Serviço de Resolução de Nomes usando DHTs para uso na Internet de Nova Geração

Matheus V. Costa, Rodolfo S. Villaça

[1]Universidade Federal do Espírito Santo (UFES)
Centro Universitário Norte do Espírito Santo (CEUNES)
Departamento de Engenharias e Computação (DECOM)
Rodovia BR 101 N, km 60, Bairro Litorâneo, CEP 29932-540, São Mateus – ES.

matvieira@gmail.com,rodolfovillaca@ceunes.ufes.br

Resumo. *A evolução da Internet trouxe alguns requisitos, tais como mobilidade, suporte a redes baseadas em conteúdo, suporte a nomes planos e personalizados e suporte a aplicações legadas. O DNS tem um papel fundamental neste cenário, uma vez que as aplicações necessitam traduzir nomes humanamente compreensíveis em endereços IP. No entanto, é amplamente aceito que o DNS tem uma boa escalabilidade, mas não suporta adequadamente estes requisitos básicos de nova geração. Devido à possibilidade de ser utilizada para fornecer serviços distribuídos de recuperação de informações em larga proporção, as DHTs surgem como uma alternativa para fornecer serviços de resolução de nomes na Internet. Este trabalho propõe um modelo para a resolução de nomes utilizando DHTs, e realiza testes para comprovar a viabilidade do uso de DHTs para implementação de um serviço de resolução de nomes na Internet.*

1. Introdução

Um problema ainda em discussão e sem apresentar uma solução consensual é a definição de um serviço de resolução de nomes que atenda aos requisitos de uma Internet de próxima geração [Ahlgren et al. 2007]. A solução atualmente adotada na Internet, o DNS (Domain Name System), não atende adequadamente estes requisitos [Pappas et al. 2006].

O uso de DHTs (Distributed Hash Tables) como solução para um serviço de resolução de nomes é uma opção a ser pesquisada [Pappas et al. 2006, Ramasubramanian and Sirer 2004]. O objetivo deste trabalho é a avaliação de diferentes modelos e implementações de DHT, comparando-os, e a proposta de um modelo para a resolução de nomes utilizando DHTs que possa atender os requisitos de uma Internet de Nova Geração.

Efetuaremos testes comparativos dentre as diferentes implementações encontradas com o objetivo de avaliar a melhor escolha para atuar como um serviço de resolução de nomes. Em seguida, proporemos um modelo para adaptação das DHTs ao serviço de nomes atualmente existente na Internet, o DNS, através da implementação do DNSHandler. O DNSHandler é um *proxy* capaz de traduzir o protocolo do DNS, em perguntas e respostas de acordo com as implementações de DHT testadas. Através do DNSHandler será possível comprovar a viabilidade do uso de DHTs em serviços de resolução de nomes na Internet.

Como resultado deste trabalho teremos um serviço de resolução de nomes baseado em DHTs pronto para a utilização. Após um processo de testes e adequação do DNSHandler, o serviço foi modelado e implementado possibilitando a substituição do DNS nas aplicações com algumas restrições.

A seguir, a Seção 2 traz uma pequena introdução sobre os sistemas de DHTs, principalmente Chord [Stoica et al. 2003] e Pastry [Rowstron and Druschel 2001] , que foram os modelos estudados neste trabalho. A Seção 3 descreve a metodologia empregada no desenvolvimento deste trabalho, incluindo a descrição do DNSHandler. Na Seção 4 são apresentados os testes realizados com o serviço implementado e alguns resultados. Por fim, a Seção 5 apresenta as conclusões do trabalho e perspectivas de trabalhos futuros.

2. Distributed Hash Tables (DHTs)

DHT é uma rede de Tabelas *Hash* Distribuídas, portanto, descentralizada. Uma rede DHT possibilita a pessoas de lugares diferentes compartilharem informações e fazerem pesquisas sobre as tais informações através de chaves geradas por funções *hash* aplicadas sobre estas informações. Tal busca é feita de forma similar a uma tabela *hash*, onde chaves e valores são armazenados na rede e qualquer nó é habilitado a recuperar um valor associado a uma determinada chave.

Existem alguns tipos diferentes de implementações de DHT [Lua et al. 2005], dentre as diferentes implementações, o que diferencia umas das outras é exatamente o mapeamento de chaves para os respectivos nós da rede. Duas implementações estudadas foram o Chord [Lua et al. 2005, Stoica et al. 2003] e o Pastry [Lua et al. 2005, Rowstron and Druschel 2001] por serem as mais populares e com uma grande variedade de trabalhos publicados na área [Ramasubramanian and Sirer 2004, Cox et al. 2002].

2.1. Chord

O Chord, desenvolvido no MIT, é simples em sua funcionalidade. Associa para cada nó um ID único, podendo ser o *hash* de seu endereço IP, por exemplo, quando caracterizamos cada nó a uma máquina na rede, e uma tabela de *fingers* com m entradas, sendo m o número de bits do espaço ID. Tal tabela contém informações sobre outros nós, possibilitando assim um menor tempo de resposta no processo de busca.

Em sua estrutura, os nós são distribuídos em um anel, de modulo 2^m. A chave k é atribuída ao nó k ou ao seu sucessor no sentido horário do anel, caracterizado por *sucessor(k)*.

A tabela de *fingers* é utilizada para que, em uma determinada busca, o caminho ao encontro do nó destino seja mais rápido. Cada entrada i na tabela corresponde ao primeiro nó com ID $\geq$ (N + (2^i mod 2^m). Porém, não há informações suficientes para encontrar o sucessor de qualquer chave, recorrendo assim a consultas sucessivas a nós que precedem a chave até que se possa encontrar o sucessor procurado.

2.2. Pastry

O Pastry, desenvolvido pela Microsoft Research e a Rice University, possui um esquema mais robusto baseado em uma rede sobreposta (*Overlay*) de nós distribuídos. Cada nó na DHT possui um NodeID de 128 bits que define a sua posição em um espaço circular de IDs que variam entre 0 a 2^{128} - 1.

O roteamento das mensagens do Pastry é feito de forma que cada nó se encarrega de enviar a mensagem para o nó cujo NodeID é numericamente mais próximo da chave dada. Para tornar isso possível, cada nó possui uma tabela que define o seu estado: a tabela de roteamento, vizinhos e folhas.

O espaço referente aos M vizinhos do nó contém os NodeIDs dos nós mais próximos fisicamente do nó local, não é utilizado no processo de roteamento, mas é útil na manutenção da DHT. De tempos em tempos os nós mandam mensagens aos seus vizinhos para verificar se ainda estão 'vivos' na rede. O espaço para os L nós folhas contém os NodeIDs dos nós numericamente mais próximos do NodeID do nó local, que por sua vez são utilizados no processo de roteamento e são diferenciados na tabela entre menores e maiores, ou seja, que precedem e sucedem numericamente o NodeID de referência.

No Pastry, cada informação é armazenada em nós cujos IDs são numericamente mais próximos que a sua chave gerada pela função *hash*.

Uma busca sobre determinada chave através de um determinado nó é roteada até o nó com NodeID mais próximo da chave. Caso a chave possa ser mapeada pela tabela de roteamento, estará 2^b mais próxima do destino a cada interação, podendo alcançar o destino em $O(log_{2^b}\ N)$ passos, onde b é a base do sistema de numeração do anel.

Por outro lado, caso a chave não possa ser mapeada diretamente, será encaminhada para o nó cujo NodeID é numericamente mais próximo.

3. Metodologia do Trabalho

Algumas implementações de DHT estão disponíveis na Internet para uso, dentre elas as duas aprofundadas no trabalho o Bamboo[1] e o FreePastry[2]. Ambas utilizam o Pastry como modelo de DHT e foram escolhidas pela simplicidade de instalação, facilidade de obtenção e documentação disponível na Internet. Optou-se por usar somente o Pastry pois não encontramos uma implementação viável do CHord possível de ser usada em nosso laboratório.

Foi necessário um estudo detalhado das duas implementações citadas. Além da compreensão da codificação destas implementações, ambas em Java, foi de grande ajuda a documentação fornecida, informações encontradas em grupos de discussões na Internet e o uso a ferramenta Wireshark[3] para o acompanhamento e depuração dos resultados obtidos com os testes realizados.

Na fase de testes e implementações, foi utilizado como base inicial para o

[1]http://bamboo-dht.org/
[2]http://freepastry.org/FreePastry/
[3]http://www.wireshark.org/

desenvolvimento deste trabalho o programa DNSHandler [Wong et al. 2007], este também implementado na linguagem Java.

Não somente as implementações do Bamboo e do FreePastry tiveram a necessidade de adaptações do código para atingir o objetivo proposto, mas também o DNSHandler, que utilizava de tabela *hash* local em sua implementação original e precisava ser adaptado para o uso de DHT.

Após a fase de alterações e adaptações dos softwares das DHTs, deu inicio a fase de testes comparativos entre estes, de onde foram obtidos diversos resultados de desempenho. Com a utilização da ferramenta *Dig* do Linux, softwares de captura de pacotes (*Wireshark*) e *scripts* para filtragem das respostas implementados em Shell Script, foi possível obter os resultados em diversas situações distintas criadas igualmente para o Bamboo e FreePastry. Os nomes eram inseridos de forma sequencial. Fizemos consultas sequenciais e aleatórias sem reiniciar os nós de tal forma a utilizar o sistema de cache das DHTs e, posteriormente, a cada consulta, reiniciavamos a DHT zerando o cache de tal forma a evitar a influência do cache nos resultados. Esses resultados foram analisados, demonstrando assim qual era a implementação mais adequada para uso nas próximas etapas propostas.

Novamente foram feitas adaptações no código do DNSHandler, possibilitando agora a resolução de nomes na Internet, tratamento de falhas existentes nas buscas na Internet, tais como ausência de respostas, e atualização automática da DHT diante de cada novo nome resolvido.

Utilizando novamente o *Dig* e *Wireshark* como ferramentas de apoio, testes foram realizados na Internet com a DHT para resolver nomes em situações de nomes validos e inexistentes no DNS da Internet. Testamos também nomes prestentes na DHT (anteriormente resolvidos).Tais situações possibilitam uma resolução sem problemas e com respostas corretas, e forçando a ocorrência de determinados problemas e tratando os mesmos adequadamente.

Diante do resultado dos testes realizados, foi possível então verificar a viabilidade na utilização da DHT como resolução de nomes na Internet.

4. Resultados e Discussões

Os testes foram feitos no Laboratório de Informática do Centro Universitário do Norte do Espirito Santo, na Universidade Federal do Espirito Santo, onde em dois computadores conectados por uma rede foi criada uma DHT com 15 nós para as duas diferentes implementações em momentos distintos.

4.1. Escolha da Implementação de DHT

Primeiramente, testes sequenciais foram realizados, ou seja, cada nome era pesquisado apenas uma vez obedecendo a uma ordem definida através de uma sequência numérica agregada a cada nome, de “num0” até “num999” para pesquisas com 1000 nomes.

Em seguida, testes aleatórios foram realizados, ou seja, consultas de nomes randômicos e possibilitando agora uma busca por nomes repetidos, o que traz uma característica adicional: a influência do sistema *cache* das DHTs. Testes com o *cache*

ativado e sem o *cache* ativado foram analisados e em ambas as DHTs este foi um fator com bastante influência nas respostas obtidas e tabeladas. Outro fator que foi analisado nos testes foi o número de pesquisas e o número de nomes armazenados nas DHTs, respeitando os limites encontrados nas máquinas.

Com os resultados obtidos através de testes de desempenho em ambas as implementações, mostramos que o FreePastry é mais eficiente e robusto que o Bamboo e menos susceptível a erros em nosso cenário de aplicação e, portanto, foi utilizado nas próximas etapas deste trabalho.

O FreePastry mostrou um limite para sobrecarga da DHT entre 1000 e 1500 nomes armazenados. Tal fator foi importante para caracterizar melhor a presença do cache, como exemplo na Tabela 2, nos testes referentes às linhas dois (sem *cache*) e quatro (com *cache*), diferenciados pelo número de consultas mas com resultados consideravelmente diferentes.

Os tempos de resposta dessas pesquisas foram obtidos pelo *Dig* e filtrados em arquivos de log com as informações mais importantes através de *scripts* implementados através de *Shell Script* do Sistema Operacional Linux. Tais informações foram então exportadas para o software Microsoft Excel, de onde foram obtidos os valores de média, mediana e desvio padrão para cada sequência de teste, como mostram as Tabelas 1 e 2.

Tabela 1. Consultas Bamboo

Descrição	Média	Mediana	Desvio Padrão
1000 consultas / 1000 nomes / sequencial	185 ms	10 ms	387 ms
150 consultas / 1000 nomes / aleatório	157 ms	7 ms	360 ms
1000 consultas / 1000 nomes / aleatório	176 ms	11 ms	383 ms

Tabela 2. Consultas FreePastry

Descrição	Média	Mediana	Desvio Padrão
1000 consultas / 1000 nomes / sequencial	13 ms	7 ms	12 ms
1000 consultas / 1000 nomes / aleatório	12 ms	6 ms	14 ms
1000 consultas / 50 nomes / aleatório	2 ms	1 ms	6 ms
10000 consultas / 1000 nomes / aleatório	3 ms	1 ms	7 ms

4.2. Testes com o DNSHandler

Na Internet, o DNS funciona de maneira em que um usuário qualquer possa acessar um determinado site através do seu nome, por exemplo: www.ufes.br, sem saber o seu endereço IP. Assim, ao fazer referência a um nome, o DNS trata de encontrar o seu endereço IP associado e permitindo à conexão entre o usuário e o destino através do protocolo TCP. Porém, por trás dessa associação, existe uma busca hierárquica em diversos servidores espalhados pelo mundo.

O DNSHandler age como um Proxy para o serviço DNS e, em sua versão original tratava de interceptar nomes vindos da aplicação do usuário, como o Internet Explorer ou o próprio *Dig* utilizado, e fazer a busca em uma tabela hash, entregando ao usuário a resposta caso fosse obtida.

Algumas modificações em sua versão original foram feitas, pois, além de não utilizar DHT o DNS Handler não se conectava a Internet para tratar os casos de nomes não encontrados em sua tabela *hash*, uma característica importante para este trabalho.

Portanto, ao se pesquisar determinado nome (exemplo: www.ceunes.ufes.br) o DNS Handler tem a responsabilidade de fazer a busca na DHT de tal endereço e responder para o cliente. Caso esse nome não exista na DHT, se fazem necessárias algumas etapas para que esse nome seja inserido na DHT, caso seja válido, e esteja disponível para uma nova busca. Todas essas etapas são mais bem detalhadas nas figuras 1, 2(a), 2(b), 3(a), 3(b):

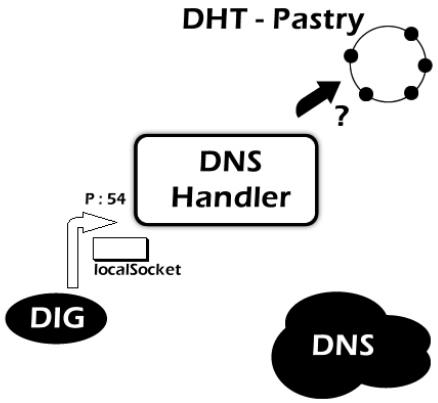


Figura 1. Etapa de pergunta do Cliente.

Na Figura 1 é possível perceber uma consulta feita pelo cliente ao DNSHandler pela porta 54, definida inicialmente para testes, no papel de substituir a porta 53, utilizada como padrão para consultas DNS. Essa consulta é feita através de um *socket* chamado *localSocket* que é responsável pela comunicação entre a aplicação cliente e o DNSHandler. Em seguida, o DNSHandler faz a consulta a DHT.

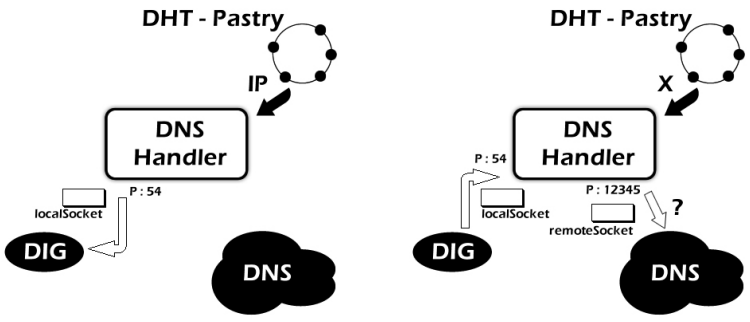
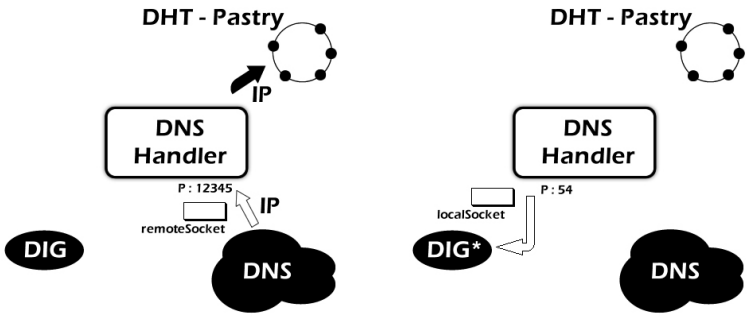


Figura 2. a)Etapa que representa uma busca com sucesso. b)Etapa que representa uma busca sem sucesso.

A Figura 2(a) representa o caso em que tal consulta foi respondida com sucesso (ou seja, o nome procurado está na DHT), onde a DHT responde ao DNSHandler com o IP associado ao nome perguntado, e na mesma comunicação pela porta 54, é feita a resposta para o Cliente.

A Figura 2(b) representa o caso em que tal pergunta não obteve resposta na DHT. Assim, o DNS Handler fica responsável por fazer a pergunta novamente, mas desta vez para o DNS na Internet. Essa comunicação é feita pela porta 1234 (definida aleatoriamente, escolhida apenas por ser superior à porta 1024, com restrições administrativas) através do *remoteSocket*.



(a) (b)

Figura 3. a)Etapa que representa como o sistema corrige uma busca sem sucesso. b)Etapa de resposta após uma busca sem sucesso.

Na sequência do caso anterior, caso a pergunta feita ao DNS seja respondida com sucesso, tal resposta será armazenada na DHT, para que assim seja possível obter a resposta para futuras perguntas sobre o mesmo nome, como é mostrado na Figura 3(a).

E finalmente, a Figura 3(b) representa a etapa final em tal sequência. O DNSHandler fica responsável também em repassar a resposta obtida em consulta ao serviço de DNS na Internet para o cliente.

5. Conclusão

Diante dos objetivos propostos e dos resultados obtidos, conseguimos informações relevantes sobre o funcionamento do serviço de resolução de nomes da Internet, o DNS, e sistema de Tabelas *Hash* Distribuídas, a DHT. Depois de etapas de estudos e prática de implementação, foi possível criar dois sistemas em DHT para resolução de nomes utilizando as implementações FreePastry e Bamboo, que por sua vez utilizam o Pastry como modelo de DHT. Após a fase de testes, foi possível observar claramente um desempenho melhor do FreePastry sobre o Bamboo no ambiente criado, que por tal motivo foi escolhido para utilização posterior.

Tendo então uma implementação que atingia a proposta inicial deste trabalho, foi necessária a adequação desta para funcionar na Internet, cujos resultados qualitativos também foram satisfatórios. Em avaliações de viabilidade, o sistema implementado demonstrou bom funcionamento quando utilizado na Internet substituindo o sistema de DNS.

Portanto, diante dos resultados neste trabalho e da experiência obtida, é possível dizer que a utilização de DHT para resolução de nomes na Internet é valida, porém, se faz necessário um maior número de testes avaliativos da utilização na Internet e de desempenho, quando comparado ao sistema DNS até então utilizado.

Outra possibilidade para projetos futuros seria a de oferecer este serviço utilizando *Cloud Computing* (Computação nas Nuvens), avaliando e comparando o desempenho do mesmo com o DNS.

Referências

Ahlgren, B., Eggert, L., Feldmann, A., Gurtov, A., and Henderson, R. T. (2007). Abstracts Collection - Naming and Addressing for Next Generation Internetworks. In *Naming and Addressing for Next-Generation Internetworks*, Dagstuhl Seminar Proceedings.

Cox, R., Muthitacharoen, A., and Morris, R. (2002). Serving DNS Using a Peer-to-Peer Lookup Service. In *IPTPS '01: Revised Papers from the First International Workshop on Peer-to-Peer Systems*, pages 155–165.

Lua, E. K., Crowcroft, J., Pias, M., Sharma, R., and Lim, S. (2005). A survey and comparison of peer-to-peer overlay network schemes. *IEEE Communications Surveys and Tutorials*, 7:72–93.

Pappas, V., Massey, D., Terzis, A., and Zhang, L. (2006). A Comparative Study of the DNS Design with DHT-Based Alternatives. In *INFOCOM 2006*.

Ramasubramanian, V. and Sirer, E. G. (2004). The design and implementation of a next generation name service for the internet. *SIGCOMM Comput. Commun. Rev.*, 34(4):331–342.

Rowstron, A. I. T. and Druschel, P. (2001). Pastry: Scalable, decentralized object location, and routing for large-scale peer-to-peer systems. In *Middleware '01: Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms Heidelberg*, pages 329–350, London, UK. Springer-Verlag.

Stoica, I., Morris, R., Liben-Nowell, D., Karger, D. R., Kaashoek, M. F., Dabek, F., and Balakrishnan, H. (2003). Chord: a scalable peer-to-peer lookup protocol for internet applications. *IEEE/ACM Trans. Netw.*, 11(1):17–32.

Wong, W., Pasquini, R., Villaça, R., de Paula, L. B., Verdi, F., and Magalhães, M. (2007). A Framework for Mobility and Flat Addressing in Heterogeneous Domains. In *Proceedings of the Brazilian Symposium of Computer Networks and Distributed System. 25th.*