High Availability: A DNS and Reverse Proxy Approach in Multi-Cloud

L. P. G. Pires and E. A. M. Fagotto

***Abstract* - In spite of Cloud Computing services seem to be very attractive as an alternative to traditional on-premise data centers, there is still some concern about the providers availability. This article describes a solution that implements high availability Multi-Cloud through the distribution of DNS access and the use of reverse proxy. The results obtained, by means of simulations indicate that, in principle, one can obtain an arbitrary Multi-Cloud availability, by using Clouds of lower availability. Specifically, it was shown that, from two Clouds of 99.43% and 99.49% availability one can obtain a 99.9971% Multi-Cloud system. It is also shown that, in this particular case, the Multi-Cloud system costs 34% less than the solution with a single Cloud.**

***Keywords* - Gestão de Redes e Serviços, Alta Disponibilidade, Cloud Computing, Multi-Cloud, Proxy Reverso e Sistema de nomes de domínio.**

I. INTRODUÇÃO

A utilização de serviços de tecnologia da informação estruturados em nuvem, conhecidos como *Cloud Computing*, vem ganhando, nos últimos anos, cada vez mais adeptos ao redor do mundo [1]. Na esfera empresarial, um dos grandes desafios enfrentados com serviços em nuvem apresenta-se com a questão da disponibilidade [2].

Provedores de *Cloud Computing* oferecem diversos serviços conhecidos como *"X as a Service"* [3], ou "tudo como serviço" que permitem uma ampla gama de possibilidades, que vão desde a utilização de um simples *software*, até complexas infraestruturas de rede como serviço. Tipicamente, provedores de *Cloud Computing* anunciam percentuais de disponibilidade de 99,95% [2]. Isso, por muitas vezes, leva os usuários a desenvolverem uma expectativa equivocada quanto à indisponibilidade (*downtime*) dos serviços oferecidos por provedores de *Cloud Computing* ser praticamente zero. Por exemplo, são excluídas dos cálculos de SLA (*Service Layer Agreement*): *i*) a indisponibilidade dos serviços de *Cloud*, causada por manutenções programadas e *ii*) falhas na rede externa, as quais, na maioria das vezes, são de responsabilidade de empresas de telecomunicações terceiras. Contudo, estes fatores afetam a taxa de disponibilidade para os usuários finais da *Cloud* [2]. Em [4] mostra-se que, por diversas vezes, falhas em serviços de *Cloud* levaram a grandes períodos de inatividade e, em alguns casos, a perda de dados de seus usuários. Por exemplo, em 2011, uma falha nos servidores de armazenamento em nuvem do provedor de *Cloud Computing* Amazon fez com que sites de vários de seus clientes ficassem fora do ar por quase de 36 horas. Neste mesmo episódio, a falha ocorrida gerou mais problemas do que sites fora do ar, levando mesmo a perda de dados que estavam armazenados na solução Amazon EBS Volumes [5].

Em [1], mostrou-se que 91% de 58 empresas consultadas, todas com representatividade expressiva em termos mundiais, migraram para serviços em *Cloud* somente ambientes "*non-productions*". Ainda assim, mesmo no caso que sistemas de produção foram migrados, estes envolviam apenas serviços de baixa criticidade, tais como sistemas de CRM (*Customer Relationship Management*) e ferramentas de escritório para a exploração dos benefícios de flexibilidade e mobilidade oferecidos por *Cloud Computing*. Desta forma, garantindo-se não afetar a continuidade dos negócios em casos de falha.

Pode-se dizer que, a despeito do entusiasmo global com *Cloud Computing*, a adesão à tecnologia enfrenta problemas de confiabilidade semelhantes aos encontrados pelos sistemas de *OutSourcing* na década de 1990 [6]. Em face de tal cenário, prover alta disponibilidade é, portanto, uma tarefa desafiadora.

Neste contexto, nossa proposta enquadra-se no âmbito de *Plataform-level SLA*, o que cobre: servidores físicos na *Cloud*, plataforma de virtualização e *hardwares* de rede.

No presente estudo, propõe-se e testa-se, utilizando-se de simulações, um modelo para o provimento de alta disponibilidade em ambientes *Multi-Cloud* [7]. Isto permite ofertar, de acordo com a necessidade do usuário, por exemplo, disponibilidades da ordem de 99,999% [8]. Para tanto, em conjunção com um ambiente *Multi-Cloud*, são utilizadas técnicas de distribuição de acessos por *Domain Name System* (DNS) e conexão de *Proxy* reverso [9].

O presente artigo organiza-se da seguinte forma: Em II, apresenta-se e discute-se o modelo proposto. Em III, apresenta-se o ambiente de experimentação para o teste do método de Alta Disponibilidade. Na Seção IV, são apresentados os resultados dos testes de desempenho e de disponibilidade do ambiente. Em V, é apresentada uma análise financeira e, finalmente, em VI, tem-se a conclusão.

II. MODELO DE ALTA DISPONIBILIDADE EM MULTI-CLOUD

Neste trabalho, propõe-se um modelo para alta disponibilidade baseado em ambiente *Multi-Cloud*, por meio da distribuição de acessos por DNS e *Proxy* Reverso. A utilização simultânea de duas ou mais *Clouds* como estrutura para um mesmo sistema de TI é conhecida como *Multi-Cloud* [10]. Esta solução pode ser implementada para sistemas que

L. P. G. Pires, Pontifícia Universidade Católica de Campinas, Campinas, luis.pires@puc-campinas.edu.br

E. A. M. Fagotto, Pontifícia Universidade Católica de Campinas, Campinas, eric@puc-campinas.edu.br

Corresponding Author: Luis Paulo Gonçalves Pires

demandem, dentre outros quesitos, disponibilidade elevada, muitas vezes não atingível se se utilizasse apenas um provedor de serviços em *Cloud*.

Segundo [7], sistemas *Multi-Cloud* podem tolerar falhas simultâneas em *N Clouds*, desde que o conteúdo seja replicado em *N+1 Clouds* distintas, o que é conhecido como espelhamento [11]-[12].

A utilização de *Proxy* para a distribuição de acessos entre *Clouds* (Fig. 1) utilizada em [7] é a forma mais comum em sistemas *Multi-Cloud.* Contudo, esta abordagem apresenta um inconveniente denominado como "ponto único de falha" [13]. Devido à sua topologia, todos os acessos passam obrigatoriamente por um único servidor *proxy*, que distribui efetivamente os acessos entre os servidores de conteúdo localizados nas diferentes *Clouds* que compõem o sistema. Neste cenário, em caso de falha deste componente, toda a estrutura de *Multi-Cloud* ficaria indisponível. De modo a resolver este inconveniente, aumentando assim a disponibilidade total do sistema, nossa proposta conta com diversos servidores *Proxy* junto às *Clouds* utilizadas, formando conexões cruzadas entre os servidores de conteúdo. Assim, mesmo que *N Proxies* apresentem falha, *N+1 Proxies* continuarão com a distribuição do acesso aos servidores de conteúdo, conforme apresentado em [7].

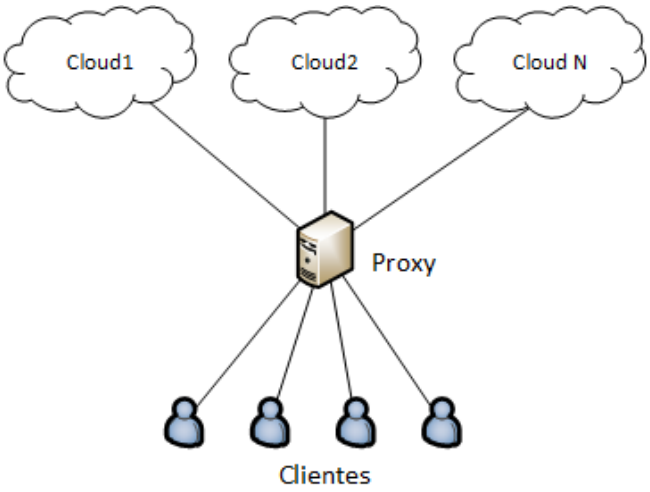

Figura 1. Distribuição de acessos com *proxy simples* em *Multi-Cloud.*

A Fig. 2 mostra o diagrama geral da solução de alta disponibilidade proposta neste trabalho para um ambiente de Infraestrutura como Serviço (IaaS) em *Multi-Cloud* com múltiplos *proxies*.

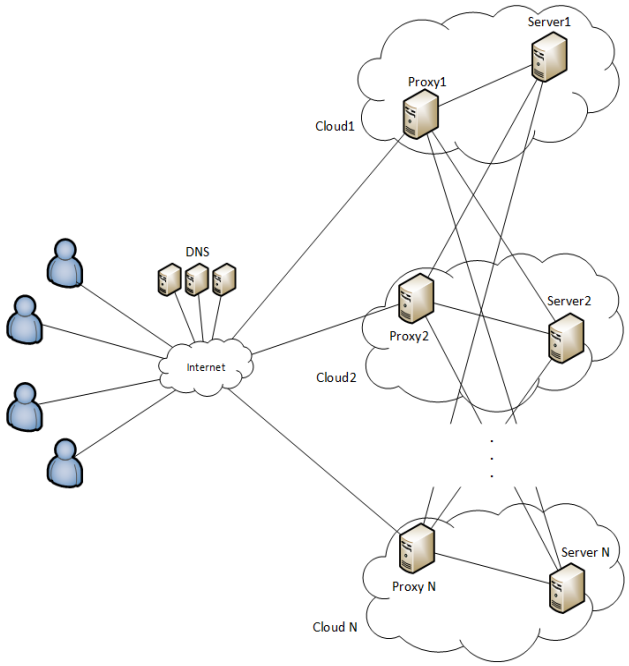

Figura. 2. Diagrama geral da proposta. – Distribuição de acessos por DNS e *Proxy* Reverso.

Neste modelo, são utilizados *N Proxies* de acesso, correspondendo *N* ao número de *Clouds* empregadas no sistema. Cada *Cloud* conta com o seu próprio Servidor *Proxy* reverso e com o servidor de conteúdo. Desta forma, todas as *Clouds* envolvidas possuem autonomia de componentes para o seu funcionamento. A requisição de acesso do usuário passa inicialmente pelo Sistema de Nomes de Domínio (DNS), que troca o nome consultado pelo IP do servidor, neste caso, o IP do servidor *Proxy*. É possível, mediante o DNS, entregar para o mesmo nome de domínio, uma lista com diferentes endereços IPs. No lado do cliente, o navegador de Internet, por exemplo, recebe a lista de IPs do domínio em questão, com os endereços dos servidores *Proxy*. Caso o *Proxy* referente ao primeiro IP da lista não esteja acessível, automaticamente, o navegador tentará acessar o domínio pelo próximo IP listado na consulta ao DNS. Desta forma, faz-se com que o acesso chegue a todos os *Proxies* de forma distribuída, de acordo com a lista de IPs entregue pelos servidores de DNS.

A disponibilidade, que trata da relação entre o tempo de atividade e o tempo de operação de um sistema de TI, pode ser calculada por meio de (1), conforme [8].

$$D_j = \frac{Ts - Tp}{Ts} \quad (1)$$

Sendo D_j, Ts, Tp, a disponibilidade total da j-ésima *Cloud*, o tempo de serviço e o tempo de parada, respectivamente.

Para o cálculo da disponibilidade total mínima em *Multi-Cloud*, propõe-se a seguinte expressão:

$$D_{T\min} = 1 - \left(\prod_{j=1}^{N} 1 - D_j \right) \quad (2)$$

Na equação (2), tem-se um produto relacionado às disponibilidades das N Clouds, que compõem o ambiente *Multi-Cloud*. Para o cálculo de $D_{T\min}$ supõe-se que as D_j das *Clouds* $j = \{1,2,3,...\}$ sejam independentes entre si.

A despeito da utilização de sistemas *Multi-Cloud* já ser uma realidade [10], no melhor de nosso conhecimento, esta é a primeira vez que este tipo de solução, conjugada com DNS e *Proxy* reverso, é utilizada para o provimento de alta disponibilidade em *Cloud Computing*.

III. AMBIENTE DE EXPERIMENTAÇÃO

A fim de se testar a presente proposta, projetou-se um sistema que provesse disponibilidade DT > 99,99% a partir de *Clouds* com menor disponibilidade. Para tanto, emularam-se duas *Clouds* com disponibilidade individuais máximas de 99,95%, um valor típico oferecido por provedores [2]. A implementação do ambiente de experimentação do método, cujo diagrama é mostrado na Fig. 3, foi realizada utilizando-se a ferramenta de virtualização Oracle Virtualbox [14]. Com esta, foi possível se emular todos os componentes de uma instância IaaS, em um microcomputador com sistema operacional Linux *Based Kernel 3.16.0-31* [15]. De forma a distribuir o acesso entre as *Clouds*, utilizou-se em nosso teste uma função do sistema de nomes de domínio (DNS - *Domain Name System*) conhecida como *Round Robin* [16], [17] implementada no Servidor DNS *Bind*, que responde a cada solicitação de consulta com uma lista de endereços IP, em ordem diferente, para determinado nome de domínio. Desta forma, um cliente ao realizar uma consulta ao DNS receberá uma lista de IPs correspondentes e, a cada nova consulta e, novo cliente, essa lista será entregue com os endereços IPs colocados de forma aleatória. Permite-se, assim, que os acessos sejam distribuídos a diversos servidores *proxies*, evitando-se a limitação imposta pelo ponto único de falha presente em outros sistemas. Uma vez configurado o servidor *Bind*, com a função *Round Robin* (DNS-RR), distribui-se, para consultas ao nome de domínio utilizado (no caso específico deste trabalho, *www.ha.lab.local*), dois endereços IP, que remetem aos servidores *Proxy1 e Proxy2,* localizados um em cada *Cloud,* que executavam o Servidor Nginx [18]. Estes servidores estão configurados como *Proxy* reverso. A função do *Proxy* é a de receber a solicitação do cliente e repassá-la aos servidores de conteúdo *Server1* e *Server2*, que executam o Servidor *Web* Apache2 [19]. Cada servidor *Proxy* foi ligado diretamente, via LAN *(Local Area Network)*, a um servidor de conteúdo localizado na mesma *Cloud*, e, via WAN *(Wide Area Network)*, a outro servidor de conteúdo localizado em outra *Cloud*. Desta forma, cada *Proxy* consegue direcionar as requisições de acesso dos clientes aos seus servidores, formando uma conexão cruzada.

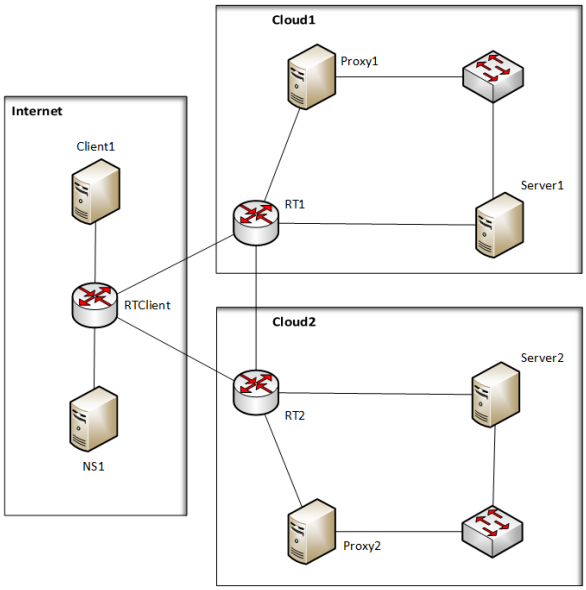

Figura. 3. Diagrama de Rede – Utilizado para o teste da proposta.

Emularam-se no ambiente de experimentação duas *Clouds*, sendo que a *Cloud1* tinha disponibilidade de 99,49% e a *Cloud2* de 99,43% (vide sessão IV, algoritmos I e II e a discussão que segue). Utilizando-se a equação (2), calculou-se que esse sistema *Multi-Cloud* proveria uma $D_{T\min} = 99{,}9971\%$. Em nosso experimento, utilizaram-se as máquinas virtuais (VM), conforme Tabela I:

Tabela I
Relação de Máquinas Virtuais.

VM	Função	Em execução
Server1	Servidor de aplicação – *Cloud1*	Apache Server V2.4.7
Server2	Servidor de aplicação – *Cloud2*	Apache Server V2.4.7
Proxy1	Servidor Proxy Reverso – *Cloud1*	Nginx V1.4.6
Proxy2	Servidor Proxy Reverso – *Cloud2*	Nginx V1.4.6
RT1	Roteador *Cloud1*	Roteamento do Kernel Linux V3.13.0-32, NetEm
RT2	Roteador *Cloud2*	Roteamento do Kernel Linux V3.13.0-32, NetEm
NS1	Servidor DNS	Bind V9 - DNS-RR
RTClient	Roteador de acesso cliente	Roteamento do Kernel Linux V3.13.0-32, Netem
Client1	Cliente gerador de requisições	Apache Benchmark V2.4.7

IV. TESTES DE DESEMPENHO E DISPONIBILIDADE

Para simular o acesso a diferentes serviços de *Cloud*, foram utilizados recursos do NetEm [20], implementados no *Kernel* do Linux, com o qual, torna-se possível inserir atrasos, perda de pacotes, *jitter* e outras características típicas de um sistema de redes de telecomunicação [20]. Neste caso, no roteador que permite acesso à *Cloud*1, foi inserido um atraso na conexão da ordem de 24 ms e, no roteador de acesso à *Cloud*2, um atraso de 140 ms. Tais atrasos foram baseados na ferramenta *Cloudping* [21], que testa a latência da conexão a diversos serviços de *Cloud Computing*. Os tempos de atraso utilizados remetem ao tempo de resposta de uma *Cloud* localizada fisicamente a 100 km de onde o acesso ao sistema foi originado, e de outra localizada a 7500 km, respectivamente, emulando um sistema distribuído entre diferentes países.

De forma a se medir o tempo de resposta dos servidores, nas diferentes *Clouds*, utilizou-se a ferramenta Apache Benchmark para servidores *Web* [22].

Na fig. 4, apresentam-se os resultados de um teste do Apache Benchmark realizado sobre o *Proxy* da *Cloud*1. Pode-se observar a diferença entre os tempos de resposta com e sem a distribuição por *Proxy reverso* entre as *Clouds*. Em média calculada, baseando-se nos resultados dos testes do Apache Benchmark, distribuindo-se os acessos, o tempo médio de resposta foi 18% menor e o tempo total da conexão (tempo de processamento + tempo de espera) foi 33% menor. Esta melhora no tempo de resposta não foi observada realizando-se o mesmo teste na *Cloud*2, que apresenta atraso na conexão de 140ms. Neste caso, praticamente não houve diferença no tempo de acesso com o *Proxy* distribuindo as requisições entre as *Clouds*, conforme se mostra na Fig. 5. Este fato se deve à maior latência na conexão, entretanto, sem prejuízo à disponibilidade. Buscou-se, com esta parte do experimento, mostrar que a distribuição de acessos não penaliza o tempo de resposta das conexões. Na verdade, em alguns casos, com o da *Cloud1*, pode até melhorá-lo.

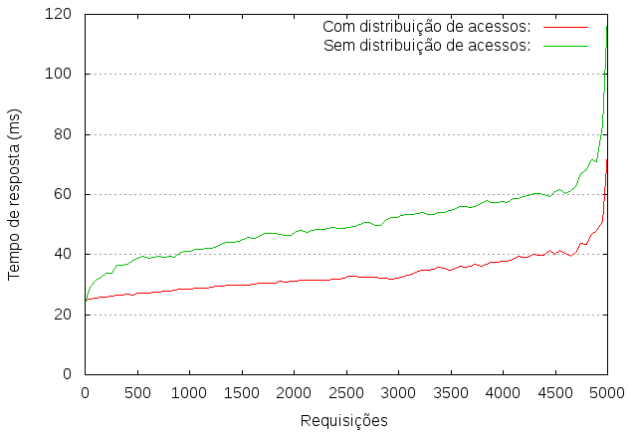

Figura. 4. Resultado do *Benchmark – Cloud*1.

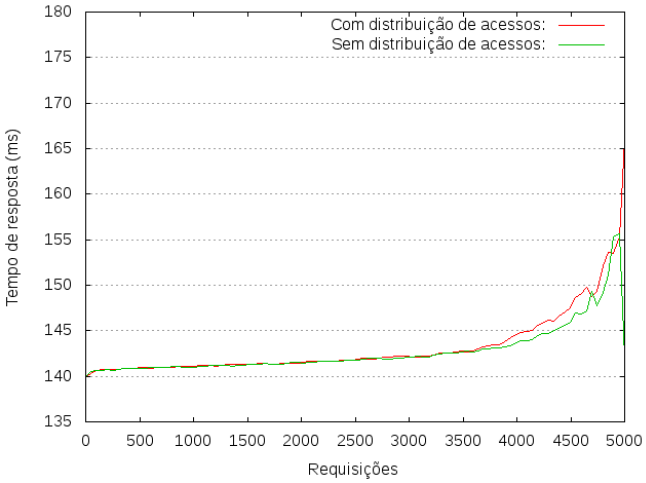

Figura. 5. Resultado do *Benchmark – Cloud*2.

Para se verificar a disponibilidade do sistema foi realizado um teste por um período de 24 horas. Durante esse tempo, geraram-se requisições aleatórias aos servidores, utilizando-se o Apache Benchmark, de acordo com o **Algoritmo1**. Os parâmetros referentes ao número de conexões totais e o número de requisições simultâneas foram determinados aleatoriamente, utilizando-se a função RANDOM do *Kernel* do Linux [23], para simular o tráfego do sistema.

Algoritmo 1. GeradorRequisições

```
1 inicio
2   enquanto verdadeiro; faça
3       execute benchmark com número de concorrência
        aleatória entre 30 e 150 com conexões totais de
        200 a 15000 em http://www.ha.lab.local
4 aguarde por um tempo aleatório entre 1 e 3 segundos
5 fim
```

Utilizando-se do **Algoritmo 2,** geraram-se *scripts*, tanto nos servidores *Proxies* como nos servidores de conteúdo, com os quais, de forma independente e aleatória, causaram-se interrupções nos serviços, emulando falhas [8] nos componentes da estrutura experimentada. Estes scripts foram responsáveis pela emulação das disponibilidades das *Clouds* 1 e 2, respectivamente, em 99,49% e 99,43%.

Algoritmo 2. GeradorInterrupção

```
1 início
2   aguarde tempo aleatório entre 3600 e 86400
    segundos.
3     se o serviço http estiver em execução então
4       pare o serviço http
5         aguarde um tempo entre 330 e 600 segundos
6       inicie o serviço http
7   se não
8         saia
9 fim
```

Durante o período de testes, coletaram-se os dados de acessos aos servidores de conteúdo, por meio do *software* de

monitoramento *Munin Monitoring* [24], com os resultados mostrados nas Fig. 6(a) e Fig. 6(b), respectivamente, para as *Clouds* 1 e 2.

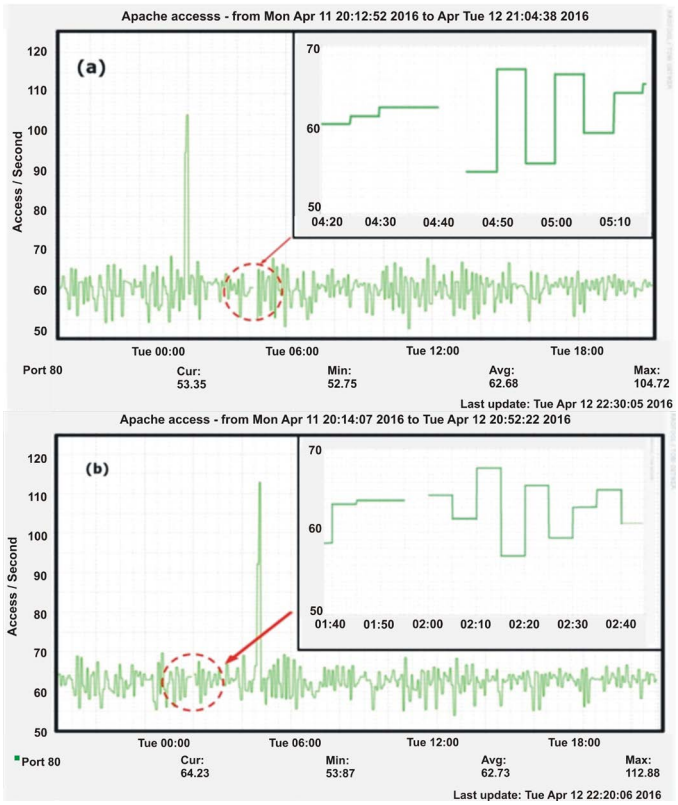

Figura. 6. Servidores de Aplicação – *Clouds* 1 e 2 – 24 horas.

A variação no número de acessos ao servidor da *Cloud1* (Fig. 6(a)), manteve-se entre 55 e 70 acessos/s, exceto às 01h55m, quando se nota um pico, correspondendo a 105 acessos/s.

Cenário semelhante foi evidenciado no monitoramento da *Cloud2* (Fig. 6(b)), sendo que o pico de acesso aconteceu às 04h40m/s, chegando a 112 acessos.

Desta forma, destacam-se dois momentos: *i*) às 01h55m registrou-se um aumento nos acessos ao Servidor de Aplicação da *Cloud*1, concomitantemente, à interrupção de acesso ao servidor de aplicação da *Cloud*2, (detalhe da Fig. 6(b)); *ii*) às 04h40m o serviço *web* do Servidor de aplicação da *Cloud*1 falhou (detalhe da Fig. 6(a)), aumentando o número de acessos ao servidor de aplicação da *Cloud*2.

A despeito das interrupções dos servidores, observa-se que jamais o acesso do usuário ao conteúdo ficou indisponível. Isso ocorreu devido ao uso da distribuição de acessos por DNS aos servidores *Proxies*, que encaminharam as solicitações aos servidores de aplicação disponíveis. Assim, qualquer que seja a origem das falhas em componentes isolados, ou mesmo da *Cloud* como um todo, o acesso ao conteúdo por parte do usuário continua de forma transparente. Desta forma, obteve-se D_T = 100%, que é maior do que $D_{T\min}$ = 99,9971%, indicando que a estratégia de projeto mostrou-se bem sucedida. Em relação à D_T = 100% observada no experimento, vê-se um aumento de 0,51% em relação à disponibilidade da melhor *Cloud* individual (*Cloud1* – 99,49%). Assim, para o usuário do sistema, tem-se a diminuição do tempo de inatividade de 7,2 minutos da *Cloud1* e 8,2 minutos da *Cloud2* para 0 (zero) minutos em ambiente *Multi-Cloud*.

Comparando-se $D_{T\min}$ com a melhor *Cloud* individual, nota-se uma melhora de 0,5% no índice de disponibilidade. Neste cenário, pode-se calcular o tempo máximo de inatividade do sistema que poderia ser experimentado pelo usuário. Assim, das 24 horas de experimentação da solução, a indisponibilidade seria da ordem de 3 segundos.

Comparativamente para o período de um ano, mantendo-se os mesmos índices, ter-se-ia a diminuição do tempo de inatividade de 1,8 dias apresentado pela *Cloud1* e 2,1 dias apresentado pela *Cloud2*, para um tempo máximo de indisponibilidade (de acordo com $D_{T\min}$) de 24 minutos no ambiente *Multi-Cloud* proposto. Isso corresponde a uma diminuição da ordem 120 vezes do tempo indisponibilidade do sistema para o usuário final.

V. ANÁLISE DE FATORES DE CUSTO ASSOCIADOS A DIFERENTES COMBINAÇÕES DE CLOUD PARA PROVIMENTO DE IAAS

De forma a verificar a viabilidade completa da solução proposta, fêz-se também uma análise financeira da mesma. De acordo com [25] atualmente, as maiores limitações, para implementação de costumam ser de ordem financeira e não técnica. Desta forma, apresenta-se a seguir, uma análise baseada em diferentes combinações de provedores de *Cloud Computing*, baseando-se no percentual de disponibilidade e custo apresentados.

Conforme apresentado na Seção II, é possível, aplicando-se o modelo proposto, utilizar diferentes serviços de *Cloud Computing* para se atingir níveis conhecidos na literatura como alta-disponibilidade (99,999%). Observa-se que não foram encontrados provedores de *Cloud Computing* que ofereçam em contrato disponibilidades da ordem de 99,999% a partir de uma única *Cloud*.

Como cenário para comparação, foram utilizados dados disponiveis em [26], referentes ao custo e as disponibilidades mensais (Tabela II).

Tabela II
Disponibildiade Mensal *vs.* Custo.

Disponibilidade (%)	**Valor (R$)**
99,95	319,00
99,90	110,00
99,00	101,20
95,00	93,50
90,00	82,50
89,90	66,00

Os valores apresentados na Tabela II foram obtidos no ano de 2016 com o valor do dólar cotado a R$ 3,38.

Na Fig. 7, apresenta-se o custo do sistema de *Cloud* em função da disponibilidade mensal. O número ao lado de cada simbolo indica a quantidade de Clouds que compõe o sistema. Dentre as combinações exibidas na Fig. 7, o melhor cenário de disponibilidade *versus* custo, consiste na utilização de duas *Clouds*, ambas com 99,90% de disponibilidade ao custo de R$ 110,00 cada. Assim, configurando-se um ambiente *Multi-Cloud*, atinge-se a disponibilidade desejada de 99,999% ao custo mensal de R$ 220,00, menor valor possível para esta disponibilidade.

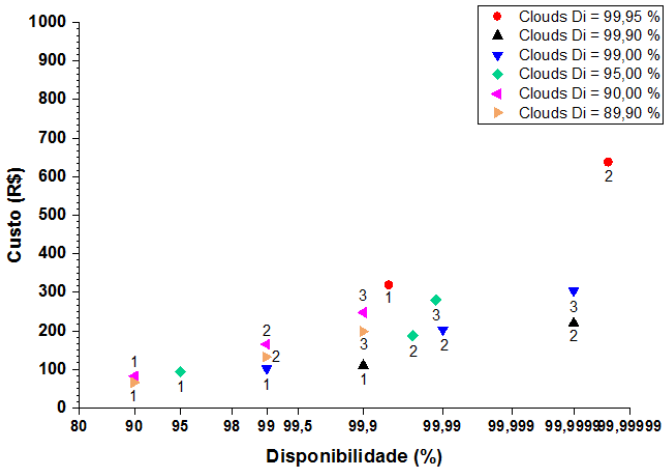

Figura. 7. Combinações de Clouds x Custo.

Destaca-se, que o ambiente *Multi-Cloud* com $D_{T\min}$ = 99,999%, formado por duas *Clouds* de 99,90% e 99,00%, apresenta um custo mensal 34% menor que um ambiente de *Cloud* simples de 99,95%. Assim mostra-se a solução ser viável não somente em quesitos técnicos, como também em aspecto financeiro.

VI. CONCLUSÃO

Neste trabalho, foi apresentada e testada uma estratégia para provimento de alta-disponibilidade para serviços em *Cloud*, utilizando-se de ambiente *Multi-Cloud* conjugado com a distribuição de acessos por DNS e *Proxy* reverso. Nossos testes foram realizados mediante simulações, com duas *Clouds* emuladas, com disponibilidades de 99,49% e 99,43%, que permitiram alcançar uma disponibilidade total mínima de 99,9971%. Este aumento de 0,5% no índice de disponibilidade corresponde a uma diminuição da ordem de 120 vezes do tempo indisponibilidade do sistema *Multi-Cloud* para o usuário final. A solução proposta permite que empresas utilizem serviços de diferentes *Clouds*, simultaneamente, de forma a não ficarem atreladas a um único provedor de serviços, aumentando assim a disponibilidade de suas aplicações. No que se refere à analise financeira, baseada em valores de serviços fornecidos por provedores de *Cloud*, mostrou-se que a solução de *Multi-Cloud* não só é economicamente viável, mas também pode ser um fator utilizado na redução de custos. Isto porque, para os testes realizados, mostrou-se que a combinação de *Clouds* com menor disponibilidade é 34% mais barata do que uma única *Cloud*, que oferece disponibilidade típica de 99,95%. Em [27], utilizou-se uma infraestrutura de Plataforma como Serviço (PaaS – *Platform as a Service*) em *Multi-Cloud* para ofertar alta-disponibilidade. Observa-se que, em seu melhor resultado, a sobrecarga inserida pelo sistema foi de 17,37%, cerca de 21s de acréscimo ao tempo de execução. Comparativamente, em nossa proposta de *Multi-Cloud,* utilizando distribuição por DNS e *proxy* reverso de forma conjugada, não houve sobrecarga para o ambiente. Na realidade, para a *Cloud* com latência de 24ms, diminuiu-se em 18% o tempo total de acesso em comparação ao teste realizado sem a utilização de *proxy* reverso. No caso da *Cloud* com maior latência (140ms), não foi evidenciada alteração no tempo de resposta na execução dos testes. Uma comparação com a tecnologia utilizada por provedores como, por exemplo, a Amazon, não é simples, visto que detalhes da infraestrutura são considerados como informação sigilosa. Entretanto, a despeito disso, [28] parece sugerir a existência de um servidor para balanceamento de carga, que pode guardar alguma semelhança com servidor *proxy* de nossa proposta. Contudo, em função de um evento recente [29], é provável que o servidor de balanceamento da Amazon, apesar de redundância interna, ainda funcione como um ponto único de falha. Certamente, entende-se a necessidade de se realizar um teste de campo com uma infraestrutura de *Clouds* comerciais, para uma melhor avaliação do potencial da proposta. Espera-se realizar este teste na continuação deste trabalho, além de configurações adicionais como *ip_hash* e *least-connected* nos servidores *proxies* para avaliação de desempenho.

AGRADECIMENTOS

À Pontifícia Universidade Católica de Campinas - PUC-Campinas pela concessão da bolsa de estudos.

REFERÊNCIAS

[1] N. Sfondrini, G. Motta, and L. You, “Service Level Agreement (SLA) in Public Cloud Environments : A Survey on the current enterprises adoption.,” *2015 5th Int. Conf. Inf. Sci. Technol.*, 2015.

[2] X. Yuan, Y. Li, T. Jia, T. Liu, and Z. Wu, “An Analysis on Availability Commitment and Penalty in Cloud SLA,” *Proc. - Int. Comput. Softw. Appl. Conf.*, vol. 2, no. 61232005, pp. 914–919, 2015.

[3] P. Mell and T. Grance, “The NIST definition of cloud computing,” *NIST Spec. Publ.*, vol. 145, p. 7, 2011.

[4] C. Pham, P. Cao, Z. Kalbarczyk, and R. K. Iyer, “Toward a High Availability Cloud : Techniques and Challenges,” *IEEE*, 2012.

[5] “Falha em serviço da Amazon faz clientes perderem dados permanentemente,” 2011. [Online]. Available: http://tecnologia.uol.com.br/ultimas-noticias/redacao/2011/04/29/falha-em-servico-da-amazon-faz-clientes-perderem-dados-permanentemente.jhtm. [Accessed: 05-Sep-2016].

[6] M. J. Earl, “The Risk of Outsourcing IT,” *Sloan Manage. Rev.*, vol. 37, no. 3, pp. 26–32, 1996.

[7] Q. Zhang, S. Li, Z. Li, Y. Xing, Z. Yang, and Y. Dai, “CHARM: A Cost-Efficient Multi-Cloud Data Hosting Scheme with High Availability,” *IEEE Trans. Cloud Comput.*, vol. 3, no. 3, pp. 372–386, 2015.

[8] K. Benz and T. Bohnert, “Dependability modeling framework: A test procedure for high availability in cloud operating systems,” *IEEE Veh. Technol. Conf.*, 2013.

[9] B. Ciciani, F. Quaglia, P. Romano, and D. Dias, “Analysis of Design Alternatives for Reverse Proxy Cache Providers,” *11TH IEEE/ACM*

Int. Symp. Model. Anal. Simul. Comput. Telecommun. Syst., 2003.
[10] M. A. Alzain, E. Pardede, B. Soh, and J. A. Thom, "Cloud Computing Security : From Single to Multi-Clouds," *2012 45th Hawaii Int. Conf. Syst. Sci.*, 2012.
[11] L. M. Pham and T. M. Pham, "Autonomic fine-grained migration and replication of component-based applications across multi-clouds," *Inf. Comput. Sci. (NICS), 2015 2nd Natl. Found. Sci. Technol. Dev. Conf.*, pp. 5–10, 2015.
[12] A. Abouzamazem and P. Ezhilchelvan, "Efficient inter-cloud replication for high-availability services," *Proc. IEEE Int. Conf. Cloud Eng. IC2E 2013*, pp. 132–139, 2013.
[13] W. Wu, K. Wang, R. Jan, and C. Huang, "A Fast Failure Detection and Failover Scheme for SIP High Availability Networks 1," *13th IEEE Int. Symp. Pacific Rim Dependable Comput.*, pp. 187–190, 2007.
[14] Oracle, "VirtuaBox," 2015. [Online]. Available: www.virtualbox.org. [Accessed: 01-Mar-2015].
[15] "Ubuntu," 2016. [Online]. Available: http://www.ubuntu.com/desktop. [Accessed: 15-May-2016].
[16] Y. S. Hong, J. H. No, and S. Y. Kim, "DNS-based load balancing in distributed web-server systems," *Proc. - Fourth IEEE Work. Softw. Technol. Futur. Embed. Ubiquitous Syst., SEUS 2006 andthe Second Int. Work. Collab. Comput. Integr., Assur. WCCIA 2006*, vol. 2006, pp. 251–254, 2006.
[17] T. Brisco, "Request for Comments: 1794 - DNS Support for Load Balancing," 1995. [Online]. Available: https://tools.ietf.org/html/rfc1794. [Accessed: 01-May-2016].
[18] "Nginx," 2016. [Online]. Available: www.nginx.com. [Accessed: 09-Mar-2016].
[19] "Apache2," 2016. [Online]. Available: httpd.apache.org. [Accessed: 01-Jan-2016].
[20] Linux Foundation, "NetEm," 2009. [Online]. Available: https://wiki.linuxfoundation.org/networking/netem. [Accessed: 01-May-2016].
[21] "Cloudping," 2015. [Online]. Available: http://www.cloudping.info/. [Accessed: 10-Dec-2015].
[22] Apache.org, "Apache Benchmark," 2015. [Online]. Available: httpd.apache.org/docs/2.4/programs/ab.html. [Accessed: 01-Mar-2015].
[23] Michael Kerrisk, "Man-pages RANDOM," 2015. [Online]. Available: http://man7.org/linux/man-pages/man4/random.4.html. [Accessed: 26-May-2016].
[24] "Munin Monitoring," 2016. [Online]. Available: http://munin-monitoring.org/. [Accessed: 10-Mar-2016].
[25] J. Steddum, "A Brief History of Cloud Computing," 2013. [Online]. Available: http://blog.softlayer.com/2013/virtual-magic-the-cloud. [Accessed: 11-Oct-2016].
[26] Embratel, "Embratel - Cloud Server," 2016. [Online]. Available: http://portal.embratel.com.br/cloud/cloud-server/. [Accessed: 31-Aug-2016].
[27] F. Paraiso, N. Haderer, P. Merle, R. Rouvoy, and L. Seinturier, "2012 IEEE Fifth International Conference on Cloud Computing A Federated Multi-Cloud PaaS Infrastructure," 2012.
[28] AWS Reference Architectures, "Fault tolerance & high availability," 2017. [Online]. Available: https://media.amazonwebservices.com/architecturecenter/AWS_ac_ra_ftha_04.pdf. [Accessed: 06-Mar-2017].
[29] D. Lee, "Amazon data centre fault knocks websites offline temporarily," *BBC News Services*, 2017. [Online]. Available: http://www.bbc.com/news/world-us-canada-39119089. [Accessed: 01-Mar-2017].

Luis Paulo Gonçalves Pires possui graduação em Informática para Gestão de Negócios pela Faculdade de Tecnologia do Estado de São Paulo - FATEC (2009). Pós-graduação Lato Sensu em Redes de Computadores pela Universidade Federal de São Carlos - UFSCar (2011). Pós-graduação Stricto Senso (Mestrado) em Engenharia Elétrica pela Pontifícia Universidade Católica de Campinas - PUC-Campinas (2016). Tem experiência na área de Ciência da Computação, com ênfase em Redes de Computadores e Infraestrutura de TI.

Eric Alberto de Mello Fagotto obteve os títulos de Bacharel em Física Aplicada em 1989, o de Mestre em Física em 1992 e o de Doutor em Ciências em 1995, todos pelo Instituto de Física Gleb Wataghin (IFGW), da Universidade Estadual de Campinas (UNICAMP). No período de 1996 a 1997, atuou como professor participante no Departamento de Telemática (DT) da Faculdade de Engenharia Elétrica e de Computação (FEEC) da UNICAMP. Desde 2001, atua como professor na Pontifícia Universidade Católica de Campinas (PUC-Campinas). Participou (2003-2006) do Programa de Tecnologia da Informação no Desenvolvimento da Internet Avançada (TIDIA) e do Projeto KyaTera (FAPESP 2003/08320-2.). Foi membro (2009-2017) do Instituto Nacional de Ciência e Tecnologia (INCT) para Comunicações Ópticas FOTONICOM (CNPq 574017/2008-9 e FAPESP 08/57857-2). No período de 2014-2015 foi Coordenador do Programa de Pós-Graduação em Engenharia Elétrica da PUC-Campinas. Sua experiência profissional compreende métodos teóricos e experimentais em física de semicondutores (sistemas 0-D, 1-D, células solares, dispositivos optoeletrônicos, espectroscopia foto- e termo-acústica, sistemas para a difusão de impurezas em semicondutores, dentre outros), aplicação de óptica não-linear a dispositivos fotônicos e processamento óptico de sinais. É revisor de periódicos do IEEE e da OSA. É membro do IEEE Photonics Society.