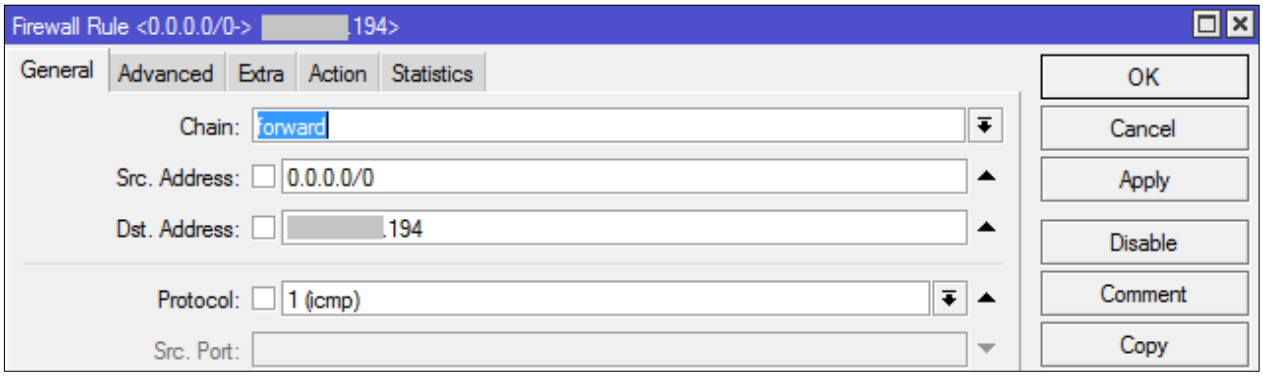
Por fim, foi necessário realizar o bloqueio a resposta ICMP (*Internet Control Message Protocol*) externas, dificultando a descoberta do serviço e evitando que o servidor receba diversas consultas de *ping,* ocasionando instabilidades no serviço. Para isso foi criado uma regra de *drop* no *firewall* mais próximo do servidor. O *firewall* utilizado neste cenário foi um *Mikrotik RouterOS*.

Mikrotik é uma empresa da Letônia fundada em 1996 construída para desenvolver roteadores e sistemas para ISP (*Internet Service Provider*). A *Mikrotik* fornece *hardware* e softwares para conectividade com a *internet* para diversos países. Em 1997 a empresa criou o sistema *RouterOS* que fornece estabilidade extensiva, controle de flexibilidades

para todos os tipos de interfaces de dados e roteamento. Em 2002 a empresa decidiu construir o próprio *hardware* com o *software RouterOS* denominado de *RouterBOARD* (MIKROTIK,2018).

Como pode ser observado pela Figura 21, em *Firewall Rule* foi definido na aba *General* a *chain forward*, em *Src.Address* sendo 0.0.0.0/0 o que determina qualquer rede, na opção *Dst. Address XXX.XXX.XXX.194* sendo o IP do servidor e na opção de protocolo foi definido 1(icmp).

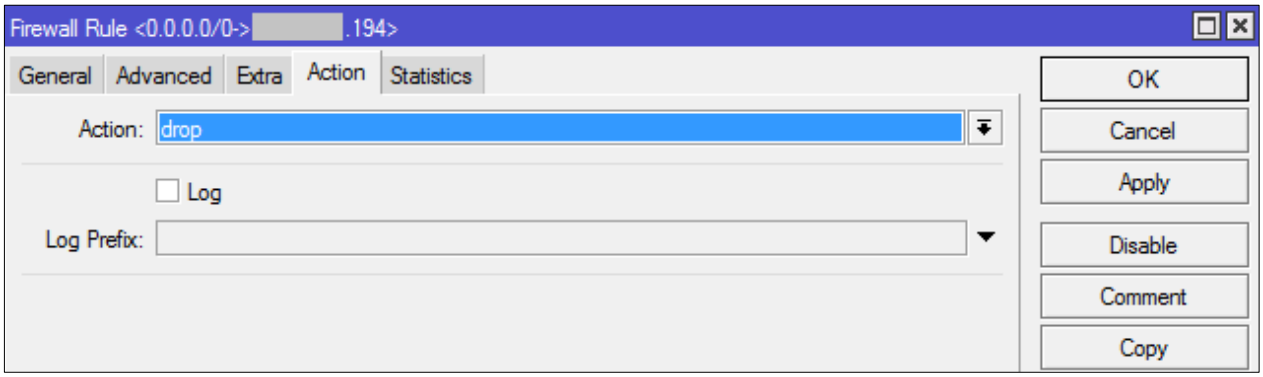
Figura 21 - Regra de *firewall* criado no *Mikrotik* aba *General*



Fonte: Próprio autor

Na aba *Action* foi definido da ação *drop*. Sendo assim, a *action drop* fez com que o *firewall* descartasse todos os pacotes com o destino e protocolo definido na aba *General*, sendo possível observar pela Figura 22.

Figura 22 - Regra de *firewall* criada no *Mikrotik* na aba *Action*



Fonte: Próprio autor

A regra criada no *firewall* consistiu em bloquear todos os pacotes ICMP com destino ao IP do servidor, evitando que chegassem consultas de *echo reply*.

O próximo capítulo tem a finalidade de abordar os assuntos referentes aos testes e demonstrar a execução em cada servidor.

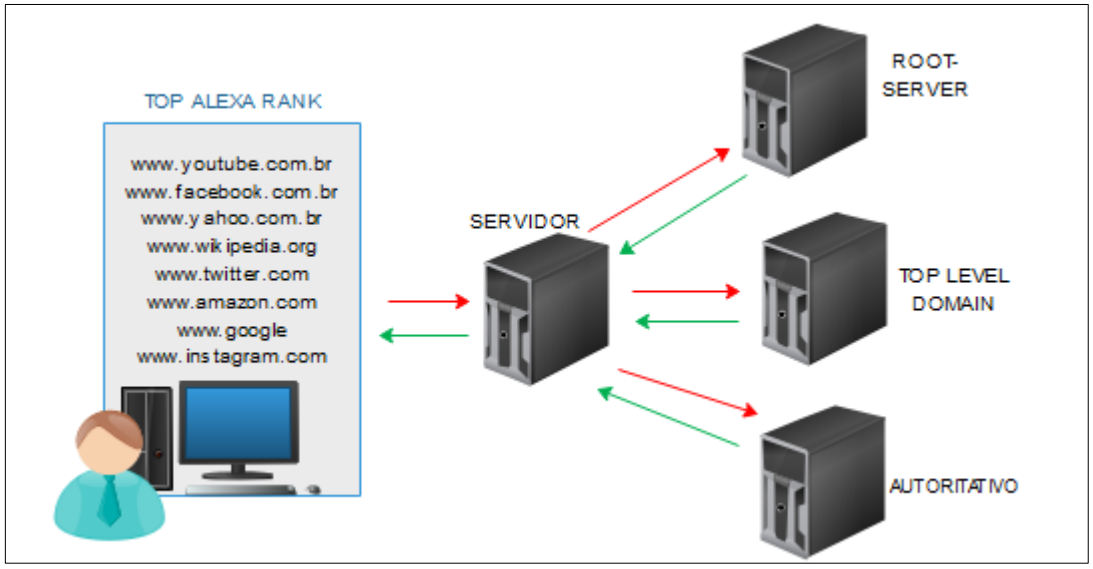
3.4 Execução dos testes

A ferramenta de teste utilizada neste trabalho foi o *Namebench* na versão 1.3.1. Os testes de *benchmark* foram definidos em três tipos no qual a própria ferramenta disponibiliza, *Top Alexa Rank*, *Cache Hit* e *Cache Miss*, que serão apresentados na próxima seção.

3.4.1 Tipos de testes

O teste *Top Alexa Rank* consistiu em requisitar uma lista dos domínios populares da *internet*, onde o servidor testado precisa realizar todo o mapeamento consultando em todos os servidores da hierarquia de domínios, podendo ser exemplificado pela Figura 23.

Figura 23 - Exemplo de funcionamento do teste *Top Alexa Rank*

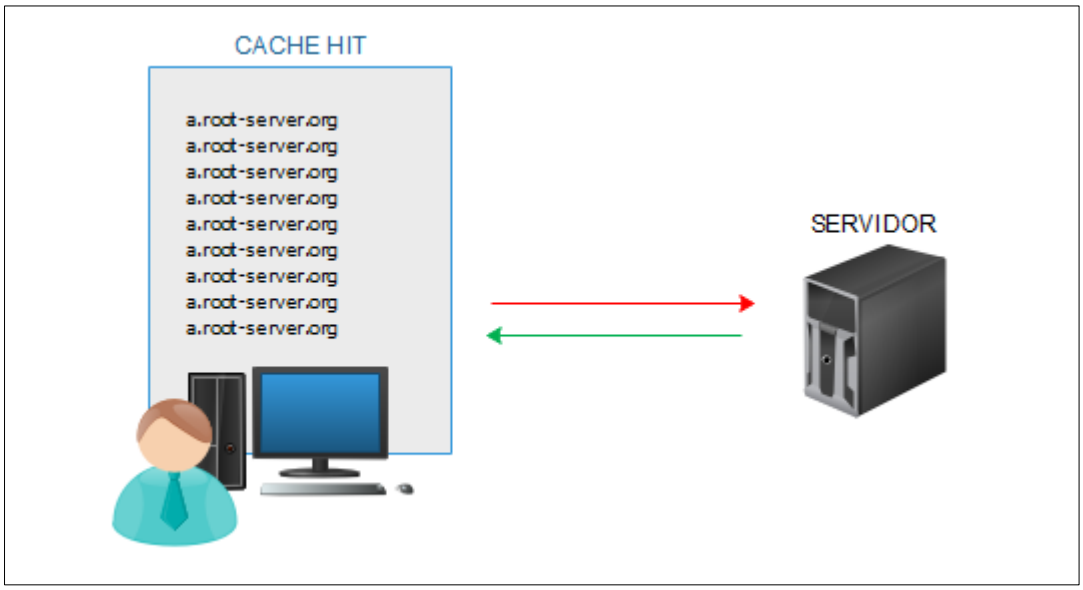


Fonte: Próprio autor

O segundo teste definido foi o de *Cache Hit*, onde o *Namebench* realizou a consulta do mesmo domínio em diversas vezes. Este teste consistiu em analisar o tempo

de resposta que servidor demandaria para realizar as consultas com informações já obtidas em *cache,* não sendo necessário realizar a recursividade, podendo ser exemplificado pela Figura 24.

Figura 24 - Exemplo de funcionamento do teste *Cache Hit*



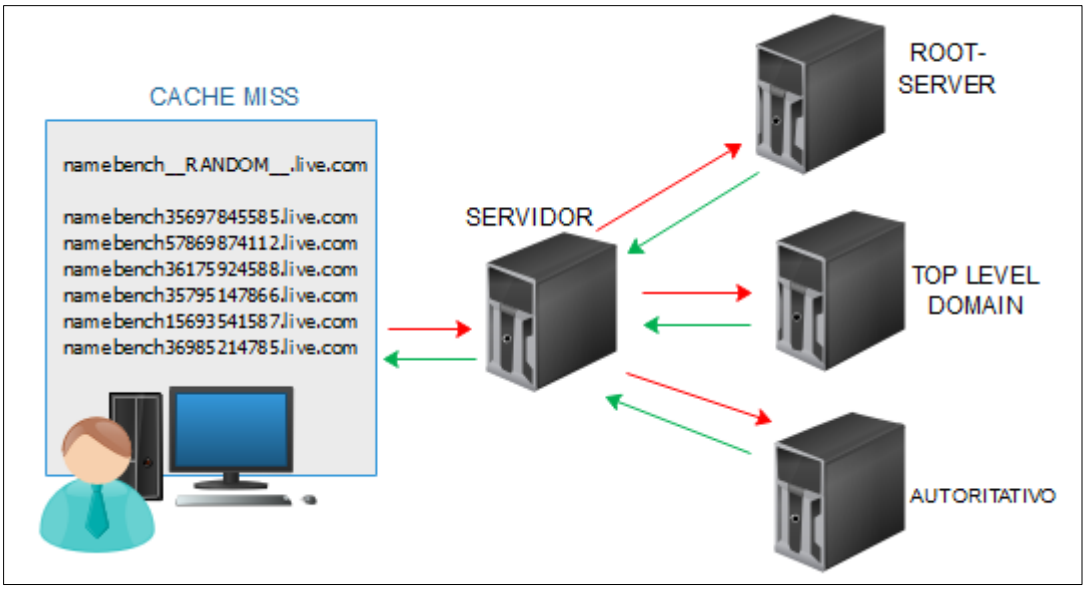
Fonte: Próprio autor

O terceiro e último teste foi definido que seria o de *Cache Miss*, onde o *Namebench* executou diversas consultas de subdomínios aleatórios. O teste de *Cache Miss* pode parecer com o *Top Alexa Rank*, porém, na prática, diversas empresas como *Google, Youtube e Facebook* utilizam diversos serviços hospedados no mesmo local.

Sendo assim, o teste de *Cache Miss* garantiu que as informações requisitadas nunca estariam em *cache*, devido fato dos subdomínios serem gerados aleatoriamente no momento da execução do teste, a fim de avaliar o desempenho da recursividade do servidor.

Nesse teste o servidor precisou analisar se as informações do domínio já estavam em *cache*, caso contrário requisitar as informações aos *root servers* realizando assim a recursividade e o mapeamento completo das informações, podendo ser observado pela Figura 25.

Figura 25 - Exemplo de funcionamento do teste de *Cache Miss*

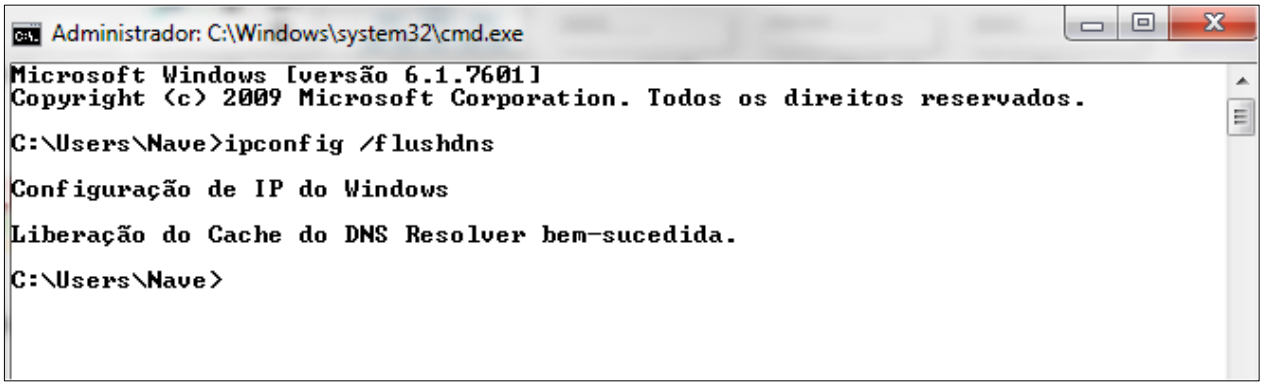


Fonte: Próprio autor

Comer (2015) cita que o *caching* é importante nos servidores de DNS locais e nos *hosts*, e que a maioria dos softwares tradutores de nome colocam entradas de DNS em *cache* no host. Neste caso, se um usuário requisitar o mesmo nome repetidamente, as mesmas poderão ser respondidas com o *cache* local sem o uso da rede.

Sendo assim, antes da execução de cada teste, o *cache* de DNS do computador que realizou o *benchmark* foi excluído, para que houvesse nenhum fator externo que pudesse influenciar nos testes. Executando o comando no terminal do *Windows 'ipconfig /flushdns',* exemplificado pela Figura 26.

Figura 26 – Comando utilizado para excluir o *cache* de DNS no *Windows*



Fonte: Próprio autor

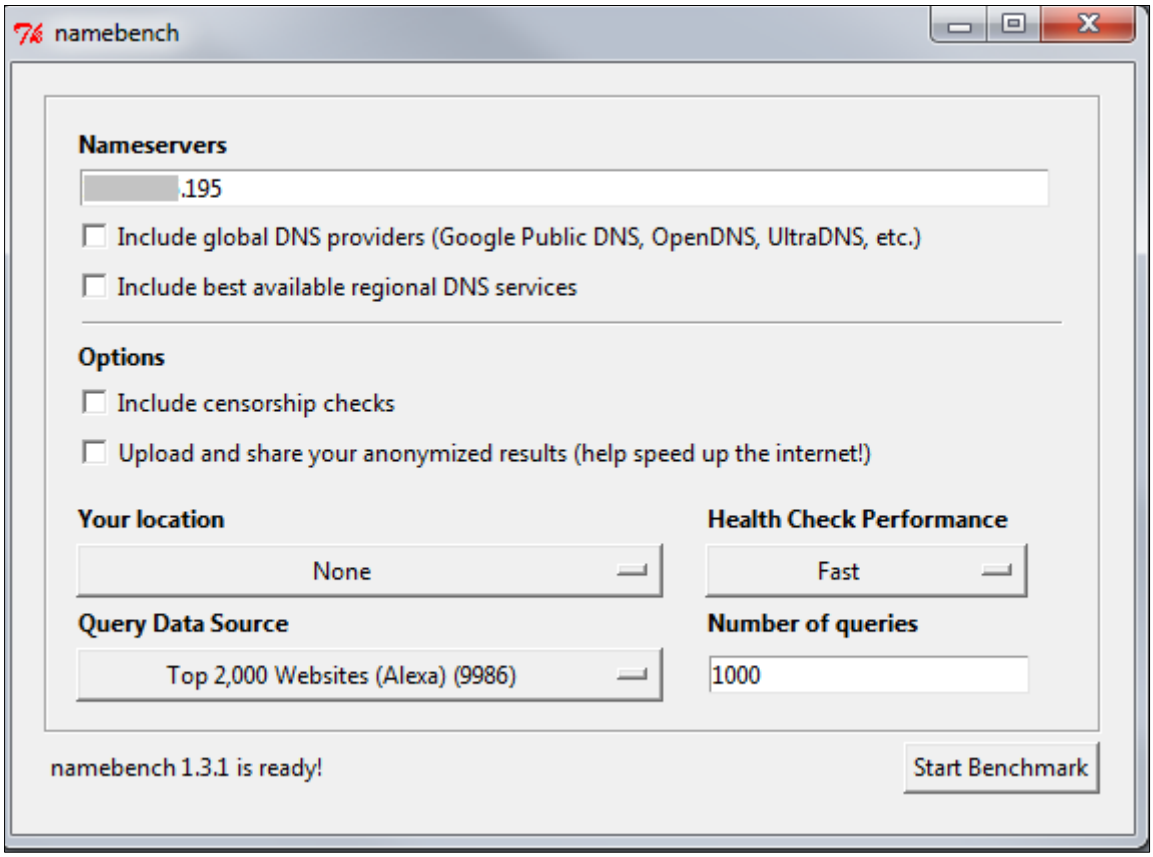
A seção seguinte demonstrará as etapas de execução dos testes que foram realizados no servidor BIND.

3.4.2 Execução do teste *Top Alexa Rank* no servidor BIND

A quantidade de consultas definida em todos os testes foram o valor 1000. Esta quantidade fez com que o servidor em todos os testes tivessem que responder em um espaço curto de tempo 1000 requisições de DNS.

Através da Figura 27 é possível observar que o campo (*Nameservers)* se refere ao IP do servidor que foi testado, o campo (*Number of queries)* se refere a quantidade de consultas que foram executadas e o campo (*Query Data Source)* determinou o tipo de teste que foi executado, sendo *Top Alexa Rank*.

Figura 27 - Teste *Top Alexa Rank* no servidor BIND



Fonte: Próprio autor

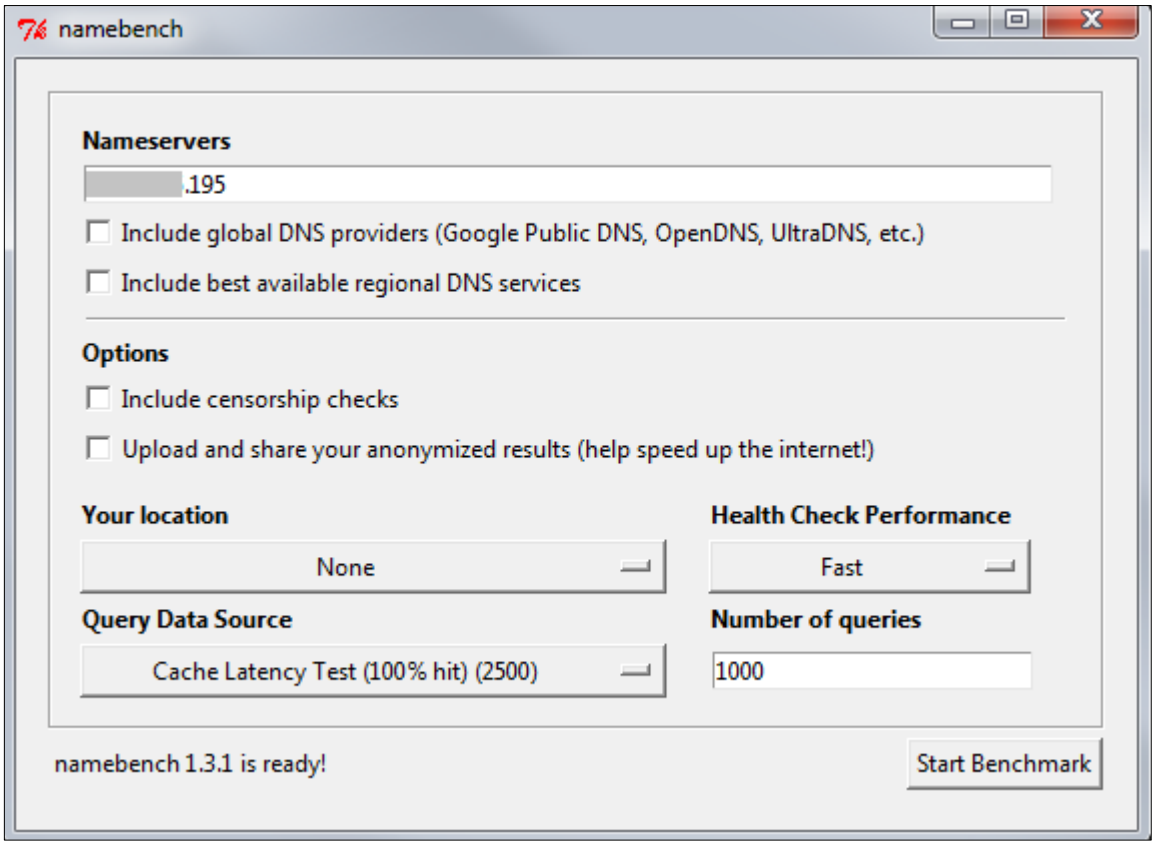
Ao final da execução do teste o sistema gerou um arquivo de *log* referente a sua execução, podendo ser observado uma amostragem através Anexo 3. Outro arquivo gerado após a execução do teste contém os valores referente ao tempo de resposta das consultas realizadas, podendo ser observado através Anexo 4.

A próxima seção abordará as informações referente ao teste de *Cache Hit* no servidor BIND.

3.4.3 Execução do teste de *Cache Hit* no servidor BIND

O segundo teste realizado no BIND, foi definido na opção de fonte de consulta (*Query Data Source)* o teste de (*Cache Latency Test (100% hit))* e no campo de quantidade de consultas (*Number of queries)* determinado o valor 1000, podendo ser observado pela Figura 28.

Figura 28 - Teste de *Cache Hit* no servidor BIND



Fonte: Próprio autor

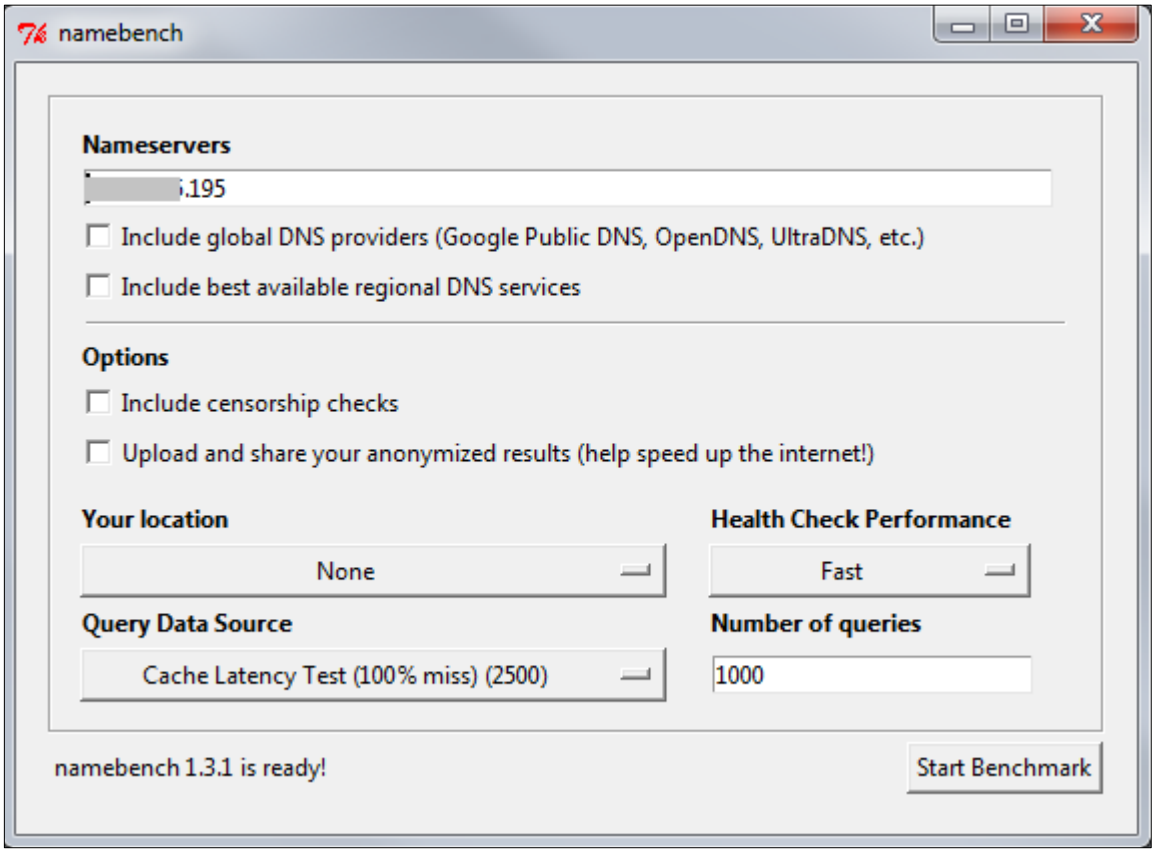
Após a execução do teste o sistema gerou um *log* de informações que podem ser observadas através do Anexo 5. Em seguida após a execução do teste o sistema gerou um arquivo contendo o tempo de respostas de cada consulta realizada, podendo ser observado pelo Anexo 6.

A seção seguinte abordará o teste de *Cache Miss* executado no servidor BIND.

3.4.4 Execução do teste de *Cache Miss* no servidor BIND

O terceiro teste realizado no servidor BIND, foi definido a quantidade de consultas (*Number of queries)* em 1000 e em fonte de consulta (*Query Data Source)* definido o teste (*Cache Latency Test (100% miss))*, podendo ser observado pela Figura 29.

Figura 29 - Teste de *Cache Miss* no servidor BIND



Fonte: Próprio autor

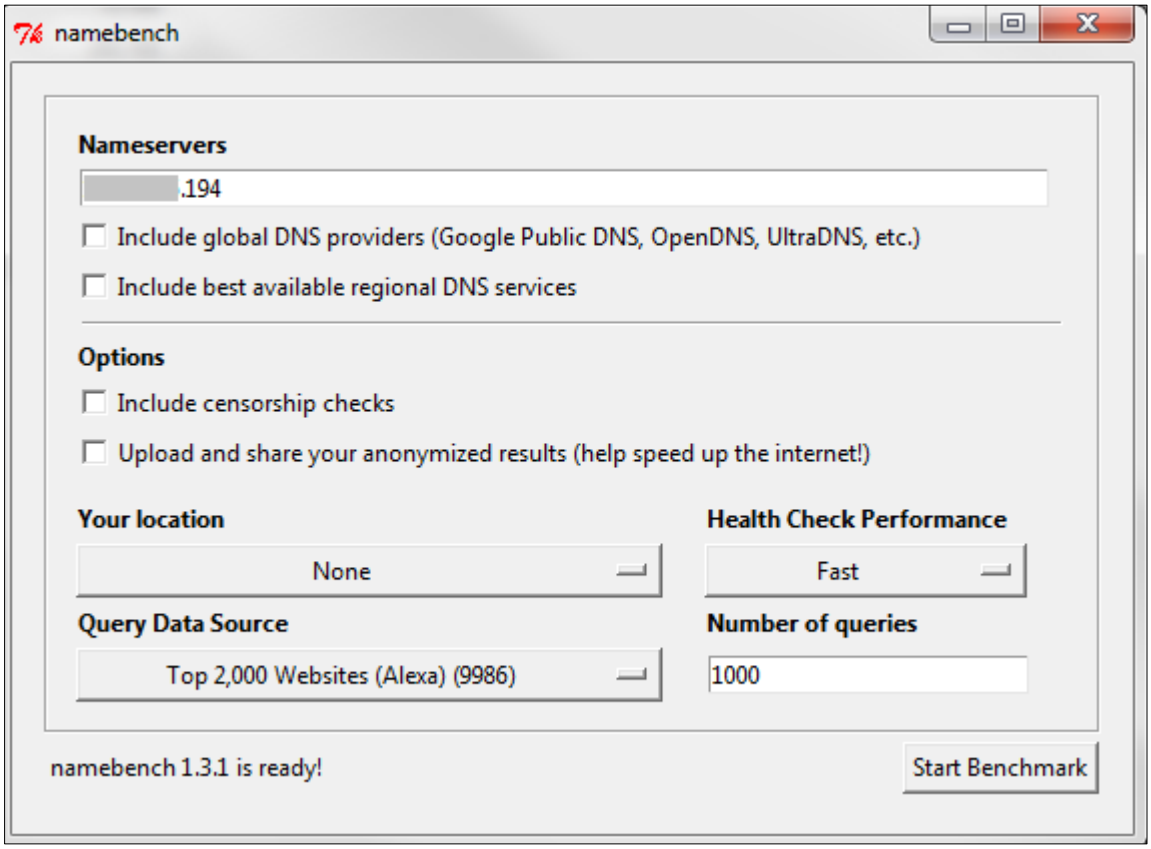
Através do Anexo 7 é possível observar o arquivo de *log* gerado após a execução do teste e através do Anexo 8 é possível observar o tempo de respostas das consultas realizadas.

A seção seguinte abordará os testes que foram executados no servidor *Unbound.*

3.4.5 Execução do teste *Top Alexa Rank* no servidor *Unbound*

O primeiro teste no servidor *Unbound* foi definido em (*Nameservers*) o endereço do servidor *XXX.XXX.XXX.194*, em fonte de consulta (*Query Data Source*) foi definido o teste (*Top Rank Alexa)* e no campo quantidade de consultas (*Number of queries*) definido o valor de 1000, podendo ser observado pela Figura 30.

Figura 30 - Teste *Top Alexa Rank* no servidor *Unbound*



Fonte: Próprio autor

Após a execução do teste, a ferramenta *Namebench* gerou o arquivo de *log* que pode ser visto através do Anexo 9. Além do arquivo de *log*, o *software* gerou o arquivo

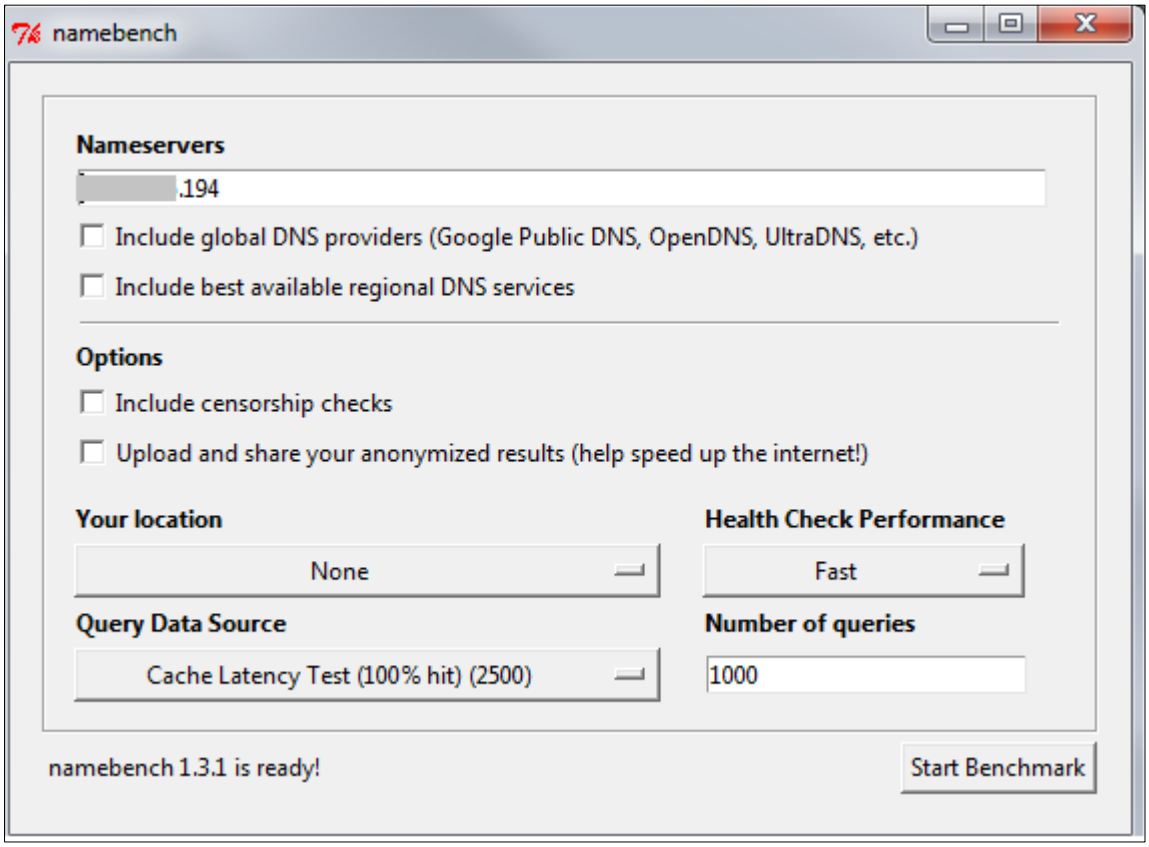
com tempo de resposta de cada consulta realizada no servidor, podendo ser observado pelo Anexo 10.

A próxima seção tem a finalidade de demonstrar os assuntos referentes a execução do teste de *Cache Hit* no servidor *Unbound*.

3.4.6 Execução do teste de *Cache Hit* no servidor *Unbound*

O segundo teste realizado no servidor *Unbound* foi definido na fonte de consulta (*Query Data Source*) o teste de (*Cache Latency Test (100% hit)*), definido em quantidade de consultas (*Number of queries*) o valor de 1000, podendo ser observado pela Figura 31.

Figura 31 - Teste de *Cache Hit* no servidor *Unbound*



Fonte: Próprio autor

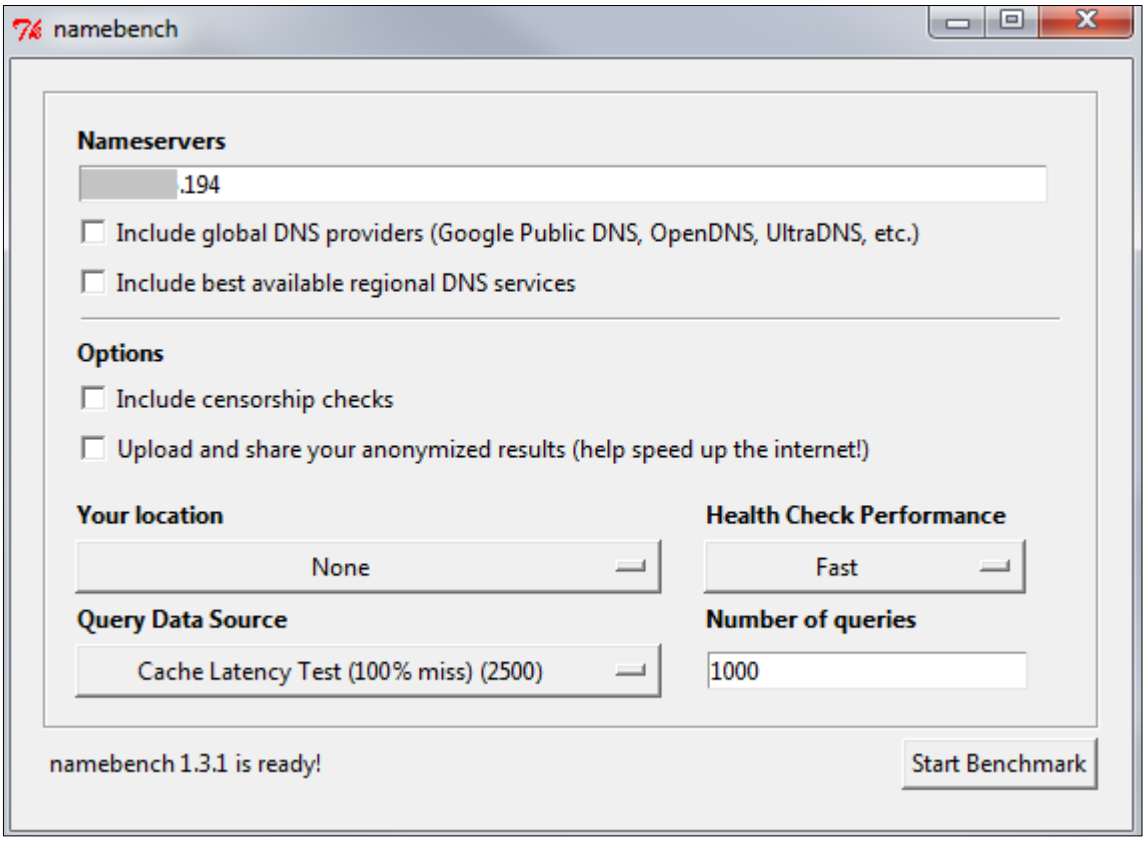
Através do Anexo 11 é possível observar as informações do arquivo de *log*, gerado pelo *software Namebench* após a execução do teste. Além do arquivo de *log* gerado é possível observar através do Anexo 12 o arquivo com o tempo de resposta das consultas.

A próxima seção trará informações referentes a execução do teste de *Cache Miss* no servidor *Unbound*.

3.4.7 Execução do teste de *Cache Miss* no servidor *Unbound*

O terceiro e último teste realizado no servidor *Unbound* foi definido na fonte de consulta (*Query Data Source*) o teste (*Cache Latency Test (100% miss)*) e a quantidade de consultas (*Number of queries*) o valor de 1000, podendo ser observado pela Figura 32.

Figura 32 - Teste de *Cache Miss* no servidor *Unbound*



Fonte: Próprio autor

Através do Anexo 13 é possível observar o arquivo de *log* gerado pelo *software Namebench* após a execução do teste. Já o Anexo 14 é possível observar o arquivo gerado contendo o tempo de resposta das consultas.

O próximo capítulo tem a finalidade de apresentar os resultados obtidos pelos testes, trazendo uma análise comparativa entre os servidores.

4 RESULTADOS

Este capítulo tem o objetivo de apresentar os resultados obtidos pela ferramenta *Namebench*, onde serão apresentados em forma gráfica o tempo de respostas que cada servidor obteve nos testes realizados, sendo possível obter uma análise detalhada do desempenho dos servidores.

Os valores obtidos pelos testes são de tempo mínimo, máximo e médio. Os valores que foram considerados para a comparação entre os servidores foram os de tempo médio, devido as consultas de DNS que não foram respondidas, excedendo o tempo máximo considerando uma perda de pacote, podendo assim, distorcer a comparação.

> Para aplicações que precisam ter controle preciso sobre o fluxo de pacotes, os erros ou a sincronização, o UDP fornece apenas aquilo que é determinado. Uma área em que ele é especialmente útil é nas situações cliente-servidor. Normalmente, o cliente envia uma solicitação curta para o servidor e espera uma resposta curta de volta. Se a solicitação ou a resposta se perderem, o cliente pode entrar em *timeout* e tentar novamente (TANENBAUM,2011).

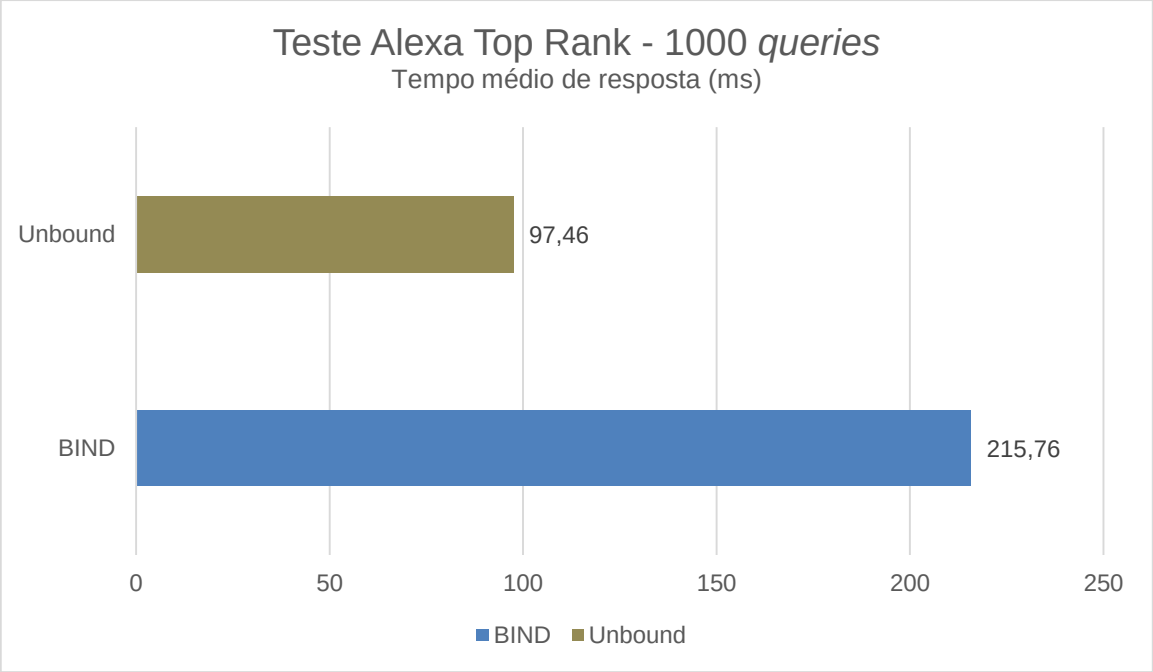
As próximas seções estão divididas em três, a seção 4.1 apresentará a comparação dos resultados do teste *Top Alexa Rank*, a seção 4.2 apresentará a comparação dos resultados do teste *Cache Miss* e a seção 4.3 a comparação dos resultados do teste *Cache Hit*.

4.1 Resultado dos testes *Top Alexa Rank*

O teste *Alexa Top Rank* teve a finalidade de realizar diversas consultas dos domínios mais acessados mundialmente em diversas localidades, classificados pela lista *Top Alexa Rank.*

Através do Gráfico 1 é possível observar que o servidor *Unbound* obteve um tempo médio de 97,46 milissegundos para realizar todas as 1000 consultas e o servidor BIND obteve o tempo de resposta médio de 215,76 milissegundos.

Gráfico 1 - Comparativo do resultado dos testes *Alexa Top Rank*



Fonte: Próprio autor

Com a proposta de obter uma análise comparativa mais detalhada em relação à porcentagem dos dados, os valores de tempo médio foram utilizados em uma regra de três simples, a fim de obter a diferença em porcentagem dos servidores, sendo assim, o resultado obtido foi o valor de 54,83, podendo afirmar que o *Unbound* foi 54,83% superior ao BIND.

LIANG et.al (2013) citam que o DNS é um componente fundamental para a maioria das aplicações da *internet*, tornando o tempo de consulta um fator crítico que afeta a qualidade da *internet* experimentada pelos usuários finais.

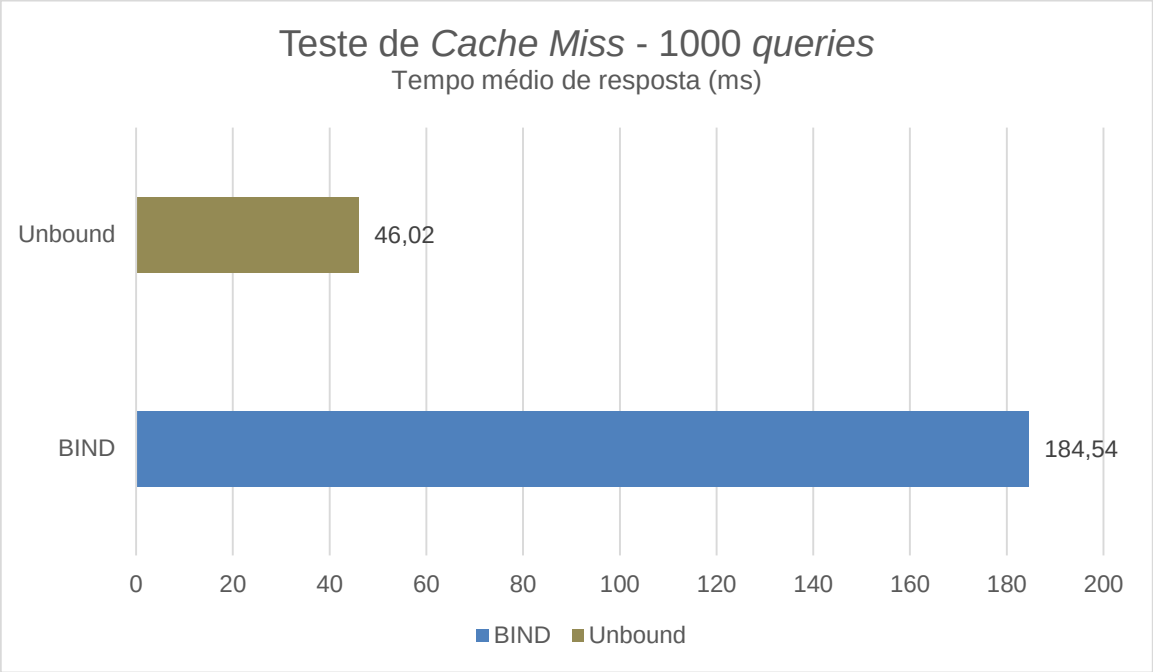
A próxima seção abordará a análise dos resultados obtidos através do teste *Cache Miss*.

4.2 Resultado dos testes *Cache Miss*

O teste *Cache Miss* teve a finalidade de executar diversas consultas com subdomínios aleatórios, fazendo com que os servidores utilizassem a recursividade em todas as consultas.

É possível observar através do Gráfico 2 que o servidor BIND obteve o tempo de resposta médio de 184,54 milissegundos das 1000 consultas realizadas e o servidor *Unbound* obteve o tempo médio de resposta o valor de 46,02 milissegundos.

Gráfico 2 - Comparativo do resultado dos testes *Cache Miss*



Fonte: Próprio autor

Novamente aplicando a regra de três simples nos valores de tempo médio, foi possível obter a diferença de 75,07, podendo afirmar que o *Unbound* foi 75,07% mais rápido em relação ao BIND.

VULIMIRI et.al (2012) citam que a baixa latência é importante para os humanos e o tempo alto de carregamento em páginas web podem reduzir significativamente as visitas e receita.

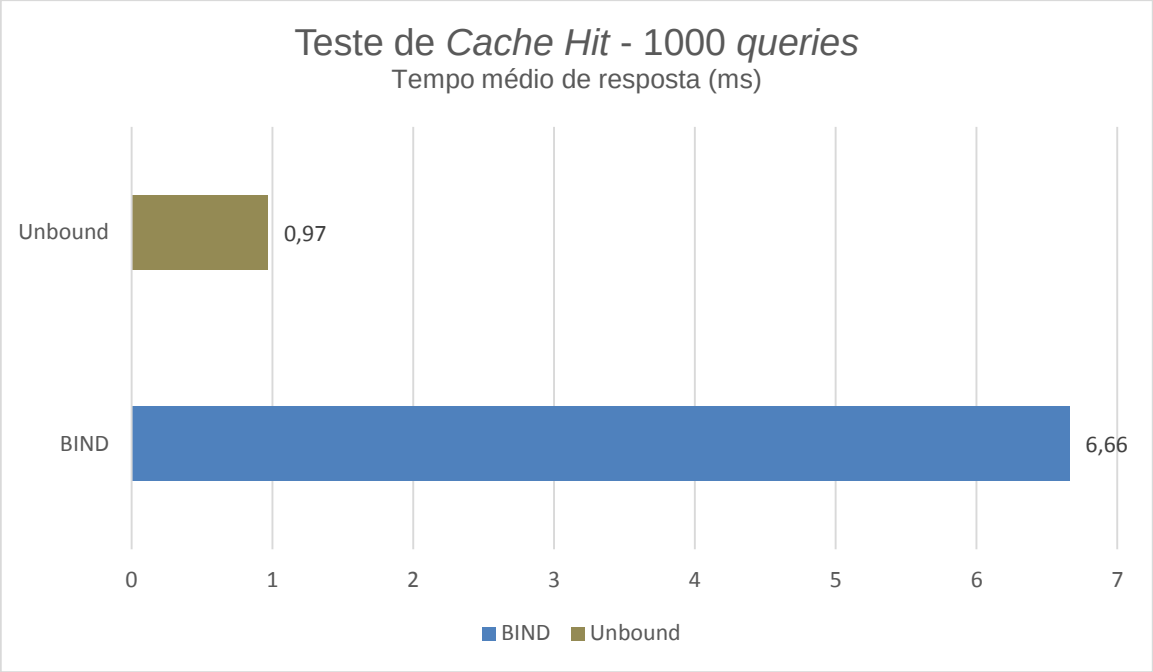
A próxima seção tem a finalidade de apresentar os resultados obtidos através do teste de *Cache Hit*.

4.3 Resultado dos testes *Cache Hit*

O teste *Cache Hit* teve a finalidade de avaliar o desempenho em *cache* dos servidores, realizando diversas consultas contendo o mesmo domínio, fazendo com que o servidor não utilizasse a recursividade e respondesse as consultas com as informações já obtidas em *cache*.

Através do Gráfico 3 é possível observar que o tempo médio de resposta das 1000 consultas realizadas no servidor *Unbound* foi de 0,97 milissegundos *e o* servidor BIND obteve o tempo médio de 6,66 milissegundos.

Gráfico 3 - Comparativo do resultado dos testes *Cache Hit*



Fonte: Próprio autor

Aplicando novamente a regra de três simples entre os valores de tempo médio, foi possível obter a diferença de 85,5. Sendo possível realçar novamente a diferença entre os servidores, onde o *Unbound* foi 85,5% mais rápido que o BIND.

Por motivos de eficiência, o DNS depende muito do armazenamento em *cache* e todas informações que um servidor entrega a um requisitante são armazenadas em *cache*. Para uma boa experiência do usuário final, os tempos rápidos de resposta do DNS são importantes (AGER; SMARAGDAKIS, 2010). “O uso de respostas em *cache* reduz

bastante as etapas de uma consulta e melhora o desempenho” (TANENBAUM,2011, p.391).

O próximo capítulo tem a finalidade de abordar assuntos referentes as conclusões obtidas pela presente pesquisa.

5 CONCLUSÃO

A produção deste trabalho possibilitou realizar a implementação de um servidor de DNS recursivo e uma análise de desempenho comparativa entre servidores de DNS, através de técnicas e testes de *benchmark*. Com a finalidade de propor melhorias nos serviços ofertados em um provedor de *internet*.

Após a análise dos resultados obtidos pelos testes foi possível concluir que, para responder as consultas dos domínios mais acessados mundialmente através do teste (*Top Alexa Rank*) o servidor *Unbound* foi 54,83% mais rápido que o servidor BIND. Com relação a recursividade (*Cache Miss*) o servidor *Unbound* foi 75,07% superior ao servidor BIND, resolvendo consultas que não estão armazenadas em *cache* realizando todo o mapeamento da hierarquia de domínios de forma mais rápida. Por fim, a velocidade de resposta utilizando as informações já contidas em *cache* através do teste de (*Cache Hit*) o *Unbound* foi superior ao BIND 85,5%, demonstrando assim, um melhor desempenho do servidor utilizando o *software Unbound*.

Neste estudo de caso o servidor *Unbound* foi mais eficiente que o BIND em todos as demandas de um servidor de DNS recursivo, seja a recursividade, armazenamento em *cache*, domínios mais acessados e a segurança com DNSSEC. O que possibilitou concluir que o *hardware* não foi um fator determinante para obter um melhor desempenho.

Além do desempenho, o servidor *Unbound* utilizando a validação de DNSSEC fez com que a segurança fosse um fator importante para determinar a sua utilização. Sendo assim, a empresa passou a utilizar dois servidores de DNS recursivos, sendo o *Unbound* como primário e o BIND como secundário, proporcionando um serviço de alta disponibilidade.

Após a produção deste estudo de caso também foi possível concluir que utilizando softwares *opensource*, foi possível realizar mudanças significativas na qualidade e segurança dos serviços ofertados pela empresa. Com isso, a empresa promove a seus clientes um serviço de DNS recursivo com maior qualidade e segurança, podendo assim,

ressaltar a eficiência do *software Unbound* como servidor de DNS recursivo. Tornando possível concluir que o objetivo deste estudo de caso foi atingido de forma satisfatória.

6 TRABALHOS FUTUROS

A implementação do servidor *Unbound* possibilitou obter uma métrica padronizada do seu arquivo de configuração, que pode ser replicada em outros servidores, sendo assim, ao decorrer do tempo será necessário avaliar essas métricas e reavaliar seu desempenho.

Uma contribuição para trabalhos futuros, seria realizar a implementação do serviço de DNS com a técnica ECMP (*Equal-Cost Multi-Path Routing*), sendo uma estratégia de roteamento e balanceamento de carga, o que poderia possibilitar em um maior desempenho.

Uma outra proposta, seria realizar um comparativo de desempenho com outros softwares de DNS que não são *opensource*.

REFERÊNCIAS

AGER, Bernhard; SMARAGDAKIS, Georgios; MUHLBAUER, Wolfgang; UHLIG, Steve. *Comparing DNS Resolvers in the Wild*. IMC'10, Nov 1–3, 2010, Melbourne, Australia.

AHMAD, Ejaz; SARWAR, Kashif. *A Comparative Analysis on Existing DNS Performance Measurement Mechanisms*. *International Journal of Computer Networks and Communications Security*. Vol 2. 05 de Maio 2014. Disponível em < http://www.ijcncs.org/published/volume2/issue5/p3_2-5.pdf >

AITCHISON, Ron. *Pro DNS and BIND 10*. 1. Ed, 2011.

ALBITZ, Paul; LIU, Cricket. *DNS and BIND*, 5th Edition. Nutshell Series. O'Reilly and Associates, Inc., 2006.

ALENCAR, Márcio Aurélio. *Fundamentos de Redes de Computadores*. Centro de Educação Tecnológica do Amazonas. 2010.

ALEXA, *ALEXA TOP RANK.* Disponível em <https://www.alexa.com/about> Acesso em 1 de Ago. 2018.

BIND. *Versatile, Classic, Complete Name Server Software*. Disponível em: < https://www.isc.org/downloads/bind/ > Acesso em 19 Set.2018.

BOULAKHRIF, Hamza, *Analysis of DNS Resolver Performance Measurements*. 2015, Disponível em <www.nlnetlabs.nl/downloads/publications/os3-2015-rp2-hamza-boulakhrif.pdf> Acesso em 29 Mar.2018.

CAMPOS, Augusto. *O que é software livre*. BR-Linux. Florianópolis, março de 2006. Disponível em <http://br-linux.org/linux/faq-softwarelivre>. Acesso em 25 de Outubro de 2018.

CERT.BR. *Sobre o Centro de Estudos, Respostas e Tratamento de Incidentes de Segurança do Brasil*. Disponível em < https://www.cert.br/sobre/ > Acesso 19 de Set.2018.

COMER, Douglas, Earl. *Interligação de Redes com TCP/IP*. 6.Ed. Rio de Janeiro,2015.

DEBIAN.ORG *Sobre o GNU/Linux Debian*. Disponível em < https://www.debian.org/intro/about >.

F5Networks, Inc. *DNSSEC: The Antidote to DNS Cache Poisoning and Other DNS Attacks* .F5 Networks, Inc, 2015.

FAIL2BAN. *FAIL2BAN Main Page*. Disponível em < https://www.fail2ban.org/wiki/index.php/Main_Page> Acesso em 25 de Set. 2018.

FOROUZAN, Behrouz. *Comunicação de dados e redes de computadores*. 4. Ed. São Paulo 2010.

Free Software Foundation, *General License Public Version 3*. Disponível em < https://www.gnu.org/licenses/gpl-3.0.html > Acesso em 25 de Set.2018.

GOOGLE DNS. *Configure your network settings to use Google Public DNS*. Disponível em: < https://developers.google.com/speed/public-dns/docs/using > Acesso em 19 Mar. 2018.

BATISTA, Gustavo. *Introdução à rede ipê*. Escola Superior de Redes. Rede Nacional de Ensino e Pesquisa. Rio de Janeiro 2013

HAHN, Erin. *An Overview of Open-Source Software Licenses and the Value of Open-Source Software to Public Health Initiatives*. JOHNS HOPKINS APL TECHNICAL DIGEST, VOLUME 32, NUMERO 4 (2014).

HERTZOG, Raphael; MAS, Roland. *The Debian Administrator's Handbook - Debian Jessie from Discovery to Mastery*. 2015.

HOEPERS, Cristine; STEDING-JESSEN, Klaus; MURILO, Nelson.; OBELHEIRO, Rafael. *Recomendações para Evitar o Abuso de Servidores DNS Recursivos Abertos V1.8 20-07-2016.* Disponível em: < https://www.cert.br/docs/whitepapers/dns-recursivo-aberto/> Acesso em 28 Mar. 2018.

HUPPLER, Karl. *The Art of Building a Good Benchmark. Performance Evaluation and Benchmarking*, volume 5895 *of Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2009.

ICANN. *FAQs – O que é o InterNIC?*. Disponível em < https://www.icann.org/resources/pages/faqs-2014-01-22-pt > Acesso em 25 Out.2018.

IDABC. *All about Open Source - An Introduction to Open Source Software for Government IT.* 2010.

INTERNET WORD STATS. *Internet Users in the World*. Disponível em: < https://www.internetworldstats.com/stats.htm > Acesso em 3 de Mar. 2018.

ISC. *Chapter 2. BIND Resource Requirements.* Disponível em < https://ftp.isc.org/www/bind/arm95/Bv9ARM.ch02.html > Acesso em 11 Set. 2018.

KISTOWSKI, Jóakim; et.al. *How to Build a Benchmark*. Disponível em < https://www.researchgate.net/publication/273133047 > Acesso em 15 de Ago.2018.

KUROIWA, Cesar. *Tutorial DNSSEC*. 2012. Disponível em :<ftp://ftp.registro.br/pub/doc/tutorial-dnssec.pdf > Acesso em 30 Mar. 2018.

KUROSE, James; ROSS, Keith. *Redes de Computadores e a Internet*. 6. Ed. São Paulo, 2014.

LIANG, Jinjin; JIANG, Jian; DUAN, Haixin; LI, Kang; WU, Jianping. *Measuring Query Latency of Top Level DNS Servers. International Conference on Passive and Active Network Measurement*. 2013. Disponível em < https://pdfs.semanticscholar.org/bfaf/e26d192d8f2102ecab45c307014e1656cc6c.pdf > Acesso em 1 de Nov 2018.

LYRA, Mauricio, R. *Governança da segurança da informaçã*o. 1. Ed, Brasília. 2015.

MARTINELO, Clériston. BELLIZE, Marcos. *Análise de Vulnerabilidades com OpenVAS e Nessus*. 2014.

MIKROTIK. *About Mikrotik*. Disponível em < https://mikrotik.com/aboutus > Acesso em 1 de Out.2018.

MITCHELL, Bradley. *Servers are the heart of the internet,* 2 de Jun, 2018. Disponível em <https://www.lifewire.com/servers-in-computer-networking-817380> Acesso em 2 Set.2018.

MITROKOTSA, Aikaterini; DOULIGERIS, Christos. *DDoS attacks and defense mechanisms: classification and state-of-the-art. Department of Informatics University of Piraeus*, Grécia. 2003.

NAMEBENCH. *Open-source DNS Benchmark Utility*. Disponível em :< https://code.google.com/archive/p/namebench/> Acesso em 31 Mar. 2018.

NLNETLABS. *About Unbound*. Disponível em: < https://www.nlnetlabs.nl/projects/unbound/about/ > Acesso em 19 Mar. 2018.

OLIVEIRA, Ronielton. *Criptografia simétrica e assimétrica: os principais algoritmos de cifragem*. Revista segurança digital. 2012. Disponível em <http://www.ronielton.eti.br/publicacoes/artigorevistasegurancadigital2012.pdf> Acesso em 26 de Out 2018.

OPEN DNS. *Security for the way the world works today.* Disponível em: < https://www.opendns.com/about > Acesso em 19 Mar. 2018.

PROJETO DEBIAN, *Debian*. Disponível em < https://www.debian.org/doc/manuals/project-history/project-history.pt.pdf> Acesso em 25 de Ago.2018.

REGISTRO.BR. *Quantidade de domínios registrado no Brasil*. Disponível em: < https://registro.br/estatisticas.html > Acesso em 16 Jun. 2018.

RFC 4180. Disponível em <https://tools.ietf.org/html/rfc4180> Acesso em 10 de Out 2018.

RFC 7871. Disponível em <https://tools.ietf.org/html/rfc7871> Acesso em 09 de Out 2018.

RIJSWIJK-DEIJ, Roland. *Deploying DNSSEC - Validation on recursive caching name servers*. V2.0. Agosto 2012.

ROSEN, Lawrence. *Open Source Licensing. Software Freedom and Intellectual Property Law*. Prentice Hall. Jul 2004.

SABINO, Vanessa; KON, Fabio. *Licenças de Software Livre História e Características*. Relatório Técnico RT-MAC-IME-USP 2009-01. IME SP. Disponível em < http://www.ccsl.org.br/files/relatorio-licencas.pdf > Acesso em 20 de Outubro 2018.

SON, Sooel; SHMATIKOV, Vitaly. *The Hitchhiker's Guide to DNS Cache Poisoning*. The *University of Texas at Austin*. 2010. Disponível em <https://www.cs.cornell.edu/~shmat/shmat_securecomm10.pdf> Acesso em 15 Ago.2018.

STENBERG, Daniel. *README cURL*. Disponível em < https://github.com/curl/curl > Acesso em 20 de Set.2018.

SUN, Hung-Min; CHANG, Wen-Hsuan; CHANG, Shih-Ying, LIN, Yue-Hsun. *DepenDNS: Dependable Mechanism against DNS Cache Poisoning.Information Security Laboratory, Department of Computer Science,National Tsing Hua University*, Taiwan R.O.C. 2009. 15 páginas.

TANENBAUM, Andrew. *Redes de Computadores*. 5. Ed. São Paulo, 2011.

THAKUR, Arjun; SANGAL, L, A; BINDRA, Harminder. *Quantitative Measurement and Comparison of Effects of Various Search Engine Optimization Parameters on Alexa*. *International Journal of Computer Applications*. Traffic Rank. Volume 26–No.5, July 2011.

TSAI, John, *For Better or worse: Introducing the GNU General Public License Version 3*, 23 Berkeley Tech. L.J. 547 (2008).

TURGUT, Tarcan; ROHPRIMARDHO. *Peeling the Google Public DNS Onion*, *Universiteit van Amsterdam System and Network Engineering*. 8 de Fev.2015. Disponível em < http://www.delaat.net/rp/2014-2015/p75/report.pdf > Acesso em 24 de Set.2018.

VERISIGN. *New Open Source DNS Server Released Today*. *Unbound - A Secure, High-Performance Alternative to BIND -Makes its Debut within Open Source Community*. Amsterdam, The Netherlands and Mountain View, CA – 20 de Maio de 2008.

VIM. *About VI IMproved*. Disponível em <https://www.vim.org/about.php> Acesso em 29 Set.2018.

VULIMIRI, Ashish; et.al. *More is Less: Reducing Latency via Redundancy*. Redmond, Washington Outubro 29-30, 2012. Disponível em <http://conferences.sigcomm.org/hotnets/2012/papers/hotnets12-final34.pdf> Acesso em 1 de Nov 2018.

WILEY, John & Sons. *Windows Server Administration Fundamentals, Exam 98-365* . 2011.

YLONEN, Tatu. *The Secure Shell (SSH) Transport Layer Protocol*. RFC4253. Disponível em < https://tools.ietf.org/html/rfc4253.html > Acesso em 29 Set.2018.

YU, Yingdi; LARSON, Matt; WESSELS, Duane; ZHANG, Lixia. *Authority Server Selection of DNS Caching Resolvers*. ACM SIGCOMM *Computer Communication Review*, Volume 42, Número 2, Abril 2012. 6 pág.

ANEXO 1 – AUTORIZAÇÃO PARA REDAÇÃO DE ESTUDO DE CASO

FORMULÁRIO DE LIBERAÇÃO PARA REDAÇÃO DE ESTUDO DE CASO

Pela presente, em nome da Prodata Informática e Cadastro Ltda, a qual represento neste ato, autorizo Jefte de Lima Ferreira a iniciar um estudo de caso para fins acadêmicos para a FACULDADE DOCTUM CARATINGA, autorizo o uso do nome empresarial para a redação, podendo distribui-lo e publicá-lo em sites, revistas, livros e coletâneas de casos que venham a ser organizados pela citada escola, sem nenhum ônus, cedendo todos os direitos inerentes a propriedade intelectual do caso à FACULDADE DOCTUM CARATINGA.

Data: 26/06/2018

Assinatura: _____

Nome completo do representante legal: Aginito Pedro da Silveira Filho

Empresa: Prodata Informática e Cadastro Ltda.

CNPJ: 22.058.192/0001-20

Endereço: Avenida Moacir de Mattos N° 229 – Caratinga – Minas Gerais

Telefone: (33) 3322-6363

ANEXO 2 – ARQUIVO *UNBOUND.CONF*

server:

 # Número de threads que serão criadas para responder as requisições.
num-threads: 4

 # Numero de 'slabs' no cache.
msg-cache-slabs: 8

 # Numero de 'slabs' no cache de RRset.
rrset-cache-slabs: 8

 # Numero de 'slabs' na infraestrutura do cache.
infra-cache-slabs: 8

 # Numero de 'slabs' na chave do cache.
key-cache-slabs: 8

 # Tamanho do cache RRset
rrset-cache-size: 512m

 # Tamanho das mensagens em cache
msg-cache-size: 512m

 # Numero de portas que serão abertas
outgoing-range: 32768

 # Definido que serão aberto sockets de escuta para consultas de entrara e saída para cada thread.
so-reuseport: yes

 # Quantidade de hosts para quais as informações são armazenadas em cache.
infra-cache-numhosts: 100000

 # Quantidade de queries que cada thread atenderá simultaneamente.
num-queries-per-thread: 4096

 # Tamanho do buffer na porta UDP 53
so-rcvbuf: 32m

 # Esconde a identidade do servidor
hide-identity: yes

 # Esconde a versão do servidor
hide-version: yes

 # Habilita a função que, os elementos do cache são pre-buscados antes de expirarem, para manter o cache atualizado.
prefetch: yes

 # Habilita a função que, busca os DNSKEYs antes do processo de validação, reduzindo a latência das consultas.
prefetch-key: yes

 # Utiliza bits aleatórios codificados por 0x20 na consulta, para impedir tentativas de falsificação.
use-caps-for-id: yes

 # Habilita a rotação da ordem dos RRsets nas respostas.
rrset-roundrobin: yes

 # Tamanho mínimo de time to live para realizar o cache das informações
cache-min-ttl: 3600

 # Tamanho máximo de time to live para realizar o cache das informações
cache-max-ttl: 86400

 # Quantidade de respostas indesejadas, quando atingido o limite uma ação defensiva é tomada e gerado uma informação no log.
unwanted-reply-threshold: 10000

 # Level de verbosidade do LOG que será explícito.
verbosity: 2

 # IP configurado na interface de rede do servidor.
interface: XXX.XXX.XXX.194

 # Definição da porta padrão do serviço de DNS
port: 53

 # Habilita o uso do protocolo de IPV4
do-ip4: yes

 # Habilita o uso do protocolo de IPV6
do-ip6: no

 # Habilita o uso do protocolo UDP

do-udp: yes

 # Habilita o uso do protocolo TCP
do-tcp: yes

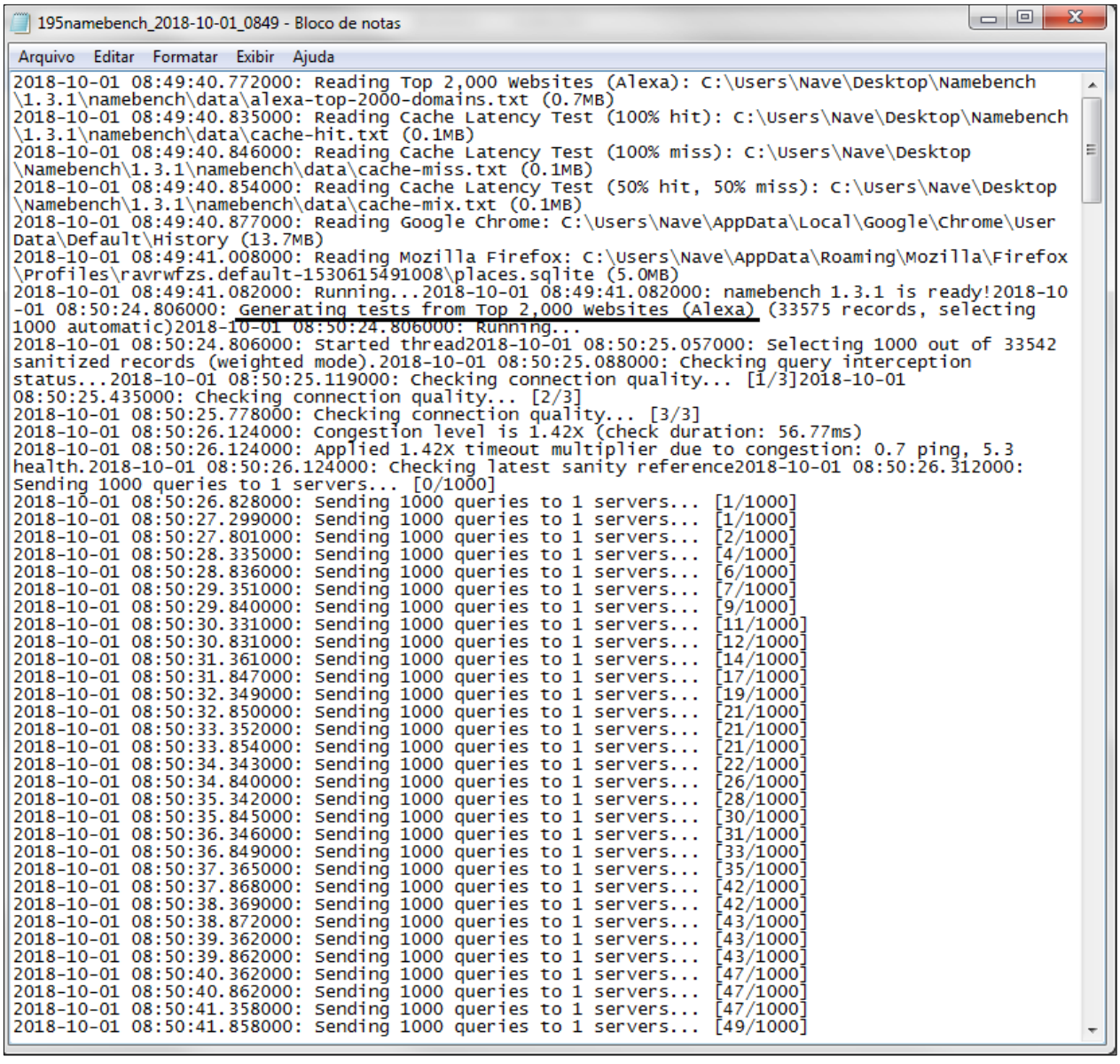
 # Especifica a rede na qual o servidor aceitará as requisições de DNS.
access-control: XXX.XXX.XXX.0/21 allow

 # Diretório e arquivo onde o servidor armazena as informações referentes ao LOG.
logfile: "/var/log/unbound/unbound.log"

 # Diretório onde está localizado as informações dos root-servers.
root-hints: "/etc/unbound/root.hints"

ANEXO 3 – LOG *TOP ALEXA RANK* SERVIDOR BIND

Arquivo gerado pela ferramenta *Namebench* após a execução do teste *Top Alexa Rank* no servidor BIND.



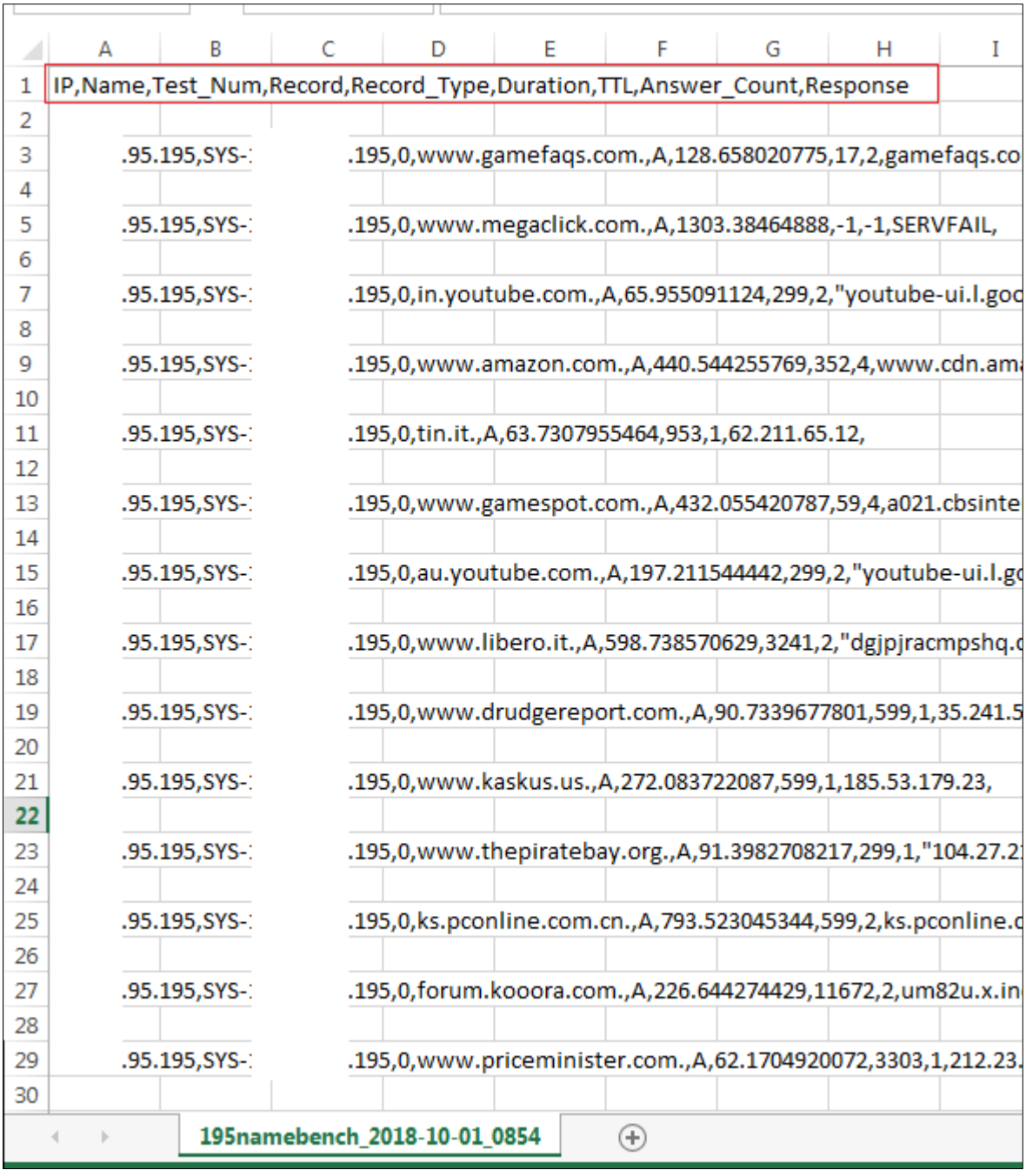
195namebench_2018-10-01_0849 - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

```
2018-10-01 08:49:40.772000: Reading Top 2,000 Websites (Alexa): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\alexa-top-2000-domains.txt (0.7MB)
2018-10-01 08:49:40.835000: Reading Cache Latency Test (100% hit): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\cache-hit.txt (0.1MB)
2018-10-01 08:49:40.846000: Reading Cache Latency Test (100% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-miss.txt (0.1MB)
2018-10-01 08:49:40.854000: Reading Cache Latency Test (50% hit, 50% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-mix.txt (0.1MB)
2018-10-01 08:49:40.877000: Reading Google Chrome: C:\Users\Nave\AppData\Local\Google\Chrome\User
Data\Default\History (13.7MB)
2018-10-01 08:49:41.008000: Reading Mozilla Firefox: C:\Users\Nave\AppData\Roaming\Mozilla\Firefox
\Profiles\ravrwfzs.default-1530615491008\places.sqlite (5.0MB)
2018-10-01 08:49:41.082000: Running...2018-10-01 08:49:41.082000: namebench 1.3.1 is ready!2018-10
-01 08:50:24.806000: Generating tests from Top 2,000 Websites (Alexa) (33575 records, selecting
1000 automatic)2018-10-01 08:50:24.806000: Running...
2018-10-01 08:50:24.806000: Started thread2018-10-01 08:50:25.057000: Selecting 1000 out of 33542
sanitized records (weighted mode).2018-10-01 08:50:25.088000: Checking query interception
status...2018-10-01 08:50:25.119000: Checking connection quality... [1/3]2018-10-01
08:50:25.435000: Checking connection quality... [2/3]
2018-10-01 08:50:25.778000: Checking connection quality... [3/3]
2018-10-01 08:50:26.124000: Congestion level is 1.42X (check duration: 56.77ms)
2018-10-01 08:50:26.124000: Applied 1.42X timeout multiplier due to congestion: 0.7 ping, 5.3
health.2018-10-01 08:50:26.124000: Checking latest sanity reference2018-10-01 08:50:26.312000:
Sending 1000 queries to 1 servers... [0/1000]
2018-10-01 08:50:26.828000: Sending 1000 queries to 1 servers... [1/1000]
2018-10-01 08:50:27.299000: Sending 1000 queries to 1 servers... [1/1000]
2018-10-01 08:50:27.801000: Sending 1000 queries to 1 servers... [2/1000]
2018-10-01 08:50:28.335000: Sending 1000 queries to 1 servers... [4/1000]
2018-10-01 08:50:28.836000: Sending 1000 queries to 1 servers... [6/1000]
2018-10-01 08:50:29.351000: Sending 1000 queries to 1 servers... [7/1000]
2018-10-01 08:50:29.840000: Sending 1000 queries to 1 servers... [9/1000]
2018-10-01 08:50:30.331000: Sending 1000 queries to 1 servers... [11/1000]
2018-10-01 08:50:30.831000: Sending 1000 queries to 1 servers... [12/1000]
2018-10-01 08:50:31.361000: Sending 1000 queries to 1 servers... [14/1000]
2018-10-01 08:50:31.847000: Sending 1000 queries to 1 servers... [17/1000]
2018-10-01 08:50:32.349000: Sending 1000 queries to 1 servers... [19/1000]
2018-10-01 08:50:32.850000: Sending 1000 queries to 1 servers... [21/1000]
2018-10-01 08:50:33.352000: Sending 1000 queries to 1 servers... [21/1000]
2018-10-01 08:50:33.854000: Sending 1000 queries to 1 servers... [21/1000]
2018-10-01 08:50:34.343000: Sending 1000 queries to 1 servers... [22/1000]
2018-10-01 08:50:34.840000: Sending 1000 queries to 1 servers... [26/1000]
2018-10-01 08:50:35.342000: Sending 1000 queries to 1 servers... [28/1000]
2018-10-01 08:50:35.845000: Sending 1000 queries to 1 servers... [30/1000]
2018-10-01 08:50:36.346000: Sending 1000 queries to 1 servers... [31/1000]
2018-10-01 08:50:36.849000: Sending 1000 queries to 1 servers... [33/1000]
2018-10-01 08:50:37.365000: Sending 1000 queries to 1 servers... [35/1000]
2018-10-01 08:50:37.868000: Sending 1000 queries to 1 servers... [42/1000]
2018-10-01 08:50:38.369000: Sending 1000 queries to 1 servers... [42/1000]
2018-10-01 08:50:38.872000: Sending 1000 queries to 1 servers... [43/1000]
2018-10-01 08:50:39.362000: Sending 1000 queries to 1 servers... [43/1000]
2018-10-01 08:50:39.862000: Sending 1000 queries to 1 servers... [43/1000]
2018-10-01 08:50:40.362000: Sending 1000 queries to 1 servers... [47/1000]
2018-10-01 08:50:40.862000: Sending 1000 queries to 1 servers... [47/1000]
2018-10-01 08:50:41.358000: Sending 1000 queries to 1 servers... [47/1000]
2018-10-01 08:50:41.858000: Sending 1000 queries to 1 servers... [49/1000]
```

ANEXO 4 – RESULTADO TESTE *TOP ALEXA RANK* SERVIDOR BIND

Arquivo gerado pela ferramenta *Namebench* após o teste *Top Alexa Rank* no servidor BIND, contendo uma amostragem dos primeiros resultados.

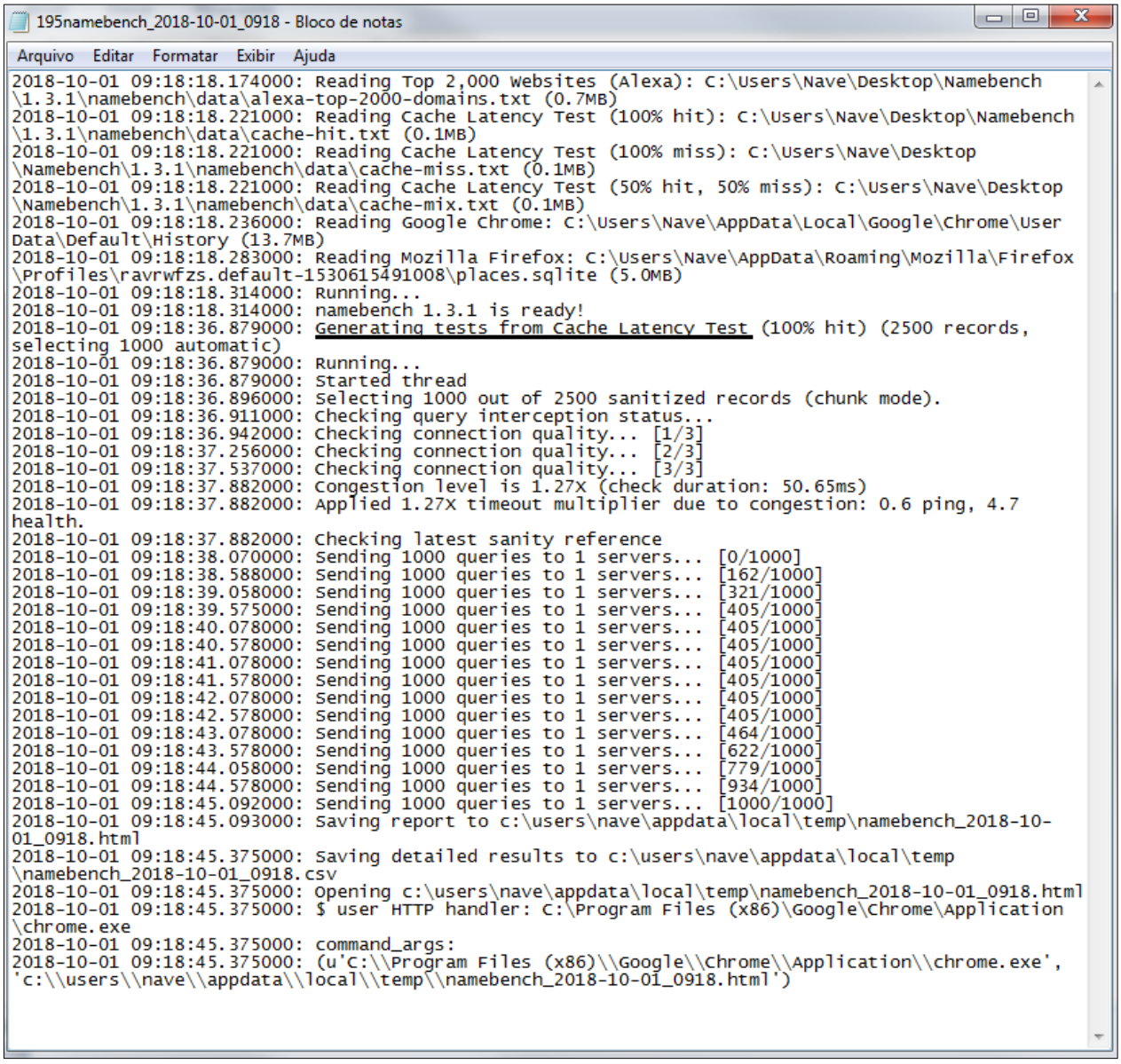


	A	B	C	D	E	F	G	H	I
1	IP,Name,Test_Num,Record,Record_Type,Duration,TTL,Answer_Count,Response								
2									
3	.95.195,SYS-:			.195,0,www.gamefaqs.com.,A,128.658020775,17,2,gamefaqs.co					
4									
5	.95.195,SYS-:			.195,0,www.megaclick.com.,A,1303.38464888,-1,-1,SERVFAIL,					
6									
7	.95.195,SYS-:			.195,0,in.youtube.com.,A,65.955091124,299,2,"youtube-ui.l.goc					
8									
9	.95.195,SYS-:			.195,0,www.amazon.com.,A,440.544255769,352,4,www.cdn.am					
10									
11	.95.195,SYS-:			.195,0,tin.it.,A,63.7307955464,953,1,62.211.65.12,					
12									
13	.95.195,SYS-:			.195,0,www.gamespot.com.,A,432.055420787,59,4,a021.cbsinte					
14									
15	.95.195,SYS-:			.195,0,au.youtube.com.,A,197.211544442,299,2,"youtube-ui.l.g					
16									
17	.95.195,SYS-:			.195,0,www.libero.it.,A,598.738570629,3241,2,"dgjpjracmpshq.c					
18									
19	.95.195,SYS-:			.195,0,www.drudgereport.com.,A,90.7339677801,599,1,35.241.5					
20									
21	.95.195,SYS-:			.195,0,www.kaskus.us.,A,272.083722087,599,1,185.53.179.23,					
22									
23	.95.195,SYS-:			.195,0,www.thepiratebay.org.,A,91.3982708217,299,1,"104.27.2					
24									
25	.95.195,SYS-:			.195,0,ks.pconline.com.cn.,A,793.523045344,599,2,ks.pconline.c					
26									
27	.95.195,SYS-:			.195,0,forum.kooora.com.,A,226.644274429,11672,2,um82u.x.in					
28									
29	.95.195,SYS-:			.195,0,www.priceminister.com.,A,62.1704920072,3303,1,212.23.					
30									

195namebench_2018-10-01_0854

ANEXO 5 – LOG *CACHE HIT* SERVIDOR BIND

Arquivo de log gerado pela ferramenta *Namebench* após a execução do teste de *Cache Hit* no servidor BIND.



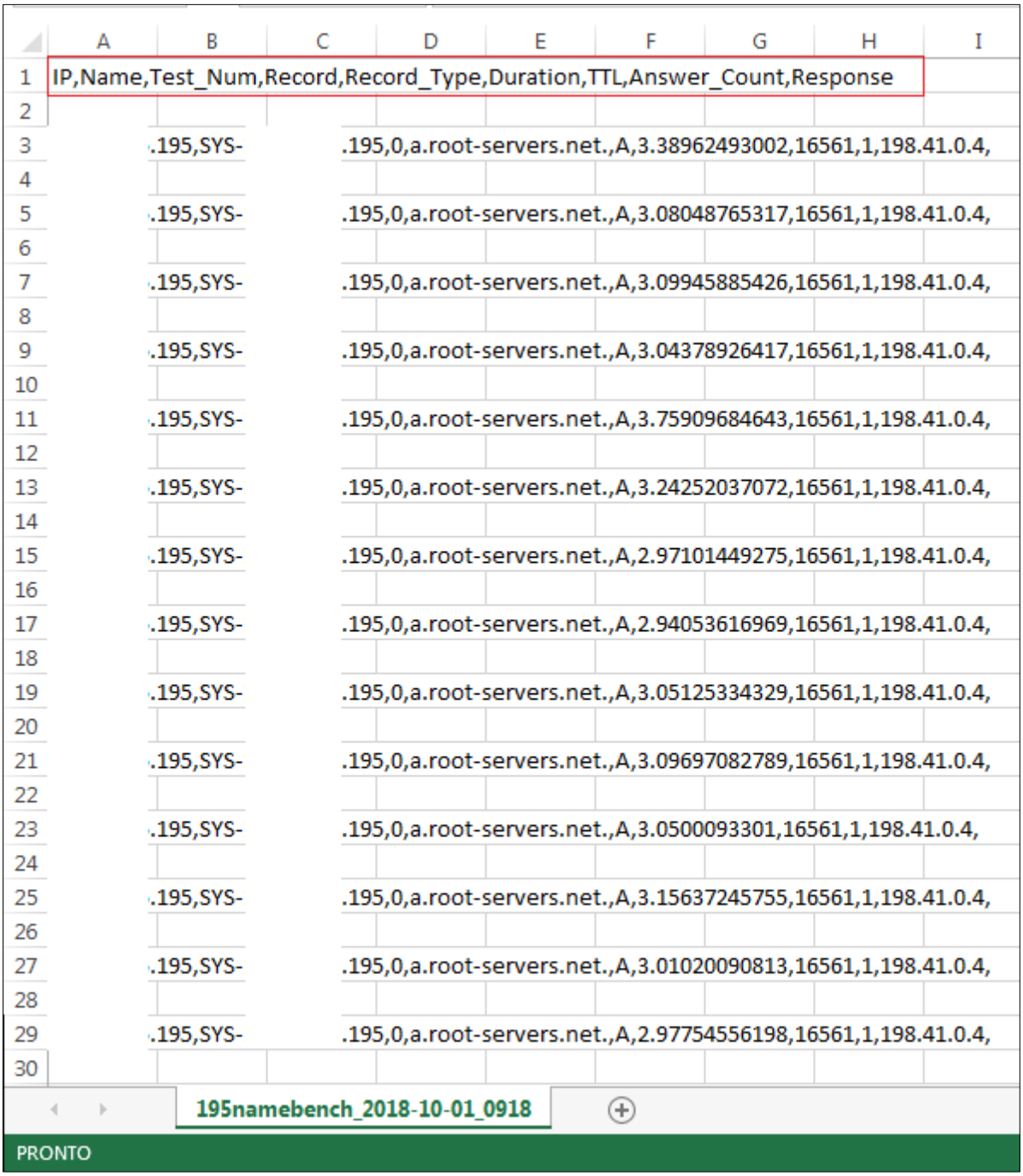
195namebench_2018-10-01_0918 - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

```
2018-10-01 09:18:18.174000: Reading Top 2,000 Websites (Alexa): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\alexa-top-2000-domains.txt (0.7MB)
2018-10-01 09:18:18.221000: Reading Cache Latency Test (100% hit): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\cache-hit.txt (0.1MB)
2018-10-01 09:18:18.221000: Reading Cache Latency Test (100% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-miss.txt (0.1MB)
2018-10-01 09:18:18.221000: Reading Cache Latency Test (50% hit, 50% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-mix.txt (0.1MB)
2018-10-01 09:18:18.236000: Reading Google Chrome: C:\Users\Nave\AppData\Local\Google\Chrome\User
Data\Default\History (13.7MB)
2018-10-01 09:18:18.283000: Reading Mozilla Firefox: C:\Users\Nave\AppData\Roaming\Mozilla\Firefox
\Profiles\ravrwfzs.default-1530615491008\places.sqlite (5.0MB)
2018-10-01 09:18:18.314000: Running...
2018-10-01 09:18:18.314000: namebench 1.3.1 is ready!
2018-10-01 09:18:36.879000: Generating tests from Cache Latency Test (100% hit) (2500 records,
selecting 1000 automatic)
2018-10-01 09:18:36.879000: Running...
2018-10-01 09:18:36.879000: Started thread
2018-10-01 09:18:36.896000: Selecting 1000 out of 2500 sanitized records (chunk mode).
2018-10-01 09:18:36.911000: Checking query interception status...
2018-10-01 09:18:36.942000: Checking connection quality... [1/3]
2018-10-01 09:18:37.256000: Checking connection quality... [2/3]
2018-10-01 09:18:37.537000: Checking connection quality... [3/3]
2018-10-01 09:18:37.882000: Congestion level is 1.27X (check duration: 50.65ms)
2018-10-01 09:18:37.882000: Applied 1.27X timeout multiplier due to congestion: 0.6 ping, 4.7
health.
2018-10-01 09:18:37.882000: Checking latest sanity reference
2018-10-01 09:18:38.070000: Sending 1000 queries to 1 servers... [0/1000]
2018-10-01 09:18:38.588000: Sending 1000 queries to 1 servers... [162/1000]
2018-10-01 09:18:39.058000: Sending 1000 queries to 1 servers... [321/1000]
2018-10-01 09:18:39.575000: Sending 1000 queries to 1 servers... [405/1000]
2018-10-01 09:18:40.078000: Sending 1000 queries to 1 servers... [405/1000]
2018-10-01 09:18:40.578000: Sending 1000 queries to 1 servers... [405/1000]
2018-10-01 09:18:41.078000: Sending 1000 queries to 1 servers... [405/1000]
2018-10-01 09:18:41.578000: Sending 1000 queries to 1 servers... [405/1000]
2018-10-01 09:18:42.078000: Sending 1000 queries to 1 servers... [405/1000]
2018-10-01 09:18:42.578000: Sending 1000 queries to 1 servers... [405/1000]
2018-10-01 09:18:43.078000: Sending 1000 queries to 1 servers... [464/1000]
2018-10-01 09:18:43.578000: Sending 1000 queries to 1 servers... [622/1000]
2018-10-01 09:18:44.058000: Sending 1000 queries to 1 servers... [779/1000]
2018-10-01 09:18:44.578000: Sending 1000 queries to 1 servers... [934/1000]
2018-10-01 09:18:45.092000: Sending 1000 queries to 1 servers... [1000/1000]
2018-10-01 09:18:45.093000: Saving report to c:\users\nave\appdata\local\temp\namebench_2018-10-
01_0918.html
2018-10-01 09:18:45.375000: Saving detailed results to c:\users\nave\appdata\local\temp
\namebench_2018-10-01_0918.csv
2018-10-01 09:18:45.375000: Opening c:\users\nave\appdata\local\temp\namebench_2018-10-01_0918.html
2018-10-01 09:18:45.375000: $ user HTTP handler: C:\Program Files (x86)\Google\Chrome\Application
\chrome.exe
2018-10-01 09:18:45.375000: command_args:
2018-10-01 09:18:45.375000: (u'C:\\Program Files (x86)\\Google\\Chrome\\Application\\chrome.exe',
'c:\\users\\nave\\appdata\\local\\temp\\namebench_2018-10-01_0918.html')
```

ANEXO 6 – RESULTADO DO TESTE *CACHE HIT* SERVIDOR BIND

Arquivo gerado pela ferramenta *Namebench* após o teste de *Cache Hit* no servidor BIND, contendo uma amostragem dos primeiros resultados.



	A	B	C	D	E	F	G	H	I
1	IP,Name,Test_Num,Record,Record_Type,Duration,TTL,Answer_Count,Response								
2									
3		.195,SYS-		.195,0,a.root-servers.net.,A,3.38962493002,16561,1,198.41.0.4,					
4									
5		.195,SYS-		.195,0,a.root-servers.net.,A,3.08048765317,16561,1,198.41.0.4,					
6									
7		.195,SYS-		.195,0,a.root-servers.net.,A,3.09945885426,16561,1,198.41.0.4,					
8									
9		.195,SYS-		.195,0,a.root-servers.net.,A,3.04378926417,16561,1,198.41.0.4,					
10									
11		.195,SYS-		.195,0,a.root-servers.net.,A,3.75909684643,16561,1,198.41.0.4,					
12									
13		.195,SYS-		.195,0,a.root-servers.net.,A,3.24252037072,16561,1,198.41.0.4,					
14									
15		.195,SYS-		.195,0,a.root-servers.net.,A,2.97101449275,16561,1,198.41.0.4,					
16									
17		.195,SYS-		.195,0,a.root-servers.net.,A,2.94053616969,16561,1,198.41.0.4,					
18									
19		.195,SYS-		.195,0,a.root-servers.net.,A,3.05125334329,16561,1,198.41.0.4,					
20									
21		.195,SYS-		.195,0,a.root-servers.net.,A,3.09697082789,16561,1,198.41.0.4,					
22									
23		.195,SYS-		.195,0,a.root-servers.net.,A,3.0500093301,16561,1,198.41.0.4,					
24									
25		.195,SYS-		.195,0,a.root-servers.net.,A,3.15637245755,16561,1,198.41.0.4,					
26									
27		.195,SYS-		.195,0,a.root-servers.net.,A,3.01020090813,16561,1,198.41.0.4,					
28									
29		.195,SYS-		.195,0,a.root-servers.net.,A,2.97754556198,16561,1,198.41.0.4,					
30									

195namebench_2018-10-01_0918

PRONTO

ANEXO 7 – LOG *CACHE MISS* SERVIDOR BIND

Arquivo de log gerado pela ferramenta *Namebench* após a execução do teste de *Cache Miss* no servidor BIND.

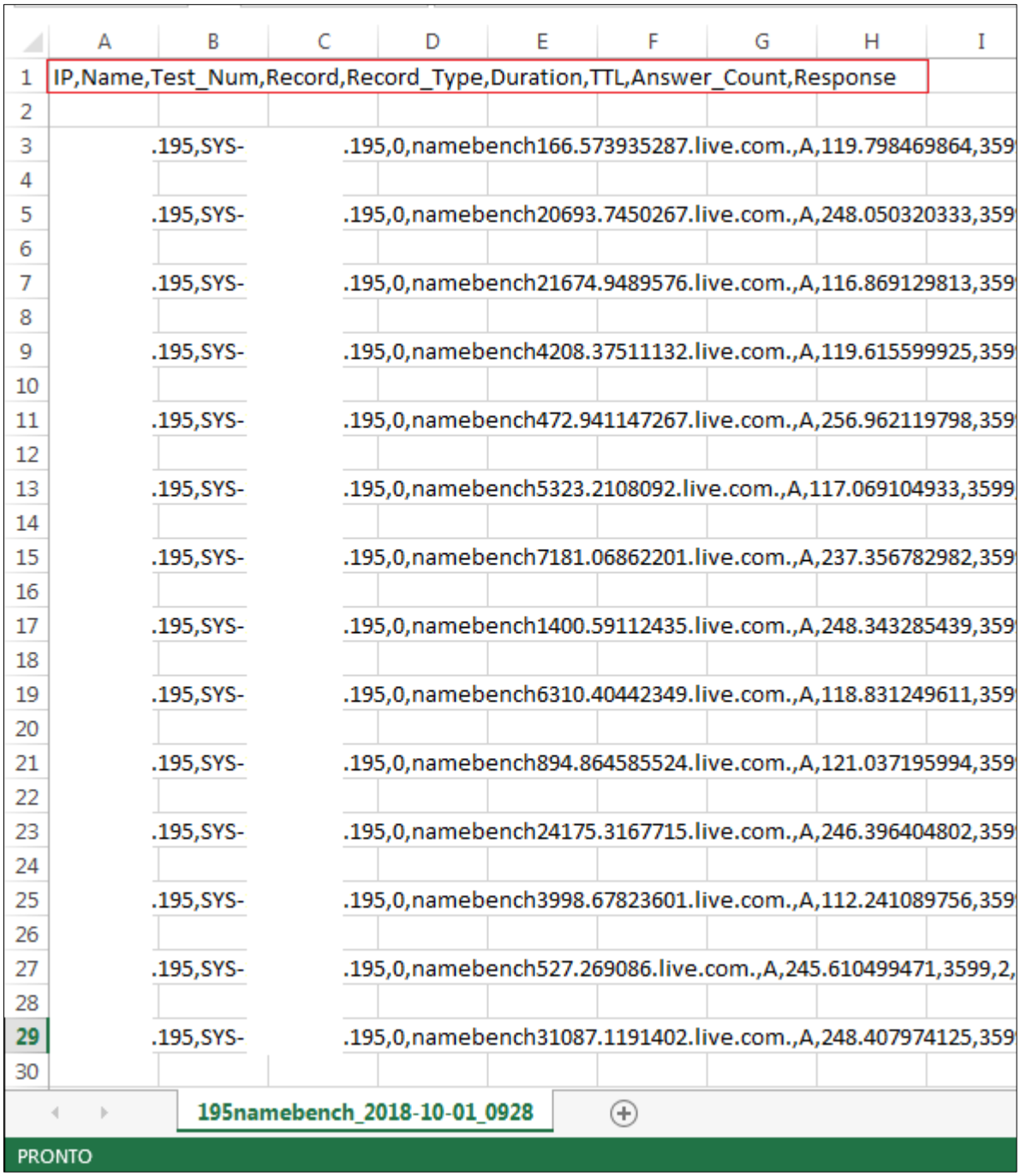
195namebench_2018-10-01_0924 - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

```
2018-10-01 09:24:53.783000: Reading Top 2,000 Websites (Alexa): C:\Users\Nave\Desktop\Nam
\data\alexa-top-2000-domains.txt (0.7MB)
2018-10-01 09:24:53.829000: Reading Cache Latency Test (100% hit): C:\Users\Nave\Desktop\
\1.3.1\namebench\data\cache-hit.txt (0.1MB)
2018-10-01 09:24:53.829000: Reading Cache Latency Test (100% miss): C:\Users\Nave\Desktop
\1.3.1\namebench\data\cache-miss.txt (0.1MB)
2018-10-01 09:24:53.829000: Reading Cache Latency Test (50% hit, 50% miss): C:\Users\Nave
\1.3.1\namebench\data\cache-mix.txt (0.1MB)
2018-10-01 09:24:53.845000: Reading Google Chrome: C:\Users\Nave\AppData\Local\Google\Chr
\History (13.7MB)
2018-10-01 09:24:53.892000: Reading Mozilla Firefox: C:\Users\Nave\AppData\Roaming\Mozill
\ravrwfzs.default-1530615491008\places.sqlite (5.0MB)
2018-10-01 09:24:53.923000: Running...
2018-10-01 09:24:53.923000: namebench 1.3.1 is ready!
2018-10-01 09:25:02.626000: Generating tests from Cache Latency Test (100% miss) (2500 re
automatic)
2018-10-01 09:25:02.626000: Running...
2018-10-01 09:25:02.626000: Started thread
2018-10-01 09:25:02.642000: Selecting 1000 out of 2500 sanitized records (chunk mode).
2018-10-01 09:25:02.657000: Checking query interception status...
2018-10-01 09:25:02.688000: Checking connection quality... [1/3]
2018-10-01 09:25:03.004000: Checking connection quality... [2/3]
2018-10-01 09:25:03.347000: Checking connection quality... [3/3]
2018-10-01 09:25:03.661000: Congestion level is 1.44X (check duration: 57.62ms)
2018-10-01 09:25:03.661000: Applied 1.44X timeout multiplier due to congestion: 0.7 ping,
2018-10-01 09:25:03.661000: Checking latest sanity reference
2018-10-01 09:25:03.849000: Sending 1000 queries to 1 servers... [0/1000]
2018-10-01 09:25:04.351000: Sending 1000 queries to 1 servers... [3/1000]
2018-10-01 09:25:04.853000: Sending 1000 queries to 1 servers... [6/1000]
2018-10-01 09:25:05.339000: Sending 1000 queries to 1 servers... [8/1000]
2018-10-01 09:25:05.872000: Sending 1000 queries to 1 servers... [11/1000]
2018-10-01 09:25:06.374000: Sending 1000 queries to 1 servers... [13/1000]
2018-10-01 09:25:06.876000: Sending 1000 queries to 1 servers... [16/1000]
2018-10-01 09:25:07.378000: Sending 1000 queries to 1 servers... [20/1000]
2018-10-01 09:25:07.863000: Sending 1000 queries to 1 servers... [23/1000]
2018-10-01 09:25:08.366000: Sending 1000 queries to 1 servers... [26/1000]
2018-10-01 09:25:08.868000: Sending 1000 queries to 1 servers... [28/1000]
2018-10-01 09:25:09.370000: Sending 1000 queries to 1 servers... [31/1000]
2018-10-01 09:25:09.869000: Sending 1000 queries to 1 servers... [34/1000]
2018-10-01 09:25:10.371000: Sending 1000 queries to 1 servers... [36/1000]
2018-10-01 09:25:10.857000: Sending 1000 queries to 1 servers... [38/1000]
2018-10-01 09:25:11.361000: Sending 1000 queries to 1 servers... [40/1000]
2018-10-01 09:25:11.861000: Sending 1000 queries to 1 servers... [44/1000]
2018-10-01 09:25:12.394000: Sending 1000 queries to 1 servers... [47/1000]
2018-10-01 09:25:12.896000: Sending 1000 queries to 1 servers... [50/1000]
2018-10-01 09:25:13.381000: Sending 1000 queries to 1 servers... [53/1000]
2018-10-01 09:25:13.883000: Sending 1000 queries to 1 servers... [57/1000]
2018-10-01 09:25:14.386000: Sending 1000 queries to 1 servers... [60/1000]
2018-10-01 09:25:14.888000: Sending 1000 queries to 1 servers... [62/1000]
2018-10-01 09:25:15.405000: Sending 1000 queries to 1 servers... [64/1000]
2018-10-01 09:25:15.907000: Sending 1000 queries to 1 servers... [67/1000]
2018-10-01 09:25:16.409000: Sending 1000 queries to 1 servers... [71/1000]
2018-10-01 09:25:16.911000: Sending 1000 queries to 1 servers... [73/1000]
2018-10-01 09:25:17.382000: Sending 1000 queries to 1 servers... [77/1000]
```

ANEXO 8 – RESULTADO DO TESTE *CACHE MISS* SERVIDOR BIND

Arquivo gerado pela ferramenta *Namebench* após o teste de *Cache Miss* no servidor BIND, contendo uma amostragem dos primeiros resultados.



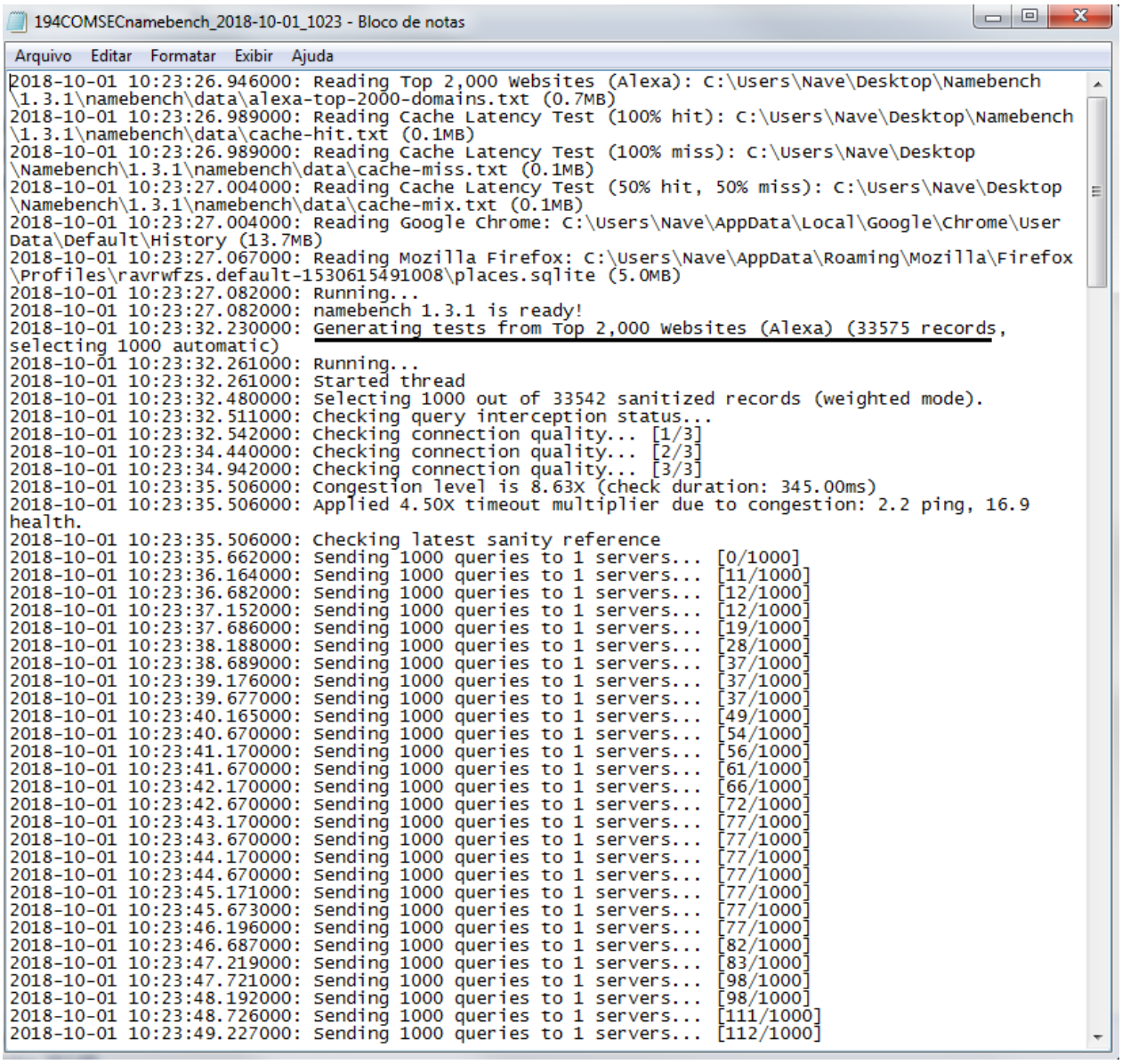
	A	B	C	D	E	F	G	H	I
1	IP,Name,Test_Num,Record,Record_Type,Duration,TTL,Answer_Count,Response								
2									
3	.195,SYS-		.195,0,namebench166.573935287.live.com.,A,119.798469864,359						
4									
5	.195,SYS-		.195,0,namebench20693.7450267.live.com.,A,248.050320333,359						
6									
7	.195,SYS-		.195,0,namebench21674.9489576.live.com.,A,116.869129813,359						
8									
9	.195,SYS-		.195,0,namebench4208.37511132.live.com.,A,119.615599925,359						
10									
11	.195,SYS-		.195,0,namebench472.941147267.live.com.,A,256.962119798,359						
12									
13	.195,SYS-		.195,0,namebench5323.2108092.live.com.,A,117.069104933,3599						
14									
15	.195,SYS-		.195,0,namebench7181.06862201.live.com.,A,237.356782982,359						
16									
17	.195,SYS-		.195,0,namebench1400.59112435.live.com.,A,248.343285439,359						
18									
19	.195,SYS-		.195,0,namebench6310.40442349.live.com.,A,118.831249611,359						
20									
21	.195,SYS-		.195,0,namebench894.864585524.live.com.,A,121.037195994,359						
22									
23	.195,SYS-		.195,0,namebench24175.3167715.live.com.,A,246.396404802,359						
24									
25	.195,SYS-		.195,0,namebench3998.67823601.live.com.,A,112.241089756,359						
26									
27	.195,SYS-		.195,0,namebench527.269086.live.com.,A,245.610499471,3599,2,						
28									
29	.195,SYS-		.195,0,namebench31087.1191402.live.com.,A,248.407974125,359						
30									

195namebench_2018-10-01_0928

PRONTO

ANEXO 9 – LOG *TOP RANK ALEXA* SERVIDOR *UNBOUND*

Arquivo de log gerado pela ferramenta *Namebench* após a execução do teste *Top Alexa Rank* no servidor *Unbound*.



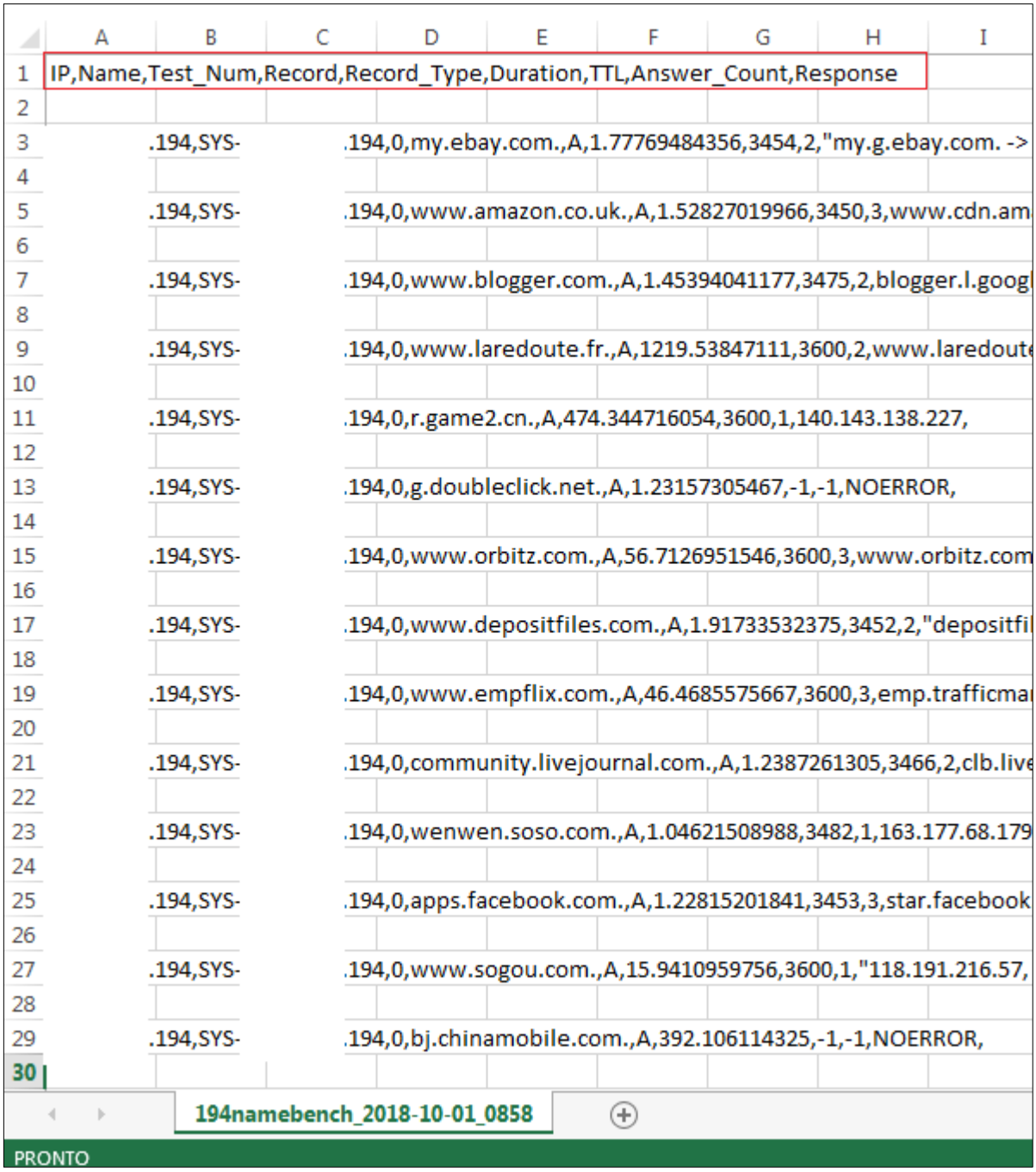
194COMSECnamebench_2018-10-01_1023 - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

```
2018-10-01 10:23:26.946000: Reading Top 2,000 Websites (Alexa): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\alexa-top-2000-domains.txt (0.7MB)
2018-10-01 10:23:26.989000: Reading Cache Latency Test (100% hit): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\cache-hit.txt (0.1MB)
2018-10-01 10:23:26.989000: Reading Cache Latency Test (100% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-miss.txt (0.1MB)
2018-10-01 10:23:27.004000: Reading Cache Latency Test (50% hit, 50% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-mix.txt (0.1MB)
2018-10-01 10:23:27.004000: Reading Google Chrome: C:\Users\Nave\AppData\Local\Google\Chrome\User
Data\Default\History (13.7MB)
2018-10-01 10:23:27.067000: Reading Mozilla Firefox: C:\Users\Nave\AppData\Roaming\Mozilla\Firefox
\Profiles\ravrwfzs.default-1530615491008\places.sqlite (5.0MB)
2018-10-01 10:23:27.082000: Running...
2018-10-01 10:23:27.082000: namebench 1.3.1 is ready!
2018-10-01 10:23:32.230000: Generating tests from Top 2,000 Websites (Alexa) (33575 records,
selecting 1000 automatic)
2018-10-01 10:23:32.261000: Running...
2018-10-01 10:23:32.261000: Started thread
2018-10-01 10:23:32.480000: Selecting 1000 out of 33542 sanitized records (weighted mode).
2018-10-01 10:23:32.511000: Checking query interception status...
2018-10-01 10:23:32.542000: Checking connection quality... [1/3]
2018-10-01 10:23:34.440000: Checking connection quality... [2/3]
2018-10-01 10:23:34.942000: Checking connection quality... [3/3]
2018-10-01 10:23:35.506000: Congestion level is 8.63X (check duration: 345.00ms)
2018-10-01 10:23:35.506000: Applied 4.50X timeout multiplier due to congestion: 2.2 ping, 16.9
health.
2018-10-01 10:23:35.506000: Checking latest sanity reference
2018-10-01 10:23:35.662000: Sending 1000 queries to 1 servers... [0/1000]
2018-10-01 10:23:36.164000: Sending 1000 queries to 1 servers... [11/1000]
2018-10-01 10:23:36.682000: Sending 1000 queries to 1 servers... [12/1000]
2018-10-01 10:23:37.152000: Sending 1000 queries to 1 servers... [12/1000]
2018-10-01 10:23:37.686000: Sending 1000 queries to 1 servers... [19/1000]
2018-10-01 10:23:38.188000: Sending 1000 queries to 1 servers... [28/1000]
2018-10-01 10:23:38.689000: Sending 1000 queries to 1 servers... [37/1000]
2018-10-01 10:23:39.176000: Sending 1000 queries to 1 servers... [37/1000]
2018-10-01 10:23:39.677000: Sending 1000 queries to 1 servers... [37/1000]
2018-10-01 10:23:40.165000: Sending 1000 queries to 1 servers... [49/1000]
2018-10-01 10:23:40.670000: Sending 1000 queries to 1 servers... [54/1000]
2018-10-01 10:23:41.170000: Sending 1000 queries to 1 servers... [56/1000]
2018-10-01 10:23:41.670000: Sending 1000 queries to 1 servers... [61/1000]
2018-10-01 10:23:42.170000: Sending 1000 queries to 1 servers... [66/1000]
2018-10-01 10:23:42.670000: Sending 1000 queries to 1 servers... [72/1000]
2018-10-01 10:23:43.170000: Sending 1000 queries to 1 servers... [77/1000]
2018-10-01 10:23:43.670000: Sending 1000 queries to 1 servers... [77/1000]
2018-10-01 10:23:44.170000: Sending 1000 queries to 1 servers... [77/1000]
2018-10-01 10:23:44.670000: Sending 1000 queries to 1 servers... [77/1000]
2018-10-01 10:23:45.171000: Sending 1000 queries to 1 servers... [77/1000]
2018-10-01 10:23:45.673000: Sending 1000 queries to 1 servers... [77/1000]
2018-10-01 10:23:46.196000: Sending 1000 queries to 1 servers... [77/1000]
2018-10-01 10:23:46.687000: Sending 1000 queries to 1 servers... [82/1000]
2018-10-01 10:23:47.219000: Sending 1000 queries to 1 servers... [83/1000]
2018-10-01 10:23:47.721000: Sending 1000 queries to 1 servers... [98/1000]
2018-10-01 10:23:48.192000: Sending 1000 queries to 1 servers... [98/1000]
2018-10-01 10:23:48.726000: Sending 1000 queries to 1 servers... [111/1000]
2018-10-01 10:23:49.227000: Sending 1000 queries to 1 servers... [112/1000]
```

ANEXO 10 – RESULTADO TOP RANK ALEXA SERVIDOR *UNBOUND*

Arquivo gerado pela ferramenta *Namebench* após o teste *Top Alexa Rank* no servidor *Unbound*, contendo uma amostragem dos primeiros resultados.



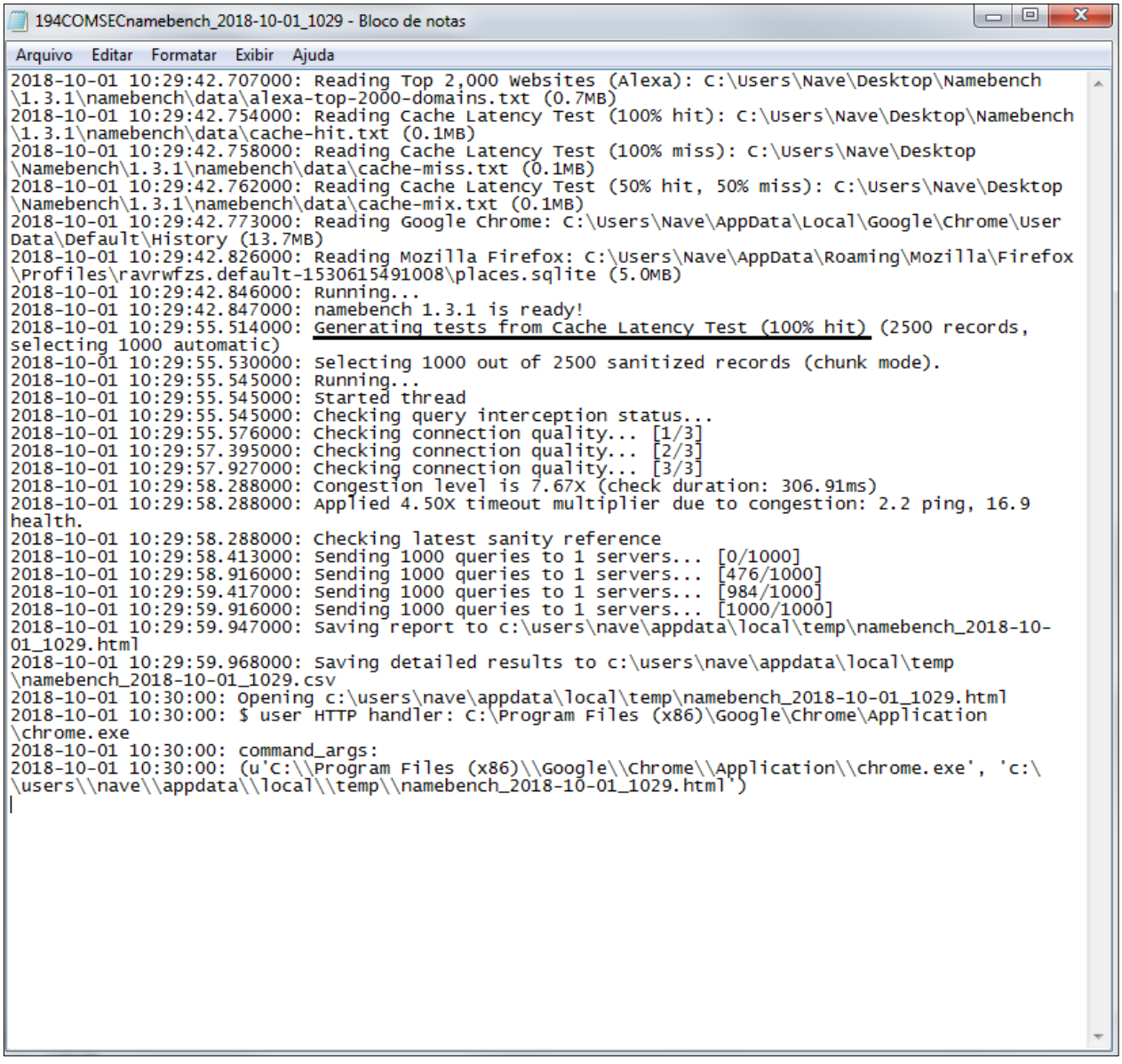
	A	B	C	D	E	F	G	H	I
1	IP,Name,Test_Num,Record,Record_Type,Duration,TTL,Answer_Count,Response								
2									
3		.194,SYS-		.194,0,my.ebay.com.,A,1.77769484356,3454,2,"my.g.ebay.com. ->					
4									
5		.194,SYS-		.194,0,www.amazon.co.uk.,A,1.52827019966,3450,3,www.cdn.am					
6									
7		.194,SYS-		.194,0,www.blogger.com.,A,1.45394041177,3475,2,blogger.l.goog					
8									
9		.194,SYS-		.194,0,www.laredoute.fr.,A,1219.53847111,3600,2,www.laredoute					
10									
11		.194,SYS-		.194,0,r.game2.cn.,A,474.344716054,3600,1,140.143.138.227,					
12									
13		.194,SYS-		.194,0,g.doubleclick.net.,A,1.23157305467,-1,-1,NOERROR,					
14									
15		.194,SYS-		.194,0,www.orbitz.com.,A,56.7126951546,3600,3,www.orbitz.com					
16									
17		.194,SYS-		.194,0,www.depositfiles.com.,A,1.91733532375,3452,2,"depositfi					
18									
19		.194,SYS-		.194,0,www.empflix.com.,A,46.4685575667,3600,3,emp.trafficma					
20									
21		.194,SYS-		.194,0,community.livejournal.com.,A,1.2387261305,3466,2,clb.live					
22									
23		.194,SYS-		.194,0,wenwen.soso.com.,A,1.04621508988,3482,1,163.177.68.179					
24									
25		.194,SYS-		.194,0,apps.facebook.com.,A,1.22815201841,3453,3,star.facebook					
26									
27		.194,SYS-		.194,0,www.sogou.com.,A,15.9410959756,3600,1,"118.191.216.57,					
28									
29		.194,SYS-		.194,0,bj.chinamobile.com.,A,392.106114325,-1,-1,NOERROR,					
30									

194namebench_2018-10-01_0858

PRONTO

ANEXO 11 – LOG DE EXECUÇÃO *CACHE HIT* SERVIDOR *UNBOUND*

Arquivo de log gerado pela ferramenta *Namebench* após a execução do teste de *Cache Hit* no servidor *Unbound*.



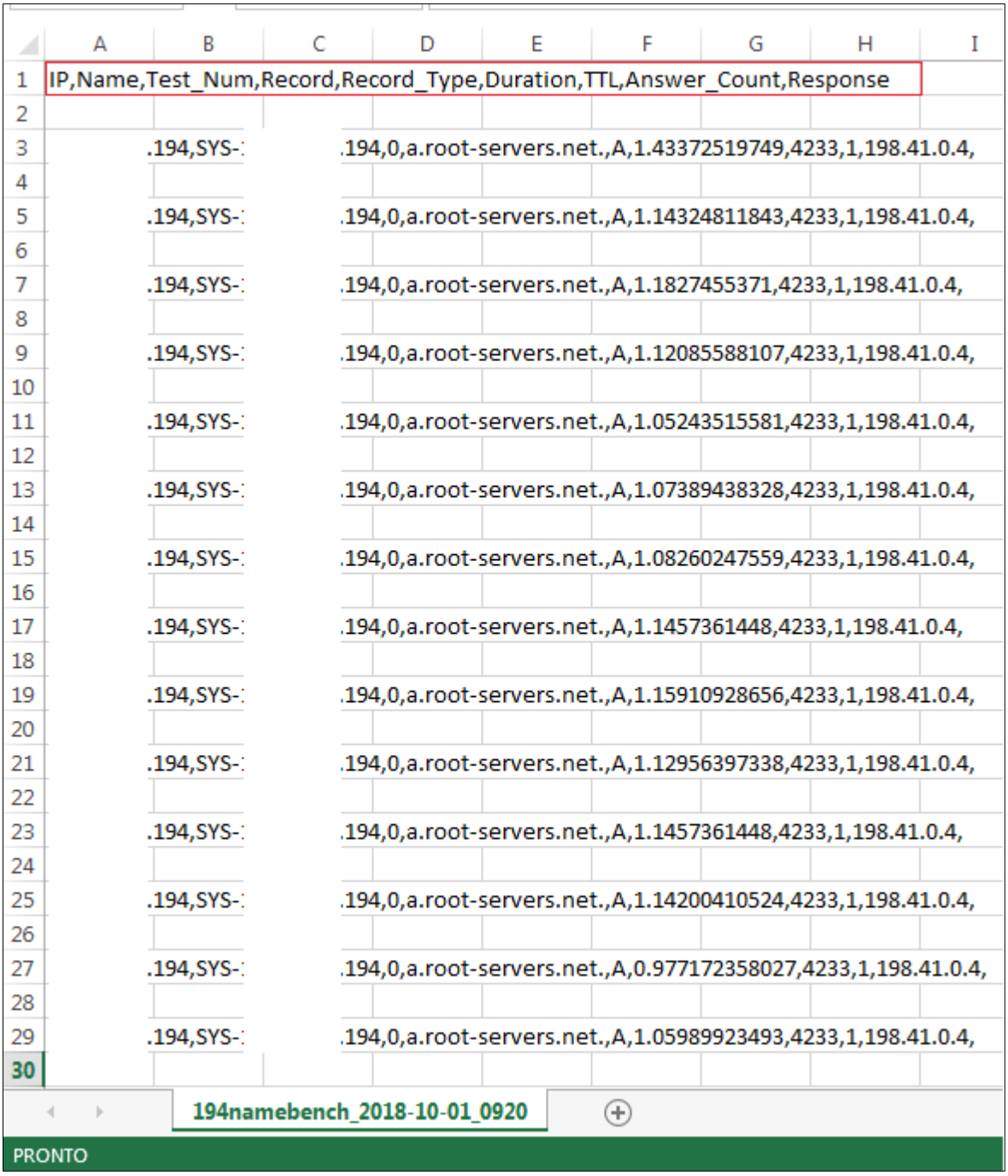
194COMSECnamebench_2018-10-01_1029 - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

```
2018-10-01 10:29:42.707000: Reading Top 2,000 Websites (Alexa): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\alexa-top-2000-domains.txt (0.7MB)
2018-10-01 10:29:42.754000: Reading Cache Latency Test (100% hit): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\cache-hit.txt (0.1MB)
2018-10-01 10:29:42.758000: Reading Cache Latency Test (100% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-miss.txt (0.1MB)
2018-10-01 10:29:42.762000: Reading Cache Latency Test (50% hit, 50% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-mix.txt (0.1MB)
2018-10-01 10:29:42.773000: Reading Google Chrome: C:\Users\Nave\AppData\Local\Google\Chrome\User
Data\Default\History (13.7MB)
2018-10-01 10:29:42.826000: Reading Mozilla Firefox: C:\Users\Nave\AppData\Roaming\Mozilla\Firefox
\Profiles\ravrwfzs.default-1530615491008\places.sqlite (5.0MB)
2018-10-01 10:29:42.846000: Running...
2018-10-01 10:29:42.847000: namebench 1.3.1 is ready!
2018-10-01 10:29:55.514000: Generating tests from Cache Latency Test (100% hit) (2500 records,
selecting 1000 automatic)
2018-10-01 10:29:55.530000: Selecting 1000 out of 2500 sanitized records (chunk mode).
2018-10-01 10:29:55.545000: Running...
2018-10-01 10:29:55.545000: Started thread
2018-10-01 10:29:55.545000: Checking query interception status...
2018-10-01 10:29:55.576000: Checking connection quality... [1/3]
2018-10-01 10:29:57.395000: Checking connection quality... [2/3]
2018-10-01 10:29:57.927000: Checking connection quality... [3/3]
2018-10-01 10:29:58.288000: Congestion level is 7.67X (check duration: 306.91ms)
2018-10-01 10:29:58.288000: Applied 4.50X timeout multiplier due to congestion: 2.2 ping, 16.9
health.
2018-10-01 10:29:58.288000: Checking latest sanity reference
2018-10-01 10:29:58.413000: Sending 1000 queries to 1 servers... [0/1000]
2018-10-01 10:29:58.916000: Sending 1000 queries to 1 servers... [476/1000]
2018-10-01 10:29:59.417000: Sending 1000 queries to 1 servers... [984/1000]
2018-10-01 10:29:59.916000: Sending 1000 queries to 1 servers... [1000/1000]
2018-10-01 10:29:59.947000: Saving report to c:\users\nave\appdata\local\temp\namebench_2018-10-
01_1029.html
2018-10-01 10:29:59.968000: Saving detailed results to c:\users\nave\appdata\local\temp
\namebench_2018-10-01_1029.csv
2018-10-01 10:30:00: Opening c:\users\nave\appdata\local\temp\namebench_2018-10-01_1029.html
2018-10-01 10:30:00: $ user HTTP handler: C:\Program Files (x86)\Google\Chrome\Application
\chrome.exe
2018-10-01 10:30:00: command_args:
2018-10-01 10:30:00: (u'C:\\Program Files (x86)\\Google\\Chrome\\Application\\chrome.exe', 'c:\
\users\\nave\\appdata\\local\\temp\\namebench_2018-10-01_1029.html')
```

ANEXO 12 – RESULTADO TESTE *CACHE HIT* SERVIDOR *UNBOUND*

Arquivo gerado pela ferramenta *Namebench* após o teste de *Cache Hit* no servidor *Unbound*, contendo uma amostragem dos primeiros resultados.



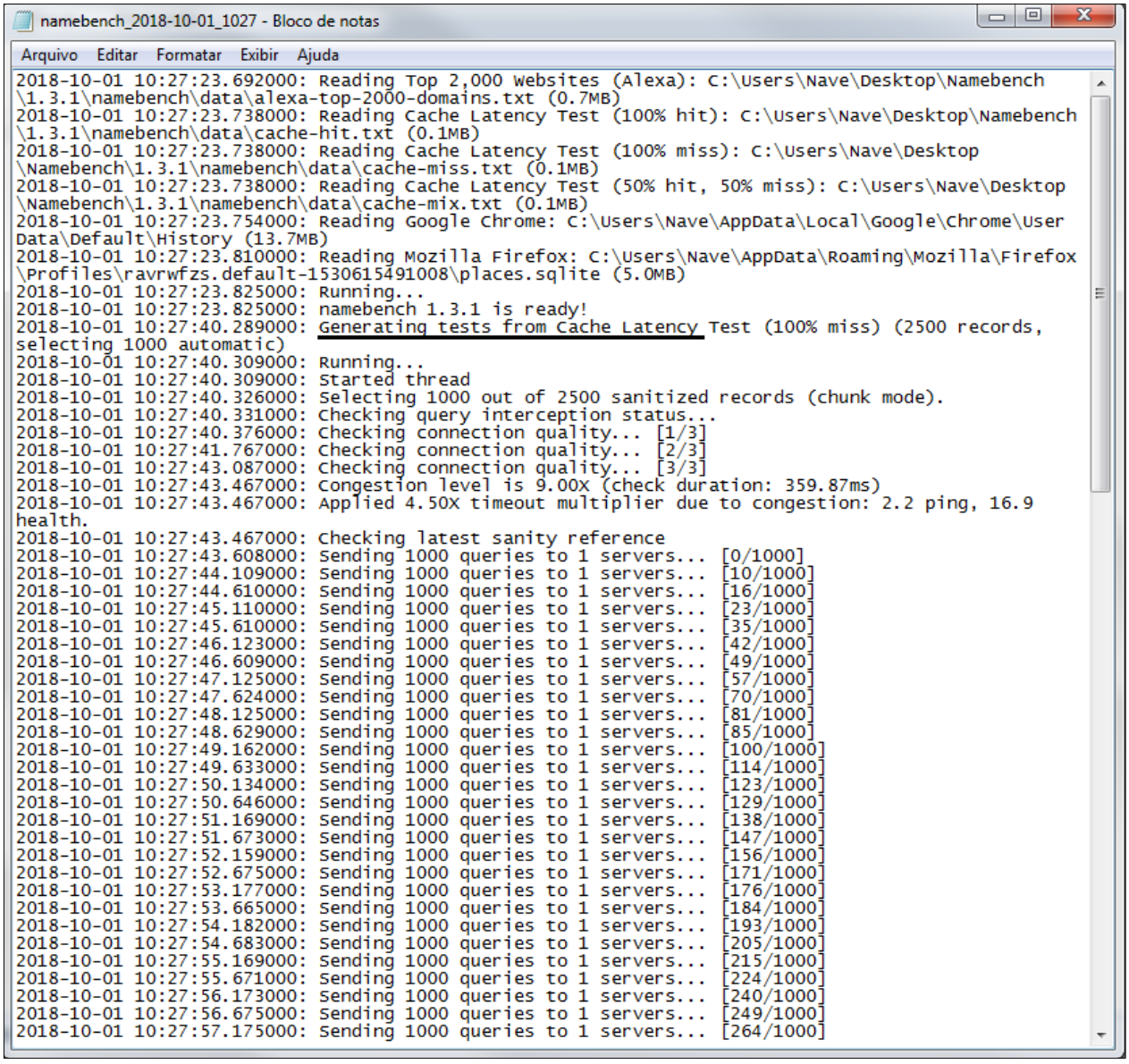
	A	B	C	D	E	F	G	H	I
1	IP,Name,Test_Num,Record,Record_Type,Duration,TTL,Answer_Count,Response								
2									
3		.194,SYS-:		.194,0,a.root-servers.net.,A,1.43372519749,4233,1,198.41.0.4,					
4									
5		.194,SYS-:		.194,0,a.root-servers.net.,A,1.14324811843,4233,1,198.41.0.4,					
6									
7		.194,SYS-:		.194,0,a.root-servers.net.,A,1.1827455371,4233,1,198.41.0.4,					
8									
9		.194,SYS-:		.194,0,a.root-servers.net.,A,1.12085588107,4233,1,198.41.0.4,					
10									
11		.194,SYS-:		.194,0,a.root-servers.net.,A,1.05243515581,4233,1,198.41.0.4,					
12									
13		.194,SYS-:		.194,0,a.root-servers.net.,A,1.07389438328,4233,1,198.41.0.4,					
14									
15		.194,SYS-:		.194,0,a.root-servers.net.,A,1.08260247559,4233,1,198.41.0.4,					
16									
17		.194,SYS-:		.194,0,a.root-servers.net.,A,1.1457361448,4233,1,198.41.0.4,					
18									
19		.194,SYS-:		.194,0,a.root-servers.net.,A,1.15910928656,4233,1,198.41.0.4,					
20									
21		.194,SYS-:		.194,0,a.root-servers.net.,A,1.12956397338,4233,1,198.41.0.4,					
22									
23		.194,SYS-:		.194,0,a.root-servers.net.,A,1.1457361448,4233,1,198.41.0.4,					
24									
25		.194,SYS-:		.194,0,a.root-servers.net.,A,1.14200410524,4233,1,198.41.0.4,					
26									
27		.194,SYS-:		.194,0,a.root-servers.net.,A,0.977172358027,4233,1,198.41.0.4,					
28									
29		.194,SYS-:		.194,0,a.root-servers.net.,A,1.05989923493,4233,1,198.41.0.4,					
30									

194namebench_2018-10-01_0920

PRONTO

ANEXO 13 – LOG DE EXECUÇÃO *CACHE MISS* SERVIDOR *UNBOUND*

Arquivo de log gerado pela ferramenta *Namebench* após a execução do teste *de Cache Miss* no servidor *Unbound*.



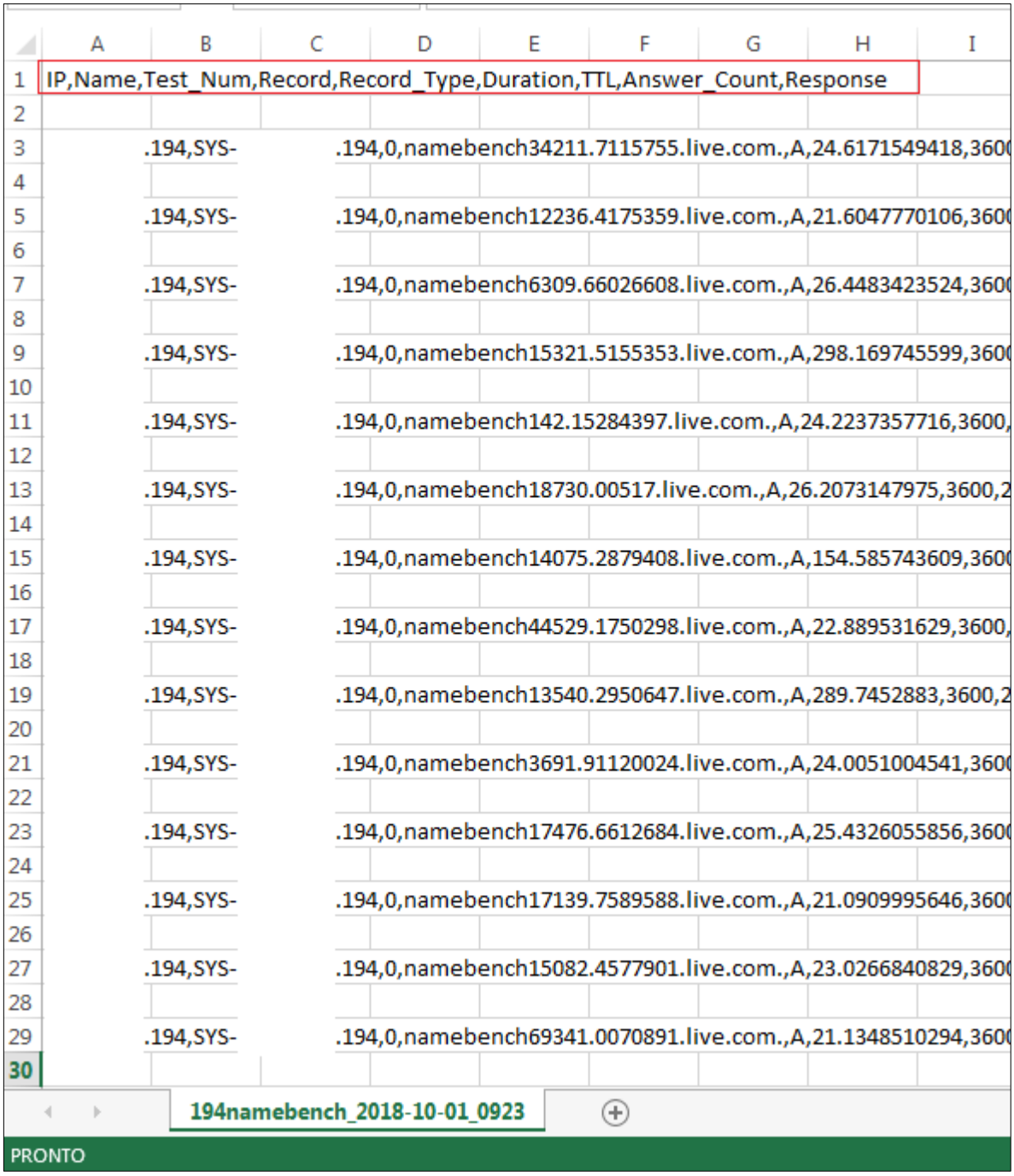
namebench_2018-10-01_1027 - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

```
2018-10-01 10:27:23.692000: Reading Top 2,000 websites (Alexa): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\alexa-top-2000-domains.txt (0.7MB)
2018-10-01 10:27:23.738000: Reading Cache Latency Test (100% hit): C:\Users\Nave\Desktop\Namebench
\1.3.1\namebench\data\cache-hit.txt (0.1MB)
2018-10-01 10:27:23.738000: Reading Cache Latency Test (100% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-miss.txt (0.1MB)
2018-10-01 10:27:23.738000: Reading Cache Latency Test (50% hit, 50% miss): C:\Users\Nave\Desktop
\Namebench\1.3.1\namebench\data\cache-mix.txt (0.1MB)
2018-10-01 10:27:23.754000: Reading Google Chrome: C:\Users\Nave\AppData\Local\Google\Chrome\User
Data\Default\History (13.7MB)
2018-10-01 10:27:23.810000: Reading Mozilla Firefox: C:\Users\Nave\AppData\Roaming\Mozilla\Firefox
\Profiles\ravrwfzs.default-1530615491008\places.sqlite (5.0MB)
2018-10-01 10:27:23.825000: Running...
2018-10-01 10:27:23.825000: namebench 1.3.1 is ready!
2018-10-01 10:27:40.289000: Generating tests from Cache Latency Test (100% miss) (2500 records,
selecting 1000 automatic)
2018-10-01 10:27:40.309000: Running...
2018-10-01 10:27:40.309000: Started thread
2018-10-01 10:27:40.326000: Selecting 1000 out of 2500 sanitized records (chunk mode).
2018-10-01 10:27:40.331000: Checking query interception status...
2018-10-01 10:27:40.376000: Checking connection quality... [1/3]
2018-10-01 10:27:41.767000: Checking connection quality... [2/3]
2018-10-01 10:27:43.087000: Checking connection quality... [3/3]
2018-10-01 10:27:43.467000: Congestion level is 9.00X (check duration: 359.87ms)
2018-10-01 10:27:43.467000: Applied 4.50X timeout multiplier due to congestion: 2.2 ping, 16.9
health.
2018-10-01 10:27:43.467000: Checking latest sanity reference
2018-10-01 10:27:43.608000: Sending 1000 queries to 1 servers... [0/1000]
2018-10-01 10:27:44.109000: Sending 1000 queries to 1 servers... [10/1000]
2018-10-01 10:27:44.610000: Sending 1000 queries to 1 servers... [16/1000]
2018-10-01 10:27:45.110000: Sending 1000 queries to 1 servers... [23/1000]
2018-10-01 10:27:45.610000: Sending 1000 queries to 1 servers... [35/1000]
2018-10-01 10:27:46.123000: Sending 1000 queries to 1 servers... [42/1000]
2018-10-01 10:27:46.609000: Sending 1000 queries to 1 servers... [49/1000]
2018-10-01 10:27:47.125000: Sending 1000 queries to 1 servers... [57/1000]
2018-10-01 10:27:47.624000: Sending 1000 queries to 1 servers... [70/1000]
2018-10-01 10:27:48.125000: Sending 1000 queries to 1 servers... [81/1000]
2018-10-01 10:27:48.629000: Sending 1000 queries to 1 servers... [85/1000]
2018-10-01 10:27:49.162000: Sending 1000 queries to 1 servers... [100/1000]
2018-10-01 10:27:49.633000: Sending 1000 queries to 1 servers... [114/1000]
2018-10-01 10:27:50.134000: Sending 1000 queries to 1 servers... [123/1000]
2018-10-01 10:27:50.646000: Sending 1000 queries to 1 servers... [129/1000]
2018-10-01 10:27:51.169000: Sending 1000 queries to 1 servers... [138/1000]
2018-10-01 10:27:51.673000: Sending 1000 queries to 1 servers... [147/1000]
2018-10-01 10:27:52.159000: Sending 1000 queries to 1 servers... [156/1000]
2018-10-01 10:27:52.675000: Sending 1000 queries to 1 servers... [171/1000]
2018-10-01 10:27:53.177000: Sending 1000 queries to 1 servers... [176/1000]
2018-10-01 10:27:53.665000: Sending 1000 queries to 1 servers... [184/1000]
2018-10-01 10:27:54.182000: Sending 1000 queries to 1 servers... [193/1000]
2018-10-01 10:27:54.683000: Sending 1000 queries to 1 servers... [205/1000]
2018-10-01 10:27:55.169000: Sending 1000 queries to 1 servers... [215/1000]
2018-10-01 10:27:55.671000: Sending 1000 queries to 1 servers... [224/1000]
2018-10-01 10:27:56.173000: Sending 1000 queries to 1 servers... [240/1000]
2018-10-01 10:27:56.675000: Sending 1000 queries to 1 servers... [249/1000]
2018-10-01 10:27:57.175000: Sending 1000 queries to 1 servers... [264/1000]
```

ANEXO 14 – RESULTADO TESTE *CACHE MISS* SERVIDOR *UNBOUND*

Arquivo gerado pela ferramenta *Namebench* após o teste de *Cache Miss* no servidor *Unbound*, contendo uma amostragem dos primeiros resultados.



	A	B	C	D	E	F	G	H	I
1	IP,Name,Test_Num,Record,Record_Type,Duration,TTL,Answer_Count,Response								
2									
3		.194,SYS-		.194,0,namebench34211.7115755.live.com.,A,24.6171549418,360(					
4									
5		.194,SYS-		.194,0,namebench12236.4175359.live.com.,A,21.6047770106,360(					
6									
7		.194,SYS-		.194,0,namebench6309.66026608.live.com.,A,26.4483423524,360(					
8									
9		.194,SYS-		.194,0,namebench15321.5155353.live.com.,A,298.169745599,360(					
10									
11		.194,SYS-		.194,0,namebench142.15284397.live.com.,A,24.2237357716,3600,					
12									
13		.194,SYS-		.194,0,namebench18730.00517.live.com.,A,26.2073147975,3600,2					
14									
15		.194,SYS-		.194,0,namebench14075.2879408.live.com.,A,154.585743609,360(					
16									
17		.194,SYS-		.194,0,namebench44529.1750298.live.com.,A,22.889531629,3600,					
18									
19		.194,SYS-		.194,0,namebench13540.2950647.live.com.,A,289.7452883,3600,2					
20									
21		.194,SYS-		.194,0,namebench3691.91120024.live.com.,A,24.0051004541,360(					
22									
23		.194,SYS-		.194,0,namebench17476.6612684.live.com.,A,25.4326055856,360(					
24									
25		.194,SYS-		.194,0,namebench17139.7589588.live.com.,A,21.0909995646,360(					
26									
27		.194,SYS-		.194,0,namebench15082.4577901.live.com.,A,23.0266840829,360(					
28									
29		.194,SYS-		.194,0,namebench69341.0070891.live.com.,A,21.1348510294,360(					
30									

194namebench_2018-10-01_0923

PRONTO