Análise de Sensibilidade do Serviço Nextcloud Hospedado em Nuvem Privada

Wenderson de Souza Leonardo[1], **Alison Silva**[2], **Thiago Bezerra**[3], **Gustavo Callou**[1]

[1]Universidade Federal Rural de Pernambuco (UFRPE)

[2]Universidade de Pernambuco (UPE)

[3]Instituto Federal de Pernambucano (IFPE)

{wenderson.leonardo,gustavo.callou}@ufrpe.br, alison.vgsilva@upe.br,

thiago.bezerra@palmares.ifpe.edu.br

Abstract. *Applications hosted on private cloud servers require high connectivity and on-demand access to resources. To prevent failures that could lead to service unavailability, it is essential for private cloud providers to have a deep understanding of the system components and to identify critical points that may cause failures. This study proposes a sensitivity analysis of the Nextcloud service, which provides a file system as a service. A case study was conducted to evaluate the availability and performance of the private cloud environment using a design of experiments (DoE) approach and Stochastic Petri Nets (SPN) models. The results indicate that component analysis contributes to identifying the system's most critical points.*

Resumo. *Aplicações hospedadas em servidores de nuvem privada demandam alta conectividade e acesso a recursos sob demanda. Para evitar falhas que possam resultar na indisponibilidade dos serviços, é fundamental que as empresas de nuvem privada possuam uma compreensão aprofundada dos componentes que formam o sistema, além de serem capazes de identificar pontos críticos que possam ocasionar falhas. O presente trabalho propõe uma análise de sensibilidade do serviço Nextcloud, que oferece, como serviço, um sistema de arquivos. Foi realizado um estudo de caso com o objetivo de avaliar, por meio de uma abordagem de projeto de experimentos (DoE) e modelos em Redes de Petri Estocásticas (SPN), a disponibilidade e o desempenho do ambiente de nuvem privada. Os resultados demonstram que a análise dos componentes contribui para a identificação dos pontos mais críticos do sistema.*

1. Introdução

Com o avanço tecnológico, os dispositivos eletrônicos têm se tornado cada vez mais acessíveis e presentes na vida cotidiana de uma parcela significativa da população mundial. No Brasil cerca de 84% dos domicílios possuem acessoa internet, o que equivale a 64 milhões de domicílios [CETIC.br 2023]. Nesse contexto, o uso de computadores e dispositivos móveis tem deixado de ser meramente pessoal, consolidando-se como ferramentas essenciais em diversos setores. Atualmente, tais tecnologias são amplamente empregadas em áreas que vão desde a indústria automotiva até os segmentos de pecuária e agricultura, contribuindo para a otimização de processos e o aumento da eficiência produtiva.

A capacidade dos dispositivos modernos de se conectarem à web conferiu um novo significado ao uso da *Internet*. Um novo uso que inclui desde a realização de pesquisas com fins educacionais até a troca de mensagens entre indivíduos. Isso gerou um novo nicho para as empresas de tecnologia que corresponde a oferta de serviços através da Internet. Esse movimento tem levado cada vez mais pessoas a permanecerem conectadas enquanto realizam suas tarefas cotidianas. Para atender a essas solicitações em expansão, os provedores de serviços precisam cada vez mais de poder computacional. Nesse contexto, ambientes de computação em nuvem são soluções eficazes para atender essas necessidades. Esses ambientes funcionam como ferramentas para o compartilhamento de recursos computacionais, além de softwares e aplicações em geral [Dantas et al. 2020].

Devido às preocupações com a segurança dos dados, muitas empresas têm se afastado do uso de serviços de computação em nuvem fornecidos por provedores públicos, optando por construir seu próprio ambiente de nuvem privada. No entanto, manter esses serviços requer alta disponibilidade e confiabilidade, uma vez que são essenciais para assegurar a continuidade das operações e a integridade dos dados. Assim, para oferecer um serviço de computação em nuvem privada, é imprescindível a análise detalhada de sua arquitetura. Este trabalho tem como objetivo realizar uma análise de sistemas em nuvem, destacando a importância da identificação dos componentes críticos. Ao abordar esses aspectos, busca-se contribuir para a construção de ambientes em nuvem mais robustos e confiáveis, capazes de atender às crescentes demandas do mercado e garantir a satisfação dos usuários. De forma mais específica, as principais contribuições deste trabalho são:

- Uma estratégia para identificação dos componentes mais relevantes tanto para a disponibilidade como para o desempenho do sistema. Para isso, é proposto a utilização de Projeto de Experimentos (DoE).
- Proposição de modelos tanto para a análise de desempenho como de disponibilidade em SPN.
- Quantificar o impacto da disponibilidade no desempenho do sistema através de modelos em SPN.

Este trabalho está organizado da maneira a seguir. A Seção 2 explana alguns conceitos necessários para uma melhor compreensão deste trabalho. A Seção 3 apresenta os trabalhos relacionados encontrados na literatura. A Seção 4 descreve a arquitetura do ambiente analisado. A Seção 5 apresenta os modelos SPN que serão utilizados neste trabalho. A Seção 6 apresenta o estudo de caso. A Seção 7 apresenta as conclusões e mostra possíveis direcionamentos da pesquisa.

2. Referencial Teórico

Esta seção busca proporcionar uma breve explanação sobre alguns conceitos, os quais são fundamentais para o entendimento deste trabalho.

2.1. Rede de Petri Estocástica

As redes de Petri (PN) são uma ferramenta de modelagem gráfica e matemática aplicável a diversos sistemas caracterizados por propriedades como concorrência, assincronicidade, distribuição, paralelismo, não determinismo e/ou estocasticidade [Murata 1989]. As Redes de Petri Estocásticas (SPN) [Haas 2006] estendem as PNs ao incorporar o conceito de tempo em seus modelos. A Figura 1 apresenta os elementos fundamentais das SPNs. Os

lugares representam os possíveis estados do sistema. As transições são ações que podem ser executadas e exigem que suas pré-condições individuais sejam atendidas, resultando em uma alteração no estado do sistema. Os arcos conectam os lugares e as transições, representando o fluxo de tokens ao longo do sistema. Os tokens simbolizam os recursos disponíveis no sistema, os quais são consumidos e gerados pelas ações do sistema (transições). A distribuição dos tokens reflete o estado atual do sistema. Na transição estocástica pode ser associado um tempo, o qual representa o intervalo correspondente ao período de execução da atividade. O arco inibidor permite que o modelo verifique a presença de tokens no lugar de origem, inibindo a ativação da transição conectada.

Figura 1. Elementos de uma SPN

A Figura 2 ilustra um exemplo de SPN. Nela, modela-se o funcionamento de um sistema, no qual os lugares representam os estados do sistema (ligado ou desligado) e as transições representam as ações que alteram o estado do sistema (ligar e desligar). A parte (a) da figura mostra a representação do sistema quando está ligado (com um token no lugar "On"). Nesse estado, a única transição habilitada para ser disparada é a transição "Desligar". Após o disparo dessa transição, o modelo transita para o estado desligado (com um token no lugar "Off"), conforme ilustrado na parte (b) da figura. Em seguida, a transição "Ligar"torna-se habilitada, e o seu disparo retorna o sistema ao estado de ligado.

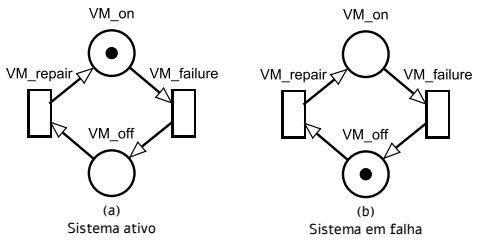

Figura 2. Exemplo de SPN

2.2. Análise de Sensibilidade

A análise de sensibilidade é um conjunto de técnicas empregadas para identificar quais componentes têm maior influência sobre uma métrica específica do sistema [Frank and Eslami 1980]. Para realizar uma análise de sensibilidade de maneira clara e objetiva, deve-se seguir alguns passos. i - Identificar e priorizar as variáveis de entrada mais influentes; ii - Determinar as variáveis de entrada não influentes para fixá-las em valores nominais; iii - Mapear o comportamento da métrica em relação aos valores de entrada, especificando um domínio de entrada; iv - Calibrar algumas variáveis de entrada utilizando dados já disponíveis ou restrições do sistema.

O Projeto de Experimentos (Design of Experiments – DoE) é uma técnica de análise de sensibilidade que usa técnicas estatísticas que possibilitam uma compreensão aprofundada sobre o objeto analisado. O DoE pode ser visto como uma sequência de experimentos nos quais o pesquisador ajusta variáveis ou fatores de entrada para observar e identificar quais mudanças nesses fatores resultam em variações nas saídas

[Montgomery and Runger 2010]. O DoE é formado por um conjunto de elementos. O primeiro elemento é o fator, que se refere a uma variável controlável que influencia o resultado. O segundo elemento é o nível, que corresponde aos diversos valores atribuídos aos fatores. O efeito representa as variações nas medições dos resultados.

O design fatorial é uma das abordagens mais comuns e eficientes para lidar com experimentos que envolvem dois ou mais fatores. Nele, os fatores são tratados como fatores cruzados, combinando os fatores em diferentes níveis. Assim, o analista é capaz de observar não só o impacto de cada fator isoladamente, mas também avaliar como as interações entre os fatores influenciam os resultados.

2.3. Disponibilidade

A disponibilidade pode ser descrita como a probabilidade de que o sistema analisado esteja operacional durante um determinado período de tempo [Maciel et al. 2010]. A disponibilidade é obtida pela divisão do tempo de funcionamento pela soma do tempo de funcionamento com o tempo de falha, conforme mostrado na Equação 1.

$$D = \frac{E[Uptime]}{E[Uptime] + E[Downtime]} \tag{1}$$

onde $E[Uptime]$ é o tempo de funcionamento do sistema e $E[Downtime]$ é o tempo em que o sistema está em falha.

Outra forma possível de calcular a disponibilidade é a partir do somatório dos tempos de falha e dos tempos de reparo dos dispositivos que compõem o sistema, conforme mostrado na Equação 2. Nessa equação, MTTF (Mean Time to Failure) representa o tempo médio até a falha do sistema, e MTTR (Mean Time to Repair) representa o tempo médio para o reparo do sistema. Esse trabalho também computa a disponibilidade em termos do número de noves, $-log_{10}(1 - D)$.

$$D = \frac{MTTF}{MTTF + MTTR} \tag{2}$$

Existem diferentes formas de aumentar a disponibilidade de um sistema, e uma das principais estratégias é a adição de redundâncias. Os dois tipos clássicos de redundância são o *hot standby* e o *cold standby*. Na técnica de redundância *hot standby*, os equipamentos redundantes permanecem ativos e em sincronia com o equipamento principal. Eles participam do processo de operação da mesma forma que o componente principal e, em alguns casos, podem até compartilhar a carga de trabalho. Dessa forma, esses equipamentos podem assumir a carga do sistema caso o componente principal falhe, sem um atraso na ativação [Johnson 1988]. Na redundância *cold standby*, os componentes redundantes iniciam desligados, e permanecem inativos aguardando até que os componentes principais falhem, momento em que são ativados [Johnson 1988].

3. Trabalhos Relacionados

Esta seção apresenta os trabalhos relacionados identificados na literatura sobre a avaliação de desempenho e a disponibilidade de ambientes de computação em nuvem. Em [Matos et al. 2020], os autores propõem uma metodologia para detectar gargalos em sistemas de computação em nuvem utilizando modelos hierárquicos. Por meio da combinação

hierárquica de modelos baseados em SPN e Cadeia de Markov de tempo contínuo (CTMC), foi possível realizar uma análise de sensibilidade e classificar os parâmetros do sistema de acordo com seu impacto nas métricas de interesse. Através de um estudo de caso que avaliou serviços web, os autores concluíram que o ranqueamento dos parâmetros permite a identificação eficiente de gargalos.

Em [Andrade et al. 2021], é apresentada uma abordagem baseada em Redes de Petri Estocásticas e Determinísticas para a avaliação de desempenho de ambientes de *Internet* das Coisas (IoT) que adotam dispositivos de *Fog*. Através dos modelos propostos, os usuários são capazes de analisar as compensações entre diferentes configurações desses ambientes *Fog-Cloud* IoT. No estudo de caso realizado, os autores demonstraram que o uso de um dispositivo de *Fog* melhora a disponibilidade e apresenta melhores resultados do que um ambiente de nuvem, especialmente quando o nível de requisições não se aproxima da capacidade máxima de limite.

Em [Santos et al. 2021], os autores propõem a avaliação de um sistema distribuído composto por três camadas: IoT, *edge* e *cloud*. Para isso, é criado um modelo em SPN, com uma representação configurável do sistema, que pode ser reutilizado em diferentes cenários. Para avaliação foi realizada uma análise de sensibilidade, a qual visou a identificação dos componentes que mais impactam a arquitetura estudada. A partir dessa análise, os autores identificaram que o MTTF possui maior impacto na disponibilidade do sistema, e que seu aumento acarreta em melhores resultados para a vazão do serviço.

Em [Rodrigues et al. 2021], os autores adotam modelos analíticos para avaliar a disponibilidade e o desempenho de sistemas de hospitais inteligentes, com o objetivo de evitar gastos antecipados com equipamentos reais. Para representar a arquitetura do sistema, foram propostos dois modelos em SPN, um voltado para a avaliação da disponibilidade e outro para o desempenho. Nos experimentos realizados, os autores identificaram que a taxa de chegada de clientes impacta diretamente no desempenho do sistema, e que a presença de redundância no sistema resulta em maior disponibilidade.

Os autores em [Melo et al. 2022] apresentam a avaliação de um ambiente privado que hospeda uma aplicação baseada em *blockchain*, utilizando métricas como disponibilidade, tempo de inatividade (*downtime*), latência e vazão. Para isso, foi proposto um modelo baseado em SPN para a avaliação combinada de disponibilidade e desempenho. Ao analisarem as métricas de desempenho, os autores observaram um aumento no consumo de recursos, o que poderia indicar um problema relacionado ao envelhecimento do software. Considerando esse aumento no consumo dentro do modelo combinado, os autores identificaram uma perda superior a 1% na disponibilidade.

Os autores em [Rolim and Sousa 2024] propõem a avaliação da disponibilidade e do desempenho de ambientes de *big data* em nuvens privadas. A abordagem adotada faz uso de modelos estocásticos e combinatórios para a análise dos ambientes estudados. Nesse contexto, um modelo em SPN foi utilizado para avaliar o desempenho, enquanto um modelo em RBD foi empregado para avaliar a disponibilidade. Nos experimentos realizados, os autores identificaram que a carga de trabalho é o fator com maior impacto nas aplicações de *big data* em nuvens privadas.

A Tabela 1 apresenta uma comparação entre este trabalho e estudos similares encontrados na literatura. Apesar de [Andrade et al. 2021] e [Rodrigues et al. 2021] anali-

Tabela 1. Trabalhos Relacionados

Artigos	Métricas	Modelagem	Análise de Sensibilidade
[Matos et al. 2020]	Tempo de Resposta, Tamanho da Fila, Uso de VMs	SPN, CTMC	sim
[Andrade et al. 2021]	Disponibilidade, Vazão, Tempo de Resposta	SPN	não
[Santos et al. 2021]	Disponibilidade, Vazão, Tempo de Resposta	SPN	sim
[Rodrigues et al. 2021]	Disponibilidade, Downtime, Tempo de Resposta	SPN	não
[Melo et al. 2022]	Disponibilidade, Downtime, Vazão, Latência	SPN	não
[Rolim and Sousa 2024]	Disponibilidade, Tempo de Execução, Uso de CPU	SPN, RBD	não
Este trabalho	Disponibilidade, Vazão, Tempo de Resposta	SPN	sim

sarem tanto a disponibilidade como o desempenho, este trabalho se distingue por adotar uma abordagem que integra a modelagem de disponibilidade impactando diretamente no desempenho do sistema analisado. Este trabalho possui semelhanças a [Matos et al. 2020] e [Santos et al. 2021] que também fazem uma análise de sensibilidade, mas se diferencia destes por aplicar a avaliação em um serviço hospedado em nuvem privada.

4. Ambiente de Experimentos

Esta seção descreve a arquitetura do ambiente configurado para a realização dos experimentos. A Figura 3 ilustra essa arquitetura, que foi baseada em [Callou and Vieira 2024], composta por um servidor que integra o poder computacional de duas máquinas físicas Dell (Host1 e Host2), ambas com as seguintes especificações: processador Intel Core i5-10400F de 10ª geração, 8 GB de memória RAM, 500 GB de armazenamento e sistema operacional CentOS 7. O Host1 foi configurado como host principal, hospedando o controlador do CloudStack, responsável pelo gerenciamento de recursos do sistema, e suas VMs de sistema (SSVM e CPVM).

Uma máquina física HP, com as seguintes configurações: processador AMD A8-550B de 3,2 GHz com 4 núcleos, 8 GB de memória RAM, 500 GB de armazenamento e sistema operacional Windows 10, foi utilizada para representar o cliente que acessa os serviços hospedados no servidor. Na máquina cliente, foi instalado o Apache JMeter, uma ferramenta para geração de carga, responsável por enviar requisições ao servidor. Um switch interconecta os hosts do servidor, o cliente e a *Internet*.

Foram criadas três VMs no Host2 para simular um serviço em nuvem, todas com a mesma configuração (processador com 1 core de 0,5 GHz, 512 MB de memória RAM e 20 GB de armazenamento). O Nextcloud, um servidor de arquivos, foi instalado em duas dessas VMs (VM1 e VM2), enquanto a terceira VM (LB) recebeu o Nginx que foi configurado para balancear a carga recebida entre o serviço ofertado pela VM1 e VM2.

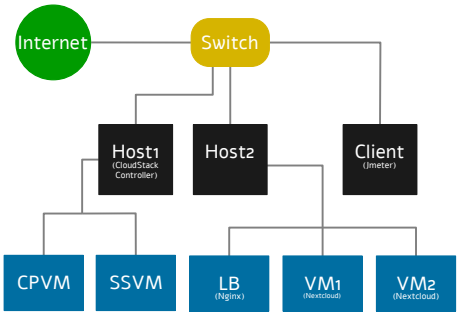

Figura 3. Arquitetura do Ambiente de Testes

Para avaliar o serviço hospedado no ambiente experimental, foi desenvolvido um plano de teste cuja função é gerar a carga de trabalho para simular o processo de acesso de usuários reais ao sistema. Foram definidas cinco atividade que os usuários fariam no sistema: acessar o serviço,fazer *login*, acessar os arquivos, fazer *upload* e fazer *logout*. Dessa forma, medições foram realizadas no ambiente real, sendo repetidas ao menos 30 vezes e os resultados da realização dessa operações é detalhado na Seção 6.

5. Modelo

A Figura 4 apresenta o modelo proposto em ([Leonardo et al. 2024]) que representa o ambiente descrito na Seção 4, e que será utilizado no estudo de caso na Seção 6.

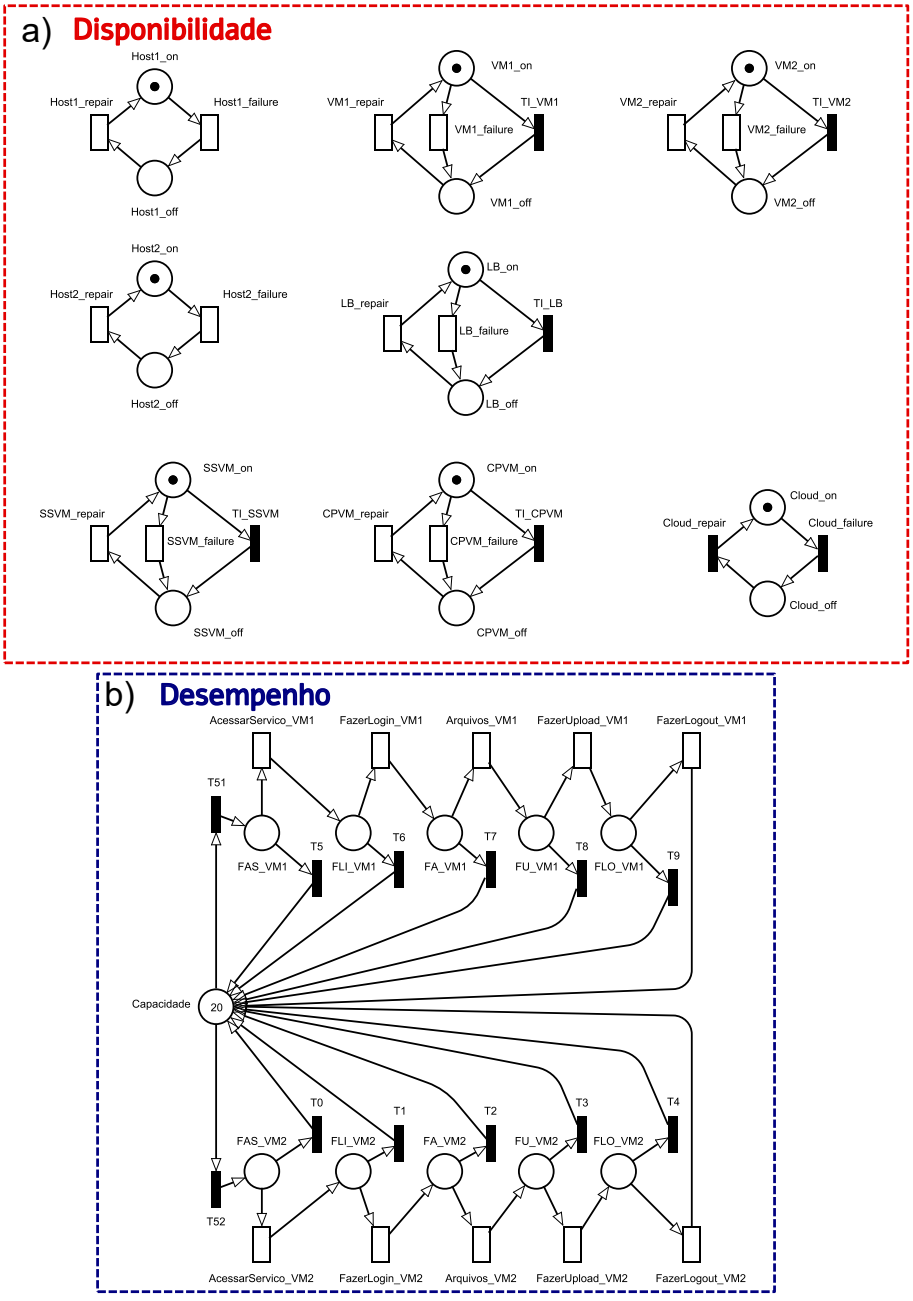

Figura 4. Modelo Proposto

Tabela 2. Valores Transições (Modelo Disponibilidade)

Componente	MTTF(h)	MTTR(h)
Host1, Host2	1259,0	0,768
SSVM, CPVM	619,56	0,05
VM1, VM2, LB	619,56	0,84

Tabela 3. Valores Transições (Modelo Desempenho)

Atividade	Tempo(h)
AcessarServico	0,00073
FazerLogin	0,00258
Arquivos	0,00064
Upload	0,00091
FazerLogoff	0,00078

5.1. Modelo de Disponibilidade

A Figura 4(a)) mostra o modelo de disponibilidade que aborda os componentes do sistema apresentado na Figura 3, representando suas falhas e reparos, a fim de avaliar a disponibilidade dos serviços e a resiliência do sistema diante de falhas. Devido as VMs estarem hospedadas em um dos hosts, elas possuem uma transição imediata que representa sua falha quando o host falhar. A Tabela 2 apresenta os valores adotados nas transições temporizadas *_failure e *_repair, que representam o tempo médio de falha (MTTF) e o tempo médio de reparo (MTTR), obtidos a partir de [Valentim et al. 2024] [Silva and Callou 2025].

Nesse modelo, o sistema é considerado operacional quando o lugar *Cloud_on* contém um token, e é considerado em falha quando o lugar *Cloud_off* possui um token. Dessa forma, a disponibilidade é computada pela expressão $P\{\#Cloud\_on = 1\}$, representando a probabilidade de termos um token no lugar *Cloud_on*. Além disso, é importante destacar que as transições imediatas associadas a esses lugares possuem as seguintes expressões de guarda, *Cloud_failure*: $((\#SSVM\_off > 0)OR(\#CPVM\_off > 0))OR((\#LB\_off > 0)OR((\#VM1\_off > 0)AND(\#VM2\_off > 0)))$; *Cloud_repair*: $((\#SSVM\_on > 0)AND(\#CPVM\_on > 0))AND((\#LB\_on > 0)AND((\#VM1\_on > 0)OR(\#VM2\_on > 0)))$.

5.2. Modelo de Desempenho

A Figura 4(b) apresenta o modelo que representa as atividades que o serviço realiza no ambiente. Esse modelo também quantifica o impacto da disponibilidade no desempenho do sistema representado pelas transições imediatas, sendo *T0* a *T4* utilizadas para interromper as atividades quando o serviço da *VM2* falhar e *T5* a *T9* fazem o mesmo, mas em relação a *VM1*. O lugar *Capacidade* representa o máximo de usuários simultâneos que o sistema pode atender. A partir deste lugar, são definidas duas transições imediatas (T51 e T52), com peso de 0.5 cada, as quais simbolizam a distribuição de carga realizada pelo balanceador de carga entre as VMs com o Nextcloud disponíveis. Os lugares *FAS_VMn*, *FLI_VMn*, *FA_VMn*, *FU_VMn* e *FLO_VMn* representam as filas de clientes em cada atividade e as transições temporizadas correspondem ao tempo necessário para realizá-las (Tabela 3).

A partir deste modelo, pode-se computar métricas de desempenho como o tempo de resposta e a vazão. A vazão é dada pela Equação 3, que é soma das esperanças de haver

tokens nas filas da última atividade de ambas as VMs (lugares *FLO_VMn*). O tempo de resposta é calculado através da Equação 4, formada pleo somatório das esperanças de todas as filas das atividades (lugares *FAS_VMn*, *FLI_VMn*, *FA_VMn*, *FU_VMn* e *FLO_VMn*) de ambas as VMs, dividido pela vazão.

$$V = ((E\{\#FLO\_VM1\}) * (\frac{1}{2,805})) + ((E\{\#FLO\_VM2\}) * (\frac{1}{2,805})) \tag{3}$$

onde E{#nome do lugar} vai computar a esperança de tokens nesse lugar, ou seja, vai retornar o número médio de tokens esperados nesse lugar.

$$TR = \frac{\begin{array}{c}((\text{E}\{\#\text{FAS\_VM1}\})+(\text{E}\{\#\text{FLI\_VM1}\})+(\text{E}\{\#\text{FA\_VM1}\})+(\text{E}\{\#\text{FU\_VM1}\})+(\text{E}\{\#\text{FLO\_VM1}\})+ \\ (\text{E}\{\#\text{FAS\_VM2}\})+(\text{E}\{\#\text{FLI\_VM2}\})+(\text{E}\{\#\text{FA\_VM2}\})+(\text{E}\{\#\text{FU\_VM2}\})+(\text{E}\{\#\text{FLO\_VM2}\}))\end{array}}{V} \tag{4}$$

6. Estudo de Caso

Este estudo de caso tem como objetivo avaliar a disponibilidade e o desempenho do sistema modelado na Figura 4, utilizando uma abordagem fundamentada no Projeto de Experimentos (DoE – Design of Experiments). Para atingir esse objetivo, optou-se pela aplicação do design fatorial de dois níveis (2^k), que contempla a análise de dois níveis distintos para cada variável, sendo um nível elevado e outro reduzido.

Este estudo busca analisar a influência da falha de cada componente nas métricas do sistema. Para tanto, os MTTFs dos componentes do sistema foram definidos como os fatores a serem manipulados nos experimentos. O nível mais baixo de cada fator corresponde aos valores padrão de tempo de falha adotados para os componentes, conforme especificado na Tabela 2. O nível elevado foi definido como uma melhoria de 50% em relação a esses valores, ou seja, um aumento de 50% nos tempos de falha.

A Tabela 4 apresenta os fatores adotados, sendo que a coluna "Nível 1 (baixo)" contém os valores padrão (MTTFs) dos fatores observados no sistema, enquanto a coluna "Nível 2 (alto)" reflete um acréscimo de 50% sobre os valores do Nível 1. Para isolar o impacto das falhas dos componentes, os tempos de reparo (MTTRs) foram mantidos constantes em todos os tratamentos experimentais.

Tabela 4. Fatores e Níveis

Fator	Nível 1	Nível 2
MTTF Host 1	1259	1888,5
MTTF Host 2	1259	1888,5
MTTF Balanceador	619,56	929,34
MTTF VMs Nextcloud	619,56	929,34

Para cada tratamento (combinação específica dos níveis dos fatores) é gerada uma versão distinta do modelo apresentado na Figura 4. Cada versão do modelo contém os valores dos MTTFs configurados conforme a combinação de níveis do respectivo tratamento, conforme detalhado na Tabela 5. Considerando que a VM2 é uma redundância idêntica à VM1, ambas serão tratadas como um único fator (VMs Nextcloud), com seus MTTFs sendo ajustados conjuntamente durante os tratamentos experimentais.

O Tratamento 1 representa o sistema conforme sua configuração atual e serve como referência para a comparação com os resultados obtidos nos demais tratamentos. Como era esperado, o aumento nos MTTFs resulta em uma melhoria nas métricas avaliadas. O Tratamento 16, no qual todos os fatores estão configurados no nível 2, obteve os melhores resultados. A Tabela 5 apresenta os resultados de disponibilidade, vazão e tempo de resposta alcançados em cada um dos tratamentos.

Tabela 5. Resultados DoE

	Fatores				Resultados		
caso	VMs Nextcloud	LB	Host2	Host1	Disp	Vazão	T_Resp
1	619,56	619,56	1259	1259	2,4271	0,921	21,644
2	929,34	619,56	1259	1259	2,4750	0,92999	21,4468
3	619,56	929,34	1259	1259	2,5017	0,92835	21,4782
4	929,34	929,34	1259	1259	2,5086	0,93399	21,3487
5	619,56	619,56	1888,5	1259	2,5258	0,92592	21,5378
6	929,34	619,56	1888,5	1259	2,5214	0,93919	21,2314
7	619,56	929,34	1888,5	1259	2,5834	0,92637	21,5338
8	929,34	929,34	1888,5	1259	2,5952	0,93569	21,3216
9	619,56	619,56	1259	1888,5	2,4295	0,9303	21,421
10	929,34	619,56	1259	1888,5	2,4437	0,93026	21,423
11	619,56	929,34	1259	1888,5	2,5200	0,93351	21,361
12	929,34	929,34	1259	1888,5	2,5719	0,93954	21,231
13	619,56	619,56	1888,5	1888,5	2,5719	0,9305	21,4373
14	929,34	619,56	1888,5	1888,5	2,5361	0,93852	21,2493
15	619,56	929,34	1888,5	1888,5	2,6655	0,93452	21,3573
16	929,34	929,34	1888,5	1888,5	2,7011	0,93795	21,2828

Note: Disponibilidade em 9s, Vazão em req/s, Tempo de Resposta em s.

Após ordenar os efeitos obtidos por meio do design fatorial de forma decrescente, é possível identificar qual componente ou combinação de componentes exerce o maior impacto positivo quando há uma alteração no valor de seu MTTF. A Tabela 6 apresenta a classificação dos efeitos conforme a métrica de disponibilidade. Nesta classificação, o MTTF do balanceador de carga (LB) demonstrou o maior impacto, com um valor de 0,000685, seguido pelo Host2, com um efeito de 0,000592. Assim, o LB configura-se como o componente cuja falha provoca o maior impacto na disponibilidade do sistema.

Tabela 6. Classificação de efeitos (Disponibilidade)

Fator/Iteração	Efeito
LB_MTTF	0,000685
Host2_MTTF	0,000592
Serv_VMs_MTTF	0,000215
Host2_MTTF*Serv_VMs_MTTF	0,000172

A Tabela 7 exibe a classificação dos efeitos segundo a métrica de vazão. De acordo com essa classificação, o MTTF do Host1 foi identificado como o mais significativo, apresentando um efeito de 0,006832, seguido pelo MTTF das VMs Nextcloud, com um efeito de 0,004325. Esses resultados indicam que a falha do Host1, que hospeda tanto o controlador na nuvem quanto as VMs de sistema, tem o maior impacto na vazão.

A Tabela 8 apresenta a classificação dos efeitos conforme a métrica de tempo de resposta. Nesta classificação, a maior influência se trata da combinação dos componentes Host2 e LB, com um efeito de 0,06945, já o segundo maior impacto vem da combinação dos componentes Host1 e VMs Nextcloud, com um valor de 0,05685. Isso significa que a falha dos componentes Host2 e LB tem o maior impacto no tempo de resposta do serviço.

Tabela 7. Classificação de efeitos (Vazão)

Fator/Iteração	Efeito
Host1_MTTF	0,006832
Serv_VMs_MTTF	0,004325
Host2_MTTF	0,00303
LB_MTTF	0,002715

Tabela 8. Classificação de efeitos (Tempo de Resposta)

Fator/Iteração	Efeito
Host2_MTTF*LB_MTTF	0,06945
Host1_MTTF*Serv_VMs_MTTF	0,05685
Host1_MTTF*Host2_MTTF*LB_MTTF	0,034
Host1_MTTF*Host2_MTTF*LB_MTTF*Serv_VMs_MTTF	0,02737

Ao cruzar os dados das tabelas de classificação dos efeitos apresentadas, é possível determinar a importância de cada fator (componente) utilizado nos experimentos. Para as métricas de disponibilidade e tempo de resposta, o balanceador de carga (LB) se destaca como o componente mais relevante, seguido pelo Host2. Dessa forma, ao utilizar versões desses componentes com maior MTTF, é possível alcançar melhores resultados. Por outro lado, para a métrica de vazão, a utilização de versões, com maior MTTF, do Host1 e das VMs com o Nextcloud gera um impacto positivo mais significativo.

7. Conclusão

Este trabalho propôs uma análise de sensibilidade para a avaliação de desempenho e disponibilidade do serviço Nextcloud hospedado em um ambiente de nuvem privada Apache CloudStack. A abordagem adotada utiliza SPN para avaliar a arquitetura proposta. O estudo utilizou o projeto de experimentos (DoE) para avaliar a relevância de um componente, considerando uma melhoria em seu tempo médio de falha (MTTR), identificando que o balanceador de carga tem o maior impacto tanto na disponibilidade quanto no desempenho do sistema. Como trabalhos futuros, pensa-se em analisar o consumo energético do sistema.

Agradecimentos

Os autores agradecem à FACEPE pelo apoio financeiro a essa pesquisa.

Referências

Andrade, E., Nogueira, B., Farias Júnior, I. d., and Araújo, D. (2021). Performance and availability trade-offs in fog–cloud iot environments. *Journal of Network and Systems Management*, 29:1–27.

Callou, G. and Vieira, M. (2024). Availability and performance analysis of cloud services. In *Proceedings of the 13th Latin-American Symposium on Dependable and Secure Computing*, LADC '24, page 262–271, New York, NY, USA. Association for Computing Machinery.

CETIC.br (2023). Tic domicílios 2023: Pesquisa sobre o uso das tecnologias de informação e comunicação nos domicílios brasileiros. Technical report, CETIC.br. Accessed: 2025-03-31.

Dantas, J., Araujo, E., Maciel, P., Matos, R., and Teixeira, J. (2020). Estimating capacity-oriented availability in cloud systems. *International Journal of Computational Science and Engineering*, 22(4):466–476.

Frank, P. M. and Eslami, M. (1980). Introduction to system sensitivity theory. *IEEE Transactions on Systems, Man, and Cybernetics*, 10(6):337–338.

Haas, P. J. (2006). *Stochastic petri nets: Modelling, stability, simulation*. Springer Science & Business Media.

Johnson, B. W. (1988). *Design & analysis of fault tolerant digital systems*. Addison-Wesley Longman Publishing Co., Inc.

Leonardo, W., Bezerra, T., and Callou, G. (2024). Stochastic petri net models for availability and performance evaluation of nextcloud service hosted in apache cloudstack. In *Proceedings of the 13th Latin-American Symposium on Dependable and Secure Computing*, pages 165–170.

Maciel, P., Trivedi, K., and Kim, D. (2010). Dependability modeling in: Performance and dependability in service computing: Concepts, techniques and research directions. *Hershey: IGI Global, Pennsylvania, USA*, 13:1380.

Matos, R., Dantas, J., Araujo, E., and Maciel, P. (2020). Bottleneck detection in cloud computing performance and dependability: Sensitivity rankings for hierarchical models. *Journal of Network and Systems Management*, 28(4):1839–1871.

Melo, C., Oliveira, F., Dantas, J., Araujo, J., Pereira, P., Maciel, R., and Maciel, P. (2022). Performance and availability evaluation of the blockchain platform hyperledger fabric. *The Journal of Supercomputing*, 78(10):12505–12527.

Montgomery, D. C. and Runger, G. C. (2010). *Applied statistics and probability for engineers*. John wiley & sons.

Murata, T. (1989). Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580.

Rodrigues, L., Gonçalves, I., Fé, I., Endo, P. T., and Silva, F. A. (2021). Performance and availability evaluation of an smart hospital architecture. *Computing*, 103:2401–2435.

Rolim, T. and Sousa, E. (2024). Performance and availability evaluation of big data environments in the private cloud. *Journal of Computer and Communications*, 12(12):266–288.

Santos, L., Cunha, B., Fé, I., Vieira, M., and Silva, F. A. (2021). Data processing on edge and cloud: a performability evaluation and sensitivity analysis. *Journal of Network and Systems Management*, 29(3):27.

Silva, A. and Callou, G. (2025). Models for availability evaluation of file servers in private clouds. *Computing*, 107(1):1–27.

Valentim, T., Callou, G., França, C., and Tavares, E. (2024). Availability and performance assessment of iomt systems: A stochastic modeling approach. *Journal of Network and Systems Management*, 32(4):95.