Um framework para avaliação de dependabilidade em sistemas de armazenamento em nuvem

Thiago de Morais Severo

Curso de Bacharelado em Sistemas de Informação – Universidade Federal Rural de Pernambuco (UFRPE) – Unidade Acadêmica de Serra Talhada (UAST)
56900-000 – Serra Talhada – PE– Brasil

thiagoeyed@gmail.com

Resumo. *Esse trabalho tem como objetivo produzir modelos que possam ser aplicados em arquiteturas de computação em nuvem, a fim de obter um sistema que ofereça disponibilidade e desempenho satisfatório. Tais modelos devem permitir avaliação de métricas relacionadas ao tempo de atividade e desempenho em sistemas que utilizem essas arquiteturas. Ao fim da construção desses modelos, foram realizados estudos de caso para a validação dos modelos gerados. Uma das principais contribuições desse trabalho é a de poder servir como base na modelagem e criação de um ambiente de computação em nuvem que tenha como características: disponibilidade e confiabilidade.*

1. Introdução

A computação em nuvem deixou de ser uma tendência ou aposta no mundo computacional para se tornar um modelo altamente utilizado na oferta de serviços de T.I. Ela está relacionada com os mercados de software como serviço (SaaS), plataforma como serviço (PaaS) e Infraestrutura como serviço (IaaS). As abordagens SaaS, PaaS e IaaS disponibilizam diferentes recursos para seus usuários. (MARINOWSKY; WEBER; PIMENTAL, 2013).

Embora a computação esteja em ampla expansão, seus projetistas possuem dificuldade de defini-la de maneira que alcance um nível de serviço operacional confiável e satisfatório, fazendo-se necessárias pesquisas sobre a avaliação de sua disponibilidade, para evitar implantá-la trazendo prejuízo para seus clientes e provedores (IBM, 2015). De acordo com (OPSERVICES, 2015), uma hora de indisponibilidade de um serviço, a partir de uma estimativa feita entre médias e grandes empresas, é de 42.000,00 de dólares /hora em média. Diante desse contexto, é vantajoso para essas empresas propor soluções a fim de minimizar os *downtimes*. Tomando como base esses problemas, nesse trabalho são propostos modelos para avaliar disponibilidade em Sistemas de Armazenamento em Nuvem.

A motivação para elaboração deste trabalho é o fato de a computação em nuvem, mesmo possuindo investimentos elevados e adoção crescente, enfrenta problemas que devem ser resolvidos para poder se tornar um modelo de computação aceito como de utilização massiva (IBM, 2015). Dentro desse cenário, projetistas de sistemas de armazenamento na nuvem têm dificuldade de defini-los de forma que possuam o nível de disponibilidade desejado pelo cliente. Principalmente os modelos de infraestrutura como serviço (IaaS), encontram dificuldades para proteger suas bases, tais serviços são vulneráveis a ataques. Nuvens podem ser afetadas por falhas estruturais como quedas de energia, falhas de hardware (MARINESCU, 2015).

Diante disso, faz-se necessária a realização de estudos sobre parâmetros a serem seguidos para que se possa atingir um nível de serviço confiável e satisfatório, evitando prejuízos tanto para os provedores quanto para os clientes. Uma maneira de realizar tais estudos é através da criação de modelos para avaliar características sobre disponibilidade de serviço na nuvem. Esses modelos podem ser usados no processo de planejamento e montagem de ambientes de computação em nuvem, permitindo que administradores e gestores possam construir ou evoluir suas arquiteturas e, consequentemente, diminuindo seus tempos de inatividade e aumentando o tempo médio para falhar.

2. Trabalhos Relacionados

(CARVALHO; GUIMARÃES, 2013) traz a abordagem para avaliação de uma infraestrutura existente onde é adicionada redundância de componentes na sua arquitetura. Nele são utilizados Diagrama de confiabilidade de blocos (RBD) como abordagem de modelagem para avaliação analítica de cenários complexos. O trabalho Realizado por (DANTAS, 2013) os benefícios de um mecanismo de replicação em um ambiente de computação em nuvem *Eucalyptus*. Apesar de considerar falhas de hardware e software em seus modelos, não utiliza a modelagem SPN como ferramenta para a análise de dependabilidade na infraestrutura estudada.

(MARINOWSKY; WEBER; PIMENTAL, 2013) Apresenta realiza a modelagem do mecanismo de tolerância a falhas do *MapReduce*, possuindo uma nova abordagem para modelar componentes distribuídos usando redes de Petri. Apesar de apresentar um ambiente de cluster distribuído, sua implantação não é feita na *cloud*. Utilizando SPN e RBD como modelos de dependabilidade para a avaliação de subsistemas, propondo mecanismos de redundância. Enquanto a proposta de um conjunto de modelos de (SOUZA, 2013) avalia o fator temperatura como impactante na disponibilidade de arquiteturas de *data centers*. Foram construídos modelos de redes de recompensa estocásticas (SRN) para o comportamento detalhado, falha e reparação do sistema, e calculam a disponibilidade do sistema através da replicação de componentes críticos do sistema.

(SILVA; MACIEL; ZIMMERMAN, 2014) Apresenta uma abordagem para avaliar a capacidade de sobrevivência em sistemas IaaS implantado em geograficamente centros de dados distribuídos. No entanto, nenhum deles faz uma abordagem voltada para o armazenamento na nuvem ou a modelagem de sistemas de armazenamento e voltadas à replicação de dados. Diferente dos trabalhos anteriores, este trabalho propõe uma análise de dependabilidade em infraestruturas de computação em nuvem por meios de modelagem utilizando modelos em RBD e SPN.

3. Redes de petri estocásticas (SPN)

É uma técnica de especificação de sistemas que possibilita representá-los matematicamente e que possui mecanismos de análise poderosos, os quais possibilitam a verificação de propriedades como disponibilidade e confiabilidade, além de permitir a averiguação da corretude do sistema especificado. Uma rede de Petri pode ser do tipo estocástica (SPN), a qual possui cada transição associada a uma variável aleatória com distribuição exponencial, que expressa o atraso de tempo necessário para a deflagração da transição. Através de Redes de Petri (SPN) é possível modelar sistemas paralelos, concorrentes, assíncronos e não determinísticos (MACIEL; TRIVEDI, 2012).

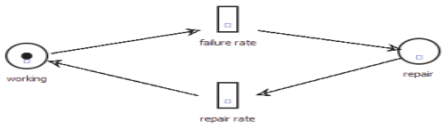


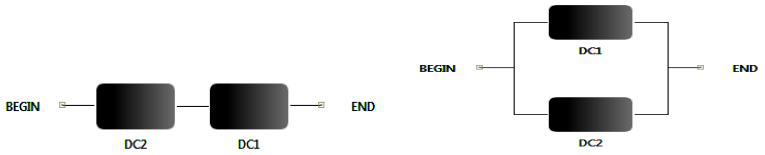
Figura 1. Modelo de Rede de Petri

Fonte: MACIEL, 2011

De acordo com (FERNANDES, 2007), a principal diferença entre um modelo SPN e uma Rede de Petri usual são as transições. A Figura 1 mostra um exemplo de modelo SPN, onde o lugar (working) está ativo pelo *token* gerando uma marcação que habilita transição de falha (failure rate) e representa o evento de falha consumindo o *token* da marcação *working* e levando o sistema ao estado de reparo (*repair*) com colocação do *token* no lugar *repair*, por consequência habilita a transição repair rate.

4. Diagrama de Blocos de Confiabilidade (RBD)

RBD define o relacionamento lógico ou sequencial entre componentes de um sistema. A proposta principal é de calcular a confiabilidade, porém, o modelo pode servir na avaliação de outras métricas, como a disponibilidade. Cada bloco representa um componente do sistema e é interligado a outros blocos (MACIEL; TRIVEDI, 2012). Em um modelo de diagramas de blocos de confiabilidade, os componentes podem ser dispostos em série, paralelo ou através de combinações dessas disposições. Uma representação que contém blocos em série necessita de todos os componentes operacionais, enquanto em paralelo pelo menos um dos componentes deve estar operacional para o sistema funcionar (TRIVEDI, 1996).



(a) (b)

Figura 2. (a) Exemplo de modelo RBD em série. (b) Exemplo de modelo RBD em paralelo

Com base na Figura 2(a), onde os blocos estão organizados em série, a confiabilidade é calculada pela Equação 1 (MACIEL; TRIVEDI, 2012)..

$$R_S = R_1 \times R_2 \quad (1)$$

Rs é a disponibilidade total do sistema, onde R1, descreve a confiabilidade do bloco DC1 e R2, descreve a confiabilidade do bloco DC2. Considerando-se a Figura 2 (b), onde os blocos estão organizados em paralelo, a confiabilidade é calculada pela Equação 2. (MACIEL; TRIVEDI, 2012)..

$$R_P = 1 - \prod_{i=1}^{2} (1 - R_i(t)).$$

(2)

5. Definição de arquiteturas e respectivos modelos de alta disponibilidade para mecanismos de acesso à nuvem.

Nessa sessa serão representados os modelos em RBD das arquiteturas propostas baseados nos estudos de características de componentes dos Sistemas de Armazenamento em nuvem do *Openstack*, através da utilização de redundância nos mecanismos de requisição ao cluster de armazenamento. As arquiteturas mostradas nas figuras 3 (a) e (b) utilizam os componentes de acesso à nuvem em nós distintos.

5.1. Arquiteturas Propostas para modelagem em RBD

A primeira dispõe de um nó para cada serviço sem redundância. O auth node (AN) contém o serviço de autenticação, enquanto o proxy node é responsável pelo serviço de proxy.

(a) (b)

Figura 3. (a) Arquitetura sem redundância nos nós de acesso. (b) Arquitetura redundante nos nós de acesso

A segunda implementa a redundância nos nós citados na arquitetura anterior em nós distintos. O serviço de autenticação está implantado no auth node redundante (ANR) assim como o serviço de proxy também está implantado no Proxy Node redundante (PXR).

5.2. Modelos RBD para as arquiteturas propostas

A figura 3(a) apresenta a primeira arquitetura. O objetivo dele é oferecer a disponibilidade do *front-end* na utilização dos serviços críticos de acesso ao cluster em máquinas físicas separadas, a falha de um dos componentes representa a falha por completo do sistema. O modelo para essa arquitetura apresenta os seguintes componentes em série: proxy node (PX), switch (SW), auth node (AN) e roteador (RT), dispostos de acordo com o modelo da figura 4.

O seguinte modelo, representa a segunda arquitetura disposto na figura 3(b), possuindo a inclusão de redundância no serviço de proxy e no serviço de autenticação. A aplicação desse modelo possui o objetivo de aumentar a disponibilidade da arquitetura através da replicação de componentes, porém, com a inclusão da redundância no mecanismo de autenticação e proxy em nós distintos . Os componentes desse modelo são: roteador (RT), switch (SW), proxy node (CN), proxy node redundante (PX), auth node(AN), auth node redundante (ANR) e storage node (SN), como mostrado na figura 7.

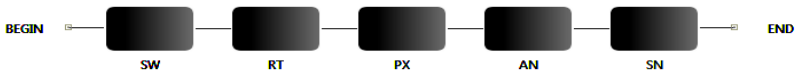


Figura 4. Modelo RBD da arquitetura sem redundância nos nós de acesso à nuvem

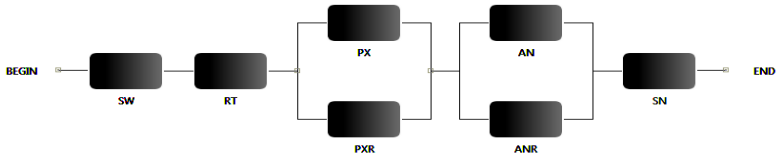


Figura 5. Modelo RBD da arquitetura sem redundância nos nós de acesso à nuvem

6. Definição da arquitetura de replicação de dados em cluster de armazenamento e respectivo modelo SPN

Seguindo as características de replicação, em que a distribuição dos dados nos nós de armazenamento é feita tentando colocar o dado o mais distante possível fisicamente, a figura 6 ilustra como tal é feita através de quatro nós de armazenamento do cluster da nuvem *Openstack*. O algoritmo, a pedido de colocação do dado pelo cliente levará três cópias, representadas pelos pontos verdes, para os locais mais distantes possíveis no cluster. O nó que não recebeu a cópia é nomeado local handoff (OPENSTACK, 2014).

A falha de um dos nós que contenha a cópia acarretará na redistribuição dos dados entre o cluster, a fim de que a durabilidade de dados triplicados permaneça. O nó *handoff* estando ativo receberia uma das cópias em caso de falha de um dos que a possuam. Quanto mais réplicas usadas, mais proteção contra a perda de dados quando houver uma falha. Isto é, grupos que possuem centros de dados separados em locais distintos, possuem maior segurança nesse quesito.

Figura 6. Distribuição das cópias no cluster de armazenamento

A figura 7 apresenta o modelo do cluster com quatro nós de armazenamento através de um modelo SPN. Em relação o modelo com três nós, além do acréscimo de um nó de armazenamento, a principal diferença está no funcionamento. Em caso de falha de um nó com cópia, não é necessário esperar a recuperação do nó para manter a cópia em três clusters, já que nessa arquitetura há a existência do nó *handoff*. A representação do modelo lógico apresenta a SN4 e suas transições imediatas SN4-H e SN4-F no modelo lógico além de SN4-ON, SN4-OFF e SN4-ON-COPY e suas transições MTTF-SN4, MTTR-SN4 e SN4-COPY. A falha de um dos nós leva a habilitação da transição SN4-H, levando o token do ring (espera) até SN4.

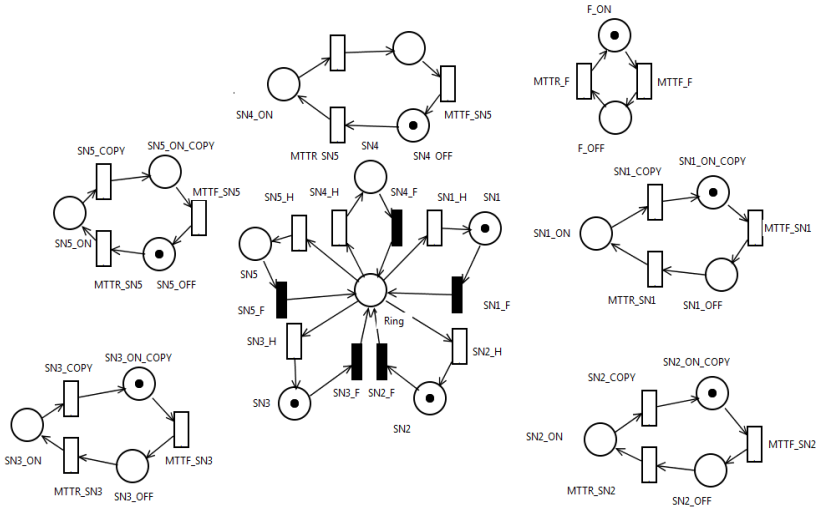


Figura 7. Modelo para a arquitetura de distribuição das cópias no cluster de armazenamento.

7. Estudos De caso

Os resultados mostrados na tabela 1, são referentes ao segundo cenário para esse estudo de caso. Eles avaliam a influência das arquiteturas de *front-end* na disponibilidade total do sistema com a utilização de cluster de baixa disponibilidade. Os clusters de armazenamento para todos os sistemas apresentam um nó de armazenamento. São avaliadas a disponibilidade no estado estacionário, downtime anual e o tempo médio para falhar (MTTF). Os maiores *downtimes* (tempo de inatividade), são encontrados nos sistemas que utilizam front-ends que não dispõem de redundância nos componentes de suas arquiteturas.

As arquiteturas com maior índice de disponibilidade, correspondem aos cenários que utilizam redundância. Apesar dos cenários que empregam redundância no subsistema de acesso à nuvem conseguirem um aumento de disponibilidade, conseguindo uma redução de 25 por cento de downtime, não é proporcional aos resultados obtidos nas arquiteturas redundantes dos subsistemas, visto que o cluster de armazenamento apresenta um disponibilidade baixa, influenciando no resultado total do sistema.

Tabela 1. Resultados com arquitetura de cluster básica

Arquitetura	**Disponibilidade**	**Downtime(Horas)**
Mecanismos de acesso não redundantes	0.98802786236591	104
Mecanismos de acesso redundantes	0.9910707317533	78

Os resultados obtidos pelos sistemas com clusters de alta disponibilidade (ver quadro 2), em relação à variedade de frontends, mostram que além da aplicação de redundância ser eficiente comparado ao não redundante, os resultados são superiores proporcionalmente ao sistema com arquitetura básica, conseguindo reduzir o *downtime* anual em 77 por cento. Os resultados mostram que a utilização de alta disponibilidade no *front-end* conseguem reduzir o tempo de inatividade de nuvemzzzz, porém para arquitetura básica do sistema de armazenamento não apresenta um grande impacto na disponibilidade devido a baixa disponibilidade do cluster de armazenamento .

7.1 Avaliações da disponibilidade de dados pela replicação de dados no cluster de armazenamento

Diante dos modelos apresentados nesse estudo de caso, foram avaliadas a disponibilidade gerada por esses modelos em alguns cenários. A disponibilidade para uma cópia representa o mínimo que o *cluster* deve ter para fornecer os dados à nuvem enquanto ter três réplicas de dados é a consistência mínima exigida para se ter um sistema de alta tolerância à desastres ou falhas. A figura 8 contém o gráfico que mostra o aumento da disponibilidade para a manutenção de pelos menos uma réplica no sistema, sendo proporcional ao aumento do número de nós no cluster, crescendo dois noves (na parte decimal) para cada nó incluso. Portanto, a replicação dos dados três vezes no cluster mostra-se eficiente, visando, visto que o *downtime* do cluster com três nós é inferior à uma hora por ano, a inclusão de nós de armazenamento varia de acordo com a necessidade da nuvem.

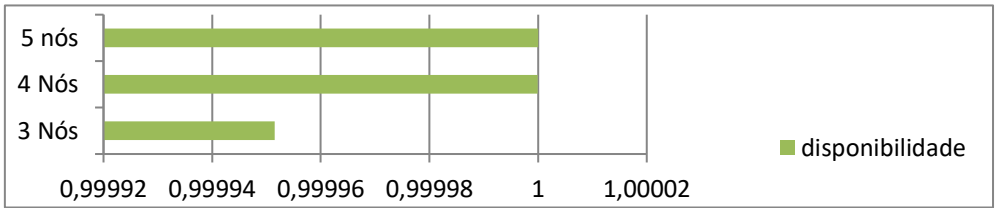


Figura 8. Disponibilidade do cluster VS número de nós.

As propostas de arquiteturas com redundância no mecanismo de acesso à nuvem (proxy), apresentam um significativo aumento nos seus valores de disponibilidade. A principal diferença de resultado no impacto causado pela utilização desse mecanismo, pode ser vista na comparação da utilização entre um e dois nós *proxy*. Enquanto a colocação de um terceiro componente não apresenta grande influência no resultado, se comparado com a segunda arquitetura. Outro aspecto analisado é que a redundância no proxy mostra-se efetiva para qualquer tipo de infraestrutura de armazenamento, no caso estudado, com três, quatro ou cinco nós de armazenamento, como pode ser visto na figura 9, onde a inclusão do terceiro *proxy* não alcança um aumento de disponibilidade tão grande como a inclusão do segundo *proxy*.

O resultado desse estudo também mostram que a disponibilidade tende a variar menos à medida que aumentamos a quantidade de componentes redundantes. Em ambos os cenários, tendo como base a replicação dos dados em diferentes locais, a utilização de um nó se mostra eficiente na manutenção dos dados replicados durante um maior tempo, além de elevar a capacidade de armazenamento da nuvem. Os dados expostos no gráfico da figura 9 validam o cenário estudado no estudo de caso anterior, em que a influência da redundância é maior em sistemas com *clusters* de alta disponibilidade.

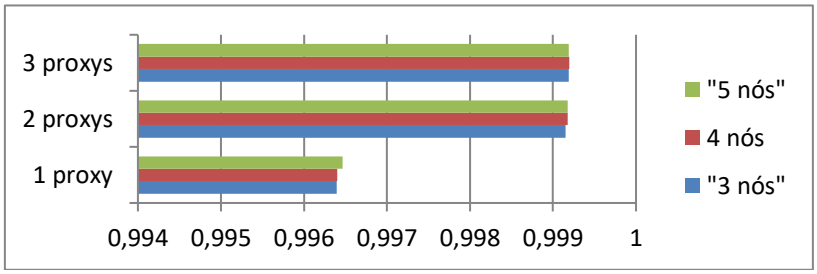


Figura 9. Disponibilidade fornt-end VS número de nós

8. Conclusão

Os resultados obtidos através desses modelos avaliam aplicação de mecanismos de replicação em pontos de acesso à nuvem e a avaliação da redundância em *clusters* de armazenamento em nuvem, através da utilização de componentes de uma infraestrutura baseada no openstack swift. Com o objetivo de avaliar os modelos propostos, estudos de caso foram realizados com o intuito de avaliar a disponibilidade de front-ends utili-zando arquiteturas redundantes e não redundante e também a influência dessas arquiteturas na disponibilidade da arquitetura básica do cluster de armazenamento.

Os resultados obtidos mostraram a grande redução do *downtime* dos front-ends redundantes, bem como o aumento da disponibilidade e tempo médio para falhar. Também é mostrada que a disponibilidade do front-end influi proporcionalmente na disponibilidade total do sistema, já que se o front-end estiver indisponível, por consequência o sistema também estará. No segundo estudo de caso são avaliadas a disponibilidade dos clusters de armazenamento a partir das técnicas de replicação de dados do *swift* e a influência do cluster na disponibilidade total do sistema. Os resultados mostraram que replicação de dados dentro do cluster eleva a disponibilidade dos dados em relação às falhas dos locais onde estão armazenados. Outro aspecto notado foi que sistemas com clusters de alta disponibilidade têm a sua própria disponibilidade afetada proporcionalmente à do front-end.

9. Referências

CARVALHO E.; GUIMARÃES A. Dependabilidade em rede de compu-tadores: uma análise baseada em importância para confiabilidade. Revista Brasileira de Administração Científica, 2013.

DANTAS J R. Modelos para Análise de Dependabilidade de Arquiteturas de Computação em Nuvem, UFPE, 2013

FERNANDES, S. M. M, Avaliação de Dependabilidade de Sistemas com Mecanismos Tolerantes a Falha. Centro de Informática, UFPE.2007. Tese (Doutorado). Programa de Pós-Graduação emCiência da Computação, UFPE, Recife, 2007.

IBM, Developer Works, Cloud computing não é o futuro, é o presente, 2013.Disponívelem:<https://www.ibm.com/developerworks/community/blogs/ctaurion/entry/cloudcomputingnaoeofuturoeopresente > Acessoem4denov2015:

MACIEL, P., TRIVEDI, K. S., Jr, R. M. DependabilityModeling. In:Performance and Dependability in Service Computing: Concepts, Techni-ques and Research Directions. IGI Global, 2012.

MARINESCU D. Computação em nuvem: vulnerabilidades na nuvem, 2013. Disponível em:<https://technet.microsoft.com/pt-br/magazine/dn271884.aspx> Acesso em: 6 de nov 2015.

Marynowski, J.E.; Pimentel, A.R.; Weber, T.S.; Mattos, A.j. Dependability Testing of MapReduce Systems. Universidade Federal do Parana, UFPR, Curitiba, Brasil, 2007.

OPENSTACK F.. Juno Instalation Openstack Object Storage. Disponível em: http://docs.openstack.org/developer/swift/. Acesso em: 20 de jan. 2016.

OPSERVICES. Qual o custo de um downtime?. Disponível em: < https://www.opservices.com.br/qual-e-o-custo-de-um-downtime>. Acesso em:15 de jul 2016.

SILVA, B.; Maciel P. R. M.; Zimmermann, A.; Brilhante, J. Survivability Evaluation of Disaster Tolerant Cloueeeeed Computing System. In: Proc. Proba-bilistic Safety Assessment Management conference, 12., 2014. Honolulu: UFPE, 2014. 12p.

TRIVEDI, K. S. et al.Reliability Analysis Techniques Explored ThroughaCommunication Network Example. [S.l.]: International Workshop on ComputerAidedDesign, Test, and Evaluation for Dependability, 1996. 241