

PCS: Private Cloud Services

Licenciatura em Engenharia Informática

João Martins Tendeiro

Miguel Francisco Lopes

Leiria, junho de 2025

PCS: Private Cloud Services

Licenciatura em Engenharia Informática

João Martins Tendeiro

Miguel Francisco Lopes

Trabalho de Projeto da unidade curricular de Projeto Informático realizado sob a orientação do Professor Doutor António Pereira, Professor Doutor Daniel Fuentes, Professor Doutor David Safadinho, Professor Doutor João Ramos e Professor Doutor Luís Frazão,

Leiria, junho de 2025

Agradecimentos

Gostaríamos de dirigir os nossos sinceros agradecimentos aos nossos orientadores, António Pereira, Daniel Fuentes, David Safadinho, João Ramos e Luís Frazão, por todo o apoio, disponibilidade e troca de conhecimentos ao longo do projeto.

Agradecemos também às nossas famílias, amigos e colegas pelo apoio e incentivo durante todo o percurso.

Resumo

A crescente necessidade de controlo sobre os dados e os recursos computacionais tem levado muitas organizações a optar por soluções de computação em nuvem privada. Este trabalho apresenta o estudo da implementação de uma infraestrutura de *Private Cloud* adaptada a contextos restritos, com foco na virtualização de serviços, armazenamento distribuído e gestão eficiente de recursos.

A solução desenvolvida integra virtualização com o Proxmox *Virtual Environment* (VE) e armazenamento distribuído com o *Ceph*, permitindo a gestão eficiente de máquinas virtuais, *containers* e dados em ambientes com recursos limitados. O principal objetivo foi conceber, implementar e avaliar uma infraestrutura de *Private Cloud*, com foco na fiabilidade, escalabilidade e tolerância a falhas.

Foram realizados diversos testes de desempenho, resiliência e de escalabilidade, que demonstraram a robustez da solução mesmo perante falhas de disco ou de nós (servidores). Os resultados obtidos confirmam que uma *Private Cloud* bem estruturada pode constituir uma alternativa sólida às *clouds* públicas, assegurando maior controlo e flexibilidade, sem comprometer o desempenho.

Entre as principais vantagens da solução desenvolvida destacam-se o controlo total sobre os dados e a infraestrutura, a aplicação de políticas de segurança personalizadas e a gestão centralizada de recursos através de uma interface web intuitiva. A segurança é reforçada pelo isolamento entre instâncias e pela flexibilidade na definição de acessos. A utilização do *Ceph* assegura replicação e tolerância a falhas, enquanto o Proxmox VE permite escalar a infraestrutura de forma eficiente, com suporte a alta disponibilidade e migração ao vivo.

Este trabalho contribui com um modelo técnico flexível e adaptável a diferentes contextos organizacionais, permitindo que entidades com necessidades específicas possam configurar e gerir internamente a sua própria infraestrutura de computação, de acordo com os seus requisitos operacionais.

Palavras-chave: computação em nuvem, *Private Cloud*, virtualização, armazenamento distri-

buído, fiabilidade, escalabilidade

Abstract

The growing need for control over data and computing resources has led many organizations to opt for private cloud computing solutions. This paper presents a study on the implementation of a Private Cloud infrastructure tailored to restricted contexts, with a focus on service virtualization, distributed storage, and efficient resource management.

The developed solution integrates virtualization with Proxmox Virtual Environment (VE) and distributed storage with Ceph, allowing efficient management of virtual machines, containers, and data in environments with limited resources. The main objective was to design, implement, and evaluate a Private Cloud infrastructure, with a focus on reliability, scalability, and fault tolerance.

Several performance, resilience, and scalability tests were conducted, demonstrating the robustness of the solution even in the face of disk or node (server) failures. The results obtained confirm that a well-structured Private Cloud can be a solid alternative to public clouds, ensuring greater control and flexibility without compromising performance.

The main advantages of the developed solution include full control over data and infrastructure, the application of customized security policies, and centralized resource management through an intuitive web interface. Security is enhanced by isolation between instances and flexibility in defining access. The use of Ceph ensures replication and fault tolerance, while Proxmox enables efficient infrastructure scaling, with support for high availability and live migration.

This work contributes a flexible technical model that can be adapted to different organizational contexts, allowing entities with specific needs to configure and manage their computing infrastructure internally, according to their operational requirements.

Keywords: cloud computing, Private Cloud, virtualization, distributed storage, reliability, scalability

Índice

Agradecimentos	i
Resumo	ii
Abstract	iv
Lista de Figuras	xvii
Lista de Tabelas	xviii
Lista de siglas e acrónimos	xviii
Lista de siglas e acrónimos	xviii
1 Introdução	1
2 Estado da arte	3
2.1 Conceitos	3
2.1.1 <i>Data Center</i>	3
2.1.2 Virtualização	4
2.1.3 Hipervisor	4
2.1.4 Virtual Machine (VM)	6
2.1.5 Container	6
2.1.6 <i>Cluster</i>	8
2.1.7 <i>Cloud</i>	8
2.1.8 Armazenamento	10
2.1.9 Anything-as-a-Service (XaaS)	11
2.1.10 Alta Disponibilidade	12
2.1.11 <i>Failover</i>	12
2.1.12 <i>Fault Tolerance</i>	12

2.1.13	Backup	13
2.1.14	Snapshot	13
2.1.15	Migração	13
2.1.16	<i>Thin Provisioning</i>	14
2.2	Hipervisor	14
2.2.1	VMware ESXi + vSphere	14
2.2.2	Proxmox VE	16
2.2.3	Microsoft Hyper-V	17
2.2.4	XenServer (Citrix Hypervisor)	19
2.2.5	Linux KVM	20
2.2.6	Comparação de Hipervisores	21
2.2.7	Escolha do Hipervisor	23
2.3	Tipos de Armazenamento do Proxmox	24
2.3.1	LVM	24
2.3.2	LVM-Thin	26
2.3.3	Diretoria	28
2.3.4	ZFS	29
2.3.5	iSCSI	33
2.3.6	NFS	35
2.3.7	Ceph	37
2.3.8	Comparação do Armazenamento	44
2.4	Soluções Existentes de <i>Private Cloud</i>	45
2.4.1	<i>The practice of developing the academic cloud using the Proxmox</i>	45
2.4.2	<i>Educational Resource Private Cloud Platform Based on OpenStack</i>	46
2.4.3	<i>Implementation - Surplus resources to Private Cloud</i>	47
2.4.4	<i>Implementation – OpenStack Multinode Deployment for Private Cloud</i>	48
2.4.5	<i>Implementation – JFE Steel Private Cloud Deployment with OpenStack</i>	49

2.5	Síntese	50
3	Private Cloud Services	51
3.1	Requisitos	51
3.1.1	Requisitos Funcionais	51
3.1.2	Requisitos Não Funcionais	52
3.2	Análise de Problemas e Abordagens	53
3.3	Proposta de Arquitetura da Solução	55
3.4	Discussão	56
3.5	Módulos da Solução	56
3.5.1	Armazenamento	56
3.5.2	Nós de Cálculo	57
3.5.3	Infraestrutura de IA	58
3.5.4	Rede	58
3.5.5	Alimentação	58
3.5.6	Backups	58
3.5.7	Monitorização	59
3.6	Síntese	59
4	Implementação	60
4.1	Objetivos da Implementação	60
4.2	Descrição da implementação	61
4.2.1	<i>Hardware</i>	61
4.2.2	<i>Software</i>	61
4.2.3	Armazenamento	62
4.2.4	Rede	62
4.3	Arquitetura de Implementação	62
4.4	Instalação do Proxmox VE em cada servidor	63

4.5	Configuração	64
4.6	Criação do <i>cluster</i>	65
4.7	Configuração do Armazenamento	65
4.7.1	Configuração do Ceph	67
4.8	Virtual Machines e Containers	69
4.8.1	Importação de ISOs e <i>templates</i> LXC	69
4.8.2	Criação de VMs	70
4.8.3	Criação de CTs	70
4.8.4	Criação a partir de <i>Templates</i> Base	71
4.9	<i>Templates</i>	71
4.9.1	Criação de <i>Templates</i> :	72
4.9.2	Utilização de <i>Templates</i> pré-configurados	72
4.10	Alta Disponibilidade (HA)	72
4.10.1	Política de encerramento de nó	73
4.10.2	Configuração do HA	74
4.10.3	Requisitos Técnicos	74
4.11	<i>Live Migration</i>	75
4.11.1	Requisitos/Funcionamento da <i>Live Migration</i>	75
4.11.2	Arquitetura de uma <i>Live Migration</i> de uma VM	75
4.11.3	<i>Live Migration</i> de CTs	76
4.11.4	Migração com Armazenamento Local	77
4.11.5	Diferença entre <i>Live Migration</i> e Alta Disponibilidade	77
4.12	Snapshots	78
4.12.1	Funcionamento dos Snapshots	78
4.12.2	Processo de Criação	78
4.13	Backups	79
4.13.1	Backups Nativos do Proxmox	79

4.13.2	Proxmox Backup Server	79
4.13.3	Processo de Criação	80
4.14	Alertas e Notificações	81
4.14.1	Configuração do SMTP para alertas por email no Proxmox VE	81
4.14.2	Alertas no Ceph com Prometheus e Alertmanager	82
4.15	Monitorização	83
4.15.1	Configuração do InfluxDB e o Grafana para métricas do Proxmox VE	83
4.15.2	Configuração do Telegraf para recolha de métricas do Ceph	85
4.16	Síntese	87
5	Testes	89
5.1	Teste 1 – HA com shutdown_policy=migrate e shutdown_policy=failover	91
5.1.1	Objetivo:	92
5.1.2	Procedimento:	92
5.1.3	Resultado:	92
5.1.4	Discussão:	93
5.2	Teste 2 – Migração em tempo real de CTs	93
5.2.1	Objetivo:	94
5.2.2	Procedimento:	94
5.2.3	Resultados:	94
5.2.4	Discussão:	96
5.3	Teste 3 – Migração em tempo real de VMs	97
5.3.1	Objetivo:	97
5.3.2	Procedimento:	97
5.3.3	Resultados:	97
5.3.4	Discussão:	99
5.4	Teste 4 – Disco cheio	100
5.4.1	Objetivo:	100

5.4.2	Procedimento:	100
5.4.3	Resultado:	101
5.4.4	Discussão:	104
5.5	Teste 5 – Falha de disco	105
5.5.1	Objetivo:	105
5.5.2	Procedimento:	105
5.5.3	Resultado:	106
5.5.4	Discussão:	107
5.6	Teste 6 – Falha de dois discos com $min_size = 2$	108
5.6.1	Objetivo:	108
5.6.2	Procedimento:	108
5.6.3	Resultado:	108
5.6.4	Discussão:	113
5.7	Teste 7 – Falha de dois discos com $min_size = 1$	113
5.7.1	Objetivo:	113
5.7.2	Procedimento:	113
5.7.3	Resultado:	113
5.7.4	Discussão:	114
5.8	Teste 8 – Adicionar um novo disco	115
5.8.1	Objetivo:	115
5.8.2	Procedimento:	115
5.8.3	Resultado:	116
5.8.4	Discussão:	116
5.9	Teste 9 – Migração do Sistema Operativo Proxmox VE para um Novo Disco	117
5.9.1	Objetivo:	117
5.9.2	Procedimento:	117
5.9.3	Resultado:	117

5.9.4	Discussão:	118
5.10	Teste 10 – Teste de adição de OSD e restabelecimento de réplicas no Ceph	118
5.10.1	Objetivo:	119
5.10.2	Procedimento:	119
5.10.3	Resultado:	119
5.10.4	Discussão:	121
5.11	Teste 11 – Escrita intensiva no Ceph ao nível RADOS com 2 e 3 réplicas	122
5.11.1	Objetivo:	122
5.11.2	Procedimento:	122
5.11.3	Análise:	123
5.11.4	Resultado:	125
5.11.5	Discussão:	126
5.12	Teste 12 – Leitura intensiva no Ceph ao nível RADOS com 2 e 3 réplicas	127
5.12.1	Objetivo:	127
5.12.2	Procedimento:	127
5.12.3	Análise:	127
5.12.4	Resultado:	130
5.12.5	Discussão:	130
5.13	Teste 13 – Escrita sequencial no Ceph RBD com 2 e 3 réplicas	131
5.13.1	Objetivo:	131
5.13.2	Procedimento:	131
5.13.3	Análise:	132
5.13.4	Resultado:	134
5.13.5	Discussão:	135
5.14	Teste 14 – Leitura sequencial no Ceph RBD com 2 e 3 réplicas	135
5.14.1	Objetivo:	136
5.14.2	Procedimento:	136

5.14.3	Análise:	136
5.14.4	Resultado:	138
5.14.5	Discussão:	139
5.15	Teste 15 – Escrita Sequencial no CephFS com 2 e 3 réplicas	139
5.15.1	Objetivo:	139
5.15.2	Procedimento:	139
5.15.3	Análise:	140
5.15.4	Resultado:	142
5.15.5	Discussão:	143
5.16	Teste 16 – Leitura Sequencial no CephFS com 2 e 3 réplicas	143
5.16.1	Objetivo:	143
5.16.2	Procedimento:	144
5.16.3	Análise:	144
5.16.4	Resultado:	145
5.16.5	Discussão:	146
5.17	Síntese	146
6	Estudo e Especificação de uma Private cloud	148
6.1	Análise Comparativa de Opções de Implementação	148
6.2	Propostas de Implementação	149
6.2.1	Opção A	150
6.2.2	Opção B	152
6.2.3	Opção C	154
6.2.4	Opção D	156
6.3	Análise Comparativa Final e Justificação da Escolha	159
6.4	Síntese	160
7	Conclusão	161

Bibliografia **163**

Anexos **168**

Lista de Figuras

1	Estrutura de um hipervisor tipo 1 e tipo 2 ¹	5
2	Diferença de estrutura de um CT e VM ²	7
3	Arquitetura de uma <i>cloud</i> ³	9
4	Arquitetura do LVM	25
5	Arquitetura do LVM-Thin	27
6	Arquitetura da Diretoria	29
7	Arquitetura do ZFS	31
8	Arquitetura do ZFS over iSCSI	35
9	Arquitetura do NFS	36
10	Arquitetura do Ceph	43
11	Arquitetura da solução proposta	56
12	Arquitetura lógica da Implementação	63
13	Arquitetura do armazenamento inicial do Proxmox	66
14	Arquitetura do armazenamento ceph	69
15	Arquitetura geral do <i>live migration</i>	76
16	Dashboard do Grafana com métricas do Proxmox	85
17	Dashboard do Grafana com métricas do Ceph	87
18	Registo da migração em tempo real de um CT com disco em armazenamento local	95
19	Registo da migração em tempo real de um CT com disco em armazenamento ceph	96
20	Configuração da migração em tempo real de uma VM com disco em armazenamento local	98
21	Registo da migração em tempo real de uma VM com disco em armazenamento local	98
22	Registo da migração em tempo real de uma VM com disco em Ceph	99
23	Estado inicial dos OSDs – antes da escrita	101
24	Gráfico de utilização inicial do volume RBD	102

25	Avisos gerados durante os testes quando OSDs e pools ultrapassaram os 85% de capacidade	102
26	Avisos gerados durante os testes quando OSDs e pools ultrapassaram os 90% de capacidade	103
27	Estado final dos OSDs – após escrita de dados	103
28	Gráfico de utilização do volume RBD – 99,21% ocupado	104
29	Estado do <i>cluster</i> após falha do OSD – <i>down/out</i> e alertas <i>HEALTH_WARN</i>	105
30	Capacidade total reduzida para 434.99 GiB com 2 OSDs ativos	106
31	Estado normalizado após reintegração	107
32	Estado dos OSDs após reinício e reintegração do disco	107
33	Distribuição normalizada dos dados no <i>cluster</i> Ceph	107
34	Estado do armazenamento antes da remoção dos discos	109
35	Exemplo da arquitetura após remoção de um disco	109
36	Estado do <i>cluster</i> após remoção de um OSD	110
37	Exemplo da arquitetura após remoção de dois discos	110
38	Capacidade de armazenamento após remoção de dois discos	111
39	Escrta bloqueada no volume CephFS com min_size=2	111
40	Escrta bem-sucedida no armazenamento local	112
41	Indisponibilidade da GUI do Proxmox após falha do <i>cluster</i> Ceph	112
42	Escrta e leitura testadas manualmente no volume CephFS	114
43	Estado do <i>ceph-osd@0</i> e mensagens de erro	114
44	Exemplo da arquitetura do teste 5	116
45	Novo disco adicionado como <i>osd 3</i>	116
46	Execução do comando dd no proxmox 3	118
47	Estado do <i>cluster</i> antes da adição do novo OSD	119
48	Capacidade total do <i>cluster</i> após a adição do novo OSD	119
49	Estado do <i>cluster</i> durante o rebalanceamento dos dados	120

50	Utilização de CPU durante o rebalanceamento	120
51	Tráfego de rede durante o rebalanceamento	121
52	Estado final do <i>cluster</i> com 3 OSDs	121
53	Distribuição equilibrada dos dados nos OSDs após o rebalanceamento	121
54	Utilização de CPU com 2 réplicas (esq.) e 3 réplicas (dir.)	123
55	Latência dos OSDs com 2 réplicas (esq.) e 3 réplicas (dir.)	123
56	Tráfego de saída com 2 réplicas (esq.) e 3 réplicas (dir.)	124
57	Tráfego de entrada com 2 réplicas (esq.) e 3 réplicas (dir.)	125
58	Escritas no Ceph com 2 réplicas (esq.) e 3 réplicas (dir.)	125
59	Espaço total ocupado na pool <i>testpool2</i> com 2 réplicas	126
60	Espaço total ocupado na pool <i>testpool3</i> com 3 réplicas	126
61	Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)	128
62	Latência dos OSDs com 2 réplicas (esq.) e 3 réplicas (dir.)	128
63	Tráfego de entrada com 2 réplicas (esq.) e 3 réplicas (dir.)	129
64	Tráfego de saída com 2 réplicas (esq.) e 3 réplicas (dir.)	129
65	Leituras do Ceph com 2 réplicas (esq.) e 3 réplicas (dir.)	130
66	Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)	132
67	Tráfego de Entrada com 2 réplicas (esq.) e 3 réplicas (dir.)	132
68	Tráfego de Saída com 2 réplicas (esq.) e 3 réplicas (dir.)	133
69	Latência dos OSDs com 2 réplicas (esq.) e 3 réplicas (dir.)	133
70	Escritas do Ceph com 2 réplicas (esq.) e 3 réplicas (dir.)	134
71	Espaço total ocupado na pool <i>testpool2</i> com 2 réplicas	134
72	Espaço total ocupado na pool <i>testpool3</i> com 3 réplicas	135
73	Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)	137
74	Tráfego de entrada com 2 réplicas (esq.) e 3 réplicas (dir.)	137
75	Tráfego de saída com 2 réplicas (esq.) e 3 réplicas (dir.)	138
76	Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)	140

77	Tráfego de Saída com 2 réplicas (esq.) e 3 réplicas (dir.)	141
78	Tráfego de Entrada com 2 réplicas (esq.) e 3 réplicas (dir.)	141
79	Latência dos OSDs com 2 réplicas (esq.) e 3 réplicas (dir.)	142
80	Escritas do Ceph com 2 réplicas (esq.) e 3 réplicas (dir.)	142
81	Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)	144
82	Tráfego de Entrada com 2 réplicas (esq.) e 3 réplicas (dir.)	145
83	Tráfego de Saída com 2 réplicas (esq.) e 3 réplicas (dir.)	145
84	Arquitetura da Opção A	151
85	Arquitetura da Opção B	153
86	Arquitetura da Opção C	155
87	Arquitetura da Opção D	157

Lista de Tabelas

1	Tipos de Armazenamento Suportados pelo VMware ESXi + vSphere	15
2	Tipos de Armazenamento Suportados pelo Proxmox VE	17
3	Tipos de Armazenamento Suportados pelo Microsoft Hyper-V	18
4	Tipos de Armazenamento Suportados pelo XenServer / Citrix Hypervisor	19
5	Tipos de Armazenamento Suportados pelo KVM em Ambiente Linux	21
6	Comparação de Funcionalidades entre Hipervisores de Tipo I	22
7	Recomendações de <i>hardware</i> para Ceph [1]	38
8	Comparação de Tipos de Armazenamento no Proxmox VE	45
9	Problemas, desafios, abordagens e métricas em ambientes de <i>datacenter/cloud</i> com Proxmox e Ceph	54
10	Especificações de <i>hardware</i> dos servidores Proxmox	61
11	Resumo dos testes realizados	91
12	Resumo das escritas com replicação 2/2 e 3/3	126
13	Resumo das leituras com replicação 2/2 e 3/3	130
14	Resumo do espaço ocupado nas pools com diferentes fatores de replicação	134
15	Comparação das métricas nos testes com replicação 2/2 e 3/3	138
16	Resumo das métricas dos testes de escrita no CephFS	143
17	Comparação dos testes de leitura no CephFS com replicação 2/2 e 3/3	146
18	Comparação entre diferentes opções para implementação da Private Cloud	149
19	Resumo de equipamentos e custos da Opção A	152
20	Resumo de equipamentos e custos da Opção B	154
21	Resumo de equipamentos e custos da Opção C	156
22	Resumo de equipamentos e custos da Opção D	158
23	Comparação final entre as opções de implementação	159

Lista de siglas e acrónimos

AI Artificial Intelligence

API Application Programming Interface

BIOS Basic Input/Output System

btrfs B-tree file system

CephFS Ceph File System

CPU Central Processing Unit

CRUSH Controlled Replication Under Scalable Hashing

CSP Cloud Service Provider

CT Container

DNS Domain Name System

EFI Extensible Firmware Interface

ext4 Extended File System 4

GPT GUID Partition Table

GPU Graphics Processing Unit

HA High Availability

HDD Hard Disk Drive

HTTP Hypertext Transfer Protocol

HTTPS Hypertext Transfer Protocol Secure

IP Internet Protocol

iSCSI Internet Small Computer System Interface

ISO Optical disc image

IT Information technology

KVM Kernel-based Virtual Machine

LUN Logical Unit Number

LV Logical Volume

LVM Logical Volume Manager

LXC Linux Containers

MDS Metadata Server

MGR Manager

ML Machine Learning

MON Monitor

NAS Network-Attached Storage

NAT Network Address Translation

NFS Network File System

NTP Network Time Protocol

OS Operating System

OSD Object Storage Daemon

PBS Proxmox Backup Server

PGs Placement Groups

POSIX Portable Operating System Interface

PV Physical Volume

RADOS Reliable Autonomic Distributed Object Store

RAID Redundant Array of Independent Disks

RAM Random Access Memory

RBD RADOS Block Device

REST Representational State Transfer

RGW RADOS Gateway

SAN Storage Area Network

SCSI Small Computer System Interface

SMART Self-Monitoring, Analysis and Reporting Technology

SMB Server Message Block

SMTP Simple Mail Transfer Protocol

SSD Solid State Drive

TCO Total Cost of Ownership

UEFI Unified Extensible Firmware Interface

UPS Uninterruptible Power Supply

URL Uniform Resource Locator

USB Universal Serial Bus

VE Virtual Environment

VG Volume Group

VLAN Virtual Local Area Network

VM Virtual Machine

VPN Virtual Private Network

XaaS Anything-as-a-Service

xfs X File System

ZFS Zettabyte File System

1 Introdução

As aplicações tradicionais, de instalação local, estão a dar lugar a serviços disponibilizados através da Internet (“*Anything-as-a-Service*”). Estes serviços, também conhecidos como serviços na Nuvem (*Cloud*), trazem vantagens para os utilizadores na medida em que o abstraem completamente da sua instalação, configuração e manutenção. Ao mesmo tempo, otimizam a utilização dos recursos, que são normalmente partilhados entre vários utilizadores e alocados mediante as suas necessidades. No entanto, as soluções na nuvem também apresentam desafios, no que diz respeito à localização da informação (não se encontra sob o controlo dos utilizadores), e da velocidade de acesso aos serviços (os serviços podem encontrar-se a centenas ou milhares de quilómetros, com o respetivo impacto nas comunicações). Para tirar partido das vantagens dos serviços na nuvem, e ao mesmo tempo colmatar os seus desafios, surge a hipótese de criar uma Nuvem Privada (*Private Cloud*), onde a infraestrutura é alojada num local privado, restrito a uma entidade e aos seus colaboradores.

O objetivo deste projeto é conceber, implementar e avaliar uma solução de *Private Cloud*. Esta visa disponibilizar diferentes serviços a um conjunto restrito de utilizadores, assegurando uma gestão centralizada, segura e eficiente da infraestrutura, dos dados e dos recursos computacionais. A solução tem como base o Proxmox VE para a virtualização e o Ceph para uma gestão do armazenamento distribuído, sendo desenhada para suportar cargas de trabalho intensivas, nomeadamente a execução de máquinas virtuais e *containers* orientados a aplicações de Inteligência Artificial, e garantir acesso redundante e fiável aos dados.

Para atingir este objetivo, primeiramente foram estudados os principais desafios associados à computação na nuvem, analisadas as abordagens tecnológicas mais adequadas e identificados os requisitos funcionais e não funcionais da infraestrutura. Tendo como base esse estudo, será definida uma arquitetura técnica detalhada, com especial atenção à forma como a solução proposta poderá resolver ou mitigar potenciais problemas operacionais, falhas de sistema e limitações de desempenho. Esta arquitetura posteriormente implementada num ambiente controlado, com recursos

limitados, foi submetida a um conjunto de testes centrados no desempenho, escalabilidade e tolerância a falhas, permitindo assim avaliar a sua viabilidade prática, robustez, capacidade de adaptação a cenários reais e situações de falha.

Este projeto contribui para demonstrar a viabilidade prática da implementação de uma infraestrutura de *Private Cloud* com recursos limitados, oferecendo uma alternativa mais controlada, segura e personalizável face às soluções de nuvem pública. Com a utilização de virtualização de serviços e com armazenamento distribuído, o projeto permite avaliar, em contexto real, o desempenho, a escalabilidade e a tolerância a falhas de uma nuvem privada. Contribuindo assim para o avanço do conhecimento aplicado na área da computação em nuvem e para apoiar organizações que pretendam adotar soluções mais autónomas e seguras para a gestão dos seus serviços e dados.

Este documento é composto por seis capítulos que descrevem todo o processo de estudo, planeamento, implementação, avaliação de uma solução de *Private Cloud*. O segundo capítulo aborda o estado da arte, introduzindo os conceitos fundamentais para uma melhor compreensão do projeto desenvolvido, bem como uma análise de soluções existentes de *Private Cloud*. O terceiro capítulo propõe a arquitetura da solução desenvolvida, com base nas tecnologias Proxmox VE e Ceph, definindo os requisitos e a estrutura do sistema. O quarto capítulo descreve em detalhe o processo de implementação da infraestrutura, desde a preparação do ambiente físico até à configuração final do *cluster*. No quinto capítulo são apresentados os testes realizados, com o objetivo de avaliar o desempenho, a robustez e a tolerância a falhas da solução. O sexto capítulo apresenta um estudo de diversas soluções de *Private Cloud*, com base em diferentes opções de hardware e orçamento, permitindo selecionar a solução mais viável e eficiente. Por fim, o documento termina com a apresentação das conclusões do trabalho e sugestões para desenvolvimentos futuros.

2 Estado da arte

Este capítulo apresenta o contexto tecnológico relativo à implementação de infraestruturas de *private cloud*, com base em soluções existentes e na análise das ferramentas mais utilizadas.

Irá ser apresentado um estudo dos principais conceitos e componentes relacionados com a virtualização e a gestão de recursos. Em seguida, serão analisados os diferentes tipos de hipervisores, com destaque para suas características técnicas e cenários típicos de aplicação.

Posteriormente, serão descritos os tipos de armazenamento disponíveis no Proxmox VE, fundamentais para a performance e escalabilidade da infraestrutura, sendo igualmente apresentadas as suas diferenças em termos de funcionalidades, desempenho e requisitos de implementação.

Por fim, é realizada uma revisão de soluções reais de *private cloud* já implementadas ou descritas na literatura técnica, com o objetivo de identificar boas práticas, tendências atuais e desafios comuns. Esta análise serve como base para o desenvolvimento de arquiteturas mais robustas e ajustadas às necessidades atuais.

2.1 Conceitos

Nesta secção abordamos alguns conceitos relevantes para o projeto, nomeadamente conceitos relativos a *Data Center*, Virtualização, Hipervisor, Máquina Virtual, *Container*, *Cluster*, *Cloud*, Armazenamento, *Anything-as-a-Service* (XaaS), Alta Disponibilidade, *Failover*, *Fault Tolerance*, Backup, *Snapshot*, Migração e *Thin Provisioning*.

2.1.1 *Data Center*

Um *data center* é uma sala física, edifício ou instalação que contém a infraestrutura de *Information technology* (IT) para criar, executar e disponibilizar aplicações e serviços. Também armazena e gere os dados associados a essas aplicações e serviços.

Os *data centers* começaram por ser instalações locais, de propriedade privada e rigidamente controladas, contendo infraestruturas de IT tradicionais para uso exclusivo de uma empresa. Recentemente, as instalações têm migrado para redes remotas geridas por fornecedores de serviços de *cloud* (Cloud Service Provider - CSP) [2].

2.1.2 Virtualização

A virtualização é um processo que permite a utilização mais eficiente do *hardware* físico do computador e é a base da computação em nuvem (*cloud*).

A virtualização utiliza *software* para criar uma camada de abstração sobre o *hardware* do computador, permitindo a divisão dos componentes de um único computador em várias máquinas virtuais (VMs).

Cada VM executa o seu próprio sistema operativo (OS) e comporta-se como um computador independente, mesmo que esteja a ser executado apenas numa parte do *hardware* do computador subjacente.

Conclui-se que a virtualização permite uma utilização mais eficiente do *hardware* físico do computador e permite um maior retorno do investimento em *hardware* de uma organização [3].

2.1.3 Hipervisor

Um hipervisor, também designado como monitor de VM, é um *software* que agrupa recursos de computação, como processamento, memória e armazenamento, e os distribui entre várias VMs. Este *software* permite criar, gerir e executar VMs a partir de uma única máquina física [4]. Na Figura 1, é apresenta uma comparação entre as arquiteturas dos hipervisores do tipo 1 e do tipo 2.

Hypervisor types

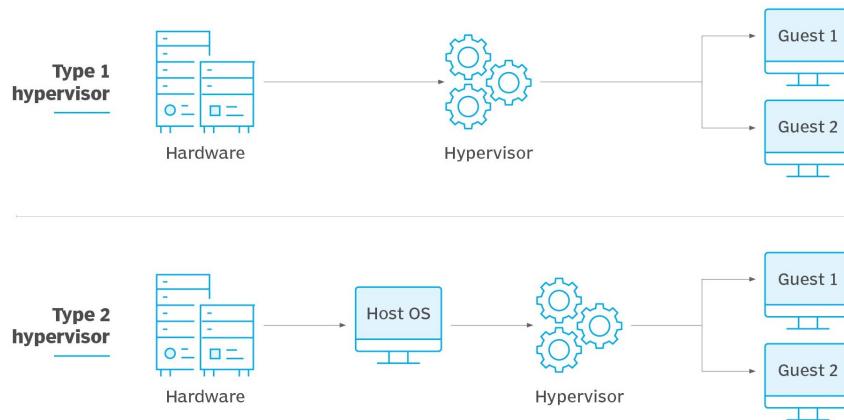


Figura 1: Estrutura de um hipervisor tipo 1 e tipo 2⁴

Hipervisor de Tipo 1 (Hipervisor Nativo ou Bare Metal): Este tipo de hipervisor é instalado diretamente no *hardware* físico do *host*. Não requer um sistema operativo subjacente, este tipo atua como um sistema operativo para gerir as VMs. Exemplos: Proxmox, VMware ESXi, Microsoft Hyper-V, XenServer e Linux KVM [5]. Este tipo de hipervisor apresenta várias vantagens, nomeadamente um desempenho superior, menor latência, maior estabilidade e segurança. No entanto, exige hardware dedicado e pode ser mais complexo de instalar e administrar, o que representa uma desvantagem em ambientes com recursos ou competências técnicas limitados.

Hipervisor de Tipo 2 (Hipervisor Hospedado): Este tipo é instalado num sistema operativo *host*. As VMs que gera são executadas no sistema operativo *host*, o que significa que o hipervisor depende do *host* para aceder aos recursos de *hardware*. Exemplos: VMware Workstation e Oracle VirtualBox [5]. A sua principal vantagem está na facilidade de instalação e utilização, sendo ideal para ambientes de desenvolvimento, testes ou uso pessoal. Não requer hardware dedicado e permite executar VMs em sistemas já existentes. No entanto, apresenta um desempenho e uma fiabilidade inferiores, uma vez que depende do sistema operativo para aceder aos recursos de hardware.

⁴<https://www.techtarget.com/searchitoperations/tip/Whats-the-difference-between-Type-1-vs-Type-2-hypervisor>

No contexto deste projeto, optou-se pelo hipervisor de tipo 1, uma vez que o objetivo era desenvolver uma solução de *Private Cloud* robusta, escalável e fiável. O controlo direto sobre os recursos físicos e a capacidade de garantir um ambiente com elevada disponibilidade e bom desempenho foram fatores decisivos nesta escolha, justificando a adoção de um hipervisor de tipo 1 como base da infraestrutura.

2.1.4 Virtual Machine (VM)

Uma VM é um recurso de computação que utiliza *software* em vez de um computador físico para executar programas e implementar aplicações. Uma ou mais VMs podem ser executadas numa máquina física designada por *host*. Cada VM executa o seu próprio sistema operativo e funciona separadamente das outras VMs, mesmo quando todas estão a ser executadas no mesmo *host*.

A tecnologia de VM é utilizada para muitos casos de utilização em ambientes locais e na *cloud*. Mais recentemente, os serviços de *cloud* pública estão a utilizar VMs para fornecer capacidades de aplicações virtuais a vários utilizadores ao mesmo tempo, para uma computação ainda mais económica e flexível [6].

2.1.5 Container

Um *container* (CT) é uma tecnologia de virtualização que permite agrupar uma aplicação com todas as suas dependências, bibliotecas e configurações necessárias à sua execução. Ao contrário das VMs, os CTs não necessitam de um sistema operativo completo para cada unidade de execução, uma vez que partilham o *kernel* do sistema operativo do *host*, o que os torna mais eficientes em termos de recursos e mais rápidos a iniciar. Na Figura 2, é apresentada as principais diferenças entre a estrutura de um CT e de uma VM.

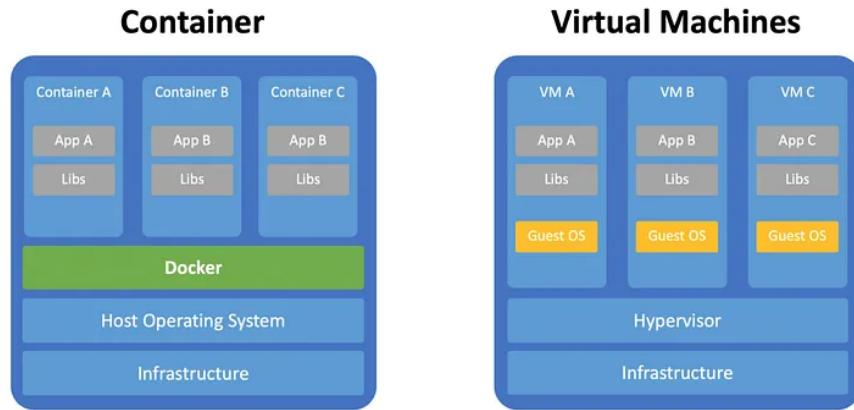


Figura 2: Diferença de estrutura de um CT e VM⁵

Os CTs destacam-se pela sua leveza e portabilidade, permitindo que diferentes aplicações sejam executadas simultaneamente no mesmo sistema sem interferências, mesmo que utilizem versões distintas das mesmas bibliotecas.

Um CT é criado a partir de uma imagem de CT, que consiste num ficheiro fixo e consistente que inclui o código da aplicação, o ambiente de execução, as bibliotecas, as ferramentas e as configurações necessárias para o seu funcionamento. Esta abordagem garante uma elevada portabilidade, permitindo que a aplicação funcione da mesma forma em diferentes ambientes – seja num computador de desenvolvimento, num servidor de testes ou numa infraestrutura na *cloud* [7].

A gestão de CTs é normalmente realizada através de plataformas como o Docker, que permitem criar, executar, parar e eliminar CTs de forma simples e eficiente. Estas plataformas disponibilizam interfaces de linha de comandos e *Application Programming Interface* (APIs) que facilitam a orquestração dos CTs, o controlo de versões das imagens, a definição de redes virtuais e a partilha de volumes de dados. Em ambientes mais complexos ou de grande escala, recorrem-se a sistemas de orquestração como o Kubernetes, que automatizam tarefas como a distribuição de CTs por múltiplos nós, a monitorização do estado das aplicações e a escalabilidade dinâmica.

⁵<https://blog.devops.dev/container-technology-docker-16fb26f1bed7>

2.1.6 *Cluster*

Num sistema informático, um *cluster* é um conjunto de servidores e outros recursos que funcionam como um único sistema, permitindo alta disponibilidade, balanceamento de carga e processamento paralelo. Estes sistemas podem variar desde configurações simples com dois nós (por exemplo, dois servidores dedicados numa pequena empresa) até arquiteturas complexas de super-computadores baseadas em *clusters* [8].

Os clusters são conjuntos de servidores geridos de forma centralizada, responsáveis pela gestão e distribuição das cargas de trabalho. Um *cluster* pode incluir nós (nodes) ou servidores de aplicações individuais. Um nó corresponde, geralmente a um sistema informático físico com um endereço IP próprio, que executa um ou mais servidores de aplicações. Os *clusters* podem ser organizados numa configuração de uma célula (cell), que associa logicamente múltiplos servidores e *clusters*, permitindo a coexistência de diferentes configurações e aplicações.

Os *clusters* têm como principal responsabilidade o balanceamento de carga (load balancing) entre os servidores que os compõem. Os servidores integrantes de um *cluster* são designados por membros do *cluster* (*cluster members*). Quando uma aplicação é instalada num *cluster*, o processo de instalação é automaticamente replicado para todos os seus membros.

2.1.7 *Cloud*

A *cloud* (nuvem) é a disponibilização de um conjunto partilhado de recursos computacionais configuráveis, incluindo servidores, armazenamento, base de dados, rede, *software*, análise e informações, através da Internet. Estes podem ser rapidamente disponibilizados e removidos com mínima gestão ou interação com o fornecedor do serviço. Essencialmente, a computação em nuvem permite que as entidades utilizem recursos de IT através da Internet, sem a necessidade de desenvolver e manter uma infraestrutura de computação própria [9].

Na Figura 3, é apresentada a arquitetura de uma *cloud*.

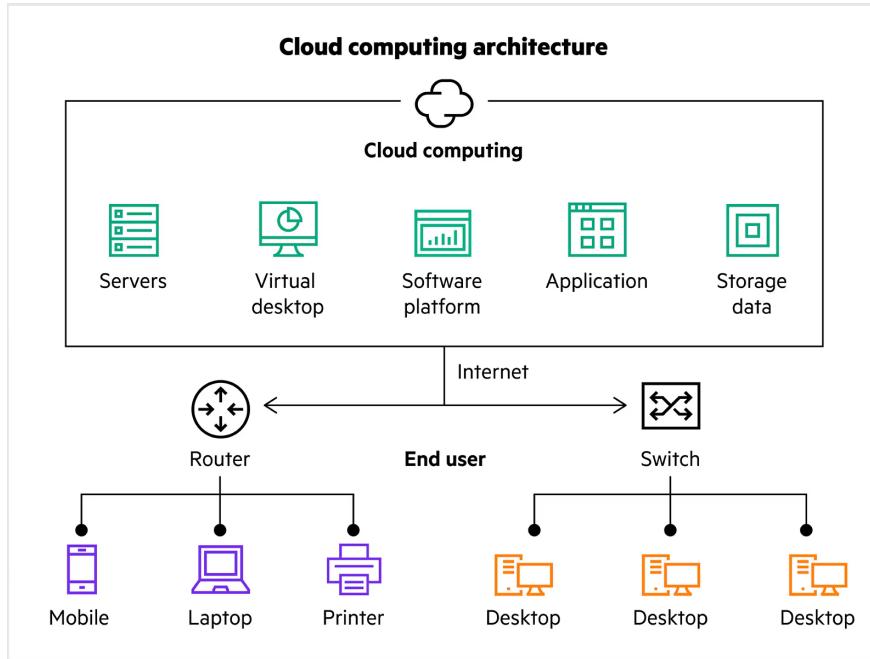


Figura 3: Arquitetura de uma *cloud*⁶

A *cloud* pode ser implementada de diferentes formas, consoante os requisitos de segurança, controlo, escalabilidade e custos de cada organização. Seguem-se os principais modelos de implementação de computação em nuvem.

Cloud Privada: Ambiente de computação em nuvem para uso exclusivo de uma única organização, que pode incluir vários utilizadores internos. Pode ser gerida pela própria organização, por um terceiro, ou por ambos, e pode estar localizada nas instalações da organização ou fora delas [9].

Oferece um elevado nível de segurança, dado que os recursos são dedicados à organização, garantindo maior controlo sobre os dados e os serviços. No entanto, implica um custo total de posse (TCO) mais elevado, uma vez que requer investimento em servidores dedicados, manutenção e gestão da infraestrutura.

Cloud Pública: Disponibilizada para múltiplos clientes. Pode ser gerida por uma entidade comercial, académica, governamental ou uma combinação destas, e encontra-se nas instalações do

⁶https://www.hpe.com/emea_europe/en/what-is/cloud-computing.html

fornecedor do serviço de *cloud*. A infraestrutura onde está alojada a *cloud* é partilhada por todos os utilizadores, o que pode resultar num nível de segurança dos dados inferior, devido à maior exposição e partilha de recursos entre diferentes entidades [9].

Cloud Híbrida: Composta por duas ou mais infraestruturas de *cloud* distintas (privada, comunitária ou pública) que, embora se mantenham como entidades separadas, estão interligadas através de tecnologias compatíveis entre elas. Esta ligação permite a portabilidade de dados e aplicações entre as diferentes *clouds*, possibilitando, por exemplo, a distribuição automática de carga (*cloud bursting*) para equilibrar o desempenho. A *cloud* híbrida oferece a flexibilidade da *cloud* pública e a segurança e controlo da *cloud* privada [9].

Cloud Comunitária: Destinada ao uso exclusivo de uma comunidade específica de organizações com preocupações ou requisitos comuns (como segurança, políticas ou conformidade). A gestão pode ser feita por uma ou mais dessas organizações, por uma entidade externa, ou por ambos, e pode estar localizada nas instalações de uma das entidades ou fora delas [9].

2.1.8 Armazenamento

O armazenamento de dados desempenha um papel fundamental numa infraestrutura de *cloud*, sendo responsável por guardar e disponibilizar a informação aos diversos serviços e utilizadores. Existem três tipos principais de armazenamento utilizados neste contexto: armazenamento em ficheiros (*File Storage*) e em blocos (*Block Storage*).

Armazenamento em Ficheiros (*File Storage*):

Este modelo organiza os dados numa estrutura hierárquica de pastas e subpastas, semelhante ao sistema de ficheiros de um computador. Os ficheiros têm nomes e extensões (ex: .docx, .jpg, .txt) e são acedidos através de caminhos específicos definidos pelo utilizador.

Os dados armazenados neste modelo são organizados e acedidos com base numa quantidade limitada de metadados, que permite ao sistema localizar com precisão o ficheiro pretendido dentro da estrutura hierárquica.

É uma solução simples e intuitiva, ideal para partilha de documentos, armazenamento de dados em rede e ambientes colaborativos. É frequentemente utilizada com servidores de ficheiros ou equipamentos Network-Attached Storage (NAS), e é compatível com sistemas como NFS ou SMB [10].

Armazenamento em Bloco (*Block Storage*):

No armazenamento em blocos, os dados são divididos em blocos de tamanho fixo, cada um com um identificador único. Estes blocos são guardados separadamente e podem ser distribuídos por diferentes sistemas, de forma a maximizar o desempenho e a eficiência do armazenamento.

Ao contrário do armazenamento em ficheiros, os blocos não são geridos como ficheiros individuais, mas sim como discos virtuais, sendo controlados ao nível do sistema operativo. O acesso aos dados é feito através da agregação dinâmica destes blocos, realizada pelo *software* de armazenamento subjacente, que os reagrupa e apresenta ao utilizador como um único volume.

Este modelo é conhecido pela sua alta performance, sendo especialmente adequado para VMs, bases de dados e sistemas que requerem acesso direto e rápido ao disco. É frequentemente utilizado em ambientes Storage Area Network (SAN), ou Rede de Área de Armazenamento, e suportado por tecnologias como iSCSI, *Fibre Channel* e volumes LVM [10].

2.1.9 Anything-as-a-Service (XaaS)

XaaS é um termo abrangente que descreve a oferta de qualquer serviço através de uma rede, geralmente a Internet. Este conceito reconhece a vasta quantidade de produtos, ferramentas e tecnologias que agora são fornecidos aos utilizadores como um serviço através da Internet.

Esta expressão geral refere-se a serviços que estão disponíveis conforme necessário e são financiados com um modelo de pagamento baseado no consumo (*pay-as-you-go*), típico da computação em nuvem. As soluções (XaaS) podem ser ajustadas para aumentar ou diminuir de acordo com as necessidades do utilizador, permitindo que um fornecedor do serviço disponibilize serviços conforme necessário [11].

2.1.10 Alta Disponibilidade

Alta disponibilidade (HA) é a eliminação de pontos únicos de falha para permitir que as aplicações continuem a funcionar mesmo que um dos componentes de IT dos quais dependem, como um servidor, falhe.

Clusters de alta disponibilidade são grupos de servidores que suportam aplicações críticas para a entidade. As aplicações são executadas num servidor primário e, em caso de falha, o funcionamento da aplicação é transferido para servidores secundários, onde continua a operar [12].

2.1.11 Failover

Failover é um mecanismo de transição automática que garante a continuidade de um sistema em caso de falha. Quando o sistema primário (como um servidor, base de dados ou rede) falha ou precisa de manutenção, o *failover* transfere automaticamente a operação para um sistema secundário (em espera) que replica o ambiente do original. Este processo é fundamental em sistemas críticos que exigem disponibilidade contínua, pois minimiza o tempo de inatividade e o impacto nos utilizadores [13].

2.1.12 Fault Tolerance

A tolerância a falhas é a capacidade de um sistema continuar a funcionar mesmo após a ocorrência de falhas em um ou mais componentes. Esta característica é essencial em diversos contextos, como sistemas informáticos, *datacenters*, *clusters* na *cloud* ou redes. A tolerância a falhas descreve a forma como um sistema operativo (OS) reage a falhas de *hardware* ou *software*, permitindo que o sistema continue a funcionar sem interrupções significativas [14].

A diferença entre *fault tolerance* e alta disponibilidade, é que num ambiente com tolerância a falhas não sofre interrupções de serviço, mas tem um custo significativamente mais elevado, enquanto um ambiente com alta disponibilidade pode ter uma interrupção mínima do serviço [15].

2.1.13 Backup

O backup consiste na criação de cópias de segurança dos dados e aplicações, guardadas num local alternativo, com o objetivo de proteger a informação contra falhas, perdas accidentais, ataques ou desastres. Posteriormente, essas cópias podem ser utilizadas para recuperar os dados ou aplicações perdidas, restabelecendo o sistema ao seu estado anterior e garantindo a continuidade das operações [16].

Os backups são criados para serem armazenados durante longos períodos de tempo e, se forem guardados fora do servidor, podem ser utilizados para restaurar servidores após uma falha [17].

2.1.14 Snapshot

Os *snapshots* representam o estado do sistema de ficheiros do servidor num momento específico. Capturam exatamente como o sistema se encontrava nesse instante. Ao restaurar um *snapshot*, o servidor regressa ao estado em que estava no momento em que o *snapshot* foi criado.

Os *snapshots* são concebidos para armazenamento de curta duração. Quando o espaço disponível se esgota, os *snapshots* mais antigos são, eventualmente, substituídos por novos. Por isso, os *snapshots* são geralmente úteis apenas quando se pretende recuperar uma versão recente do servidor [17].

2.1.15 Migração

A migração de VMs é o processo de transferir uma VM de um *host* para outro. Este processo pode ser realizado com ou sem interrupção do serviço, dependendo da abordagem utilizada. A migração é comum em contextos como manutenção planeada, otimização de recursos, balanceamento de carga ou recuperação após falhas. Envolve a transferência da memória, armazenamento e conectividade de rede da VM para um novo destino dentro do *cluster* [18].

2.1.16 *Thin Provisioning*

O *Thin Provisioning* é um método de alocação de armazenamento que possibilita a criação de volumes lógicos com uma capacidade superior à efetivamente disponível em termos físicos. Ao contrário do provisionamento tradicional, que reserva antecipadamente todo o espaço atribuído, esta abordagem apenas aloca armazenamento físico à medida que os dados são efetivamente gravados [19].

Este método é utilizado em redes de armazenamento (SAN), discos centralizados e em sistemas de virtualização de armazenamento, permitindo uma utilização mais eficiente dos recursos disponíveis. No entanto, requer monitorização do sistema, nomeadamente do armazenamento disponível e da taxa de utilização dos volumes, de forma a prevenir a indisponibilidade de armazenamento. Esta monitorização pode ser realizada através de ferramentas como o Grafana, o Zabbix ou interfaces nativas dos sistemas de armazenamento.

2.2 Hipervisor

Neste capítulo, analisam-se as principais soluções existentes de hipervisores de tipo 1 utilizadas em ambientes de virtualização. É apresentada uma comparação detalhada entre as diferentes opções. Com base nesta análise, é justificada a escolha do hipervisor mais adequado para a implementação da infraestrutura da *private cloud* proposta neste projeto.

2.2.1 VMware ESXi + vSphere

O VMware ESXi é um hipervisor bare-metal do tipo 1, com licenciamento proprietário, que é instalado diretamente no *hardware* do servidor para executar VMs de forma segura e eficiente. Uma das principais vantagens do VMware ESXi é o seu conjunto abrangente de funcionalidades para gestão de ambientes virtualizados.

O vSphere, a plataforma de gestão associada, permite o controlo e monitorização centralizada de múltiplos *hosts* ESXi. Através do vSphere, os administradores podem alocar e equilibrar recursos de

forma eficiente, gerir o armazenamento e garantir alta disponibilidade, com funcionalidades como o vMotion (para migração em tempo real de VMs) e o vSphere High Availability (reinício automático de VMs em caso de falha do *host*).

Todo o conjunto vSphere é um produto pago, licenciado por subscrição (normalmente “por vCPU/core”) e disponível em diferentes edições (*Standard, Enterprise Plus, Foundation*) [20, 21].

Na Tabela 1 são apresentados os tipos de armazenamento suportados pelo VMware ESXi em conjunto com o vSphere.

Storage Type	Level	Shared	Snapshots	Thin Provisioning
VMFS (VMware File System)	Bloco	Sim	Sim	Sim
vSAN	Bloco	Sim	Sim	Sim
NFS	Ficheiro	Sim	Sim	Requer suporte no sistema de armazenamento
iSCSI	Bloco	Sim	Sim	Requer suporte no sistema de armazenamento
Fibre Channel / FCoE	Bloco	Sim	Sim	Requer suporte no sistema de armazenamento
vVOLs (Virtual Volumes)	Bloco	Sim	Sim	Sim
Armazenamento Local	Bloco	Não	Sim	Sim
SMB	Ficheiro	Sim	Sim	Depende do servidor

Tabela 1: Tipos de Armazenamento Suportados pelo VMware ESXi + vSphere

A tabela apresentada resume os diferentes tipos de armazenamento compatíveis com o VMware ESXi e a plataforma de gestão vSphere, destacando as suas principais características técnicas. São considerados tanto sistemas baseados em bloco (como VMFS, vSAN, iSCSI, Fibre Channel/FCoE e vVOLs) como em ficheiros (como NFS e SMB), bem como o armazenamento local. Para cada tipo, é indicado se suporta partilha entre *hosts*, *snapshots* e alocação dinâmica de espaço (*thin pro-*

visioning).

2.2.2 Proxmox VE

O Proxmox Virtual Environment (Proxmox VE ou PVE) é uma plataforma *open-source* de gestão de servidores, concebida para virtualização empresarial. Trata-se de um hipervisor bare-metal de tipo 1 que permite executar sistemas operativos Linux e Windows em *hardware* x64.

Integra de forma nativa tanto o hipervisor Kernel-based Virtual Machine (KVM) como os Linux CTs (LXC), assim como funcionalidades de armazenamento definido por *software* (ZFS, Ceph, LVM, iSCSI, NFS) e capacidades avançadas de rede (bridges, VLANs, bonds, Open vSwitch), tudo em uma única solução.

Através de uma interface web integrada e intuitiva que permite gerir VMs e CTs, criar *snapshots* e backups automatizados (vzdump).

Um recurso notável do Proxmox VE é sua capacidade de *clustering*, permitindo a criação de *clusters* de alta disponibilidade para maior confiabilidade. Além disso, o Proxmox VE inclui migração em tempo real (*live migration*), permitindo migrar VMs entre nós físicos sem qualquer interrupção de serviço [22, 21].

Na Tabela 2 são apresentados os tipos de armazenamento suportados pelo Proxmox VE.

Storage Type	Level	Shared	Snapshots	Thin Provisioning
ZFS (local)	Ambos ¹	Não	Sim	Sim
Diretoria	Ficheiro	Não	Não ²	Sim
BTRFS	Ficheiro	Não	Sim	Sim
NFS	Ficheiro	Sim	Não ²	Sim
CIFS	Ficheiro	Sim	Não ²	Depende do servidor
Proxmox Backup Server	Ambos	Sim	n/a	Não se aplica
GlusterFS	Ficheiro	Sim	Não ²	Sim

Storage Type	Level	Shared	Snapshots	Thin Provisioning
CephFS	Ficheiro	Sim	Sim	Sim
LVM	Bloco ³	Não	Não	Não
LVM-thin	Bloco	Não	Sim	Sim
iSCSI/kernel	Bloco	Sim	Não	Depende do array
iSCSI/libiscsi	Bloco	Sim	Não	Depende do array
Ceph RBD	Bloco	Sim	Sim	Sim
ZFS over iSCSI	Bloco	Sim	Sim	Sim

Tabela 2: Tipos de Armazenamento Suportados pelo Proxmox VE

¹ As imagens de disco das VMs são guardadas em datasets ZFS do tipo volume (zvol), que se apresentam como equipamentos de bloco.

² Em armazenamentos baseados em ficheiros, os *snapshots* só são suportados em imagens no formato qcow2.

³ É possível criar um armazenamento LVM partilhado sobre back-ends iSCSI ou Fibre Channel, montando volumes LVM em cada nó.

A tabela apresentada resume os diferentes tipos de armazenamento suportados pelo Proxmox VE, incluindo soluções baseadas em ficheiros, bloco e sistemas híbridos. Estão incluídos sistemas locais como ZFS, Diretoria e LVM, bem como soluções partilhadas como NFS, CephFS, GlusterFS e iSCSI. A tabela indica para cada tipo de armazenamento se permite partilha entre nós, suporte para *snapshots* e *thin provisioning*. Destaca-se o suporte avançado do Proxmox a tecnologias como ZFS e Ceph, que oferecem funcionalidades robustas de *snapshots* e alocação dinâmica, contribuindo para maior flexibilidade e resiliência.

2.2.3 Microsoft Hyper-V

O Microsoft Hyper-V é um hipervisor de tipo 1 desenvolvido pela Microsoft para ambientes Windows Server e sistemas operativos Windows. O Microsoft Hyper-V é um componente integrado em todas as versões modernas do Windows, pelo que os utilizadores não necessitam de adquirir *software* adicional da Microsoft para o utilizar. Ao ter uma licença de um sistema operativo Microsoft

atual, o Hyper-V estará automaticamente disponível para utilização.

O Hyper-V executa-se diretamente sobre o *hardware* do computador do utilizador. Para os utilizadores do Windows, uma vantagem clara é a sua integração nativa com o sistema operativo. Pode ser gerido através do Hyper-V Manager, uma interface gráfica, ou através do PowerShell, para administração por linha de comandos.

Adicionalmente, o Hyper-V pode ser integrado com o *System Center Virtual Machine Manager* (SCVMM) da Microsoft para gestão centralizada e funcionalidades avançadas como migração em tempo real (*live migration*), virtualização de redes e otimização dinâmica de recursos [23, 21].

Na Tabela 3 são apresentados os tipos de armazenamento suportados pelo Microsoft Hyper-V.

Storage Type	Level	Shared	Snapshots	Thin Provisioning
Disco local (NTFS/ReFS)	Ficheiro	Não	Sim	Sim
SMB 3.0	Ficheiro	Sim	Sim	Sim
iSCSI	Bloco	Sim	Sim	Depende do array
Fibre Channel	Bloco	Sim	Sim	Depende do array
Storage Spaces Direct (S2D)	Bloco	Sim	Sim	Sim
Cluster Shared Volumes (CSV)	Ambos	Sim	Sim	Sim
VHD/VHDX	Ficheiro	Sim	Sim	Sim
Pass-through Disk	Bloco	Não	Não	Não

Tabela 3: Tipos de Armazenamento Suportados pelo Microsoft Hyper-V

A tabela apresentada resume os diferentes tipos de armazenamento suportados pelo Microsoft Hyper-V, incluindo soluções baseadas em ficheiros, em bloco e híbridas. Estão incluídos sistemas locais como discos NTFS/ReFS, partilhas SMB 3.0, volumes VHD/VHDX e opções avançadas como iSCSI, *Fibre Channel* e *Storage Spaces Direct* (S2D). A maioria destes tipos suporta *snapshots* e *thin provisioning*, com exceção dos discos *pass-through*, que não oferecem estas fun-

cionalidades. Destaca-se ainda o suporte a *Cluster Shared Volumes* (CSV), que permite alta disponibilidade e partilha de armazenamento entre nós do *cluster*.

2.2.4 XenServer (Citrix Hypervisor)

O XenServer é um hipervisor de tipo 1, concebido para virtualização de servidores. Importa referir que XenServer e Citrix Hypervisor referem-se ao mesmo produto, desenvolvido pela Citrix Systems. Inicialmente conhecido como XenServer, por ser baseado no projeto *open-source* Xen Project, o produto passou a designar-se Citrix Hypervisor, à medida que a Citrix continuou a desenvolvê-lo e a melhorá-lo ao longo do tempo.

Uma das suas principais vantagens é o suporte à virtualização completa (HVM) com aceleração por *hardware* (Intel VT-x/AMD-V), bem como paravirtualização, que permite melhor desempenho [24, 21].

Na Tabela 4 são apresentados os tipos de armazenamento suportados pelo XenServer (Citrix Hypervisor).

Storage Type	Level	Shared	Snapshots	Thin Provisioning
Storage Local (EXT3/EXT4)	Ficheiro	Não	Sim	Sim
LVM over Storage Local	Bloco	Não	Não	Não
NFS	Ficheiro	Sim	Sim	Depende do servidor
iSCSI	Bloco	Sim	Não	Depende do array
Fibre Channel	Bloco	Sim	Não	Depende do array
SMB/CIFS	Ficheiro	Sim	Não	Depende do servidor
LVM over iSCSI ou FC	Bloco	Sim	Sim	Sim

Tabela 4: Tipos de Armazenamento Suportados pelo XenServer / Citrix Hypervisor

A tabela apresentada resume os diferentes tipos de armazenamento suportados pelo XenServer / Citrix Hypervisor, incluindo soluções baseadas em ficheiros, em bloco e configurações híbridas. Estão incluídas opções locais como EXT3/EXT4 e LVM, bem como soluções partilhadas como NFS, iSCSI, *Fibre Channel* e SMB/CIFS. A tabela indica, para cada tipo, se permite partilha entre nós, suporte para *snapshots* e *thin provisioning*. Destaca-se o suporte a configurações como LVM sobre iSCSI ou *Fibre Channel*, que combinam desempenho com funcionalidades avançadas, permitindo adaptar a infraestrutura às necessidades de ambientes virtualizados com requisitos de alta disponibilidade e escalabilidade.

2.2.5 Linux KVM

O Linux KVM é um hipervisor de tipo 1 *open-source* integrado diretamente no kernel do Linux. Esta é uma das suas principais vantagens: por fazer parte do kernel principal, o KVM beneficia continuamente do desenvolvimento e suporte da vasta comunidade Linux.

O KVM suporta vários sistemas operativos, incluindo várias distribuições Linux, Windows, entre outros. A gestão de ambientes virtualizados com KVM é frequentemente realizada através de ferramentas como o libvirt e o QEMU. O libvirt fornece uma interface padronizada para gestão de plataformas de virtualização e suporta várias tecnologias, incluindo o KVM. O QEMU, por sua vez, é um emulador *open-source* que o KVM utiliza para executar e gerir VMs [25, 21].

Na Tabela 5 são apresentados os tipos de armazenamento suportados pelo Linux KVM.

Storage Type	Level	Shared	Snapshots	Thin Provisioning
LVM	Bloco	Não	Não	Não
LVM-thin	Bloco	Sim	Sim	Sim
ZFS	Ambos	Sim	Sim	Sim
Btrfs	Ficheiro	Sim	Sim	Sim
Ceph RBD	Bloco	Sim	Sim	Sim
CephFS	Ficheiro	Sim	Sim	Sim
iSCSI	Bloco	Sim	Não	Depende do array

Storage Type	Level	Shared	Snapshots	Thin Provisioning
Fibre Channel	Bloco	Sim	Não	Depende do array
NFS	Ficheiro	Sim	Sim	Depende do servidor
GlusterFS	Ficheiro	Sim	Sim	Sim
SMB/CIFS	Ficheiro	Sim	Não	Depende do servidor
DRBD	Bloco	Sim	Sim	Não

Tabela 5: Tipos de Armazenamento Suportados pelo KVM em Ambiente Linux

A tabela apresentada resume os diferentes tipos de armazenamento suportados pelo KVM em ambientes Linux, incluindo soluções baseadas em ficheiros, em bloco e sistemas híbridos. Estão incluídas opções locais como LVM, *LVM-thin*, ZFS e btrfs, bem como soluções distribuídas e partilhadas como CephFS, Ceph RBD, NFS, GlusterFS, e protocolos como iSCSI e *Fibre Channel*. A tabela indica, para cada tipo, se permite partilha entre nós, suporte para *snapshots* e *thin provisioning*. Destaca-se a flexibilidade do ecossistema Linux em suportar tecnologias modernas como Ceph, btrfs e ZFS, que oferecem funcionalidades avançadas.

2.2.6 Comparação de Hipervisores

A Tabela 6 apresenta uma comparação detalhada entre algumas das principais soluções de virtualização de hipervisores Tipo 1, nomeadamente VMware ESXi com vSphere, Proxmox VE, Microsoft Hyper-V, XenServer (Citrix Hypervisor) e Linux KVM. A análise incide sobre um conjunto alargado de funcionalidades essenciais em ambientes de virtualização, incluindo suporte a alta disponibilidade, migração em tempo real, integração com CTs, gestão através de interfaces web e por linha de comandos. Esta comparação visa apoiar a tomada de decisão relativamente à escolha da plataforma de virtualização mais adequada às necessidades específicas da infraestrutura em estudo, tendo em consideração critérios como a disponibilidade de código aberto, funcionalidades nativas e exigências de licenciamento.

Funcionalidade	VMware ESXi + vSphere	Proxmox VE	Microsoft Hyper-V	XenServer / Citrix Hypervisor	Linux KVM
Tipo de Hipervisor	Tipo 1 (bare-metal)	Tipo 1	Tipo 1	Tipo 1	Tipo 1 (integrado no kernel)
Código Aberto	Não (versão gratuita limitada)	Sim	Não	Parcialmente (base Xen é <i>open-source</i>)	Sim
Gestão Web UI	Sim (vSphere Client)	Sim	Sim (Windows Admin Center/SCVMM)	Sim (XenCenter, Xen Orchestra)	Sim (Cockpit, virt-manager, etc.)
Migração em tempo real	Sim (vMotion)	Sim	Sim	Sim	Sim
Snapshots	Sim	Sim	Sim (Checkpoints)	Sim	Sim (com QCOW2, ZFS, Btrfs)
Alta Disponibilidade (HA)	Sim	Sim	Sim	Sim	Sim (via pacemaker, corosync)
<i>Fault Tolerance</i>	Sim (Enterprise Plus)	Não	Não nativo	Não	Não nativo
CTs	Não	Sim (LXC)	Não	Não	Sim (LXC, systemd-nspawn, etc.)
vGPU / GPU Passthrough	Sim (vGPU GRID)	Sim	Sim (DDA, RemoteFX)	Sim	Sim (VFIO, SR-IOV)
Backup Integrado	Parcial (vSphere Replication)	Sim (PBS)	Sim (Windows Backup, VSS)	Não nativo	Não (via ferramentas externas)
Gestão CLI/API	Sim (PowerCLI, API REST)	Sim (CLI + API REST)	Sim (PowerShell, WMI)	Sim (XAPI, xe CLI)	Sim (virsh, libvirt, Ansible, etc.)
<i>Cluster de Hosts</i>	Sim	Sim	Sim (Failover Cluster)	Sim	Sim (com ferramentas manuais)
Armazenamento definido por <i>software</i>	Sim (vSAN)	Sim (Ceph, ZFS)	Sim (S2D)	Parcial	Sim (Ceph, GlusterFS, ZFS, DRBD)
Licenciamento	Pago (gratuito com limitações)	Gratuito (<i>open source</i>)	Incluído no Windows Server / licenças adicionais	Gratuito (versão base) + pago (Citrix)	Gratuito (GPL)

Tabela 6: Comparação de Funcionalidades entre Hipervisores de Tipo I

2.2.7 Escolha do Hipervisor

Tendo com base a pesquisa realizada sobre os principais hipervisores de tipo 1, e a comparação efetuada na Tabela 6, esta análise foi fundamental para a tomada de decisão sobre o hipervisor a utilizar. Essa pesquisa efetuada fundamentou a decisão de selecionar o Proxmox VE como solução de virtualização para a *private cloud*.

A escolha recaiu sobre o Proxmox VE pelas seguintes razões:

- É uma solução *open-source* e sem custos de licenciamento;
- Oferece suporte nativo a VMs (KVM) e CTs (LXC);
- Disponibiliza uma interface de gestão web intuitiva e suporte a API REST;
- Suporta funcionalidades como migração em tempo real, alta disponibilidade (HA) e *snapshots*;
- É compatível com armazenamento definido por *software*, como Ceph e ZFS, proporcionando resiliência, escalabilidade e eficiência;
- Inclui ferramentas de backup integradas, como o *Proxmox Backup Server*, facilitando a proteção e recuperação de dados.

Estas características são particularmente relevantes no contexto deste projeto, que tem como objetivo a construção de uma *private cloud* funcional e escalável, capaz de garantir a alta disponibilidade e a redundância dos serviços.

Em comparação com outras soluções analisadas, como o VMware ESXi (com funcionalidades avançadas sujeitas a licenciamento) ou o Microsoft Hyper-V (integrado em ambientes Windows), o Proxmox VE apresenta um equilíbrio entre funcionalidades, simplicidade de gestão e ausência de custos, o que o torna particularmente atrativo neste projeto.

O Proxmox VE revelou-se a solução mais completa, oferecendo um vasto conjunto de funcionalidades. A sua integração permite não só maximizar o aproveitamento dos recursos disponíveis,

como também garantir a escalabilidade e fiabilidade necessárias ao bom funcionamento da *private cloud*.

2.3 Tipos de Armazenamento do Proxmox

Nesta secção são apresentados e explicados os principais tipos de armazenamento suportados pelo Proxmox VE , destacando as suas características, aplicações, funcionamento e arquitetura. Serão abordados os seguintes tipos de armazenamento: LVM, LVM-Thin, ZFS, iSCSI, NFS e Ceph.

2.3.1 LVM

O LVM (Logical Volume Manager) é uma camada de abstração leve que se situa entre os discos físicos ou partições e o sistema operativo. Permite agrupar equipamentos físicos em grupos de volumes (*Volume Groups*, ou VGs) e, a partir destes, criar volumes lógicos (*Logical Volumes*, ou LVs) que funcionam como discos virtuais. Esta abordagem oferece uma gestão de armazenamento mais flexível e poderosa, especialmente útil em ambientes virtualizados como o Proxmox VE [26].

Aplicações:

No Proxmox, o LVM é amplamente utilizado para gerir o armazenamento de VMs. Os volumes lógicos podem ser facilmente redimensionados, migrados entre discos ou utilizados para a criação de *snapshots* (no LVM-Thin), sendo essenciais para a administração de sistemas virtualizados com alta disponibilidade.

O LVM pode ser aplicado tanto sobre discos locais como sobre equipamentos de armazenamento remoto, como Logical Unit Number (LUN) iSCSI. Nestes casos, o LVM torna possível organizar e gerir o espaço disponível com maior flexibilidade e controlo, algo que o protocolo iSCSI, por si só, não proporciona. Esta capacidade permite, por exemplo, que um único LUN seja dividido em múltiplos volumes lógicos, otimizando o uso do armazenamento.

Além disso, o LVM pode ser utilizado em conjunto com sistemas de armazenamento em rede, como *iSCSI targets* ou *NFS shares*, permitindo que múltiplos nós de um *cluster* accedam ao mesmo

armazenamento partilhado.

Funcionamento:

O LVM funciona sobre três níveis de abstração tal como ilustrado na Figura 4:

- **Physical Volume (PV)**: representa os equipamentos físicos, como discos ou partições, que são adicionados ao LVM.
- **Volume Group (VG)**: conjunto de PVs agrupados, formando um espaço comum de armazenamento.
- **Logical Volume (LV)**: unidades criadas dentro de um VG, que podem conter sistemas de ficheiros e ser usadas como discos virtuais pelas VMs/CTs [27].

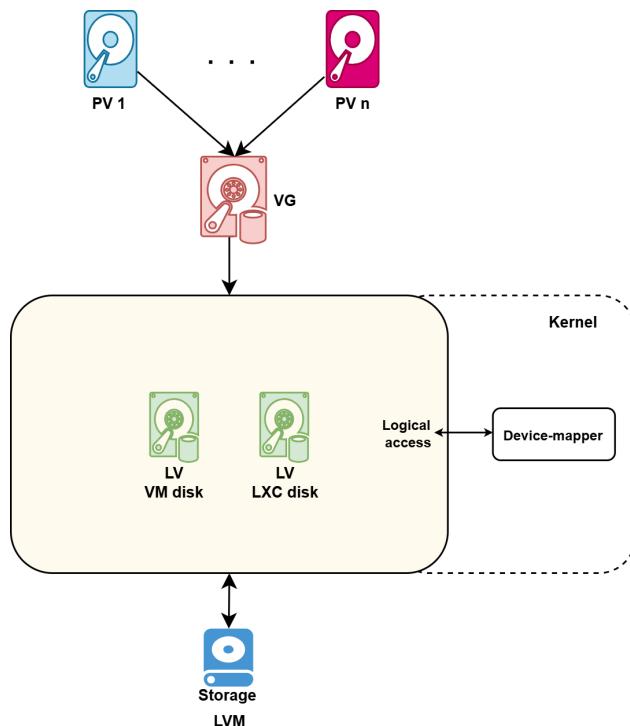


Figura 4: Arquitetura do LVM

A Figura 4 apresenta a arquitetura de um sistema LVM, onde cada volume lógico funciona como um disco separado que pode ser formatado, montado ou atribuído diretamente a uma VM ou CT (LXC). A capacidade de redimensionar LVs e adicionar novos PVs aos VGs sem interrupções

oferece grande flexibilidade, permitindo a expansão dinâmica do armazenamento.

Por baixo desta estrutura está o *device-mapper*, um componente do kernel Linux que faz o mapeamento entre volumes lógicos e físicos, permitindo a gestão eficiente dos volumes pelo LVM [27].

2.3.2 LVM-Thin

O LVM-Thin é uma extensão do LVM que implementa o conceito de *thin provisioning* no armazenamento em LVs. Ao contrário do LVM tradicional, onde o espaço físico é reservado integralmente no momento da criação do volume, o LVM-Thin aloca o espaço apenas quando os dados são efetivamente escritos no volume [28].

Aplicações:

No Proxmox, o LVM-Thin é muito valorizado em ambientes virtualizados, pois permite criar LVs com tamanhos virtuais superiores ao espaço físico disponível, otimizando a utilização dos recursos de armazenamento. Esta funcionalidade é especialmente útil quando se gerem múltiplas VMs que nem sempre utilizam o espaço total alocado a cada uma delas [29].

Além disso, o LVM-Thin facilita a criação de *snapshots* eficientes e leves, consumindo espaço adicional somente para as alterações realizadas após o *snapshot*, o que melhora o desempenho e a gestão dos backups.

Funcionamento:

O LVM-Thin funciona através da criação de um *thin pool*, que é um LV especial dentro de um VG, gerido pelo LVM. Este *thin pool* serve como base para os LVs *thin* e contém dois LVs auxiliares tal como ilustrado na Figura 5:

- **VG-ThinPool_tmeta**: volume que armazena os metadata do thin pool, contendo os mapas de alocação, estrutura e controlo dos volumes thin.
- **VG-ThinPool_tdata**: volume que guarda os dados reais escritos pelos volumes thin [28].

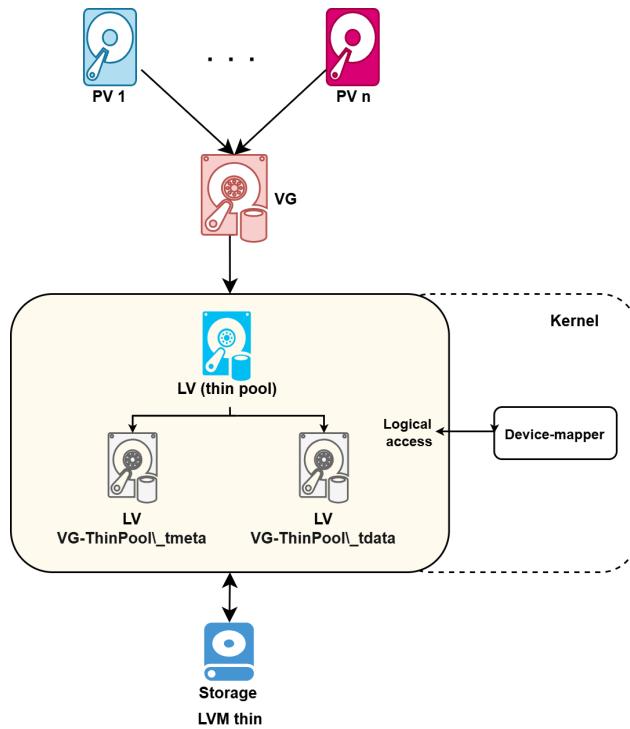


Figura 5: Arquitetura do LVM-Thin

A Figura 5 apresenta a arquitetura de um LVM-Thin, onde o LV principal, *thin pool*, é o volume sobre o qual os LVs *thin* (discos de VMs/CTs) são efetivamente criados, ocultando os volumes auxiliares (*_tmeta* e *_tdata*), que são geridos automaticamente pelo LVM.

Por trás do funcionamento do *thin pool* está o *device-mapper*, um componente do kernel Linux responsável por mapear o acesso a equipamentos virtuais, controlando dinamicamente a alocação do espaço físico conforme os dados são escritos e permitindo o *thin provisioning* e a criação eficiente de *snapshots* [30].

Esta abordagem oferece maior flexibilidade e eficiência na gestão de armazenamento, tornando o LVM-Thin uma solução ideal para sistemas virtualizados que exigem escalabilidade e otimização dos recursos. Especialmente em ambientes onde o espaço físico é um recurso limitado e deve ser maximizado, sendo que cada disco virtual é um LV thin individual.

2.3.3 Diretoria

O tipo de armazenamento diretoria refere-se a um armazenamento baseado em sistema de ficheiros, utilizando uma diretoria local montada num ponto específico do sistema. Esse tipo de armazenamento é um dos mais simples e comuns, ideal para armazenar discos de VMs, CTs, ISOs, backups e *templates* [31].

Aplicações:

O armazenamento por diretoria é muito utilizado em servidores que possuem discos formatados com sistemas de ficheiros como ext4, xfs ou btrfs. É ideal para cenários onde se pretende ter facilidade de acesso aos dados através do sistema de arquivos tradicional do Linux, e não exige configurações avançadas.

No Proxmox, a configuração de uma diretoria consiste em montar um sistema de ficheiros sobre um PV (disco ou partição) e adicioná-lo via interface web, especificando o caminho da diretoria.

Além disso, é possível criar um LV a partir do LVM, formatado com um sistema de ficheiros e montado como uma diretoria. Dessa forma, combina-se a flexibilidade do LVM com a simplicidade do *backend* baseado em diretoria. Essa abordagem permite redimensionar facilmente o espaço do armazenamento conforme necessário, aproveitando as vantagens da gestão dinâmica dos LVs.

Funcionamento:

O armazenamento baseado em diretoria, ao ser configurado no proxmox, é automaticamente organizado numa estrutura de pastas no caminho especificado, com subdiretorias destinadas a diferentes tipos de dados:

- Discos de VMs e CTs (images/).
- ISOs de instalação (iso/).
- Backups realizados via *vzdump* (dump/).
- *Templates* de CTs (template/).

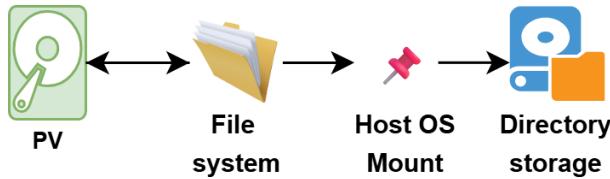


Figura 6: Arquitetura da Diretoria

Como apresentado na Figura 6, a arquitetura da Diretoria envolve a utilização de um PV, que pode ser uma partição, um disco ou um LV, sobre o qual é criado um sistema de ficheiros. Este sistema é montado pelo sistema operativo anfitrião (*Host OS*) e disponibilizado como uma diretoria de armazenamento, que serve de backend para os dados.

Diferente de soluções como LVM ou ZFS, o armazenamento diretoria não possui funcionalidades nativas de *snapshots* ou replicação. No entanto, sua simplicidade e compatibilidade com ferramentas de backup de arquivos fazem dele uma opção prática e direta.

É importante destacar que, em armazenamentos baseados em arquivos, os *snapshots* são suportados apenas para imagens de disco no formato *qcow2*. Este formato oferece funcionalidades internas de *snapshot*, permitindo capturar o estado da VM sem necessidade de suporte nativo do sistema de ficheiros. Por outro lado, formatos como *raw* não possuem essa capacidade e dependem de soluções externas para *snapshots* [31].

2.3.4 ZFS

O ZFS é um sistema de ficheiros e gestor de volumes lógicos combinados. No Proxmox VE, o ZFS é suportado nativamente a partir da versão 3.4, permitindo que seja usado tanto como sistema de ficheiros adicional como também como sistema de ficheiros de raiz. Não há necessidade de compilar módulos manualmente, pois todos os dados necessários já vêm incluídos no sistema [32].

O ZFS distingue-se por integrar num único sistema, funcionalidades que tradicionalmente exigiam várias camadas separadas. Além de gerir ficheiros, o ZFS também gere volumes lógicos (zvols), permitindo trabalhar ao nível de blocos, o que é essencial para discos virtuais usados por

VMs.

É possível obter funcionalidades avançadas mesmo com *hardware* de baixo custo, mas também alcançar sistemas de alto desempenho através do uso de cache com SSD ou mesmo configurações exclusivamente com SSD. O ZFS pode substituir controladores RAID de *hardware* dispendiosos, impondo apenas uma carga moderada ao processador e à memória, além de oferecer uma gestão simplificada [32].

Partilha e Replicação de Dados com ZFS:

No entanto, importa referir que o ZFS no Proxmox VE funciona localmente em cada nó — os pools ZFS (*zpool*) não são partilhados automaticamente entre nós de um *cluster*. Cada nó mantém o seu próprio armazenamento independente. No entanto, é possível partilhar volumes ZFS entre nós utilizando protocolos como iSCSI ou NFS. Com o iSCSI, pode-se criar um volume ZFS num nó e disponibilizá-lo como armazenamento em bloco para outros nós do *cluster*. Já com o NFS, é possível partilhar diretórios ou sistemas de ficheiros ZFS entre nós, permitindo o acesso simultâneo a ficheiros através da rede.

Adicionalmente, para replicação de dados entre nós, pode ser utilizada a funcionalidade de replicação nativa do Proxmox (*ZFS Replication*), que copia volumes de forma assíncrona, útil para alta disponibilidade ou recuperação em caso de falha.

Funcionamento:

O funcionamento do ZFS assenta em quatro conceitos principais (Figura 7):

- **Zpool:** É a base do armazenamento ZFS. Um *zpool* é um conjunto de equipamentos de armazenamento (discos, SSDs, etc.) combinados para formar uma unidade lógica. Toda a gestão de dados no ZFS parte do *zpool*. Os dados são distribuídos pelos discos com base na política de redundância definida (como RAIDZ ou mirror).
- **Dataset:** No ZFS, um *dataset* é qualquer estrutura de dados gerida dentro de um *zpool*. Pode assumir diferentes formas, como sistemas de ficheiros (*file systems*), volumes em bloco

(*zvols*), *snapshots* ou clones. Cada dataset possui propriedades configuráveis individualmente, como compressão, permissões e encriptação, permitindo uma gestão detalhada do armazenamento.

- **File system:** É um tipo específico de dataset que fornece um sistema de ficheiros ZFS. Comporta-se como uma diretoria independente montável, ideal para armazenamento de ficheiros, partilhas em rede (via NFS ou SMB), cópias de segurança ou utilização local. Os sistemas de ficheiros ZFS permitem aplicar políticas de gestão como *snapshots*, compressão e limitação de espaço, de forma isolada por diretoria.
- **Zvol:** É um dataset do tipo volume de bloco, criado dentro de um *zpool*. Um *zvol* apresenta-se ao sistema como um equipamento de blocos (semelhante a um disco físico), sendo adequado para utilização com VMs, CTs ou exportações com iSCSI.

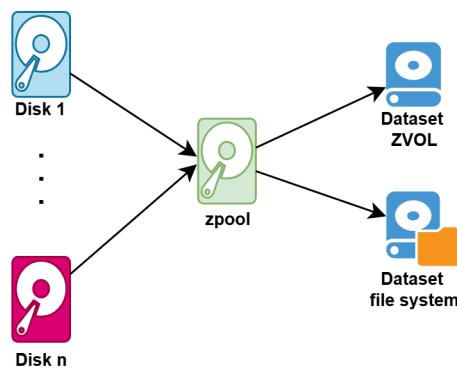


Figura 7: Arquitetura do ZFS

A Figura 7 ilustra a arquitetura do ZFS. Apresenta vários discos físicos que formam uma *zpool*, onde é escolhida a política de redundância. Dentro do *zpool* criam-se dois tipos principais de *datasets*: o *filesystem* e o *zvol*.

Redundância no ZFS:

O ZFS oferece várias opções de redundância para garantir a integridade e a disponibilidade dos dados. A redundância pode ser configurada durante a criação da pool de armazenamento(*zpool*), utilizando diferentes níveis de RAID, permitindo que o sistema se recupere automaticamente de

falhas de disco sem perda de dados.

As principais opções de redundância são:

- **RAID0 (striping):** Combina a capacidade de vários discos num único volume, sem redundância. Ou seja, a falha de um disco resulta na perda de todos os dados.
- **RAID1 (mirroring):** Grava os dados de forma idêntica em dois discos. A capacidade total do volume é a de um único disco, mas oferece redundância, pois os dados são espelhados.
- **RAID10:** Uma combinação de RAID0 e RAID1, que exige pelo menos quatro discos. Oferece boa performance e redundância, pois utiliza tanto mirroring como striping.
- **RAIDZ-1:** Uma variação do RAID5, com paridade simples. Requer pelo menos três discos e pode tolerar a falha de um disco sem perda de dados.
- **RAIDZ-2:** Uma variação do RAID5, com paridade dupla. Exige pelo menos quatro discos e pode tolerar a falha de até dois discos sem perda de dados.
- **RAIDZ-3:** Uma variação do RAID5, com paridade tripla. Requer pelo menos cinco discos e pode tolerar a falha de até três discos sem perda de dados [32].

Nota: A **paridade** é uma técnica de proteção de dados que utiliza cálculos matemáticos para gerar informações redundantes.

Além da redundância configurada através dos níveis de RAID, o ZFS também oferece a possibilidade de usar dRAID (RAID declusterizado), uma opção avançada para grandes volumes de discos, que oferece maior velocidade de reconstrução de dados em caso de falha (ideal para configurações com 10 a 15 discos). Para menos discos, a configuração RAIDZ pode ser mais adequada.

O dRAID no ZFS utiliza unidades de disco de reserva que fazem parte do RAID, funcionando de forma semelhante a um RAID5 + *spare*. A capacidade dos discos de reserva consiste em serem utilizados para restaurar rapidamente os dados em caso de falha de um dos discos. Isso proporciona, dependendo da configuração, uma recuperação mais eficiente em comparação com o RAIDZ.

Observação: A interface gráfica do Proxmox VE exige um disco adicional em relação ao mínimo necessário. Por exemplo, o dRAID1 requer 3 discos, sendo um deles um disco adicional de reserva.

Configurações de dRAID:

- **dRAID1:** Requer pelo menos 2 discos. Um disco pode falhar antes que os dados sejam perdidos.
- **dRAID2:** Requer pelo menos 3 discos. Até dois discos podem falhar sem perder dados.
- **dRAID3:** Requer pelo menos 4 discos. Até três discos podem falhar sem perda de dados [32].

Além das opções de redundância, o ZFS também oferece criptografia nativa. A criptografia pode ser aplicada a datasets (unidades de armazenamento dentro de uma pool), e os dados são criptografados antes de serem gravados no disco. Para aceder aos dados, é necessária a chave correta, o que proporciona uma camada adicional de segurança contra acessos não autorizados .

2.3.5 iSCSI

O iSCSI (Internet Small Computer System Interface) é um protocolo que permite a ligação de equipamentos de armazenamento através de uma rede utilizando TCP/IP. O iSCSI possibilita a transmissão de comandos SCSI por redes IP, facilitando a criação de redes de armazenamento (Storage Area Networks (SANs)) e permitindo o acesso a equipamentos de armazenamento remotos como se estivessem ligados localmente [33].

Aplicações:

O iSCSI é frequentemente usado em conjunto com sistemas de ficheiros avançados, como o ZFS, para a partilha de armazenamento em blocos. Por exemplo, um servidor com ZFS pode criar volumes (zvols) e disponibilizá-los como targets iSCSI para outros servidores acederem pela rede. Desta forma, combina-se a robustez do ZFS com a flexibilidade de acesso remoto proporcionada

pelo iSCSI.

No Proxmox VE, o iSCSI é utilizado para conectar o ambiente de virtualização a equipamentos de armazenamento remotos, permitindo que os volumes disponibilizados através da rede sejam utilizados como se fossem discos locais. O Proxmox atua como initiator, ligando-se a um servidor remoto (target), como um NAS ou um servidor com ZFS, que fornece volumes via iSCSI. Esses volumes podem ser utilizados diretamente ou em conjunto com tecnologias como LVM ou ZFS.

O suporte a iSCSI no Proxmox é considerado semi-nativo, já que a funcionalidade de initiator está integrada diretamente na interface do Proxmox. No entanto, para criar targets iSCSI, é necessário instalar pacotes adicionais e configurar a solução através da linha de comandos.

Funcionamento:

O iSCSI encapsula comandos SCSI dentro de pacotes TCP/IP, permitindo que estes sejam transportados por redes IP normais. Este processo envolve quatro componentes principais:

- **Initiator:** o sistema que inicia a ligação e envia os comandos, normalmente um servidor ou *host* que necessita de aceder ao armazenamento.
- **Target:** o equipamento ou sistema que disponibiliza o armazenamento remoto e responde a esses comandos [33].
- **Portal:** endereço de rede (IP) e porta TCP (por padrão, a porta 3260) onde o serviço iSCSI está disponível. É através deste portal que o initiator contacta o target.
- **LUN (Logical Unit Number):** cada target iSCSI pode disponibilizar uma ou mais unidades lógicas de armazenamento, chamadas LUNs. Cada LUN é visto pelo initiator como um disco em bloco, podendo ser utilizado diretamente ou em conjunto com tecnologias como LVM ou ZFS.

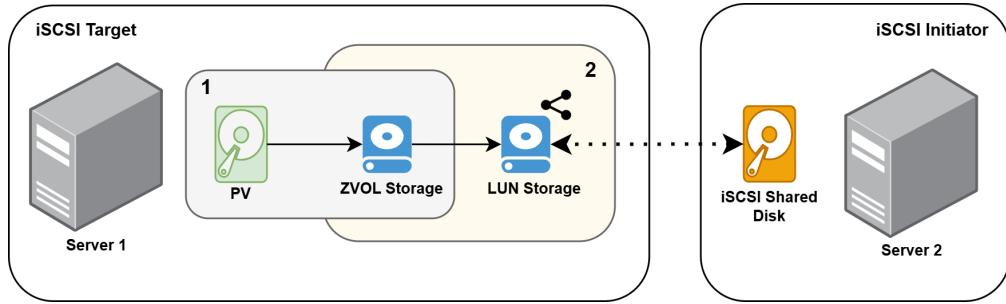


Figura 8: Arquitetura do ZFS over iSCSI

A Figura 8 apresenta a arquitetura do ZFS sobre iSCSI, onde o iSCSI é o responsável por fazer a partilha do armazenamento ZFS através da rede. Nesta arquitetura, o initiator é o sistema que inicia a ligação, enquanto o target disponibiliza o armazenamento remoto. Dentro do target existem dois módulos principais: o módulo identificado com o número 1, responsável pela criação e gestão do armazenamento ZFS; e o módulo identificado com o número 2, que exporta o *zvol* como LUN através do protocolo iSCSI, permitindo o seu acesso remoto pelo initiator.

2.3.6 NFS

O Network File System (NFS) é um protocolo que permite que sistemas de ficheiros sejam partilhados entre diferentes máquinas através de uma rede. O NFS possibilita que um sistema aceda a diretórios e ficheiros localizados noutra servidor como se estivessem no seu próprio sistema de ficheiros, permitindo a criação de soluções de armazenamento em rede simples e eficientes [34].

Aplicações:

O NFS é frequentemente utilizado em ambientes virtualizados, especialmente em *clusters*, por permitir que múltiplos nós tenham acesso simultâneo a um sistema de ficheiros partilhado.

A sua principal vantagem é a simplicidade na configuração e a capacidade de ser partilhado entre vários nós sem a necessidade de sistemas de ficheiros distribuídos ou protocolos de replicação complexos.

No Proxmox, a configuração de um armazenamento NFS pode ser feita diretamente através

da interface web. Basta fornecer o endereço IP do servidor NFS, o caminho da exportação e selecionar os tipos de conteúdo suportados. O sistema montará automaticamente o NFS na diretoria `/mnt/pve/<nome_do_storage>`, ficando imediatamente disponível para uso [35].

Embora o NFS opere a nível de ficheiros (*file-level*), ao contrário de protocolos como iSCSI que trabalham a nível de blocos (*block-level*), é adequado para a maioria das aplicações de virtualização, mas com exceção de cenários que exigem alta performance de I/O ou funcionalidades como *snapshots* a nível de bloco.

Funcionamento:

O funcionamento do NFS baseia-se na partilha de diretórias pela rede (Figura 9), com os seguintes componentes principais:

- **Servidor NFS:** máquina que disponibiliza o sistema de ficheiros. Pode ser uma NAS ou um servidor Linux configurado com NFS.
- **Exportação (Export):** diretoria partilhada pelo servidor, definida geralmente no ficheiro `/etc/exports` [34].
- **Cliente NFS:** sistema que acede à diretoria exportada.

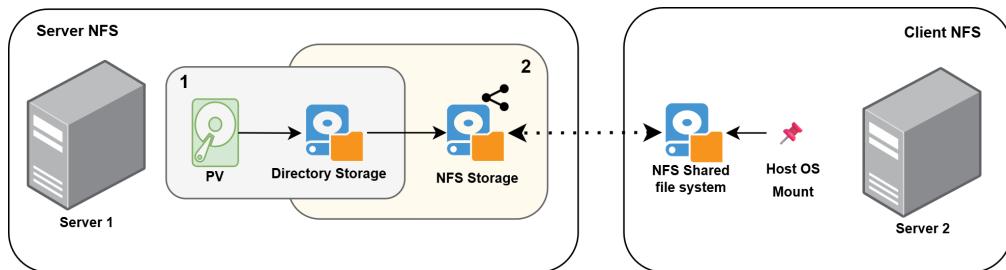


Figura 9: Arquitetura do NFS

A Figura 9 apresenta a arquitetura do NFS, onde o protocolo NFS é responsável por partilhar sistemas de ficheiros através da rede. Nesta arquitetura, o cliente NFS é o sistema que acede remotamente à diretoria partilhada e a monta localmente no seu sistema operativo como se fosse uma pasta normal. Por sua vez, o servidor NFS disponibiliza esse armazenamento. Dentro do servidor,

existem dois módulos principais: o módulo identificado com o número 1, responsável pela criação e gestão da diretoria local (por exemplo, Diretoria ou num sistema de ficheiros ZFS); e o módulo identificado com o número 2, que exporta essa diretoria como armazenamento NFS acessível pela rede.

Quando é montado, o armazenamento NFS passa a ser utilizado como backend de armazenamento, e o sistema cria automaticamente as subpastas necessárias.

Apesar de o NFS não suportar *snapshots* nativos, é possível armazenar imagens de disco no formato qcow2, que oferecem suporte interno a *snapshots*. No entanto, o desempenho e a robustez desta solução dependem da qualidade da rede e do servidor NFS utilizado [35].

2.3.7 Ceph

O Ceph é uma plataforma de armazenamento distribuído de código aberto, desenvolvida para oferecer elevada escalabilidade, desempenho e fiabilidade em ambientes de armazenamento de dados de grande dimensão. Trata-se de um sistema extremamente versátil que permite armazenar objetos, blocos e ficheiros num único ambiente unificado, proporcionando assim flexibilidade para diversas aplicações, como VMs, backups, sistemas de ficheiros partilhados e outras soluções.

O sistema foi concebido para ser auto-recuperável e auto-gerido, com o objetivo de minimizar o tempo necessário para administração e reduzir os custos operacionais. Isto significa que o Ceph consegue detetar automaticamente falhas em discos ou servidores e recuperar os dados perdidos a partir de cópias existentes, tudo de forma automática, sem necessidade de intervenção manual imediata. Esta característica torna o sistema altamente resiliente, mantendo a integridade dos dados mesmo em caso de falhas de *hardware*. [36]

Uma das grandes vantagens do Ceph é a sua capacidade de crescer horizontalmente. Basta adicionar mais discos ou servidores para expandir o armazenamento disponível, sem necessidade de interrupções ou reconfigurações complexas. Além disso, o Ceph não tem um ponto único de falha, o que significa que o sistema continua a funcionar mesmo que um dos seus componentes

fique indisponível.

O Ceph é amplamente utilizado em ambientes como *datacenters*, infraestruturas de *private cloud* e plataformas de virtualização, nomeadamente o Proxmox VE, onde se integra de forma nativa. O seu caráter *open-source* permite ainda que seja adotado sem custos de licenciamento, o que o torna particularmente atrativo para organizações que procuram soluções robustas e económicas.

O Ceph é uma solução que requer um volume significativo de recursos e, por isso, exige um *hardware* adequado. As recomendações para a sua implementação estão apresentadas na Tabela 7.

Requisitos de Hardware	Descrição
CPU	Processador moderno (mínimo 4 cores por nó).
Memória RAM	4 GB de RAM por OSD + 2 GB adicionais para o sistema e serviços Ceph.
Armazenamento para OSDs	Discos dedicados ao Ceph; SSDs NVMe para DB/WAL e HDDs ou SSDs para dados.
Placa de rede	Interface de rede de 10 GbE, dedicada ao tráfego Ceph.
Número mínimo de nós	3 nós para garantir resiliência e quorum.

Tabela 7: Recomendações de *hardware* para Ceph [1]

RADOS

O RADOS (*Reliable Autonomic Distributed Object Store*) é o componente central do Ceph, responsável pelo armazenamento de dados em formato de objetos distribuídos. Concebido para ser altamente escalável, distribuído e auto-gerido, serve de base para os serviços de mais alto nível do Ceph, como o RBD (*RADOS Block Device*), o CephFS (*Ceph File System*) e o RGW (*RADOS Gateway*). Estes serviços utilizam a biblioteca **libRADOS**, que oferece uma API para acesso direto à camada de objetos do RADOS. Além disso, essa biblioteca pode ser utilizada diretamente por aplicações desenvolvidas à medida, oferecendo maior flexibilidade e desempenho. [1]

Todos os dados geridos por esses serviços são armazenados internamente como objetos no RADOS, organizados logicamente em **pools**, que são agrupamentos virtuais que definem políticas de

replicação, codificação, e distribuição. Essa estrutura facilita a gestão detalhada dos dados e permite que diferentes pools partilhem o mesmo *cluster* com configurações personalizadas.

O RADOS utiliza o algoritmo CRUSH (*Controlled Replication Under Scalable Hashing*) para determinar, de forma eficiente e determinística, em quais OSDs (*Object Storage Daemon*) cada objeto deve ser armazenado.

A consistência e a integridade dos dados são asseguradas por meio de mecanismos internos de replicação, deteção de falhas e auto-recuperação. Quando um OSD falha, o RADOS reorganiza automaticamente os dados, com base nas regras do CRUSH, para restaurar a resiliência configurada.

Daemons RADOS

Ceph consiste em vários daemons em execução sobre a camada RADOS, cada um com uma função específica dentro do *cluster*:

- **MON (Monitor):** o daemon de monitorização do Ceph gere o estado global do *cluster*, mantendo os mapas de configuração e assegurando o consenso entre os nós.
- **MGR (Manager):** fornece informações de monitorização do *cluster*, métricas de desempenho e suporte para interfaces de gestão e ferramentas de orquestração. Trabalha em conjunto com os MONs para fornecer uma visão mais rica e detalhada do estado do *cluster*.
- **OSD (Object Storage Daemon):** responsável por armazenar fisicamente os dados, replicá-los, equilibrar a carga e proceder à recuperação dos dados em caso de falha. Uma das características mais relevantes do Ceph é que os clientes podem comunicar diretamente com os OSDs que possuem a cópia primária dos dados, evitando a necessidade de recorrer a um servidor centralizado para obter metadados. Isto elimina potenciais congestionamentos de desempenho e pontos únicos de falha, garantindo maior escalabilidade e resiliência. [1]

Replicação no Ceph:

A replicação no Ceph garante que os dados sejam duplicados em vários locais dentro do *clus-*

ter, protegendo-os contra falhas de discos, nós ou outros componentes físicos, permitindo ainda criptografia dos dados ao nível dos OSDs, protegendo dados sensíveis.

Por padrão, o Ceph cria três réplicas de cada objeto, propagando-os por servidores diferentes. Contudo, o administrador pode ajustar o número de réplicas e escolher onde estas são armazenadas (em discos diferentes, nós distintos ou até em *racks* separados) por meio das configurações definidas em cada pool, de acordo com as necessidades de desempenho e segurança. O algoritmo CRUSH é responsável por gerir essas decisões.

Além da replicação simples, o Ceph utiliza uma técnica que permite dividir os dados em pedaços e adicionar ”paridade”, permitindo a recuperação completa das informações mesmo com perdas de dados.

No processo de leitura, o Ceph normalmente tenta acessar os dados diretamente a partir do OSD primário (o primeiro OSD designado pelo algoritmo CRUSH como responsável principal pelo objeto). Essa preferência existe porque o OSD primário mantém a visão mais atual e consistente do estado do objeto, especialmente em operações simultâneas de leitura e escrita. [37]

Durante uma operação de escrita, o OSD primário recebe a atualização primeiro, aplica a modificação e, em seguida, propaga as alterações para as réplicas secundárias, garantindo a consistência entre as cópias.

Caso o OSD primário não esteja disponível durante a leitura, o sistema redireciona o acesso para uma réplica secundária, realizando verificações adicionais de integridade para assegurar que os dados estão íntegros e atualizados. Essa lógica pode ser ajustada via linha de comando.

Algoritmo CRUSH

O CRUSH (*Controlled Replication Under Scalable Hashing*) é o algoritmo utilizado pelo Ceph para distribuir os dados de forma eficiente, escalável e sem pontos únicos de falha. Em vez de recorrer a uma tabela centralizada para indicar onde cada fragmento de dados está armazenado, o CRUSH utiliza uma função matemática que calcula, de forma determinística, onde um determinado

objeto deve ser guardado no *cluster*. Esta abordagem permite que qualquer cliente ou daemon do Ceph saiba imediatamente onde se encontram os dados, sem necessidade de consultar uma base de dados central.

O funcionamento do CRUSH baseia-se em regras definidas pelo administrador, que indicam como os dados devem ser replicados e distribuídos.

Quando o Ceph necessita de guardar ou aceder a um objeto, o CRUSH utiliza informações como o nome do objeto, o número de réplicas, a topologia do *cluster* (OSDs, nós, bastidores, etc.) e as políticas de colocação para calcular exatamente onde o objeto deve ser armazenado. Este processo é rápido e eficiente e, como é determinístico, o sistema pode sempre calcular a localização correta dos dados sem comunicações adicionais. Além disso, ao decidir onde guardar os dados, o CRUSH considera o peso atribuído a cada OSD, um valor que normalmente é definido proporcionalmente à capacidade total de armazenamento de cada disco. Esse peso influencia a frequência com que um OSD é escolhido para armazenar dados, fazendo com que discos com maior peso (e, geralmente, maior capacidade) recebam mais dados, garantindo uma distribuição equilibrada no *cluster* (o valor do peso pode ser alterado manualmente). [36]

Graças ao CRUSH, o Ceph consegue escalar para milhares de discos e nós, mantendo uma distribuição equilibrada dos dados, uma elevada resiliência e uma recuperação automática eficiente juntamente com o RADOS. Este algoritmo é uma das principais razões pelas quais o Ceph é tão robusto, auto-gerido e adequado para ambientes de armazenamento distribuído em grande escala.

Tipos de Armazenamento Ceph

O Ceph disponibiliza três tipos principais de armazenamento, cada um adaptado a diferentes necessidades e cenários de utilização. Todos estes tipos funcionam sobre o RADOS, o motor de armazenamento distribuído do Ceph:

- **Armazenamento de Objetos (Ceph Object Gateway ou RADOS Gateway - RGW):** permite armazenar e aceder a dados sob a forma de objetos, sendo ideal para aplicações web, bac-

kups e grandes volumes de dados não estruturados. Integra-se com os protocolos S3 (Amazon) e Swift (OpenStack) e fornece acesso via HTTP através de uma interface RESTful.

Nota: Apesar de ser uma componente oficial do ecossistema Ceph, o Ceph Object Gateway (RGW) não é suportado nativamente pelo Proxmox VE. A sua utilização requer configuração manual externa e não está integrada na interface de gestão do Proxmox, ao contrário do RBD e do CephFS, que são totalmente suportados.

- **Armazenamento em Bloco (RADOS Block Device - RBD):** fornece volumes de armazenamento que podem ser montados como discos em sistemas operativos. É usado em ambientes de virtualização como Proxmox e OpenStack, sendo ideal para VMs ou bases de dados que exijam armazenamento persistente e de alto desempenho. Cada volume é dividido em objetos e distribuído automaticamente pelo *cluster*.
- **Sistema de Ficheiros (Ceph File System - CephFS):** é um sistema de ficheiros distribuído e compatível com POSIX, permitindo o acesso simultâneo a ficheiros por múltiplos clientes. É especialmente indicado para partilhas de ficheiros, ambientes científicos e outras aplicações que beneficiem de acesso concorrente. O CephFS utiliza servidores próprios, chamados *Metadata Server* (MDS), para gerir os metadados do sistema de ficheiros, como nomes de ficheiros, permissões e estrutura de diretórios. Estes servidores MDS são daemons próprios que podem correr em nós dedicados ou partilhados com outros serviços do *cluster*. Os metadados que os MDS gerem, são armazenados em uma pool específica (pool de metadados), enquanto os dados dos ficheiros propriamente ditos são armazenados como objetos RADOS em outra pool (pool de dados). Esta separação permite otimizar o acesso aos dados, abstraindo a complexidade do sistema de ficheiros e garantindo um acesso mais rápido e eficiente aos dados armazenados. [1]

O BlueStore é uma camada de armazenamento de baixo nível no Ceph, projetada para otimizar a maneira como os dados são armazenados diretamente nos equipamentos de armazenamento (OSDs), como discos rígidos (HDDs) ou discos de estado sólido (SSDs). Sendo introduzido para substituir

o antigo método de armazenamento utilizado pelo FileStore, que dependia de sistemas de ficheiros tradicionais como xfs ou ext4. [38]

O BlueStore tem como objetivo melhorar o desempenho e reduzir a sobrecarga associada ao uso de sistemas de ficheiros tradicionais.

Na Figura 10 apresenta a arquitetura do armazenamento Ceph.

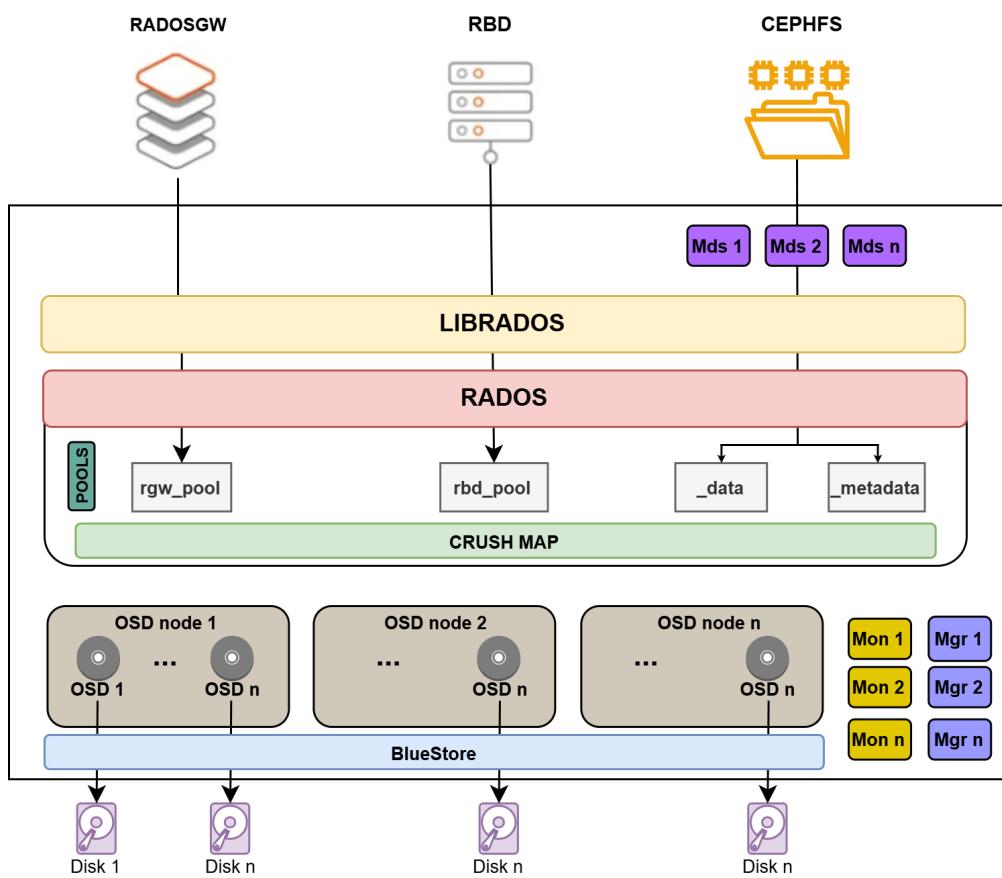


Figura 10: Arquitetura do Ceph

A Figura 10 representa a arquitetura do Ceph, juntamente com a estrutura dos diferentes tipos de armazenamento suportados: objetos (RGW), blocos (RBD) e sistemas de ficheiros (CephFS). Na base do sistema encontram-se os OSD nodes, que correspondem a nós distintos do *cluster*. Cada OSD node contém um ou mais OSDs, responsáveis por armazenar os dados em discos físicos através do sistema BlueStore. Acima desta camada está o RADOS, que gere a distribuição e replicação dos

dados entre os OSDs com base no CRUSH map, organizando-os em pools. Sobre o RADOS, o Librados fornece uma interface que permite o acesso aos dados pelos diferentes clientes. O sistema é ainda suportado pelos nós de monitorização (MON), pelos gestores de *cluster* (MGR) e pelos servidores de metadados (MDS) no caso do uso do CephFS [36].

2.3.8 Comparação do Armazenamento

O Ceph destaca-se como a solução mais escalável e resiliente disponível no Proxmox VE, sendo ideal para ambientes distribuídos. Permite alta disponibilidade, replicação automática e migração ao vivo de VMs. No entanto, essa robustez exige uma infraestrutura mais exigente, com pelo menos três nós físicos, ligação de rede de alto desempenho e maior consumo de recursos, tanto em termos de CPU como de RAM. Os restantes tipos de armazenamento apresentam configurações mais simples e são mais adequados a contextos com menor complexidade.

Para uso local e em ambientes com poucos recursos, o LVM Thin pode ser a opção mais indicada, pois permite *snapshots* e um melhor aproveitamento do espaço em disco. Quando há necessidade de redundância e integridade dos dados, o ZFS surge como uma excelente alternativa, graças à sua fiabilidade, *snapshots* eficientes e compressão integrada, além de suportar diferentes níveis de redundância, garantindo proteção contra falhas de disco. Contudo, exige uma maior quantidade de memória RAM.

O iSCSI e o NFS são protocolos essenciais para a partilha de armazenamentos locais entre diferentes nós, sendo o NFS mais simples de configurar.

A Tabela 8 apresenta uma comparação entre os principais tipos de armazenamento suportados pelo Proxmox VE, tendo em conta critérios relevantes para ambientes de virtualização, como suporte a *thin provisioning*, *snapshots*, partilha entre nós, alta disponibilidade, migração ao vivo de VMs e suporte backups. Esta comparação visa auxiliar na escolha do sistema de armazenamento mais adequado, dependendo das necessidades e recursos de cada infraestrutura.

Storage	Level	Thin	Snapshots	Shared	HA	Live Migration	Backups
LVM	Bloco	Não	Não	Não	Não	Não	Sim
LVM-Thin	Bloco	Sim	Sim	Não	Sim	Sim	Sim
ZFS (local)	Ambos	Sim	Sim	Não	Sim	Sim	Sim
iSCSI	Bloco	Depende	Não	Sim	Sim	Sim	Não
NFS	Ficheiro	Depende	Sim	Sim	Sim	Sim	Sim
Ceph RBD	Bloco	Sim	Sim	Sim	Sim	Sim	Sim
CephFS	Ficheiro	Sim	Sim	Sim	Sim	Sim	Sim
Directory	Ficheiro	Depende	Depende	Não	Não	Depende	Sim

Tabela 8: Comparação de Tipos de Armazenamento no Proxmox VE

Nota: As células marcadas com *Depende* indicam que a funcionalidade não é garantida pelo tipo de armazenamento em si, mas pode estar disponível dependendo da configuração.

2.4 Soluções Existentes de *Private Cloud*

Com o objetivo de compreender melhor as abordagens mais comuns no mercado e no meio académico, esta secção analisa três soluções e arquiteturas de *private cloud* já implementadas, explorando as tecnologias utilizadas, os modelos de gestão adotados, os desafios enfrentados e as melhores práticas identificadas. Pretende-se, com esta análise, compreender as abordagens mais comuns no mercado e no meio académico, bem como extraír aprendizagens relevantes para o desenvolvimento e implementação de novas soluções neste contexto.

2.4.1 *The practice of developing the academic cloud using the Proxmox*

Este estudo foi desenvolvido por Vasyl P. Oleksiuk e Olesia R. Oleksiuk (2021) [39], descreve a criação e evolução de uma *cloud* académica privada na Ternopil Volodymyr Hnatiuk National Pedagogical University, na Ucrânia.

O principal objetivo deste estudo é descrever um modelo de *cloud* aplicado à formação de

professores de informática, sistematizando a experiência da sua implementação com recurso à plataforma Proxmox VE.

A *private cloud* teve como base inicial o Apache CloudStack. No entanto, após vários anos de utilização, começaram a surgir dificuldades técnicas significativas, o que levou à decisão de migrar para o Proxmox VE. Entre os principais problemas identificados estavam a má alocação de recursos computacionais, a complexidade na gestão, a transferência de VMs entre utilizadores, limitações no sistema de backups e a inexistência de uma aplicação móvel.

Face a estas limitações, os autores optaram por adotar uma nova infraestrutura baseada no Proxmox VE. A escolha recaiu sobre esta plataforma por ser gratuita e *open-source*, suportar o modelo IaaS, apresentar simplicidade técnica na sua implementação e manutenção. Adicionalmente, permite autenticação através de bases de dados externas, suporte a *templates* e backups de discos virtuais, bem como escalabilidade através da adição de novos nós e armazenamento.

O estudo demonstra como é possível, mesmo num contexto de recursos limitados, implementar uma infraestrutura de *cloud* académica eficaz e escalável. A migração do Apache CloudStack para o Proxmox VE superou as limitações antes existentes. A experiência partilhada pelos autores evidencia o potencial do Proxmox VE como base para ambientes educativos virtualizados. Este caso de estudo representa uma referência útil para outras instituições que pretendam adotar soluções *open-source* no desenvolvimento de *clouds* privadas orientadas à educação [39].

2.4.2 Educational Resource Private Cloud Platform Based on OpenStack

Este estudo foi desenvolvido por Linchang Zhao, Guoqing Hu e Yongchi Xu (2024) [40], descreve a criação e implementação de uma *cloud* académica privada na Guiyang University, na China. O trabalho propõe uma solução de infraestrutura digital orientada para o ensino, baseada em tecnologias *open-source*, OpenStack e Ceph.

O principal objetivo foi fornecer um ambiente de ensino flexível, seguro e escalável, permitindo a gestão eficiente de recursos educativos, personalização de serviços, controlo dos dados e alta

disponibilidade.

A infraestrutura implementada segue uma arquitetura modular distribuída com suporte a alta disponibilidade (HA), dividida em várias camadas: infraestrutura, gestão de recursos e aplicação. É composta por 13 servidores físicos, interligados para garantir redundância e desempenho.

A *private cloud* criada permite o acesso em modo *self-service* a recursos como VMs, sistemas operativos personalizados, ambientes de ensino, plataformas Learning Management Systems (LMS) e ferramentas de simulação. A camada de armazenamento, baseada em Ceph, garante redundância dos dados, tolerância a falhas e migração em tempo real de VMs.

O estudo demonstra que a implementação do OpenStack em conjunto com o Ceph permite alcançar níveis elevados de desempenho, fiabilidade e flexibilidade. Esta solução constitui, assim, uma referência valiosa para outras instituições de ensino que pretendam implementar infraestruturas de *private cloud* robustas, seguras e economicamente sustentáveis, com recurso a tecnologias *open-source* [40].

2.4.3 *Implementation - Surplus resources to Private Cloud*

Este estudo foi desenvolvido por Harsh Oswal, Shivani Bhalsakle e Minal Swami (2022) [41], descreve a criação e avaliação de uma *private cloud* construída a partir de recursos computacionais excedentários (surplus), como forma de reaproveitar equipamentos funcionais, mas obsoletos.

O principal objetivo consistiu em integrar diversos equipamentos antigos numa única *private cloud*, capaz de executar VMs e CTs, avaliando o respetivo desempenho com diferentes hipervisores. A solução proposta visa não só reutilizar *hardware*, como também oferecer uma alternativa viável e económica para instituições com recursos limitados.

A infraestrutura foi composta por três servidores físicos, cada um com 8 GB de RAM, usando hipervisores de tipo 1 — nomeadamente Proxmox VE 7.0 (*open-source*) e VMware ESXi 6.7.

O estudo incluiu uma análise comparativa do desempenho de CTs e VMs em ambos os hipervi-

sores, bem como uma comparação entre os recursos de memória e armazenamento. Foram também derivadas fórmulas matemáticas que permitem estimar a quantidade de recursos utilizáveis numa *cloud* construída com equipamentos semelhantes.

Os resultados demonstraram que os hipervisores de tipo 2 são ineficazes neste contexto, devido ao desperdício de recursos provocado pela necessidade de executar um sistema operativo base. Por outro lado, os hipervisores de tipo 1 revelaram-se mais adequados, oferecendo melhor desempenho e menor sobrecarga. O Proxmox destacou-se pelo suporte nativo a CTs e pela facilidade de gestão via interface web, enquanto o ESXi apresentou melhor desempenho na execução de VMs, embora sem suporte nativo a CTs.

Este trabalho demonstra que é possível construir uma *private cloud* funcional e eficiente a partir de recursos excedentários, contribuindo para a sustentabilidade digital e para a redução dos resíduos eletrónicos [41].

2.4.4 Implementation – OpenStack Multinode Deployment for Private Cloud

Este estudo, desenvolvido por Ciptaningtyas et al.(2023) [42], descreve a implementação e avaliação de uma infraestrutura de *private cloud* construída com recurso à plataforma *open-source* OpenStack, recorrendo a um método de implantação multinó baseado na ferramenta Kolla Ansible.

O principal objetivo consistiu em implementar uma solução de virtualização privada no Departamento de Tecnologias de Informação (ITS), capaz de executar múltiplas instâncias de máquinas virtuais com diferentes configurações e de permitir a criação de ambientes de treino cibernético com a plataforma KYPO Cyber Range Platform. A abordagem adotada visou igualmente demonstrar a viabilidade da utilização de hardware convencional e de ferramentas de automatização para facilitar a gestão e a escalabilidade da *cloud*.

A infraestrutura foi composta por quatro tipos de nós: um nó de controlo, dois nós de computação, um nó de armazenamento em bloco e um nó gateway NAT. Cada equipamento utilizava sistemas modestos, com CPUs Intel Core i3/i5 e memória RAM entre 4 GB e 12 GB. A rede foi

segmentada entre a rede de gestão (interna ao OpenStack) e a rede de acesso externo, com NAT configurado para contornar as restrições de acesso à Internet da instituição.

A implementação recorreu ao Kolla Ansible para automatizar a configuração dos serviços OpenStack em *containers*, simplificando o processo e aumentando a portabilidade do sistema. O desempenho foi avaliado através de ferramentas como LINPACK (CPU), STREAM (memória), IPerf e Speedtest (rede), e Rally-OpenStack (escala e concorrência).

Os resultados demonstraram que o desempenho dos recursos virtuais degrada com o aumento do número de instâncias ativas, com uma redução média de 1,41% no desempenho do CPU e de 4,46% na memória. A criação de instâncias manteve-se estável mesmo com múltiplos pedidos simultâneos, mas a alteração de configurações de VMs revelou instabilidade a partir de cinco pedidos diferentes. A infraestrutura permitiu ainda a execução da plataforma KYPO, embora limitada a um único *sandbox pool*, devido à escassez de recursos físicos.

Este estudo evidencia a eficácia da utilização de OpenStack em contextos educativos ou institucionais com recursos limitados, demonstrando que a automatização com Kolla Ansible e o aproveitamento de hardware convencional permitem criar uma *private cloud* funcional e escalável [42].

2.4.5 Implementation – JFE Steel Private Cloud Deployment with OpenStack

No setor industrial, a JFE Steel, uma das maiores empresas siderúrgicas do Japão, descreveu em 2023 a implementação de uma infraestrutura de *private cloud* corporativa, denominada “J-OSCloud”, baseada na plataforma *open-source* OpenStack [43].

A arquitetura da solução contempla a instalação da nuvem privada em dois data centers geograficamente redundantes, localizados nas regiões leste e oeste da empresa, para garantir alta disponibilidade e resiliência. A infraestrutura foi projetada para automatizar o provisionamento de servidores e permitir a padronização de imagens e templates de máquinas virtuais, facilitando a escalabilidade e a gestão dos recursos computacionais internos.

A plataforma utiliza diversos módulos essenciais do OpenStack, incluindo Nova para orques-

tração de computação, Glance para gerenciamento de imagens, Cinder para armazenamento em bloco, Keystone para autenticação, Horizon para a interface web e Neutron para a rede virtualizada. Além disso, o ambiente suporta múltiplos hipervisores padrão do mercado, como KVM e Xen, bem como soluções comerciais como VMware ESX e Microsoft Hyper-V, conferindo flexibilidade na escolha da camada de virtualização.

A implantação dessa arquitetura *on-premise* permitiu à JFE Steel reduzir significativamente o tempo de configuração da infraestrutura, de meses para apenas alguns dias, melhorando a agilidade na disponibilização dos serviços internos e promovendo uma gestão eficiente da nuvem privada da empresa [43].

2.5 Síntese

Neste capítulo foram abordados os principais conceitos e tecnologias relevantes para a implementação de infraestruturas de *private cloud*. Apresentaram-se os fundamentos da virtualização e os diferentes tipos de hipervisores, com especial ênfase na comparação entre hipervisores de tipo 1, destacando-se o Proxmox VE como a solução mais adequada para este projeto, pela sua flexibilidade, funcionalidades avançadas e natureza *open-source*. Foram igualmente analisados os principais tipos de armazenamento suportados pelo Proxmox VE, com especial destaque para o Ceph, pela sua escalabilidade e resiliência. A comparação entre soluções permitiu identificar as opções mais adequadas consoante os requisitos e recursos disponíveis. Por fim, a análise de casos reais contribuiu para a identificação de boas práticas, desafios recorrentes e para observar quais as ferramentas que estão a ser usadas atualmente no mercado.

No próximo capítulo é apresentada a arquitetura e os mecanismos de implementação da solução proposta *Private Cloud Services*, baseada em Proxmox VE e Ceph, com o objetivo de criar uma infraestrutura robusta, resiliente, segura e escalável, adequada a ambientes críticos como os académicos ou empresariais.

3 Private Cloud Services

Neste capítulo, é apresentada a arquitetura e os mecanismos de implementação da solução proposta *Private Cloud Services* baseada em Proxmox VE e Ceph. Esta proposta visa criar uma infraestrutura altamente robusta, resiliente, segura e escalável, adequada a ambientes críticos, como ambientes académicos ou empresariais.

A solução assume um cenário ideal, permitindo a integração das melhores práticas e tecnologias disponíveis. O objetivo é garantir disponibilidade contínua dos serviços, resiliência total a falhas físicas ou lógicas, desempenho elevado mesmo em situações de carga intensa, e gestão centralizada inteligente, com capacidades de monitorização proativa e automação de tarefas.

Nas secções seguintes, descrevemos os requisitos funcionais e não funcionais da solução, detalhamos os módulos principais, e apresentamos a arquitetura final proposta, integrando todos os componentes num sistema coeso e de elevada fiabilidade.

3.1 Requisitos

Os requisitos desta proposta estão organizados em duas categorias: funcionais, que definem as funcionalidades esperadas do sistema, e não funcionais, que estabelecem os níveis de desempenho, segurança e fiabilidade.

3.1.1 Requisitos Funcionais

- Permitir a criação, gestão e monitorização de máquinas virtuais (KVM) e CTs (LXC) em alta disponibilidade.
- Suportar migração automática e manual de VMs entre nós sem interrupção de serviço (*live migration*).
- Disponibilizar armazenamento distribuído e replicado via Ceph (RBD para blocos, CephFS para partilha de ficheiros).

- Implementar backups incrementais e automáticos com o Proxmox Backup Server.
- Oferecer redundância física e lógica a nível de rede, alimentação elétrica e armazenamento.
- Permitir a execução de cargas computacionais intensivas (AI/ML) nos servidores especializados.
- Fornecer acesso seguro e remoto à infraestrutura via VPN, com suporte a autenticação de dois fatores (2FA).
- Disponibilizar um painel de administração.
- Suportar escalabilidade horizontal com integração dinâmica de novos nós ao *cluster* e ao Ceph.
- Monitorizar continuamente o estado da infraestrutura.
- Possibilitar a replicação de serviços críticos e a recuperação rápida de desastres.

3.1.2 Requisitos Não Funcionais

- Garantir alta disponibilidade dos serviços com failover automático e *quorum* distribuído.
- Assegurar resiliência a falhas de hardware (discos, servidores, switches) sem perda de dados.
- Fornecer desempenho elevado, com baixa latência e elevada capacidade de IOPS, mesmo em condições de carga intensa.
- Minimizar o tempo de indisponibilidade durante atualizações e manutenções (zero *downtime* planeado).
- Garantir a consistência e integridade dos dados com replicação síncrona e controlo de versões.
- Suportar isolamento de tráfego de gestão, dados e Ceph através de VLANs dedicadas.
- Utilizar UPS em todos os segmentos críticos para garantir continuidade em caso de falha elétrica.

- Garantir a segurança dos dados com encriptação dos dados armazenados (em disco) e dos dados transmitidos pela rede (via VPN e firewall).
- Suportar auditoria completa com logs centralizados e capacidade de análise detalhada.
- Fornecer flexibilidade de expansão, tanto ao nível do armazenamento como do processamento.
- Garantir compatibilidade com ferramentas *open-source*, sem dependência de soluções proprietárias.

3.2 Análise de Problemas e Abordagens

A definição dos requisitos e a estrutura da arquitetura proposta foram baseadas na análise da literatura e em boas práticas identificadas em projetos e implementações reais de ambientes de *datacenter* e *cloud*. A Tabela 9 apresenta esta análise, relacionando cada problema identificado com abordagens específicas através do Proxmox VE e Ceph , bem como as respetivas métricas de avaliação. Desta forma, serve como um suporte direto na tomada de decisão para a escolha da solução mais adequada.

Problemas Comuns	Desafios	Abordagens	Métricas de Comparação
Alta disponibilidade	Falhas de nó ou rede	Proxmox HA + <i>watchdog</i> ¹ + <i>quorum</i> ² de 3 nós + Ceph + failover automático	Tempo de failover
Armazenamento confiável	Tolerância a falhas de disco	Ceph replicado (3x/2x) + auto-recovery + <i>hot swapping</i> ³	<i>IOPS</i> ⁴ , latência (ms), <i>RTO</i> ⁵ / <i>RPO</i> ⁶
Atualizações seguras	Evitar <i>downtime</i> nas VMs	Live migration + manutenção em fases	<i>Downtime</i> médio, nº de VMs afetadas
Balanceamento de carga	Nós sobrecarregados	Migração automática baseada na carga	Utilização CPU/RAM, nº migrações

Problemas Comuns	Desafios	Abordagens	Métricas de Comparação
Segurança da infraestrutura	Acesso não autorizado	VPN, 2FA, firewall, Ceph encriptado	Nº vulnerabilidades, tempo de resposta
Escalabilidade horizontal	Adicionar nós sem impacto	Integração dinâmica com rebalanceamento Ceph + <i>hot swapping</i> ³	Tempo de integração, impacto do desempenho
Backups consistentes	Restauração rápida após falhas	Proxmox Backup Server com snapshots	Tempo de backup/restauro, uso do disco
Sincronização de tempo	Problemas com <i>quorum</i> ² /logs	NTP sincronizado em todos os nós	Desvio horário (ms), estabilidade do <i>quorum</i> ²
Failover automático	Continuidade do serviço	<i>Fencing</i> ⁷ automático + Live migration para VMs e CTs	Tempo de recuperação, nº de falhas detetadas
Ausência de <i>fault tolerance</i> ⁸	Perda de estado em falhas inesperadas	Snapshots em Ceph + backups externos + replicação de serviços críticos	Tempo de indisponibilidade, perda de dados
Monitorização contínua	Deteção precoce de problemas	InfluDB + Prometheus + Grafana + alertas (Alertmanager)	Tempo de resposta, nº de alertas críticos
Conformidade e auditoria	Requisitos legais e técnicos	Logs centralizados, auditoria contínua, RBAC ⁹	Nº de não conformidades, tempo de auditoria
Migração entre <i>datacenters/clouds</i>	Manter consistência e continuidade	Replicação de VMs, DNS dinâmico, scripts de failover	Tempo de migração, consistência de dados

Tabela 9: Problemas, desafios, abordagens e métricas em ambientes de *datacenter/cloud* com Proxmox e Ceph

¹**Watchdog:** serviço que deteta falhas e reinicia automaticamente os nós que deixaram de responder.

²**Quorum:** número mínimo de nós que precisam de estar ativos e em comunicação para tomar decisões no *cluster*.

³**Hot swapping:** capacidade de adicionar ou remover componentes (ex: discos) com o sistema ligado, sem desligar o servidor.

⁴**IOPS – Input/Output Operations Per Second:** Quantidade de operações de leitura/escrita que o sistema consegue realizar por segundo.

⁵**RTO – Recovery Time Objective:** Tempo máximo tolerável para restaurar o serviço após uma falha.

⁶**RPO – Recovery Point Objective:** Quantidade máxima de dados que se pode perder em caso de falha.

⁷**Fencing automático:** é uma técnica usada em clusters de alta disponibilidade (como no Proxmox com HA) para isolar ou reiniciar automaticamente um nó que deixou de responder, a fim de proteger a integridade do *cluster* e dos dados.

⁸**Fault tolerance:** capacidade de continuar as operações sem interrupções mesmo em caso de falha de hardware/-software, mantendo o estado da aplicação (ex: VMware FT). Proxmox VE não implementa esta funcionalidade.

⁹**RBAC (Role-Based Access Control):** controla o acesso com base em funções pré-definidas.

3.3 Proposta de Arquitetura da Solução

Na Figura 11 é apresentada a arquitetura global da solução ideal proposta, composta por um ou mais clusters de servidores interligados por uma infraestrutura de rede redundante. Cada cluster integra, no mínimo, três servidores híbridos, responsáveis simultaneamente pelo armazenamento distribuído (Ceph) e pela execução de VMs/CTs, e dois servidores dedicados exclusivamente a tarefas de AI/ML com recurso a GPUs dedicadas.

A rede foi desenhada para garantir isolamento de tráfego e elevado desempenho, utilizando *bonding* com *failover* e switches redundantes. A alimentação elétrica é assegurada por UPS independentes, com monitorização integrada. Todos os elementos da infraestrutura estão integrados num *cluster* de alta disponibilidade, com suporte a migração de serviços, replicação de dados e monitorização centralizada.

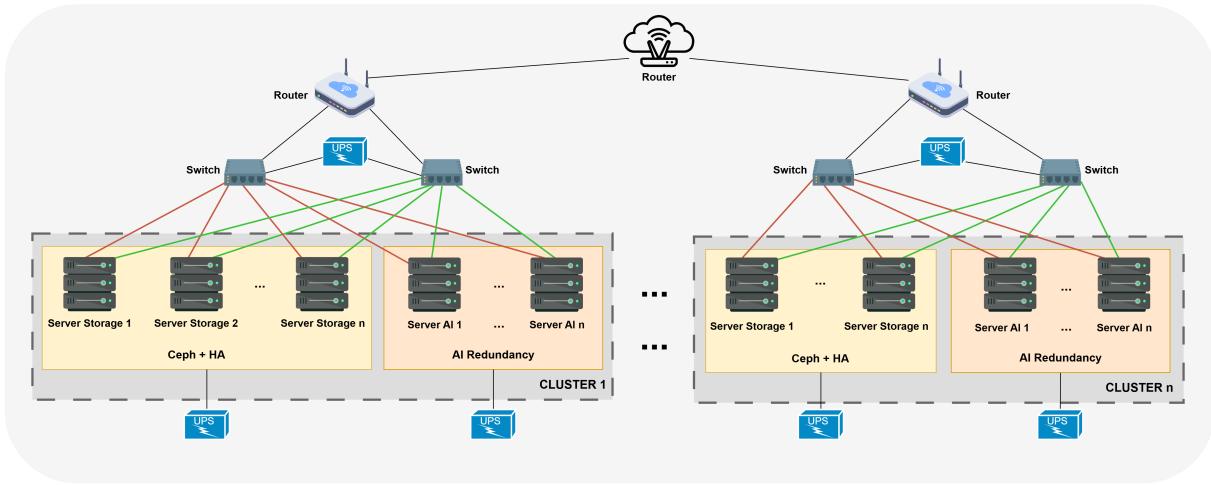


Figura 11: Arquitetura da solução proposta

3.4 Discussão

A proposta apresentada neste capítulo define uma solução ideal da *Private Cloud Services*, baseada em tecnologias como o Proxmox VE e o Ceph, e pensada para garantir elevada disponibilidade, resiliência e escalabilidade. Através da análise de problemas comuns, da definição rigorosa de requisitos e da organização modular da infraestrutura, foi possível conceber uma arquitetura robusta, adaptada a ambientes críticos e exigentes. Esta proposta representa o cenário de referência sobre o qual será desenvolvida a implementação real, considerando os recursos disponíveis e limitações práticas, que serão abordadas no capítulo seguinte.

3.5 Módulos da Solução

A solução proposta está organizada em seis módulos principais, cada um com responsabilidades específicas para garantir a robustez, fácil escalabilidade e eficiência da infraestrutura.

3.5.1 Armazenamento

O módulo de armazenamento Ceph+HA apresentado na Figura 11, assenta em pelo menos três servidores dedicados, responsáveis por executar os serviços OSD, MON e MGR do Ceph. Estes

servidores constituem a base do armazenamento distribuído, com replicação 3x, escolhida para garantir um equilíbrio entre segurança dos dados e eficiência no uso do espaço, assegurando que cada bloco de dados é armazenado em três locais distintos para tolerância a falhas sem comprometer demasiado a capacidade total. O sistema suporta tanto RBD, utilizado para fornecer volumes de blocos virtuais a VMs e CTs, quanto o CephFS, destinado à disponibilização de um sistema de ficheiros distribuído com suporte a diretorias.

A escolha do Ceph justifica-se pelas suas capacidades avançadas de escalabilidade, replicação automática, alta disponibilidade e integração com funcionalidades como *live migration* e *snapshots*. Estas características, como descrito na comparação de soluções de armazenamento (mencionado na secção 2.3.8), o destacam como a opção mais robusta e adequada para ambientes distribuídos, apesar das exigências superiores de hardware.

Além disso, todos os servidores presentes na arquitetura proposta incluem uma camada de armazenamento local baseada em LVM, utilizada para alojar ficheiros essenciais ao funcionamento do Proxmox VE, imagens ISO, *templates* de VMs/CTs e outros recursos que não necessitam de ser distribuídos. Esta camada local é necessária para garantir desempenho otimizado e disponibilidade imediata destes ficheiros, além de assegurar que o sistema operativo dos nós não seja partilhado entre eles, evitando possíveis conflitos e problemas de sincronização.

3.5.2 Nós de Cálculo

Os nós de cálculo são os mesmos três servidores que compõem o módulo de armazenamento. Para além de suportarem o *cluster* Ceph, estes servidores são também responsáveis pela execução de VMs e CTs com requisitos gerais de processamento. Estão integrados num *cluster* Proxmox VE com *quorum* e oferecem suporte a alta disponibilidade, migração em tempo real e balanceamento de carga. Sendo utilizados para executar tarefas comuns, como bases de dados, serviços web e outras aplicações que não requerem uma GPU dedicada.

3.5.3 Infraestrutura de IA

A infraestrutura de Inteligência Artificial é composta por dois servidores identificados dentro do módulo AI Redundancy, dedicados exclusivamente à execução de tarefas de treino e inferência em AI/ML. Estes servidores estão equipados com GPUs de alto desempenho e configurados com o Proxmox VE, permitindo a execução de tarefas computacionalmente intensivas em paralelo. São isolados dos restantes nós para garantir estabilidade e elevado desempenho durante operações de treino e inferência. Além disso, estes servidores têm acesso ao armazenamento distribuído via Ceph, podendo consumir dados diretamente dos volumes partilhados, facilitando o processamento de grandes conjuntos de dados sem necessidade de replicação local.

3.5.4 Rede

A infraestrutura de rede deve estar segmentada por VLANs específicas para tráfego de gestão, Ceph e serviços. São utilizadas ligações agregadas (*bonding*) com *failover* ativo-passivo, e todos os servidores estão ligados a switches redundantes, garantindo elevada disponibilidade e desempenho. Os três servidores principais e os dois servidores dedicados a AI comunicam entre si através de ligações otimizadas, permitindo que todos acedam de forma eficiente aos dados partilhados pelo Ceph, com baixa latência e sem interferências no desempenho da rede.

3.5.5 Alimentação

Todos os servidores, switches e equipamentos críticos devem estar ligados a UPSs independentes. Os servidores de armazenamento estão alimentados por diferentes circuitos elétricos, garantindo resiliência a falhas energéticas localizadas. O estado das UPS é monitorizado via SMTP, com geração automática de alertas em caso de anomalia.

3.5.6 Backups

A proteção de dados é assegurada através do *Proxmox Backup Server (PBS)*, que centraliza e optimiza a gestão dos backups. O PBS é executado numa VM armazenada no Ceph, garantindo

disponibilidade mesmo em caso de falha de um dos nós. Para reforçar a resiliência e segurança, é recomendada a realização de cópias externas adicionais (em *cloud* ou outro servidor físico).

3.5.7 Monitorização

A infraestrutura é monitorizada através da integração das ferramentas Prometheus, Grafana, InfluxDB e Telegraf. O Prometheus recolhe métricas em tempo real, enquanto o InfluxDB armazena séries temporais para análise histórica. O Telegraf funciona como agente de recolha instalado em cada nó e equipamentos relevantes. O Grafana disponibiliza *dashboards* visuais, e os alertas são geridos pelo Alertmanager, com envio automático por e-mail.

3.6 Síntese

Este capítulo apresentou a proposta de arquitetura ideal da solução *Private Cloud Services*, baseada em Proxmox VE e Ceph. Foram definidos os requisitos funcionais e não funcionais da infraestrutura, tendo por base boas práticas e soluções adotadas em ambientes reais. Procedeu-se à análise dos principais problemas enfrentados em contextos de *cloud* e *datacenter*, relacionando-os com abordagens concretas que maximizam a disponibilidade, segurança, desempenho e escalabilidade da solução. A arquitetura proposta foi estruturada em módulos funcionais distintos, incluindo armazenamento, nós de cálculo, infraestrutura de Artificial Intelligence (AI), rede, alimentação, backups e monitorização, garantindo uma solução coesa, resiliente e adaptável a diferentes exigências operacionais.

Esta proposta servirá de base para a implementação prática que será descrita no capítulo seguinte, com a apresentação das escolhas técnicas, dos procedimentos de instalação e configuração realizados e da forma como a arquitetura ideal foi adaptada aos recursos disponíveis.

4 Implementação

Este capítulo descreve a implementação prática da solução de virtualização distribuída baseada em Proxmox VE, de acordo com a arquitetura definida no capítulo anterior. O objetivo é apresentar, de forma sistemática, as etapas realizadas, os componentes utilizados e as configurações aplicadas para construir um *cluster* funcional e resiliente.

A implementação abrange desde a preparação do ambiente físico até à configuração de serviços essenciais, como o sistema de armazenamento distribuído Ceph, mecanismos de *backup*, *snapshots*, entre outros. Serão abordados os objetivos do projeto, o *hardware* e *software* envolvidos, a topologia de rede, bem como os procedimentos de instalação, configuração e validação do *cluster*.

4.1 Objetivos da Implementação

O principal objetivo da implementação desta solução é criar um ambiente de virtualização distribuído baseado em Proxmox VE, com suporte a alta disponibilidade, tolerância a falhas, escalabilidade, backups, gestão centralizada e monitorização, utilizando os recursos disponíveis. Pretende-se replicar, na medida do possível, a arquitetura proposta no capítulo anterior, adaptando-a às limitações e capacidades do *hardware* e *software* acessíveis.

Este ambiente visa compreender e testar as funcionalidades oferecidas pelo Proxmox, incluindo a gestão de VMs e CTs, a configuração e funcionamento de *live migration*, a implementação de políticas de alta disponibilidade (HA), bem como a criação de *snapshots*, *templates*, backups e sistemas de monitorização.

Implementar uma solução distribuída de armazenamento, nomeadamente o Ceph, para avaliar a sua integração, fiabilidade e desempenho no contexto do *cluster*, considerando os recursos mais limitados disponíveis.

Esta solução será utilizada para a realização de um conjunto de testes de desempenho e resiliência do sistema, de modo a validar a eficácia da implementação e identificar possíveis pontos críticos,

garantindo uma infraestrutura o mais preparada possível para situações de falhas de *hardware* ou *software*.

4.2 Descrição da implementação

Nesta secção, apresenta-se uma visão detalhada da implementação do ambiente de virtualização. Serão descritos os principais componentes da infraestrutura, incluindo o *hardware* utilizado, as soluções de *software* adotadas, o armazenamento utilizado e a arquitetura de rede estabelecida. Esta descrição visa fornecer uma compreensão clara dos recursos e das configurações que sustentam a solução implementada.

4.2.1 *Hardware*

A infraestrutura é composta por três servidores físicos com especificações semelhantes, destinados à virtualização e ao armazenamento distribuído via Ceph. Na Tabela 10, são apresentadas as características técnicas principais de cada nó:

Nó	CPU	Disco Principal (sda)	Disco Secundário (sdb)	RAM
Proxmox nó 1	Intel(R) Core(TM)2 Quad CPU Q9550 @ 2.83GHz	250 GB	250 GB	8 GB
Proxmox nó 2	Intel(R) Core(TM)2 Quad CPU Q9400 @ 2.66GHz	500 GB	500 GB	8 GB
Proxmox nó 3	Intel(R) Core(TM)2 CPU 6600 @ 2.40GHz	500 GB	250 GB	8 GB

Tabela 10: Especificações de *hardware* dos servidores Proxmox

4.2.2 *Software*

Cada servidor executa o sistema operativo Proxmox VE 8.3, uma plataforma de virtualização baseada em Debian Linux, otimizada para a gestão de VMs e CTs. Adicionalmente, foi instalado

e configurado o Ceph Quincy (versão 17.2), utilizado como solução de armazenamento distribuído no *cluster*.

4.2.3 Armazenamento

O *cluster* utiliza o Ceph como solução principal de armazenamento distribuído, permitindo alta disponibilidade, replicação automática e acesso partilhado entre os nós. Paralelamente, cada nó possui um volume LVM local, utilizado para armazenar imagens ISO, *templates* de VMs e outros recursos que não necessitam de ser partilhados.

4.2.4 Rede

Todas as ligações Ethernet foram realizadas através de cabos RJ45, utilizando interfaces de rede limitadas a 100 Mbps. A comunicação entre os nós ocorre dentro da mesma gama de endereços IP, sem a utilização de sub-redes adicionais nem VLANs.

4.3 Arquitetura de Implementação

A arquitetura é composta por três nós físicos interligados em rede (*cluster*) e integrados a um sistema de armazenamento distribuído Ceph. Cada nó fornece recursos de computação e armazenamento, formando uma infraestrutura coesa e resiliente.

A comunicação entre os nós ocorre através de uma rede local dedicada, 172.79.79.0/24, com endereços IP estáticos atribuídos a cada servidor.

Cada nó dispõe de dois discos: o disco principal (sda) é utilizado exclusivamente para a instalação do sistema operativo (Proxmox VE), enquanto o disco secundário (sdb) é dedicado ao armazenamento distribuído via Ceph.

A Figura 12 apresenta uma visão geral da arquitetura lógica do *cluster*, incluindo os principais componentes e a interligação entre os nós. Esta representação permite compreender melhor a estrutura da solução implementada.

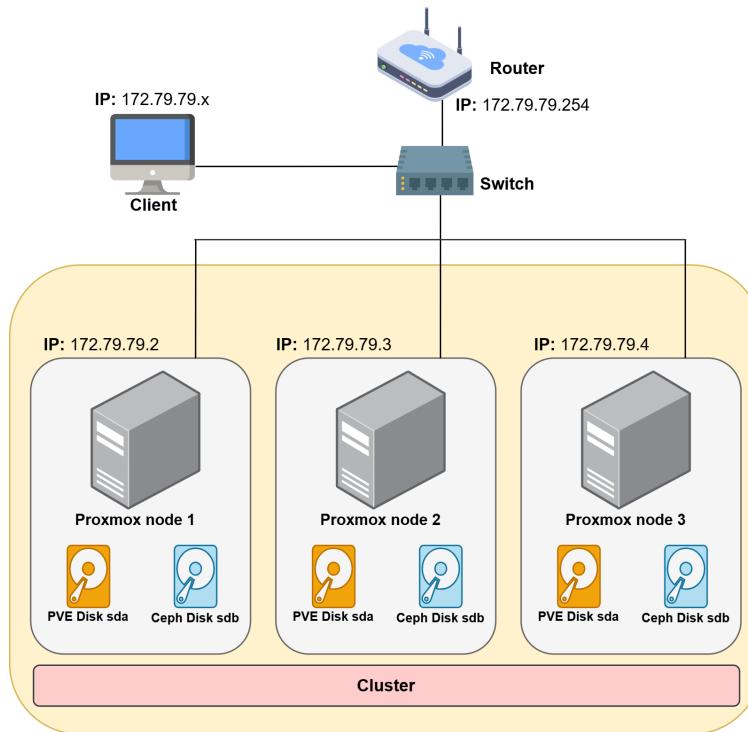


Figura 12: Arquitetura lógica da Implementação

4.4 Instalação do Proxmox VE em cada servidor

A instalação do Proxmox VE foi realizada localmente em cada um dos três servidores físicos, utilizando uma pen USB com a imagem oficial do sistema. A versão 8.3 do Proxmox VE, em formato ISO, foi descarregada diretamente do site oficial do Proxmox e transferida para uma pen USB de *boot* com recurso a uma ferramenta dedicada. Antes da instalação, acedeu-se à BIOS de cada máquina para garantir que a opção de virtualização estava ativada e que a inicialização por USB estava permitido.

Durante o processo de instalação, foi selecionado como disco para alojar o OS o disco principal (sda) de cada servidor, conforme descrito na Tabela 10. Foram igualmente definidos os parâmetros básicos de rede (Figura 12), nomeadamente o nome do *host* (*hostname*), o endereço IP estático, o *gateway* e o servidor DNS. Cada servidor recebeu o identificador único (proxmox1, proxmox2, proxmox3) e foi integrado na rede.

Após a conclusão da instalação, cada máquina foi reiniciada e, posteriormente, acedida remotamente através da interface web do Proxmox, disponível por defeito na porta 8006 via protocolo HTTPS. O acesso foi efetuado introduzindo o endereço correspondente a cada nó no browser, por exemplo, <https://172.79.79.x:8006>. A partir da interface gráfica, foi possível validar o estado da instalação, concluir as configurações iniciais e preparar cada nó para a criação no *cluster*.

4.5 Configuração

Após a instalação inicial do Proxmox VE em cada servidor, foram realizados apenas os ajustes essenciais para garantir o funcionamento correto da plataforma e preparar os nós para a criação do *cluster*.

Em primeiro lugar, os sistemas foram atualizados com os pacotes mais recentes, utilizando o gestor de pacotes APT. Esta atualização foi importante para assegurar a segurança e a estabilidade da distribuição:

```
apt update && apt dist-upgrade -y
```

De seguida, foi removido o repositório empresarial, utilizado por defeito nas instalações do Proxmox VE, uma vez que a versão em uso não estava associada a uma subscrição ativa. Esta alteração teve como objetivo eliminar o aviso apresentado na interface gráfica e permitir o acesso aos repositórios comunitários. Para tal, foi comentada a linha correspondente no ficheiro */etc/apt/sources.list.d/pve-enterprise.list* e ativado o repositório *pve-no-subscription*.

Durante os testes iniciais, foi identificado um problema de sincronização horária entre os servidores, provocado pela ausência de ligação a um servidor NTP funcional na rede local. Para contornar esta limitação, foi desenvolvido um *script* que permitia ajustar manualmente a hora em cada máquina, garantindo que todos os nós se encontravam temporariamente sincronizados e prontos para a formação do *cluster*.

Nota: apesar da execução do *script* de sincronização horária, os servidores mantiveram

uma diferença residual de 0.1s.

4.6 Criação do *cluster*

Após a configuração inicial dos três servidores Proxmox, procedeu-se à criação do *cluster*, permitindo a gestão centralizada e futura configuração de alta disponibilidade. A criação do *cluster* foi iniciada no nó proxmox1, através da interface gráfica, selecionando a opção *Datacenter > Cluster > Create Cluster*. Nesta etapa, foi atribuído um nome e indicado o endereço IP da rede do *cluster*, previamente definida para a comunicação entre os nós.

Uma vez criado o *cluster* no proxmox1, foi gerada a informação de adesão (*Join Information*) necessária para que os restantes nós se pudessem integrar no *cluster*. A partir dos nós proxmox2 e proxmox3, acedeu-se à mesma secção da interface gráfica (*Datacenter > Cluster*), onde foi utilizada a opção *Join Cluster*. Introduziu-se a informação do nó principal (endereço IP, nome do *cluster* e chave de autenticação), permitindo assim que os dois nós adicionais se integrassem corretamente no *cluster* já existente.

A criação do *cluster* foi finalizada com sucesso, tendo sido possível confirmar a presença dos três nós através da interface do *Datacenter*. No separador *Summary*, os três servidores são apresentados como nós ativos do *cluster*, com os respetivos nomes e estado atual. Esta confirmação validou a criação e sincronização correta do *cluster*, ficando os nós prontos para a configuração do armazenamento distribuído.

4.7 Configuração do Armazenamento

Após a conclusão da instalação do Proxmox VE, é criada automaticamente uma arquitetura de armazenamento, ilustrada na Figura 13.

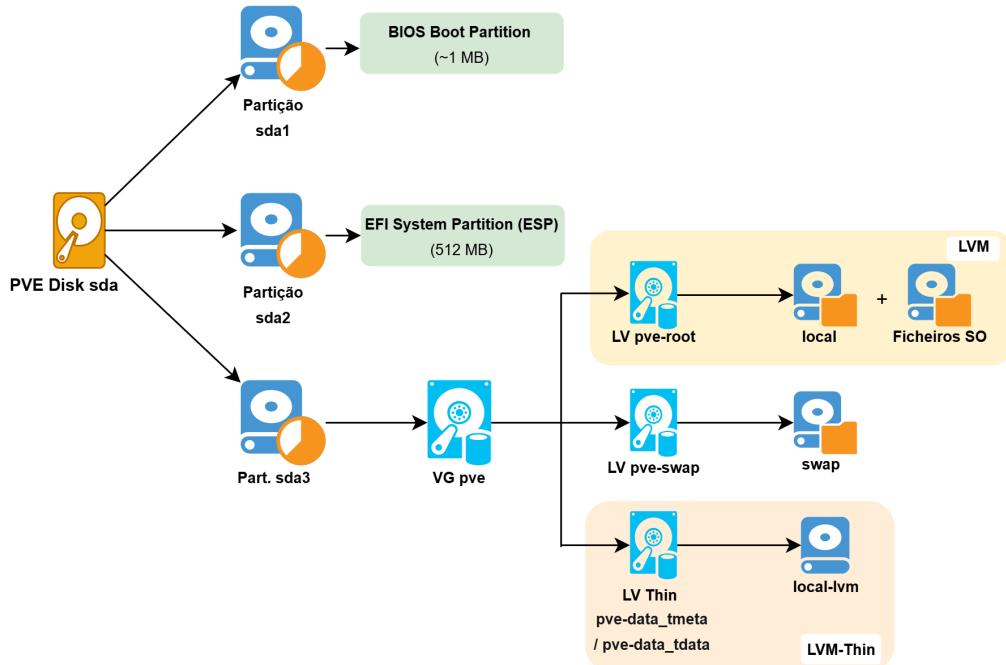


Figura 13: Arquitetura do armazenamento inicial do Proxmox

Essa configuração baseia-se no disco principal selecionado durante a instalação e segue uma estrutura padrão com três partições distintas:

- A **primeira partição** é dedicada ao BIOS boot, necessária para sistemas que arrancam em modo *legacy* com tabela de partições GPT.
- A **segunda partição** é reservada como EFI System Partition (ESP), utilizada em sistemas que arrancam em modo UEFI.
- A **terceira partição** ocupa o restante do disco e contém um Volume Group (VG) LVM denominado ”pve”, onde são criados três volumes lógicos (LVs):
 - **pve-root**: volume lógico do tipo LVM com sistema de ficheiros ext4, onde estão armazenados os ficheiros do sistema operativo, comportando-se como um armazenamento do tipo *Directory* no Proxmox.
 - **pve-swap**: volume dedicado à memória *swap*, utilizada quando a memória RAM está totalmente ocupada.

- **pve-data**: volume lógico do tipo LVM-Thin, criado a partir dos LVs *pve-data_tmeta* (metadados) e *pve-data_tdata* (dados).

Contudo, como foi utilizado o Ceph como sistema de armazenamento principal, o volume lógico pve-data foi eliminado, uma vez que não seria necessário armazenamento local em LVM-Thin para discos de VMs. Para melhor aproveitamento do espaço, o volume pve-root foi expandido de forma a ocupar toda a área libertada da terceira partição/VG . Este procedimento foi efetuado por linha de comandos, uma vez que a interface do Proxmox VE não disponibiliza funcionalidades nativas para esta operação.

Os principais comandos utilizados foram:

```
#Remover o volume lógico 'pve-data' (LVM-Thin)
Nota: pode ser feito via interface em "Nome do nó > Disks > LVM Thin > data > Remove"
lvremove /dev/pve/data

#Expandir o volume lógico 'pve-root' (LVM) com todo o espaço livre disponível no grupo de
volume 'pve'
lvextend -l +100%FREE /dev/pve/root

#Redimensionar o sistema de ficheiros para usar o novo espaço atribuído ao 'pve-root'
resize2fs /dev/pve/root
```

Com estes comandos, o volume pve-root passou a ocupar todo o espaço anteriormente atribuído ao pve-data, garantindo maior capacidade disponível para o sistema operativo e futuras configurações/armazenamentos locais.

4.7.1 Configuração do Ceph

Embora os requisitos definidos (Tabela 10) estejam abaixo dos valores recomendados (Tabela 7), nomeadamente ao nível da memória RAM disponível por nó, para a implementação do ceph, procedeu-se à instalação da solução para fins de aprendizagem e testes.

A configuração iniciou-se com a transferência, através da interface gráfica do Proxmox, do módulo Ceph em cada um dos nós, *Nome do nó > Ceph > "Install Ceph"*. Este processo instala os pacotes necessários e ativa o suporte ao Ceph dentro do *cluster*. Após a instalação, foi possível aceder ao menu Ceph em cada nó e iniciar a configuração dos componentes principais (podendo a criação e gestão destes ser realizada a partir de qualquer nó do *cluster*, não sendo necessário aceder individualmente a cada um dos nós), nomeadamente:

- **Monitor (MON)**
- **Manager (MGR)**
- **Object Storage Daemon (OSD)**
- **Metadata Server (MDS)**

Para a configuração dos OSDs, não é aconselhado utilizar o mesmo disco onde se encontra instalado o sistema operativo. Por esse motivo, é necessário que existam discos dedicados exclusivamente ao Ceph. Assim, os segundos discos (sdb) de cada host Proxmox (Figura 12) foram configurados como OSDs na secção *Nome do nó > Ceph > OSD*.

Após a criação dos OSDs, foram configurados os dois principais tipos de armazenamento suportados pelo Ceph: Ceph RBD e CephFS. A configuração de ambos foi realizada a partir de um único nó do *cluster*. O armazenamento do tipo RBD foi criado através da secção *Nome do nó > Ceph > Pool*, onde se definiu uma nova pool de armazenamento, com 3 réplicas e um valor mínimo de 2 réplicas. Por outro lado, o CephFS foi configurado na secção *Nome do nó > Ceph > CephFS*, processo que criou automaticamente as duas pools necessárias ao seu funcionamento — uma para os dados e outra para os metadados — também com 3 réplicas e mínimo de 2 (Figura 14).

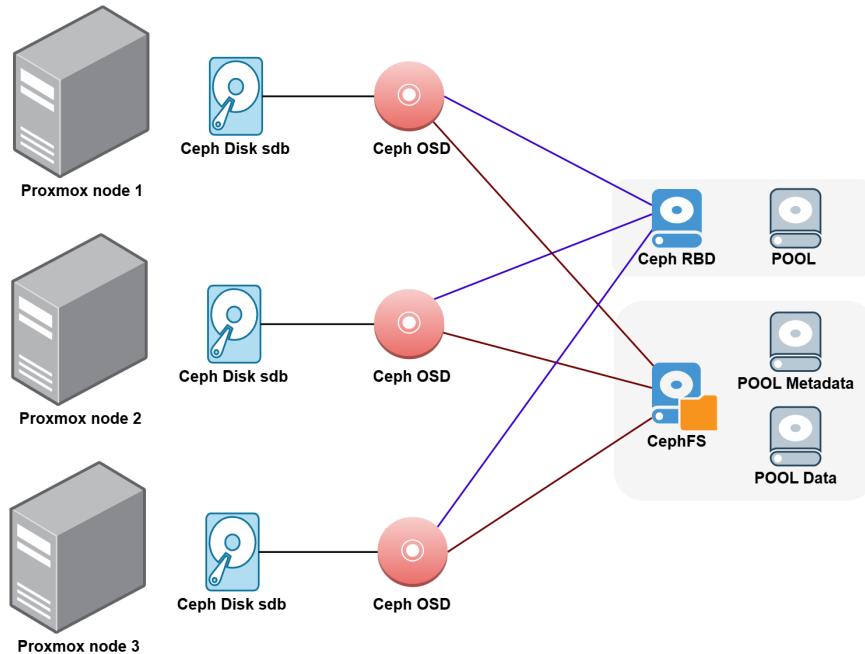


Figura 14: Arquitetura do armazenamento ceph

4.8 Virtual Machines e Containers

A criação de novas VMs e CTs no Proxmox VE pode ser feita manualmente ou a partir de *templates* base. Quando não se recorre a *templates*, o processo inicia-se com a inserção de imagens base, com ficheiros .iso para VMs ou *templates* de CTs LXC para CTs.

4.8.1 Importação de ISOs e *templates* LXC

- Os ficheiros .iso devem ser carregados manualmente para o armazenamento local ou de rede através da interface do Proxmox. Em *Nome_Armazenamento > ISO Images > Upload*, seleciona-se o ficheiro a carregar.
- Os *templates* do CT podem ser obtidos de duas formas em *Nome_Armazenamento > CT Templates*, a partir do download de *templates* existentes diretamente via interface do proxmox (*Download from URL*) ou carregados a partir de ficheiros (*Upload*).

4.8.2 Criação de VMs

- No Proxmox, o processo inicia-se clicando em *Create VM* e preenchendo o ID (identificador único) e o nome da máquina virtual.
- Na etapa OS, é necessário selecionar um ficheiro .iso previamente carregado para o armazenamento. Este ficheiro representa o instalador do sistema operativo desejado (por exemplo, Debian, Ubuntu, Windows).
- Nas etapas seguintes, define-se a configuração do sistema: número de CPUs, memória RAM, disco virtual (tamanho e tipo de armazenamento), tipo de interface de rede, entre outros parâmetros.
- Ao finalizar a configuração, a VM é criada e pode ser iniciada. A instalação do sistema operativo decorre como num computador físico.
- Após a instalação, a VM está pronta a ser utilizada ou convertida num template, caso se pretenda reutilizar a configuração.

4.8.3 Criação de CTs

- Para criar um CT, é necessário dispor de um template LXC previamente descarregado. Estes *templates* são ficheiros compactados com imagens de sistemas Linux otimizadas para execução em CTs.
- O processo inicia-se com a opção *Create CT*, onde se preenche o ID, o nome, a palavra-passe de root e outras configurações básicas.
- Em seguida, seleciona-se o template de CT desejado (por exemplo, *debian-11-standard*).
- Tal como nas VMs, define-se a quantidade de CPU, memória, armazenamento (tipo e tamanho) e configurações de rede.
- Após concluir o processo, o CT pode ser iniciado e está pronto a ser utilizado.

4.8.4 Criação a partir de *Templates* Base

A criação de novas instâncias de VMs ou CTs pode ser facilitada recorrendo a *templates* previamente configurados, criados a partir de VMs ou CTs (mencionado na subsecção 4.9). Esta abordagem permite poupar tempo e garantir consistência entre diferentes instâncias.

Para criar novas VMs ou CTs com base num template, seleciona-se o template desejado na interface do Proxmox e utiliza-se a opção *Clone*. Neste processo, pode-se escolher entre dois tipos de clonagem:

- **Cópia completa (*Full Clone*)**: Cria uma cópia totalmente independente do template, duplicando todos os dados e ficheiros. Este tipo de clonagem consome mais espaço em disco, mas permite que a nova instância funcione de forma autónoma e sem dependência do template original.
- **Cópia leve (*Linked Clone*)**: Cria uma nova instância que partilha os dados base do template, utilizando ligações ao disco original. Esta abordagem é mais rápida e eficiente em termos de espaço, mas mantém uma dependência do template, o que pode limitar algumas operações (por exemplo, mover ou apagar o template original).

A escolha entre os dois tipos depende das necessidades específicas de desempenho, espaço de armazenamento e independência da nova instância.

4.9 *Templates*

Os *templates* são imagens base pré-configuradas que permitem criar novas VMs ou CTs de forma rápida e consistente, evitando a necessidade de reinstalar e configurar sistemas operativos a partir do zero.

4.9.1 Criação de *Templates*:

- **CTs:** É possível converter um CT existente num template. Para isso, seleciona-se o CT desejado e utiliza-se a opção *Convert to Template* na interface do Proxmox. A partir deste template, podem ser criados novos CTs de forma rápida e eficiente.
- **VMs:** Também é possível criar *templates* a partir de VMs. Para tal, seleciona-se a VM base pretendida e converte-se num template através da opção *Convert to Template* na interface do Proxmox.

Para a criação de *templates*, tanto as VMs como os CTs não podem ter *snapshots* associados nem podem estar em execução. Depois de criado o template, a máquina original deixa de poder ser usada diretamente, sendo necessário criar novas instâncias a partir do template.

Os *templates* ficam armazenados no mesmo tipo de armazenamento que estava anteriormente, mas agora, em vez de serem do tipo VM ou CT, são do tipo base para outras instâncias.

4.9.2 Utilização de *Templates* pré-configurados

Além da criação de *templates* a partir de CTs locais, o Proxmox VE permite importar *templates* pré-configurados para CTs, disponibilizados oficialmente ou por terceiros, através da funcionalidade de *Download Template* na secção de *templates* do armazenamento de ficheiros (local e no CephFS). Estes *templates* incluem diversas distribuições Linux otimizadas para utilização em CTs e facilitam a criação rápida de ambientes padronizados.

Para VMs, não existem *templates* pré-configurados oficiais para download direto na interface do Proxmox.

4.10 Alta Disponibilidade (HA)

A funcionalidade de Alta Disponibilidade (HA) no Proxmox VE permite que VMs e CTs sejam automaticamente reiniciados noutro nó do *cluster* em caso de falha do nó onde estavam a correr.

Este mecanismo é possível graças à uma integração entre o gestor do *cluster* (pmxcfs), o sistema de gestão de recursos (corosync) e o serviço pve-ha-manager, que monitoriza continuamente o estado dos nós e dos serviços configurados para alta disponibilidade [44].

Nota: A solução de HA nativa do Proxmox VE, baseada no ha-manager, permite atingir níveis de disponibilidade de até 99,999% por ano. No entanto, devido aos tempos típicos de deteção de falhas e failover (cerca de 2 minutos), alcançar valores superiores, como 99,9999% ou mais, torna-se impraticável sem soluções externas complementares. Isto deve-se ao facto de o Proxmox não suportar nativamente mecanismos de *fault tolerance*, nos quais a execução contínua da VM é mantida sem interrupções.

4.10.1 Política de encerramento de nó

O mecanismo de HA do Proxmox VE permite definir a política de gestão das VMs ou CTs durante o processo de encerramento de um nó, através do parâmetro ”shutdown_policy” em *Datacenter > Options > HA Settings*. Estão disponíveis quatro opções principais:

- **failover**: Trata o encerramento como uma falha; a VM/CT será automaticamente iniciada noutra nó do grupo de HA.
- **freeze**: Mantém o serviço em estado ”suspenso” durante o encerramento. Ao reiniciar o nó original, a VM/CT retoma o estado anterior, sem ser migrada.
- **migrate**: Tenta realizar uma migração em tempo real (*live migration*) da instância para outro nó do grupo HA antes de o nó ser desligado.
- **conditional**: Executa *migrate* se for possível; caso contrário, aplica *freeze*.

Esta configuração permite adaptar o comportamento do *cluster* HA consoante o tipo de manutenção ou cenário de encerramento, maximizando a continuidade dos serviços.

4.10.2 Configuração do HA

A configuração do HA é realizada através da interface web, na secção *Datacenter > HA*. As VMs ou CTs a incluir no sistema de HA são adicionadas nesta mesma secção. Durante este processo, estão disponíveis diversos parâmetros de configuração, como:

- **Group:** Grupo de nós com prioridades definidas para realocação. Cada grupo é previamente criado em *Datacenter > HA > Groups*, onde são selecionados os nós que farão parte do grupo, atribuindo-se uma prioridade numérica a cada um. Quanto maior o valor, maior a prioridade na escolha do nó para execução da VM/CT.
- **Max. Restart:** Número máximo de tentativas de reinício da instância no mesmo nó antes de ser movida.
- **Max. Relocate:** Número máximo de tentativas de migração para outro nó em caso de falha.
- **Requested State:** Define o estado pretendido da VM/CT sob gestão do HA. Por exemplo, “Started” garante que o serviço será mantido ativo, tentando reiniciar ou migrar em caso de paragem inesperada.

Este sistema proporciona uma camada adicional de resiliência e continuidade de serviço, especialmente importante em ambientes de produção ou laboratórios de teste com simulação de falhas.

4.10.3 Requisitos Técnicos

Para garantir um funcionamento estável, o sistema de HA depende de um quorum funcional (mínimo de 3 nós) e de um sistema de armazenamento partilhado ou distribuído entre os nós. Sem armazenamento acessível por múltiplos nós, as VMs ou CTs não poderão ser iniciados noutra nó em caso de falha, comprometendo o objetivo da alta disponibilidade.

4.11 *Live Migration*

A *Live Migration* permite mover uma máquina virtual em execução de um nó para outro dentro do *cluster* sem interrupção perceptível do serviço. Este processo é útil para manutenção programada, balanceamento de carga ou otimização de recursos.

4.11.1 Requisitos/Funcionamento da *Live Migration*

Para realizar a *Live Migration*, é aconselhado utilizar armazenamento partilhado entre os nós (por exemplo, armazenamento distribuído como Ceph, NFS ou SAN). Isto garante que os dados das VMs/CTs estão acessíveis simultaneamente em ambos os nós, permitindo a migração sem necessidade de copiar o disco inteiro durante o processo. Apenas os dados da memória (RAM) e o estado da execução (no caso das VMs) são transferidos possibilitando a continuação da execução no novo nó sem interrupções perceptíveis nem perda de dados. Já no caso dos CTs, não seria necessário transferir nada.

4.11.2 Arquitetura de uma *Live Migration* de uma VM

Na Figura 15 é apresentada a arquitetura geral de uma *live migration*, exemplificando a migração de uma VM do nó 1 para o nó 2. Durante este processo, apenas o conteúdo da memória RAM é transferido para o novo nó, enquanto o disco é acedido diretamente pelo Ceph, garantindo acesso imediato aos dados. A VM é brevemente pausada para concluir a transferência da memória e, em seguida, é restaurada no nó de destino.

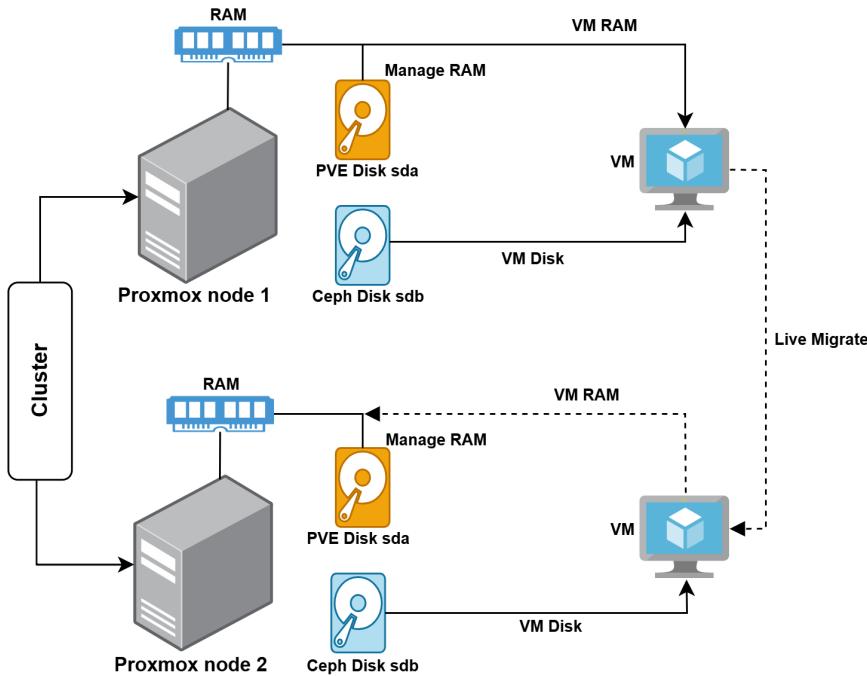


Figura 15: Arquitetura geral do *live migration*

4.11.3 *Live Migration* de CTs

Nos CTs, a *live migration* propriamente dita não é suportada de forma fiável como nas VMs. Como os CTs partilham o kernel do host, não é possível preservar o estado da memória durante a migração. Assim, o CT tem de ser desligado, o disco copiado (caso não esteja num armazenamento partilhado) e reiniciado no novo nó.

Com armazenamento distribuído (Ceph), a migração consiste apenas na transferência do controlo de execução do CT entre os nós. Como o ficheiro de configuração dos CTs encontra-se centralizado e sincronizado em todos os nós através do sistema de ficheiros distribuído do *cluster* (pmxcfs), não é necessária a sua transferência durante a migração.

Embora exista suporte experimental para migração via CRIU (Checkpoint/Restore In Userspace), tecnologia que possibilita a captura e restauro do estado completo de um CT noutro nó, sem ser necessário reiniciar. Esta funcionalidade permanece instável e com pouca adoção em ambientes de produção.

4.11.4 Migração com Armazenamento Local

Nesta subsubsecção, analisa-se o processo de migração de VMs e CTs quando utilizam armazenamento local.

VM:

Quando o armazenamento é local, o Proxmox permite a *Live Migration* de VMs através da cópia integral do disco local para o nó de destino.

Para reduzir o *downtime*, esta cópia é realizada em segundo plano enquanto a VM continua a executar-se no nó original. Apenas na fase final da migração (quando a cópia do disco está concluída), a memória e o estado de execução da VM são transferidos, permitindo retomar a sua execução no novo nó. Após a migração bem-sucedida, a cópia do disco no nó de origem é eliminada.

Esta técnica, apesar de reduzir o tempo de indisponibilidade, pode causar uma sobrecarga significativa nos recursos do sistema, devido ao elevado consumo de CPU, I/O e rede durante a cópia em segundo plano.

CT:

No caso dos CTs com armazenamento local, não existe um mecanismo de pré-cópia em segundo plano como acontece com as VMs. A migração exige sempre a paragem completa do CT antes da transferência dos seus dados, o que implica maior *downtime*.

4.11.5 Diferença entre *Live Migration* e Alta Disponibilidade

Ao contrário do sistema de Alta Disponibilidade (HA), a *Live Migration* não requer a deteção automática de falhas nem reinicia a VM em caso de falha inesperada. Trata-se de um processo controlado pelo administrador, geralmente utilizado para evitar paragens programadas. No contexto do HA, pode ser configurada como parte da política de encerramento (`shutdown_policy`), através das opções `migrate` ou `conditional`, que permitem mover a VM para outro nó antes do encerramento do nó original.

4.12 Snapshots

Os *snapshots* permitem capturar o estado de uma máquina virtual ou CT num dado momento, incluindo o conteúdo do disco e, opcionalmente, da memória. Esta funcionalidade é fundamental para tarefas como testes, atualizações críticas ou alterações no sistema, possibilitando a reversão rápida em caso de falhas.

4.12.1 Funcionamento dos Snapshots

No ambiente implementado, os *snapshots* são suportados graças às funcionalidades dos sistemas de armazenamento utilizados:

- **Ceph RBD:** que aloja a maioria das VMs e CTs no sistema, permite a criação de *snapshots* consistentes e eficientes ao nível de bloco.
- **CephFS:** embora suporte *snapshots*, não é utilizado neste contexto para armazenar discos de VMs ou CTs, e por isso não desempenha um papel direto nesta funcionalidade.
- **Armazenamento local:** pode vir a conter VMs ou CTs, que não necessitem de ser distribuídas. No entanto, os *snapshots* neste tipo de armazenamento apenas são possíveis se os discos estiverem em formato QCOW2.

Os *snapshots* ficam diretamente associados à respetiva VM ou CT, sendo armazenados no mesmo volume onde se encontra o disco. Esta associação permite que, ao restaurar um *snapshot*, a máquina seja revertida ao estado exato em que se encontrava no momento da sua criação. No entanto, também implica que a eliminação da VM/CT resultará na perda dos seus *snapshots*.

4.12.2 Processo de Criação

A criação de *snapshots* no Proxmox é um processo simples, acessível através da interface web. Pode ser realizada tanto com a máquina ligada — criando um *snapshot* do disco e da memória no caso das VMs, e somente do disco no caso dos CTs — como desligada, guardando apenas o estado

do disco. O procedimento consiste em aceder a: *Nome do nó > Nome da VM/CT > Snapshots > Take Snapshot*, e preencher o nome e uma descrição opcional para o *snapshot*.

4.13 Backups

Os *backups* são essenciais para garantir a integridade e a recuperação dos dados em caso de falhas, erros humanos ou atualizações mal sucedidas. No contexto do Proxmox VE, existem dois mecanismos principais para realizar cópias de segurança: o sistema de backups nativo do Proxmox e a integração com o Proxmox Backup Server (PBS).

4.13.1 Backups Nativos do Proxmox

O Proxmox VE permite realizar backups de VMs e CTs de forma nativa, sem necessidade de instalar ou configurar serviços adicionais. No entanto, este sistema apenas suporta backups completos, o que pode resultar num uso elevado de espaço e em tempos de execução mais prolongados. Ainda assim, oferece suporte a compressão dos dados para mitigar parcialmente o impacto no armazenamento.

4.13.2 Proxmox Backup Server

O Proxmox Backup Server (PBS) é uma solução dedicada, desenvolvida pela mesma equipa do Proxmox, que permite realizar backups incrementais, deduplicados e comprimidos de forma eficiente. Após um primeiro backup completo, os seguintes são incrementais, o que reduz significativamente o uso de espaço. A deduplicação evita redundância entre backups e a compressão optimiza ainda mais o armazenamento. Além disso, o PBS inclui uma interface web própria, que suporta verificações de integridade e políticas de retenção automáticas configuráveis (por exemplo, manter os últimos 7 backups).

O PBS pode ser instalado numa máquina dedicada fora do *cluster*, num CT ou numa VM dentro do próprio *cluster*, sendo que cada abordagem apresenta vantagens e desvantagens. Nenhuma configuração está totalmente livre de falhas, pelo que é recomendável manter múltiplas versões de

backups e, sempre que possível, realizar cópias externas ou para outro PBS remoto, de modo a garantir maior resiliência e segurança dos dados.

Processo de Configuração:

1. Instalar o PBS a partir da imagem ISO disponível no site oficial do Proxmox.
2. Aceder à interface web do PBS e configurar o armazenamento para os backups através da criação de um *Datastore*, em *Datastore > Create Datastore*, atribuindo um nome e selecionando a diretoria onde os backups serão guardados.
3. No Proxmox VE, adicionar o PBS como um destino de backups através de *DataCenter > Storage > Add > Proxmox Backup Server*, preenchendo os dados de ligação (nome, endereço, porta, token ou credenciais, e nome do *datastore*).

4.13.3 Processo de Criação

Existem duas formas principais de criação de *backups* através da interface web do Proxmox: accedendo diretamente à VM/CT (*Nome da VM/CT > Backup > Backup now*) ou através da criação de tarefas programadas (*Datacenter > Backup > Add*). Sendo especificado os seguintes campos:

Os backups podem ser realizados nos seguintes modos:

- **Stop:** a VM/CT é desligada durante o processo, garantindo consistência total dos dados.
- **Suspend:** a VM/CT é pausada temporariamente, com um curto tempo de indisponibilidade.
- **Snapshot:** o backup é efetuado com a VM/CT em funcionamento, recorrendo à funcionalidade de *snapshot* do sistema de armazenamento.

Nota: este modo só está disponível se o armazenamento utilizado suportar *snapshots*.

Os backups são armazenados nos destinos previamente configurados, que podem ser armazenamento local, partilhado como CephFS ou soluções como o Proxmox Backup Server. Independen-

temente do local onde são guardados, os backups permanecem associados às respetivas VMs/CTs, podendo ser visualizados em *Nome da VM/CT > Backup*.

No caso dos *backup jobs*, é necessário especificar adicionalmente a frequência de execução, as VMs/CTs incluídas, o modo de compressão, o destino e a política de retenção.

4.14 Alertas e Notificações

A configuração de alertas é essencial para garantir a rápida deteção e resposta a eventos críticos numa infraestrutura virtualizada. O Proxmox VE e o Ceph oferecem mecanismos próprios para notificação de falhas, degradações de desempenho ou alterações no estado do sistema, com envio automatizado de alertas por email.

4.14.1 Configuração do SMTP para alertas por email no Proxmox VE

Para permitir o envio de alertas por email no Proxmox VE, é necessário configurar um servidor SMTP através da interface web:

1. Aceder a *Datacenter > Notifications > Notification Targets*.
2. Criar um novo *Notification Target* do tipo SMTP.
3. Introduzir os detalhes do servidor SMTP: endereço, porta (normalmente 465), nome de utilizador e palavra-passe.
4. Definir o endereço de email de origem no campo *From Address*.
5. Usar o botão *Test* para validar o envio de notificações para o *target* criado.

Depois de configurar os destinos da notificação, é possível criar regras de alerta em *Datacenter > Notifications > Notification Matchers*, definindo as condições que geram alertas.

4.14.2 Alertas no Ceph com Prometheus e Alertmanager

Por se tratar de um módulo separado da infraestrutura Proxmox, a configuração dos alertas do Ceph exige uma configuração específica. O Ceph expõe métricas, incluindo o estado de integridade do *cluster*, através do módulo Prometheus integrado no Ceph Manager.

Os passos para configurar os alertas no Ceph, num dos nós do *cluster* (*ceph*), são os seguintes:

- Ativar o módulo Prometheus no Ceph Manager:

```
ceph mgr module enable prometheus
```

- Configurar o Prometheus para recolher métricas do Ceph.

```
# Ficheiro de configuração:  
/etc/prometheus/prometheus.yml  
  
# Exemplos e documentação sobre a configuração do Prometheus, consultar:  
https://prometheus.io/docs/prometheus/latest/configuration/configuration/
```

- Definir regras de alerta no Prometheus que detetam estados de *Warning* e *Error*.

```
# Ficheiro de configuração:  
/etc/prometheus/ceph_alerts.yml  
  
# Exemplos de regras de alertas que podem ser encontrados na documentação do Ceph:  
https://docs.ceph.com/en/latest/rados/operations/monitoring/#prometheus
```

- Configurar o Alertmanager para gerir e enviar notificações desses alertas para email via SMTP.

```
# Normalmente não está instalado por defeito:  
apt install alertmanager  
  
# Ativar e iniciar o serviço:  
systemctl enable alertmanager  
systemctl start alertmanager  
  
# Ficheiro de configuração:  
/etc/alertmanager/alertmanager.yml
```

```
# A configuração do Alertmanager está documentada em:  
https://prometheus.io/docs/alerting/latest/configuration/
```

4.15 Monitorização

A monitorização é um elemento fundamental na administração de infraestruturas virtualizadas, permitindo acompanhar o estado dos recursos físicos e virtuais, diagnosticar anomalias e planear a capacidade futura.

O Proxmox VE disponibiliza, na sua interface gráfica, diversos gráficos e estatísticas de utilização, sem que seja necessária qualquer configuração adicional. Estes dados são recolhidos automaticamente e armazenados localmente.

Além disso, o Proxmox VE inclui suporte nativo para exportação de métricas para sistemas externos de monitorização, como é o caso do *InfluxDB*. O *InfluxDB* é uma base de dados otimizada para o armazenamento de métricas e séries temporais, amplamente utilizada em ambientes de monitorização modernos.

Estas métricas podem ser visualizadas e analisadas de forma dinâmica através de plataformas como o Grafana, que se integra nativamente com o InfluxDB. Esta integração permite criar *dashboards* personalizados e configurar alertas com base em condições específicas, o que é especialmente útil em ambientes com múltiplos utilizadores ou com requisitos de alta disponibilidade [44].

4.15.1 Configuração do InfluxDB e o Grafana para métricas do Proxmox VE

Para recolher métricas do Proxmox VE, o InfluxDB pode ser instalado numa VM/CT dentro do Proxmox, isolando a aplicação e facilitando a sua gestão.

Passos para a configuração:

1. Criar e configurar um VM/CT no Proxmox para o InfluxDB, garantindo que tem acesso à rede e recursos necessários.

2. Instalar o InfluxDB:

```
apt update  
apt install influxdb  
systemctl enable influxdb  
systemctl start influxdb
```

3. Criar a base de dados para as métricas do Proxmox no InfluxDB (pode ser feito também pela interface web):

```
influx  
CREATE DATABASE "name_Database"  
exit
```

4. Configurar o Proxmox VE para enviar métricas para o InfluxDB, em *Datacenter > Metric Server > Add > InfluxDB*, definindo o nome, o endereço IP da VM/CT e a porta (normalmente 8086).

5. Verificar que as métricas estão a ser enviadas corretamente para o InfluxDB, acedendo à interface web do InfluxDB no endereço: <http://<IP-da-VM-ou-CT>:8086>.

Para visualizar e analisar as métricas armazenadas no InfluxDB, o Grafana pode ser instalado numa VM/CT dentro do Proxmox, proporcionando uma interface intuitiva para a criação de *dashboards* personalizados, como ilustrado na Figura 16.

1. Criar e configurar uma VM/CT no Proxmox para o Grafana, garantindo acesso à rede e recursos necessários.

2. Instalar o Grafana:

```
apt update  
apt install grafana  
systemctl enable grafana-server  
systemctl start grafana-server
```

3. Aceder à interface web do Grafana no endereço: <http://<IP-da-VM-ou-CT>:3000> (por defeito, o utilizador é admin e a password admin).

4. Configurar o Grafana para ler os dados do InfluxDB:

- No Grafana, ir a *Configuration > Data Sources > Add data source*.
- Selecionar InfluxDB.
- Definir o URL do InfluxDB, por exemplo: <http://<IP-da-VM-ou-CT>:8086>.
- Inserir o nome da base de dados criada no InfluxDB.
- Salvar e testar a ligação.

5. Criar dashboards personalizados no Grafana para visualização das métricas recolhidas do Proxmox VE, ou utilizar *templates* já existentes no site oficial do Grafana: <https://grafana.com/grafana/dashboards/>.

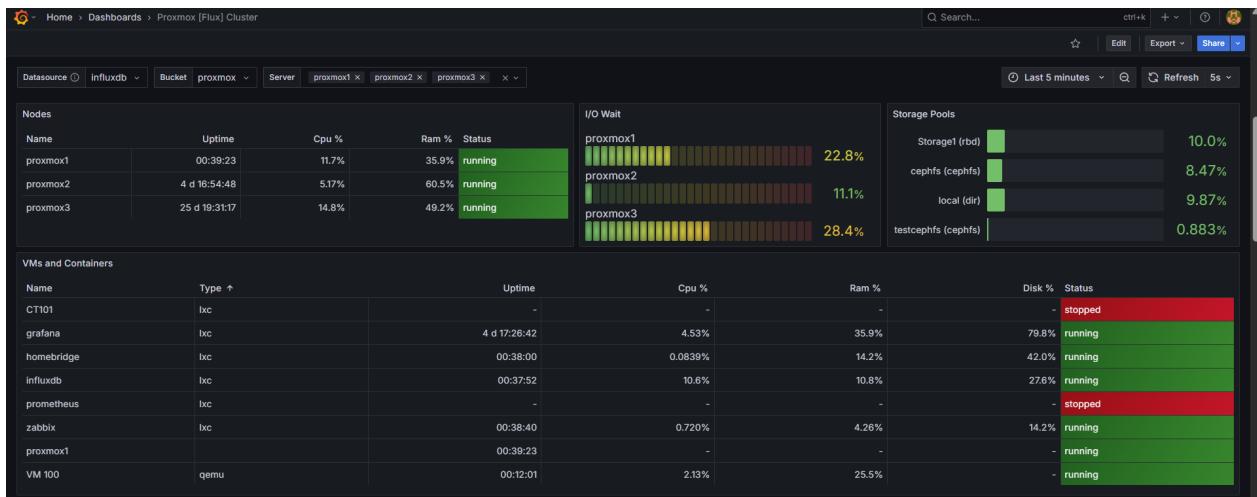


Figura 16: Dashboard do Grafana com métricas do Proxmox

4.15.2 Configuração do Telegraf para recolha de métricas do Ceph

A exportação de métricas do Ceph requer configuração adicional, uma vez que estas não são fornecidas diretamente pelo Proxmox. Para tal, é necessário ativar o módulo de exportação apropriado no daemon de gestão do Ceph (exportação para o influxdb é necessário telegraf + prometheus),

denominado *ceph-mgr*. Após essa configuração, tornam-se disponíveis métricas como:

- Estado e desempenho dos OSDs;
- Latência de leitura/escrita;
- Taxas de replicação e rebalanceamento;
- Espaço utilizado e disponível por pool;
- Estado dos MON e do *cluster*.

Uma vez que o módulo do prometheus já está configurado por ser necessário na subseção 4.14.2, é necessário configurar o Telegraf para recolher as métricas do Ceph (via prometheus) e enviá-las para o InfluxDB.

Os passos principais são os seguintes:

1. Instalar o Telegraf na VM/CT onde está instalado o InfluxDB (caso ainda não esteja instalado):

```
apt update  
apt install telegraf  
systemctl enable telegraf  
systemctl start telegraf
```

2. Configurar o Telegraf para recolher métricas do Ceph através do endpoint do Prometheus (no nó onde foi configurado) e as enviar para a VM/CT do InfluxDB.

```
# Ficheiro de configuração  
/etc/telegraf/telegraf.conf  
  
# Configuração de Prometheus  
[[inputs.prometheus]]  
urls = ["http://<IP-do-node-Proxmox>:9283/metrics"]  
  
# Configuração do InfluxDB
```

```
[[outputs.influxdb]]
urls = ["http://localhost:8086"]
database = "name_Database"

# Se necessário, adicionar autenticação do InfluxDB:
username = "user"
password/token = "password"
```

3. Reiniciar o serviço Telegraf para aplicar as alterações:

```
systemctl restart telegraf
```

4. Confirmar no InfluxDB que as métricas do Ceph estão a ser recebidas corretamente e atualizar a dashboard do Grafana para as visualizar, como ilustrado na Figura 17.

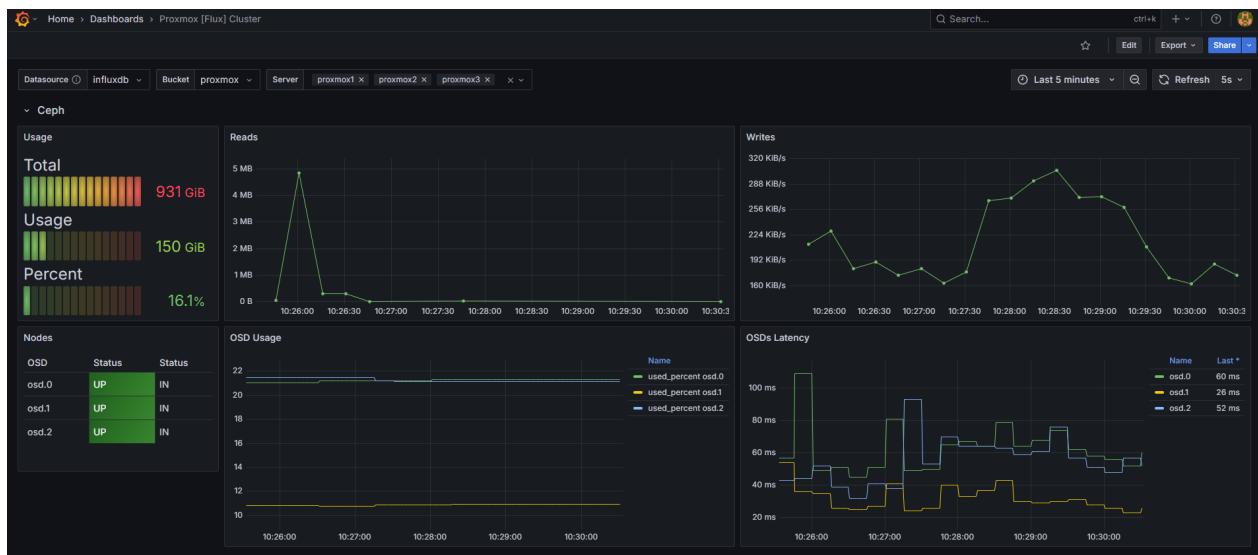


Figura 17: Dashboard do Grafana com métricas do Ceph

4.16 Síntese

Este capítulo descreveu, de forma detalhada, a implementação prática da solução de virtualização distribuída baseada em Proxmox VE e Ceph. Foram apresentados os objetivos da implementação, os componentes de *hardware* e *software* utilizados, a arquitetura física e lógica do *cluster*, bem

como todos os procedimentos de instalação, configuração e validação da infraestrutura.

A solução implementada integra funcionalidades como armazenamento distribuído com Ceph, migração em tempo real de VMs, alta disponibilidade, *snapshots* e monitorização centralizada. Apesar das limitações do *hardware* disponível, foi possível construir um ambiente funcional e resiliente, capaz de simular cenários reais de produção e suportar testes de tolerância a falhas e desempenho.

No capítulo seguinte, serão descritos os testes realizados sobre esta infraestrutura, com o objetivo de avaliar a robustez, o desempenho e a fiabilidade do sistema em diferentes cenários operacionais, destacando o comportamento do sistema de armazenamento Ceph.

5 Testes

Neste capítulo, são apresentados os testes realizados à infraestrutura de armazenamento baseada em Ceph, com o objetivo de avaliar a sua robustez, desempenho e capacidade de tolerância a falhas. Os testes incidiram sobre diferentes cenários, começando pela avaliação dos mecanismos de alta disponibilidade (HA) e da migração em tempo real de VMs e CTs, seguindo-se simulações de falhas de discos e nós, e por fim, cenários de carga intensiva de escrita e leitura.

Antes de apresentar os resultados detalhados, a Tabela 11 resume os principais testes realizados ao sistema de armazenamento distribuído Ceph. Para cada caso de teste são indicados os respetivos objetivos, as métricas recolhidas e o cenário simulado. Estes testes permitiram validar o comportamento, a fiabilidade e o desempenho da arquitetura, com especial destaque para o Ceph, em diferentes cenários.

Caso de Teste	Objetivos	Output (Métricas)	Cenário
1	Avaliar o comportamento da HA com <i>shutdown_policy=migrate</i> e <i>shutdown_policy=failover</i>	Tempo de recuperação (minutos), velocidade de migração (MiB/s), <i>downtime</i> (ms)	Interrupção manual e corte de energia do nó
2	Avaliar a migração em tempo real de CTs	Tempo total de migração (minutos, segundos), <i>downtime</i> (segundos)	Migração entre nós com armazenamento local e Ceph
3	Avaliar a migração em tempo real de VMs	Tempo total de migração (minutos, segundos), velocidade de migração (MiB/s), <i>downtime</i> (ms)	Migração entre nós com armazenamento local e Ceph
4	Avaliar o comportamento do Ceph quando os discos atingem a capacidade máxima.	Armazenamento total dos OSDs (GiB)	Disco cheio
5	Avaliar o comportamento do Ceph perante a falha de um único disco.	Armazenamento total (GiB), Dados do Ceph	Falha de disco

Caso de Teste	Objetivos	Output (Métricas)	Cenário
6	Avaliar o desempenho do Ceph (leitura/escrita) quando dois discos falham.	Armazenamento total dos OSDs (GiB), Estado dos OSDs, Escrita/Leitura	Falha dupla com min_size=2
7	Avaliar o desempenho do Ceph (leitura/escrita) quando dois discos falham.	Escrita/Leitura, Estado do OSD	Falha dupla com min_size=1
8	Verificar se o sistema reconhece e integra automaticamente um novo disco, sem reinicialização.	Estado dos OSDs	Novo disco físico no nó
9	Testar a migração do Proxmox VE para um novo disco, preservando a configuração e garantindo o arranque.	Velocidade da migração (MB/s)	Migração do sistema operativo
10	Avaliar a reação e integridade do Ceph ao adicionar um novo OSD ao cluster e o rebalanceamento dos dados na garantia das réplicas.	Armazenamento total dos OSDs (GiB), Degradão dos dados (%), CPU (%), Memória RAM, Estado dos OSDs	Adicionar OSD
11	Avaliar a escrita intensiva no RADOS com múltiplos fluxos.	CPU (%), OSDs Latency (ms), Nodes Traffic Out/In (MB), Ceph Writes (MiB/s), Armazenamento nas Pools (GiB)	Escrita - 2 e 3 réplicas (RADOS)
12	Avaliar a leitura intensiva no RADOS com múltiplos fluxos.	CPU (%), OSDs Latency (ms), Nodes Traffic Out/In (MB), Ceph Reads (MB/s)	Leitura - 2 e 3 réplicas (RADOS)
13	Avaliar a escrita sequencial no volume RBD com carga contínua e múltiplas tarefas.	CPU (%), OSDs Latency (ms), Nodes Traffic Out/In (MB), Ceph Writes (MiB/s), Armazenamento nas Pools (GiB)	Escrita - 2 e 3 réplicas (RBD)

Caso de Teste	Objetivos	Output (Métricas)	Cenário
14	Avaliar a leitura sequencial no volume RBD com múltiplos fluxos paralelos.	CPU (%), Nodes Traffic Out/In (MB), Armazenamento nas Pools (GiB)	Leitura - 2 e 3 réplicas (RBD)
15	Avaliar a escrita sequencial no CephFS sob carga prolongada e acesso direto.	CPU (%), OSDs Latency (ms), Nodes Traffic Out/In (MB), Ceph Writes (MiB/s), Armazenamento nas Pools (GiB)	Escrita - 2 e 3 réplicas (CephFS)
16	Avaliar a leitura sequencial no CephFS com tarefas paralelas e acesso direto.	CPU (%), Nodes Traffic Out/In (MB), Armazenamento nas Pools (GiB)	Leitura - 2 e 3 réplicas (CephFS)

Tabela 11: Resumo dos testes realizados

Nos testes de 1 a 10, as métricas de desempenho foram recolhidas diretamente a partir da interface do próprio Proxmox. Para os testes de 11 a 16, foi utilizado um sistema de monitorização com o InfluxDB a correr num CT no Proxmox1, e o Grafana a correr num CT no Proxmox3. Esta configuração permitiu a recolha e visualização em tempo real das métricas relevantes, como a utilização de CPU, latência dos OSDs, tráfego de rede e volume de leituras/escritas no *cluster* Ceph. É relevante mencionar que a execução dos serviços de monitorização nos próprios nós pode ter introduzido uma carga adicional nos sistemas, o que poderá ter tido um impacto residual nas métricas de desempenho recolhidas.

5.1 Teste 1 – HA com `shutdown_policy=migrate` e `shutdown_policy=failover`

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.1.1 Objetivo:

Avaliar o comportamento do mecanismo de Alta Disponibilidade (HA) do Proxmox VE, aplicando as políticas *shutdown_policy=migrate* e *shutdown_policy=failover*, em dois cenários distintos: interrupção manual e falha abrupta (corte de energia) de um nó com uma VM em execução (com proteção HA ativa). A VM foi criada sobre o armazenamento distribuído Ceph.

5.1.2 Procedimento:

Foi criada uma VM com HA ativo. O teste foi realizado em dois cenários distintos, para cada política de falha ou encerramento de um nó (*shutdown_policy*):

- **Desligar manualmente o servidor:** a ação é interpretada como uma operação controlada, permitindo a migração da VM antes da paragem do nó.
- **Desligar o servidor da corrente:** simula uma falha de energia do nó. O *cluster* deteta a falha e volta a iniciar a VM noutro nó disponível.

A VM, criada sobre o armazenamento distribuído Ceph, encontrava-se em funcionamento, mas sem serviços ou cargas de trabalho significativas a decorrer durante os testes.

5.1.3 Resultado:

Com *shutdown_policy=migrate*:

- **Desligar manualmente:** a VM é migrada com sucesso para outro nó antes do encerramento, recorrendo ao método de *live migration*.

Velocidade média de migração: 36.9 MiB/s, *downtime* de apenas 58 ms, duração total: 1 minuto e 10 segundos.

- **Desligar da corrente:** a VM não é migrada, mas é automaticamente reiniciada noutro nó após a deteção da falha (*failover*).

Tempo total até retomar o funcionamento normal: ~5 minutos.

Com *shutdown_policy=failover*:

- **Desligar manualmente:** a VM não é migrada, mas é automaticamente reiniciada noutro nó, antes do encerramento do nó original, com o funcionamento normal restabelecido em ~3 minutos.
- **Desligar da corrente:** a VM é reiniciada automaticamente noutro nó. Tempo total até retomar o funcionamento normal: ~5 minutos.

5.1.4 Discussão:

Os resultados demonstram que a política *shutdown_policy=migrate* permite um tempo de indisponibilidade (*downtime*) mínimo quando a paragem do nó é efetuada de forma controlada, beneficiando da funcionalidade de *live migration* da VM. No entanto, em caso de falha súbita, o comportamento aproxima-se do verificado com a política de encerramento *failover*, registando-se um maior tempo até à retoma da VM.

Em situações de falha abrupta, existe a possibilidade de perda de dados que não tenham sido previamente gravados no armazenamento persistente antes da falha do nó.

Os testes foram realizados com uma VM, mas o comportamento esperado para os CTs é semelhante, exceto que, com *shutdown_policy=migrate* e desligamento manual do nó, é sempre necessário o CT reiniciar, uma vez que a *live migration* não é suportada.

A *shutdown_policy=migrate* revela-se mais adequada em cenários onde seja possível antecipar a paragem do nó, assegurando uma maior continuidade do serviço e minimizando o tempo de indisponibilidade.

5.2 Teste 2 – Migração em tempo real de CTs

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.2.1 Objetivo:

Avaliar o comportamento da migração em tempo real (*live migration*) de CTs entre nós do *cluster*, comparando os tempos de migração e de indisponibilidade entre dois tipos de armazenamento: local (Directory/LVM) e distribuído (Ceph).

5.2.2 Procedimento:

Foram realizados dois testes distintos:

- Migração de um CT com disco armazenado localmente (local).
- Migração de um CT com disco armazenado em armazenamento partilhado (Ceph).

Em ambos os casos, os CTs foram criados com um disco de 8 GiB e encontravam-se em execução. A migração foi realizada entre dois nós distintos, sem falhas ou paragens manuais. Contudo, importa referir que os CTs tinham alguns serviços em execução, embora estes não estivessem a ser utilizados durante o processo.

5.2.3 Resultados:

CT com armazenamento local:

- Duração total da migração: **13 minutos**.
- Tempo de indisponibilidade (*downtime*): **13 minutos**.

Na Figura 18 é apresentado o registo da migração em tempo real de um CT com disco armazenado localmente. É possível observar métricas como o tempo total da migração, o tempo de indisponibilidade (*downtime*) e a velocidade média de transferência. Pode-se também verificar a transferência da cópia do disco para o novo nó, seguida da eliminação dos dados no nó de origem.

```

Task viewer: CT 109 - Migrate (proxmox1 --> proxmox3)

Output Status
Stop Download

2025-06-17 10:52:48 6490087424 bytes (6.5 GB, 6.0 GiB) copied, 500 s, 11.4 MB/s
2025-06-17 10:52:58 6604161024 bytes (6.6 GB, 6.2 GiB) copied, 570 s, 11.4 MB/s
2025-06-17 10:53:08 6718222336 bytes (6.7 GB, 6.3 GiB) copied, 590 s, 11.4 MB/s
2025-06-17 10:53:18 6830149632 bytes (6.8 GB, 6.4 GiB) copied, 600 s, 11.4 MB/s
2025-06-17 10:53:48 7171067904 bytes (7.2 GB, 6.7 GiB) copied, 630 s, 11.4 MB/s
2025-06-17 10:54:18 7511494656 bytes (7.5 GB, 7.0 GiB) copied, 660 s, 11.4 MB/s
2025-06-17 10:54:48 7854157824 bytes (7.9 GB, 7.3 GiB) copied, 690 s, 11.4 MB/s
2025-06-17 10:55:18 8197558272 bytes (8.2 GB, 7.6 GiB) copied, 720 s, 11.4 MB/s
2025-06-17 10:55:48 8540983296 bytes (8.5 GB, 8.0 GiB) copied, 750 s, 11.4 MB/s
2025-06-17 10:55:52 2097152+0 records in
2025-06-17 10:55:52 2097152+0 records out
2025-06-17 10:55:52 8589934592 bytes (8.6 GB, 8.0 GiB) copied, 754.273 s, 11.4 MB/s
2025-06-17 10:55:52 440+277364 records in
2025-06-17 10:55:52 440+277364 records out
2025-06-17 10:55:52 successfully imported 'local:109/vm-109-disk-0.raw'
2025-06-17 10:55:52 volume 'local:109/vm-109-disk-0.raw' is 'local:109/vm-109-disk-0.raw' on the target
2025-06-17 10:55:52 # /usr/bin/ssh -e none -o 'BatchMode=yes' -o 'HostKeyAlias=proxmox3' -o 'UserKnownHostsFile=/etc/pve/nodes/proxmox3/ssh_known_hosts'
2025-06-17 10:56:00 start final cleanup
2025-06-17 10:56:02 start container on target node
2025-06-17 10:56:02 # /usr/bin/ssh -e none -o 'BatchMode=yes' -o 'HostKeyAlias=proxmox3' -o 'UserKnownHostsFile=/etc/pve/nodes/proxmox3/ssh_known_hosts'
2025-06-17 10:56:11 migration finished successfully (duration 00:13:00)
TASK OK

```

Figura 18: Registo da migração em tempo real de um CT com disco em armazenamento local

CT com armazenamento em Ceph:

- Duração total da migração: **2 segundos**.
- Tempo de indisponibilidade (*downtime*): **2 segundos**.

Na Figura 19 é apresentado o registo da migração em tempo real de um CT com disco armazenado em Ceph. Observa-se que, devido ao armazenamento distribuído, não é necessária a transferência do disco durante a migração, sendo apenas realizada a sincronização dos dados essenciais para o controlo da execução do CT.

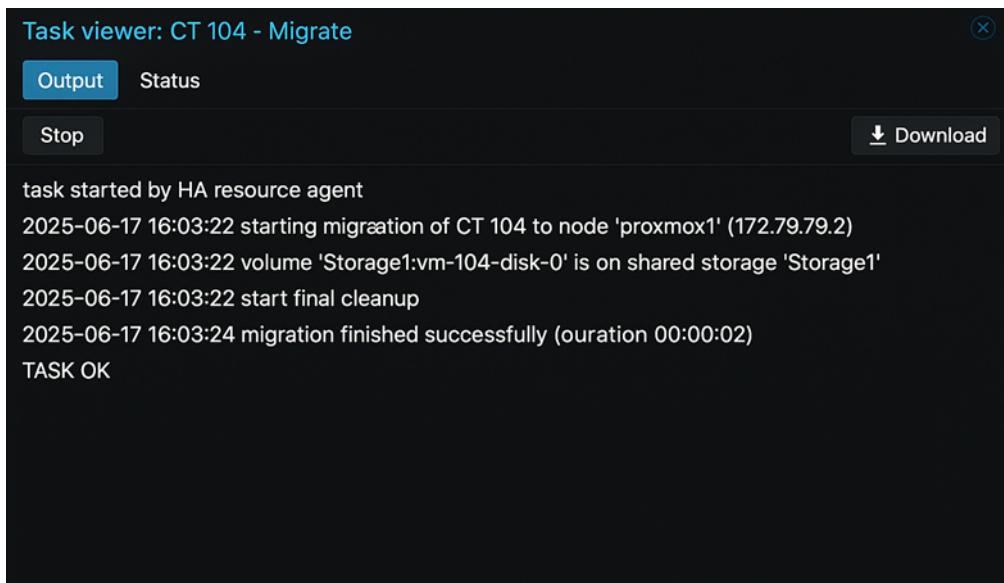


Figura 19: Registo da migração em tempo real de um CT com disco em armazenamento ceph

5.2.4 Discussão:

Os resultados demonstram uma diferença bastante grande entre os dois tipos de armazenamento. No caso do armazenamento local, a migração exigiu a cópia integral do disco do CT para o nó de destino, o que resultou num tempo de migração e de indisponibilidade muito elevados.

Com armazenamento distribuído (Ceph), a migração dos CTs ocorre de forma bastante rápida. Como o volume de armazenamento está disponível e acessível por todos os nós do *cluster*, não é necessário transferir o disco do CT. Apenas os ficheiros de configuração e os metadados necessários são sincronizados, o que reduz significativamente o tempo de migração.

Apesar disso, a *live migration* implica uma breve interrupção do CT no nó de origem e o seu reinício no nó de destino. No entanto, devido à natureza leve dos CTs e aos seus tempos de arranque rápidos, o período de indisponibilidade mantém-se bastante reduzido.

Este comportamento reforça a vantagem da utilização de armazenamento distribuído em ambientes que exigem elevada disponibilidade e migrações frequentes.

5.3 Teste 3 – Migração em tempo real de VMs

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.3.1 Objetivo:

Avaliar o comportamento da migração em tempo real (*live migration*) de VMs entre nós do *cluster*, comparando o desempenho entre dois tipos de armazenamento: local (Directory/LVM) e distribuído (Ceph).

5.3.2 Procedimento:

Foram realizadas duas migrações em tempo real:

- Uma VM com disco armazenado localmente, utilizando armazenamento do tipo local e ficheiro no formato qcow2.
- Uma VM com disco armazenado no sistema distribuído Ceph.

Em ambos os casos, a VM criada com um disco de 8.5 GiB, estava em execução, porém sem serviços ou cargas de trabalho significativas durante os testes. A migração foi realizada entre dois nós distintos, sem necessidade de interrupção manual ou ocorrência de falha no nó de origem.

5.3.3 Resultados:

Migração da VM em armazenamento local:

- Duração total da migração: 16 minutos e 25 segundos.
- Velocidade média de migração: 11.5 MiB/s.
- Tempo de indisponibilidade (*downtime*): 171 ms.

Na Figura 20, observa-se a janela utilizada para efetuar a *live migration* de uma VM. É possível visualizar um aviso (*warning*) a indicar que será necessário migrar o disco da VM, o que poderá

prolongar o tempo total do processo e tornar a migração mais demorada.

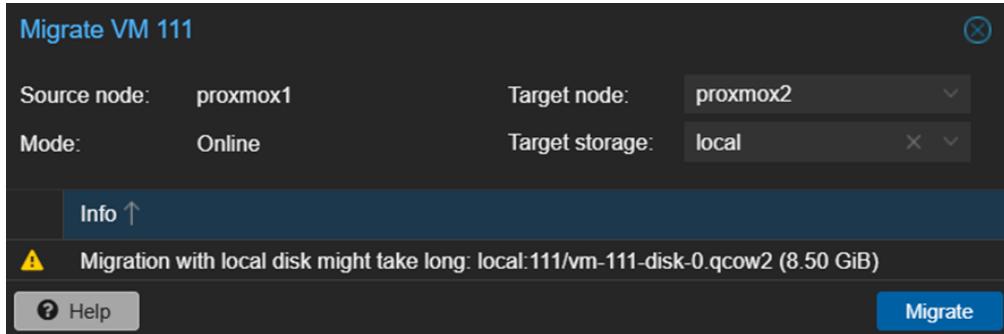


Figura 20: Configuração da migração em tempo real de uma VM com disco em armazenamento local

Na Figura 21 são apresentados os registos da migração em tempo real de uma VM com disco em armazenamento local. Podemos observar as métricas referentes ao tempo de indisponibilidade (*downtime*), duração total e velocidade média da transferência. Também é possível verificar que a migração envolveu a transferência tanto do disco quanto da memória RAM da VM para o nó de destino.

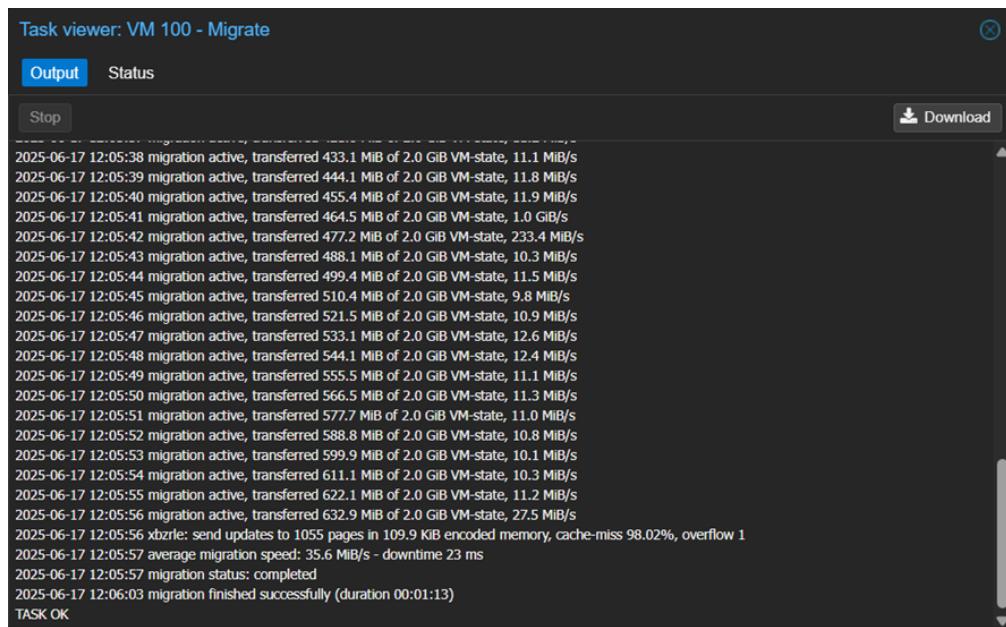
```
Task viewer: VM 111 - Migrate (proxmox1 --> proxmox2)
Output Status
Stop Download
2025-06-17 11:51:50 migration active, transferred 1.7 GiB of 2.0 GiB VM-state, 10.9 MiB/s
2025-06-17 11:51:51 migration active, transferred 1.8 GiB of 2.0 GiB VM-state, 11.2 MiB/s
2025-06-17 11:51:52 migration active, transferred 1.8 GiB of 2.0 GiB VM-state, 11.2 MiB/s
2025-06-17 11:51:53 migration active, transferred 1.8 GiB of 2.0 GiB VM-state, 10.8 MiB/s
2025-06-17 11:51:54 migration active, transferred 1.8 GiB of 2.0 GiB VM-state, 11.2 MiB/s
2025-06-17 11:51:55 migration active, transferred 1.8 GiB of 2.0 GiB VM-state, 10.8 MiB/s
2025-06-17 11:51:56 migration active, transferred 1.8 GiB of 2.0 GiB VM-state, 11.2 MiB/s
2025-06-17 11:51:57 migration active, transferred 1.8 GiB of 2.0 GiB VM-state, 9.9 MiB/s
2025-06-17 11:51:58 migration active, transferred 1.8 GiB of 2.0 GiB VM-state, 11.2 MiB/s
2025-06-17 11:51:59 migration active, transferred 1.8 GiB of 2.0 GiB VM-state, 10.8 MiB/s
2025-06-17 11:52:00 migration active, transferred 1.9 GiB of 2.0 GiB VM-state, 11.0 MiB/s
2025-06-17 11:52:01 migration active, transferred 1.9 GiB of 2.0 GiB VM-state, 10.6 MiB/s
2025-06-17 11:52:02 migration active, transferred 1.9 GiB of 2.0 GiB VM-state, 11.1 MiB/s
2025-06-17 11:52:03 migration active, transferred 1.9 GiB of 2.0 GiB VM-state, 11.4 MiB/s
2025-06-17 11:52:04 migration active, transferred 1.9 GiB of 2.0 GiB VM-state, 10.0 MiB/s
2025-06-17 11:52:05 average migration speed: 11.5 MiB/s - downtime 171 ms
2025-06-17 11:52:05 migration status: completed
all 'mirror' jobs are ready
drive-scsi0: Completing block job...
drive-scsi0: Completed successfully.
drive-scsi0: mirror-job finished
2025-06-17 11:52:07 stopping NBD storage migration server on target.
2025-06-17 11:52:13 migration finished successfully (duration 00:16:25)
TASK OK
```

Figura 21: Registo da migração em tempo real de uma VM com disco em armazenamento local

Migração da VM no Ceph:

- Duração total da migração: 1 minuto e 13 segundos.
- Velocidade média de migração: 35.6 MiB/s.
- Tempo de indisponibilidade (*downtime*): 23 ms.

Na Figura 22 são apresentados os registos da migração em tempo real de uma VM com disco armazenado em Ceph. É possível observar as métricas relacionadas ao tempo de indisponibilidade (*downtime*), duração total e velocidade média da transferência, sendo transferida somente a memória RAM durante a migração.



The screenshot shows a terminal window titled "Task viewer: VM 100 - Migrate". The "Output" tab is selected. The log output is as follows:

```

Task viewer: VM 100 - Migrate
Output Status
Stop Download
2025-06-17 12:05:38 migration active, transferred 433.1 MiB of 2.0 GiB VM-state, 11.1 MiB/s
2025-06-17 12:05:39 migration active, transferred 444.1 MiB of 2.0 GiB VM-state, 11.8 MiB/s
2025-06-17 12:05:40 migration active, transferred 455.4 MiB of 2.0 GiB VM-state, 11.9 MiB/s
2025-06-17 12:05:41 migration active, transferred 464.5 MiB of 2.0 GiB VM-state, 1.0 GiB/s
2025-06-17 12:05:42 migration active, transferred 477.2 MiB of 2.0 GiB VM-state, 233.4 MiB/s
2025-06-17 12:05:43 migration active, transferred 488.1 MiB of 2.0 GiB VM-state, 10.3 MiB/s
2025-06-17 12:05:44 migration active, transferred 499.4 MiB of 2.0 GiB VM-state, 11.5 MiB/s
2025-06-17 12:05:45 migration active, transferred 510.4 MiB of 2.0 GiB VM-state, 9.8 MiB/s
2025-06-17 12:05:46 migration active, transferred 521.5 MiB of 2.0 GiB VM-state, 10.9 MiB/s
2025-06-17 12:05:47 migration active, transferred 533.1 MiB of 2.0 GiB VM-state, 12.6 MiB/s
2025-06-17 12:05:48 migration active, transferred 544.1 MiB of 2.0 GiB VM-state, 12.4 MiB/s
2025-06-17 12:05:49 migration active, transferred 555.5 MiB of 2.0 GiB VM-state, 11.1 MiB/s
2025-06-17 12:05:50 migration active, transferred 566.5 MiB of 2.0 GiB VM-state, 11.3 MiB/s
2025-06-17 12:05:51 migration active, transferred 577.7 MiB of 2.0 GiB VM-state, 11.0 MiB/s
2025-06-17 12:05:52 migration active, transferred 588.8 MiB of 2.0 GiB VM-state, 10.8 MiB/s
2025-06-17 12:05:53 migration active, transferred 599.9 MiB of 2.0 GiB VM-state, 10.1 MiB/s
2025-06-17 12:05:54 migration active, transferred 611.1 MiB of 2.0 GiB VM-state, 10.3 MiB/s
2025-06-17 12:05:55 migration active, transferred 622.1 MiB of 2.0 GiB VM-state, 11.2 MiB/s
2025-06-17 12:05:56 migration active, transferred 632.9 MiB of 2.0 GiB VM-state, 27.5 MiB/s
2025-06-17 12:05:56 xbzrle: send updates to 1055 pages in 109.9 kB encoded memory, cache-miss 98.02%, overflow 1
2025-06-17 12:05:57 average migration speed: 35.6 MiB/s - downtime 23 ms
2025-06-17 12:05:57 migration status: completed
2025-06-17 12:06:03 migration finished successfully (duration 00:01:13)
TASK OK

```

Figura 22: Registo da migração em tempo real de uma VM com disco em Ceph

5.3.4 Discussão:

A migração de VMs com discos armazenados em Ceph revelou-se muito mais rápida do que com armazenamento local. No caso do Ceph, como o volume já está acessível por todos os nós do *cluster*, apenas o estado da memória RAM e da execução da VM necessita ser transferido, o que reduz drasticamente o tempo total da migração e a quantidade de dados a transferir.

Por outro lado, no caso do armazenamento local, embora o tempo de indisponibilidade (*downtime*) seja apenas ligeiramente superior ao do Ceph, a operação implicou a cópia completa do disco

para o nó de destino. Essa cópia é realizada em segundo plano, com a VM ainda em execução no nó original, utilizando recursos de rede e de entrada/saída do *cluster*, o que pode resultar numa maior sobrecarga do sistema durante o processo. Esta diferença evidencia a vantagem do armazenamento distribuído em cenários onde a migração de VMs é frequente e o desempenho do *cluster* é essencial.

Comparando com o teste anterior (sec. 5.2), podemos observar que o tempo de migração de VMs é mais rápido do que o dos CTs. Contudo, os CTs são ligeiramente mais demorados, pois precisam ser reiniciados durante a migração. No entanto, ao contrário das VMs, não é necessário enviar dados da memória RAM pela rede, o que evita a transferência de grandes volumes de dados e pode reduzir o impacto na rede.

5.4 Teste 4 – Disco cheio

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.4.1 Objetivo:

Avaliar o comportamento do sistema de armazenamento Ceph quando os OSDs se aproximam da sua capacidade máxima. Pretende-se verificar se a ocupação do espaço é distribuída uniformemente e se o mecanismo de replicação está a funcionar corretamente.

5.4.2 Procedimento:

Foi criada uma imagem RBD com 250 GiB e montada como volume de bloco no sistema de ficheiros:

```
# Criar uma imagem RBD com 250 GiB
rbd create Storage1/test-image1 --size 250G
# Tornar a imagem visível como um dispositivo de bloco
rbd map Storage1/test-image1
# Formatar
```

```
mkfs.ext4 /dev/rbd0
# Montar o volume
mkdir /mnt/test-rbd
mount /dev/rbd0 /mnt/test-rbd
```

De seguida, foi utilizado o comando dd para preencher o volume com dados reais:

```
# Usar /dev/zero para escrever dados reais:
dd if=/dev/zero of=/mnt/test-rbd/fill.img bs=1G count=250 status=progress
```

5.4.3 Resultado:

A Figura 23 e Figura 24 mostram o estado inicial do *cluster*, onde os OSDs apresentam uma ocupação do espaço bastante reduzida. Por outro lado, a Figuras 27 e Figura 28, capturadas após a execução do teste, evidenciam um aumento significativo da ocupação, com os OSDs a atingirem até 95% da sua capacidade, reduzindo drasticamente o espaço disponível.

ID	CLASS	WEIGHT	REWEIGHT	SIZE	RAW USE	DATA	OMAP	META	AVAIL	%USE	VAR	PGS	STATUS
0	hdd	0.22739	1.00000	233 GiB	21 GiB	20 GiB	18 KiB	1.2 GiB	212 GiB	9.14	1.36	112	up
1	hdd	0.45479	1.00000	466 GiB	21 GiB	20 GiB	21 KiB	680 MiB	445 GiB	4.45	0.66	129	up
2	hdd	0.22739	1.00000	233 GiB	21 GiB	20 GiB	17 KiB	672 MiB	212 GiB	8.90	1.32	114	up
TOTAL				931 GiB	63 GiB	60 GiB	58 KiB	2.5 GiB	869 GiB	6.74			
MIN/MAX VAR: 0.66/1.36 STDDEV: 2.29													

Figura 23: Estado inicial dos OSDs – antes da escrita

A pool do Ceph utilizada é composta por três OSDs, sendo um disco de 500 GB e dois discos de 250 GB cada (Figura 23). Como o número de réplicas configurado é três, o Ceph grava uma cópia dos dados em cada OSD. Por isso, o espaço efetivamente utilizável é calculado dividindo a soma da capacidade disponível dos discos inicialmente (aproximadamente 869 GiB) pelo número de réplicas (3), resultando em cerca de 293,15 GB disponíveis para armazenamento útil (Figura 24).

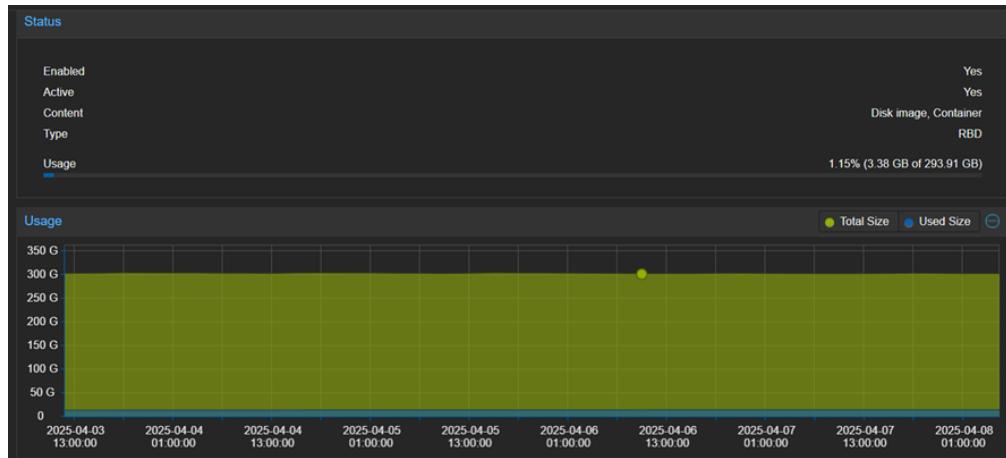


Figura 24: Gráfico de utilização inicial do volume RBD

Durante a realização do teste, o sistema Ceph entrou em estado de *HEALTH_WARN* (Figura 25), sinalizando níveis elevados de utilização em alguns componentes. O aviso foi emitido quando os OSDs *osd.0* e *osd.2*, bem como as pools *.mgr*, *cephfs_data*, *cephfs_metadata* e *Storage1*, ultrapassaram os 85% da sua capacidade. Este *warning* indica que o espaço disponível se encontra criticamente reduzido, exigindo atenção para evitar impactos no desempenho. Nessa situação, o Ceph ativa o modo de *backfilling*, tentando redistribuir e replicar os dados para garantir integridade e redundância. No entanto, este processo pode causar sobrecarga no *cluster*, degradar a performance e, caso os limites continuem a ser excedidos, levar ao bloqueio de operações de escrita para prevenir perda ou corrupção de dados.

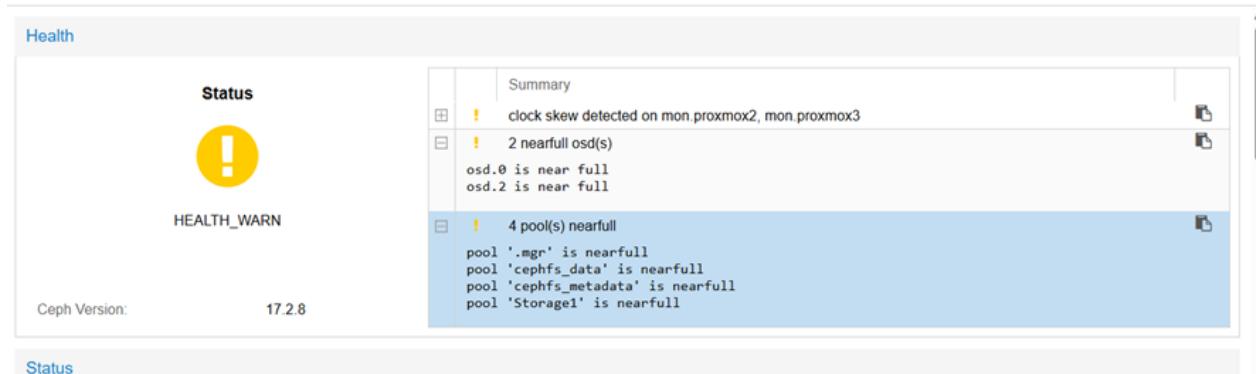


Figura 25: Avisos gerados durante os testes quando OSDs e pools ultrapassaram os 85% de capacidade

Numa fase posterior do teste, foi atingido um estado mais crítico, em que os OSDs *osd.0* e *osd.2*, bem como as pools, ultrapassaram os 90% de ocupação, originando os avisos *OSD_BACKFILLFULL* e *POOL_BACKFILLFULL* (Figura 26).

```
[WRN] OSD_BACKFILLFULL: 2 backfillfull osd(s)
  osd.0 is backfill full
  osd.2 is backfill full
[WRN] POOL_BACKFILLFULL: 4 pool(s) backfillfull
  pool '.mgr' is backfillfull
  pool 'cephfs_data' is backfillfull
  pool 'cephfs_metadata' is backfillfull
  pool 'storage1' is backfillfull
```

Figura 26: Avisos gerados durante os testes quando OSDs e pools ultrapassaram os 90% de capacidade

ID	CLASS	WEIGHT	REWEIGHT	SIZE	RAW USE	DATA	OMAP	META	AVAIL	%USE	VAR	PGS	STATUS
0	hdd	0.22739	1.00000	233 GiB	221 GiB	220 GiB	28 KiB	1.4 GiB	12 GiB	95.00	1.33	97	up
1	hdd	0.45479	1.00000	466 GiB	222 GiB	220 GiB	20 KiB	2.1 GiB	244 GiB	47.65	0.67	97	up
2	hdd	0.22739	1.00000	233 GiB	221 GiB	220 GiB	23 KiB	1.2 GiB	12 GiB	94.95	1.33	97	up
				TOTAL 931 GiB	664 GiB	659 GiB	72 KiB	4.7 GiB	267 GiB	71.31			
MIN/MAX VAR: 0.67/1.33				STDDEV: 23.66									

Figura 27: Estado final dos OSDs – após escrita de dados

Apesar de ainda existir espaço disponível na pool do Ceph (Figura 27), quando a ocupação dos dois OSDs de 250 GB se aproxima do limite, o Ceph tenta continuar a garantir o nível mínimo de duas réplicas (definido na criação da pool). Contudo, não o consegue fazer, pois apenas resta um OSD com espaço suficiente e o sistema não permite armazenar duas cópias do mesmo objeto no mesmo OSD. Assim, o Ceph ajusta o espaço total considerado disponível na pool para o limite imposto pelos OSDs mais pequenos (Figura 28), mesmo que ainda subsista espaço livre noutros OSDs.

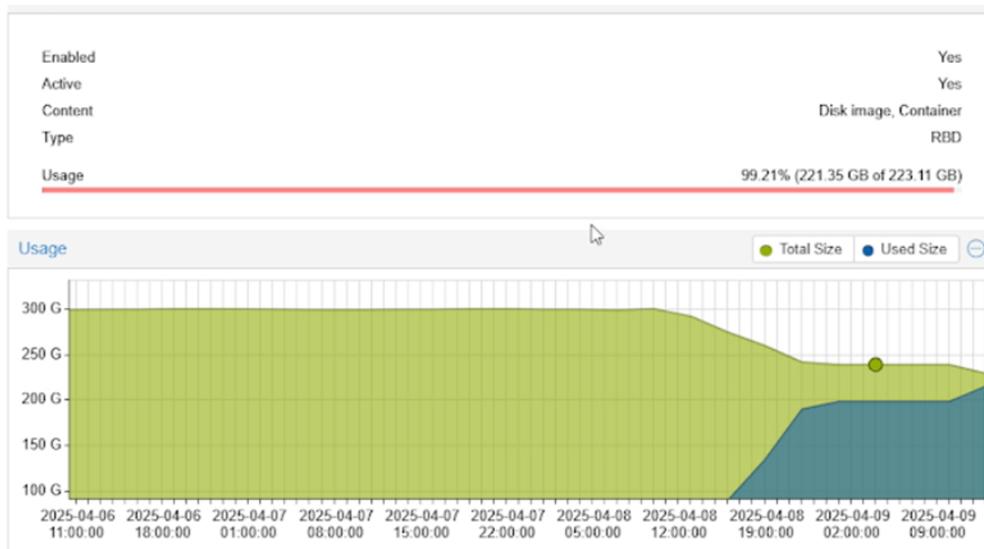


Figura 28: Gráfico de utilização do volume RBD – 99,21% ocupado

5.4.4 Discussão:

À medida que os dados foram armazenados nos OSDs, verificou-se uma redução proporcional do espaço disponível em todos os equipamentos. Este comportamento confirma que o mecanismo de replicação do Ceph está ativo e funcional, replicando os dados nos três OSDs e garantindo a existência de três réplicas.

O Ceph emite alertas (*warnings*) conforme a ocupação dos OSDs e pools atinge limites críticos: aos 85% surge o alerta *nearfull*, indicando que o espaço disponível está reduzido; aos 90% aparece o alerta *backfillfull*, que sinaliza o modo de recuperação intensiva, podendo afetar a performance.

Quando a ocupação ultrapassa os 90%, o Ceph bloqueia a escrita no OSD para evitar perda ou corrupção de dados. No nosso caso, como dois OSDs de igual capacidade ultrapassaram esse limite simultaneamente, não foi possível garantir o mínimo de duas réplicas (definido na criação da pool), comprometendo a escrita no ceph.

5.5 Teste 5 – Falha de disco

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.5.1 Objetivo:

Avaliar o comportamento do *cluster* Ceph perante a falha súbita de um dos discos que suporta um OSD. Pretende-se verificar como o sistema lida com a perda de um OSD, que alterações ocorrem no estado do *cluster* e como se processa a reintegração do disco.

5.5.2 Procedimento:

Foi removido o cabo de alimentação do disco pertencente ao ceph no nó proxmox3, simulando uma falha física. Essa ação causou a indisponibilidade imediata do OSD associado. A partir da interface de monitorização do ceph, observou-se que o OSD passou de estado *up/in* para *down/out*, conforme ilustrado na Figura 29.

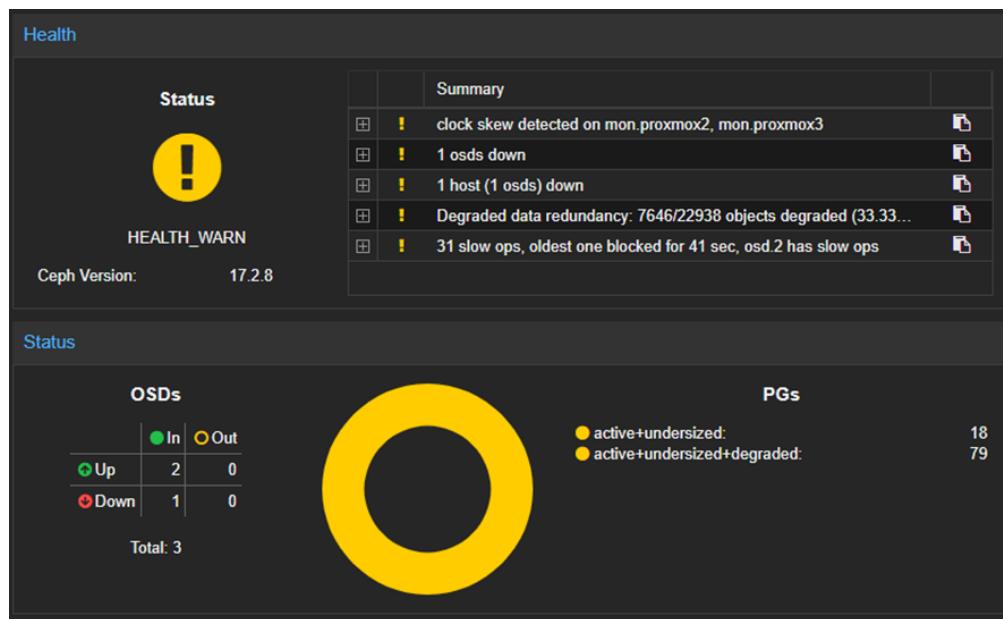


Figura 29: Estado do *cluster* após falha do OSD – *down/out* e alertas *HEALTH_WARN*

5.5.3 Resultado:

Durante esse período, o Ceph passou a operar apenas com dois OSDs, o que provocou uma redistribuição de dados e degradação da redundância, tal como indicado pelo alerta HEALTH_WARN. Com a falha de um dos OSDs, o sistema deixou de conseguir garantir as três réplicas definidas, mantendo temporariamente apenas duas cópias dos dados (o mínimo necessário para continuar a operar). A capacidade total reportada da pool foi aumentado para 434,99 GiB (Figura 30), uma vez que, ao realizar apenas duas réplicas, o Ceph passa a dispor de mais espaço útil. Importa referir que o Proxmox, apesar da falha de um dos OSDs, continua a apresentar na interface a capacidade total considerando ainda o espaço do OSD inativo.

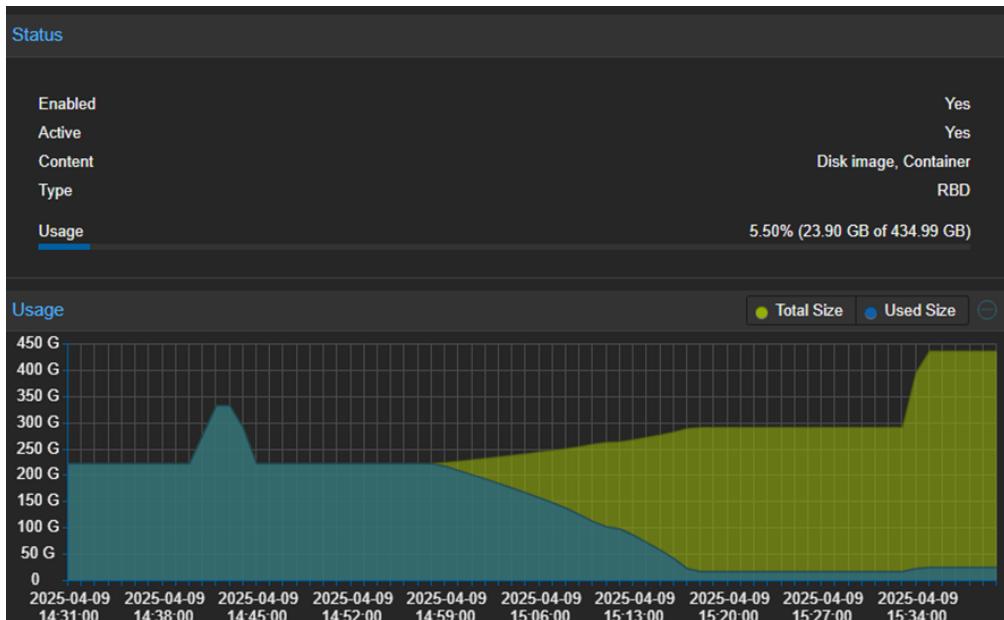


Figura 30: Capacidade total reduzida para 434.99 GiB com 2 OSDs ativos

De seguida, o disco foi reconectado com a máquina em funcionamento. Através da análise do comando `dmesg | grep -i sd`, confirmou-se que o disco `/dev/sdb` foi detetado, mas apresentou erros de I/O relacionados com o Ceph. Como solução, a máquina foi reiniciada.

Após o reinício do nó, o OSD foi reintegrado no *cluster* e passou novamente para o estado *up/in*. De imediato, o Ceph iniciou o processo de recuperação, garantindo a reposição das réplicas em falta para voltar a assegurar as três cópias definidas na configuração da pool. A Figura 31

mostra os valores atualizados dos OSDs após essa reintegração. Durante este processo, o sistema ainda apresentava valores anormais de ocupação, que foram gradualmente normalizados à medida que o rebalanceamento de dados decorria e o Ceph restabelecia a distribuição equilibrada (Figura 32 e Figura 33).

ceph osd df													
ID	CLASS	WEIGHT	REWEIGHT	SIZE	RAW USE	DATA	OMAP	META	AVAIL	%USE	VAR	PGS	STATUS
0	hdd	0.22739	1.00000	233 GiB	23 GiB	22 GiB	19 KiB	1.1 GiB	210 GiB	9.84	1.33	97	up
1	hdd	0.45479	1.00000	466 GiB	23 GiB	22 GiB	20 KiB	1.3 GiB	443 GiB	4.96	0.67	97	up
2	hdd	0.22739	1.00000	233 GiB	23 GiB	22 GiB	28 KiB	597 MiB	210 GiB	9.84	1.33	93	up
				TOTAL	931 GiB	69 GiB	66 GiB	69 KiB	3.0 GiB	862 GiB	7.40		
MIN/MAX VAR: 0.67/1.33 STDDEV: 2.44													

Figura 31: Estado normalizado após reintegração

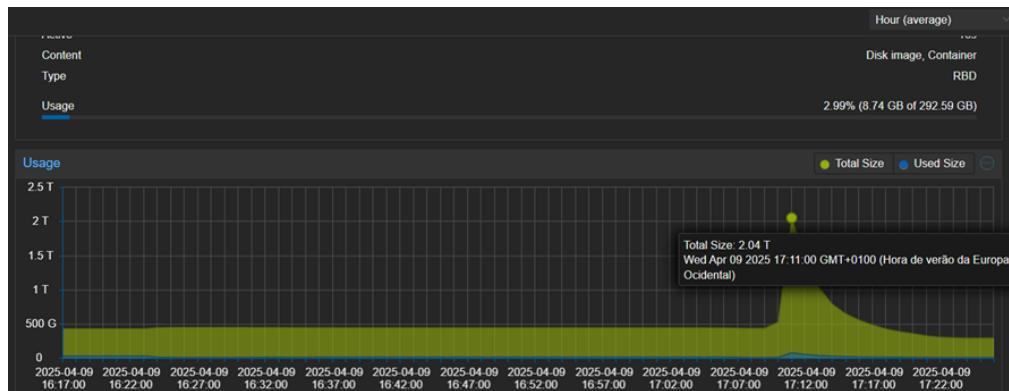


Figura 32: Estado dos OSDs após reinício e reintegração do disco

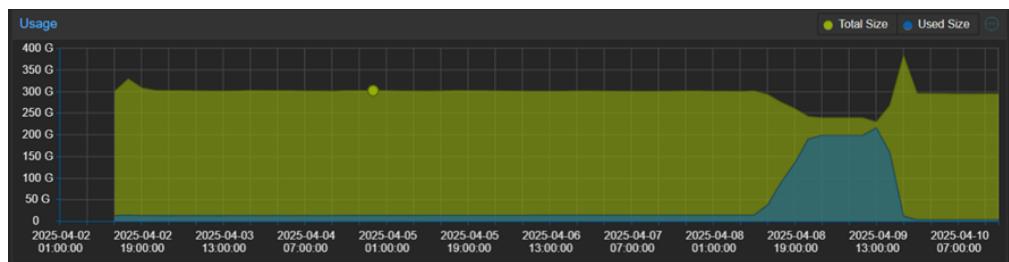


Figura 33: Distribuição normalizada dos dados no *cluster* Ceph

5.5.4 Discussão:

A falha de um OSD provocou uma degradação do estado do *cluster*, conforme esperado, mas não interrompeu a operação. O Ceph respondeu automaticamente, redistribuindo os dados disponíveis e emitindo os alertas apropriados, garantindo sempre o mínimo de duas réplicas necessárias

para o funcionamento da pool. A reinserção do disco obrigou ao reinício do nó, devido a dificuldades de reconhecimento inicial pelo Ceph, mas concluiu na reintegração correta do OSD no *cluster*.

Este teste permitiu observar o comportamento resiliente do Ceph perante falhas de OSDs e evidenciar que não se deve confiar exclusivamente nos valores de capacidade de armazenamento apresentados pela interface de gestão: mesmo com um OSD inativo, o sistema continua a contabilizar o seu espaço como disponível na apresentação global da capacidade do armazenamento.

5.6 Teste 6 – Falha de dois discos com *min_size* = 2

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.6.1 Objetivo:

Avaliar o comportamento do sistema Ceph quando dois discos (OSDs) são removidos fisicamente, num cenário em que o parâmetro *min_size* = 2. Pretende-se validar a capacidade de leitura, bem como a política de bloqueio de escrita para garantir a integridade dos dados.

5.6.2 Procedimento:

Foi removido o cabo de alimentação de um dos discos pertencentes ao Ceph no nó *proxmox2*, simulando uma falha física. Esta ação causou a indisponibilidade imediata do OSD associado (*osd 1*). Posteriormente, foi removido o cabo de alimentação de um segundo disco Ceph (*osd 0*), pertencente ao nó *proxmox1*, reduzindo o número de OSDs ativos para apenas um.

5.6.3 Resultado:

Inicialmente, o *cluster* encontrava-se estável, com todos os OSDs ativos e uma utilização de 5,55% do volume RBD (~16 GiB), como apresentado na Figura 34.

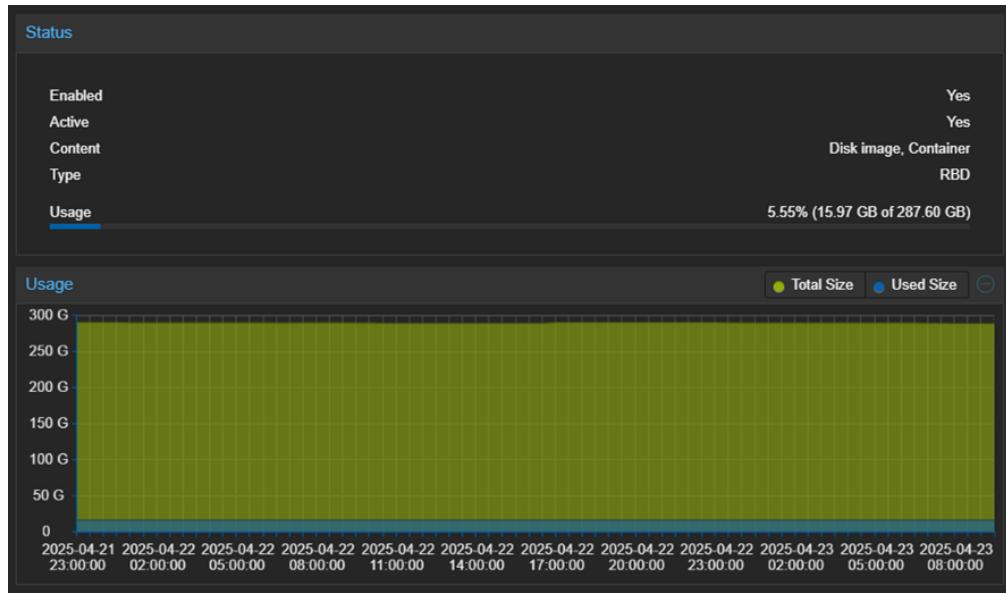


Figura 34: Estado do armazenamento antes da remoção dos discos

Após remover um disco, o *cluster* passou a operar com apenas dois OSDs ativos (Figura 35 e Figura 36). Nesta fase, o Ceph manteve as operações de escrita, uma vez que ainda conseguia garantir o número mínimo de réplicas definido na pool ($\text{min_size}=2$).

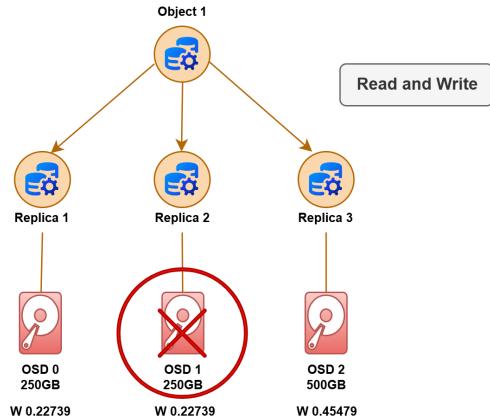


Figura 35: Exemplo da arquitetura após remoção de um disco

Name	Class	OSD Type	Status	Version	weight	reweight	Used (%)	Total	Apply/Commit Latency (ms)	PGs ↓
default										
proxmox3	osd.2	hdd	bluestore	up ○ / in ●	17.2.8	0.22739	1.00	12.69	232.83 GiB	41 / 41
proxmox2	osd.1	hdd	bluestore	down ○ / out ○	17.2.8	0.45479	0.00	0.00	0 B	0 / 0
proxmox1	osd.0	hdd	bluestore	up ○ / in ●	17.2.8	0.22739	1.00	13.51	232.83 GiB	61 / 61

Figura 36: Estado do *cluster* após remoção de um OSD

Após remover o segundo OSD, o sistema passou a funcionar apenas com um OSD ativo (Figura 37). Apesar do *cluster* operar temporariamente com apenas um OSD disponível, o campo *Usage* continua a exibir a capacidade total teórica do Ceph (Figura 38). Este comportamento é normal no Ceph, pois o valor reportado resulta da soma das capacidades dos OSDs definidos na pool, dividida pelo número de réplicas que no momento consegue garantir, independentemente do estado atual dos OSDs. No entanto, como o número mínimo de réplicas definido na pool (*min_size=2*) não podia ser satisfeito, as operações de escrita foram automaticamente bloqueadas, assegurando a integridade dos dados.

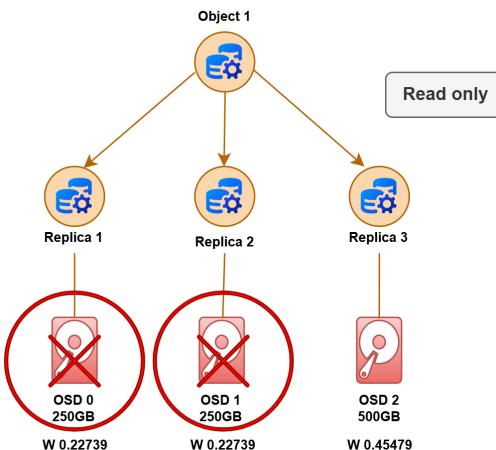


Figura 37: Exemplo da arquitetura após remoção de dois discos

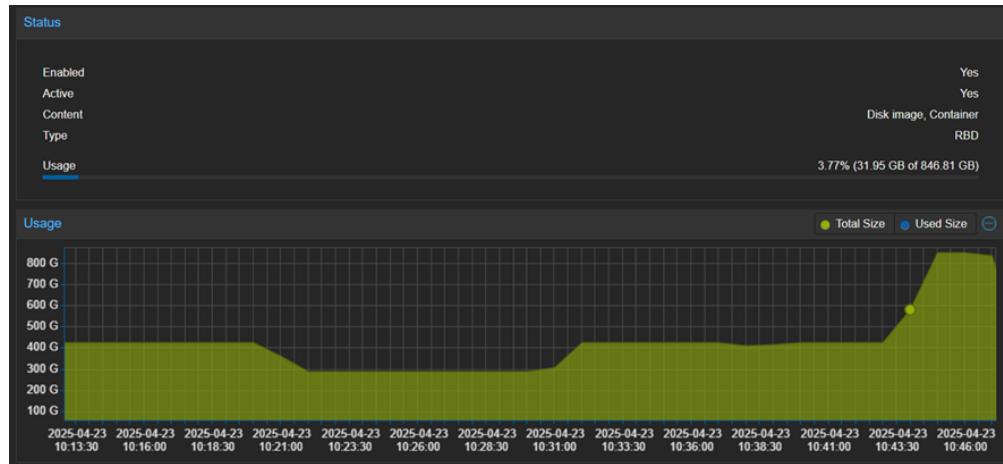


Figura 38: Capacidade de armazenamento após remoção de dois discos

Foram realizados testes de escrita tanto no volume CephFS como no armazenamento local. A tentativa de escrita no Ceph (*/mnt/pve/cephfs/teste1.txt*) ficou bloqueada, sem erro explícito, como mostrado na Figura 39. Esta reação é coerente com a política de integridade do Ceph, que impede a escrita sempre que o número mínimo de réplicas não é atingido.

Por contraste, a escrita no armazenamento local (`/home/teste`) decorreu com sucesso imediato (Figura 40), confirmando que o problema se restringia ao *cluster* Ceph.

```
172.79.79.3 - PuTTY
login as: root
root@172.79.79.3's password:
Linux proxmox2 6.8.12-8-pve #1 SMP PREEMPT_DYNAMIC PMX 6.8.12-8 (2025-01-24T12:32Z) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

ast logged in: Wed Apr 23 11:19:37 2025 from 172.79.79.232
root@proxmox2:~# echo "teste" > /mnt/pve/cephfs/testel.txt

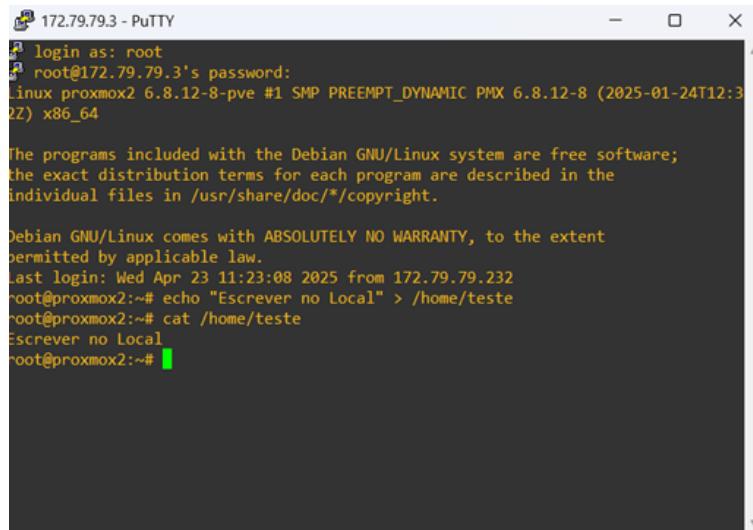
172.79.79.3 - PuTTY
root@172.79.79.3's password:
Linux proxmox2 6.8.12-8-pve #1 SMP PREEMPT_DYNAMIC PMX 6.8.12-8 (2025-01-24T12:32Z) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.

Last login: Wed Apr 23 11:19:32 2025 from 172.79.79.232
root@proxmox2:~# ps aux | grep "echo"
root      15529  0.0  0.0  6336  2048 pts/3   S+   11:19  0:00 grep echo
root@proxmox2:~# ps aux | grep "echo"
root      16492  0.0  0.0  6336  2048 pts/3   S+   11:23  0:00 grep echo
root@proxmox2:~# ps aux | grep "echo"
root      16497  0.0  0.0  6336  2048 pts/3   S+   11:23  0:00 grep echo
root@proxmox2:~# ps aux | grep "echo"
root      16535  0.0  0.0  6336  2304 pts/3   S+   11:23  0:00 grep echo
root@proxmox2:~# ps aux | grep "echo"
root      20017  0.0  0.0  6336  2176 pts/3   S+   11:37  0:00 grep echo
root@proxmox2:~# ps aux | grep "echo"
root      20040  0.0  0.0  6336  2048 pts/3   S+   11:37  0:00 grep echo
root@proxmox2:~#
```

Figura 39: Escrita bloqueada no volume CephFS com min_size=2



```

172.79.79.3 - PuTTY
login as: root
root@172.79.79.3's password:
Linux proxmox2 6.8.12-8-pve #1 SMP PREEMPT_DYNAMIC PMX 6.8.12-8 (2025-01-24T12:3
2Z) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Apr 23 11:23:08 2025 from 172.79.79.232
root@proxmox2:~# echo "Escrever no Local" > /home/teste
root@proxmox2:~# cat /home/teste
Escrever no Local
root@proxmox2:~#

```

Figura 40: Escrita bem-sucedida no armazenamento local

Como consequência, os serviços do Proxmox que dependiam de volumes Ceph (RBD ou CephFS) ficaram bloqueados em operações de I/O. Esta situação causou a indisponibilidade da interface gráfica (GUI) e impediu o login, mesmo com credenciais corretas, como ilustrado na Figura 41.

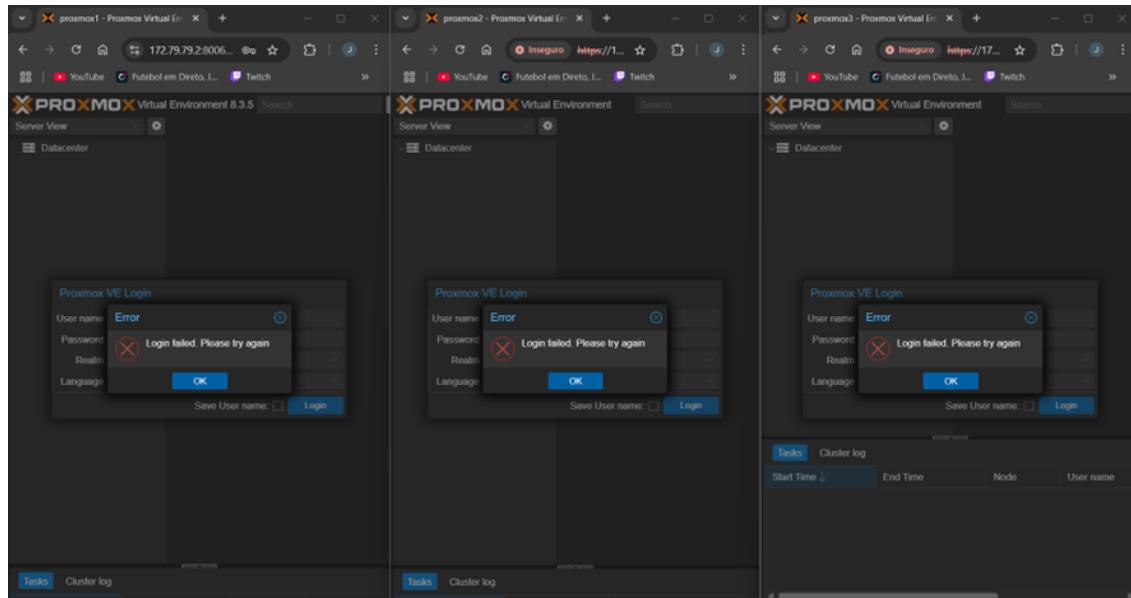


Figura 41: Indisponibilidade da GUI do Proxmox após falha do *cluster Ceph*

5.6.4 Discussão:

A remoção de dois OSDs provocou a degradação severa do *cluster*, limitando o acesso aos dados a operações de leitura. O Ceph respondeu corretamente, impedindo a escrita para proteger a integridade dos dados, de acordo com a configuração de *min_size=2*. Esta limitação refletiu-se diretamente na estabilidade dos serviços do Proxmox, tornando a interface web de administração inacessível até à recuperação do *cluster*. Assim, torna-se fundamental garantir que o número mínimo de réplicas definido seja igual ou inferior ao número mínimo de OSDs ativos para evitar falhas semelhantes.

5.7 Teste 7 – Falha de dois discos com *min_size = 1*

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.7.1 Objetivo:

Avaliar o comportamento do sistema Ceph quando dois discos (OSDs) são removidos fisicamente, num cenário em que o parâmetro *min_size = 1*. Pretende-se validar a capacidade de leitura, bem como a política de bloqueio de escrita para garantir a integridade dos dados.

5.7.2 Procedimento:

Neste teste seguiu-se o mesmo procedimento que o teste anterior mas desta vez com o *min_size = 1*.

5.7.3 Resultado:

Durante o teste, verificou-se que a interface gráfica (GUI) do Proxmox ficou extremamente lenta, alguns dados não carregavam e os CTs ou VMs não conseguiam ser iniciados. Esta degradação de serviço ocorreu porque os volumes estavam armazenados na pool Storage1, e o *cluster* funcionava apenas com um OSD ativo.

Apesar disso, como o parâmetro *min_size* estava configurado com o valor 1, o Ceph permitiu operações de leitura e escrita mesmo com uma única réplica. A Figura 42 mostra a execução bem-sucedida de comandos de escrita e leitura num volume CephFS, demonstrando que, tecnicamente, o armazenamento permaneceu funcional neste cenário.

```
root@proxmox1:~# echo "teste cephfs" > /mnt/pve/cephfs/teste-ceph.txt
root@proxmox1:~# cat /mnt/pve/cephfs/teste-ceph.txt
teste cephfs
```

Figura 42: Escrita e leitura testadas manualmente no volume CephFS

Após o teste, os discos foram reintegrados. O Proxmox reconheceu corretamente os discos, mas o serviço correspondente *ceph-osd@0.service* (exemplo para o OSD 0) não arrancou automaticamente (tal como já tinha ocorrido no teste da secção 5.5). Isto deveu-se ao facto de o serviço se encontrar em estado de falha (Figura 43). Embora o *hardware* estivesse funcional, foi necessário proceder ao reinício manual do serviço. Contudo, a falha persistiu inicialmente, uma vez que o ficheiro *keyring*, essencial para a autenticação do OSD junto do *cluster*, não estava presente, impossibilitando a sua inicialização correta. Face a esta situação, foi necessário remover completamente o OSD do Ceph e voltar a adicioná-lo ao *cluster* como um novo OSD, utilizando o mesmo disco.

```
root@proxmox1:~# systemctl status ceph-osd@0 -l
● ceph-osd@0.service - Ceph object storage daemon osd.0
   Loaded: loaded (/lib/systemd/system/ceph-osd@.service; disabled; preset: enabled)
   Drop-In: /usr/lib/systemd/system/ceph-osd@.service.d
             └─ceph-after-pve-cluster.conf
     Active: failed (Result: exit-code) since Wed 2025-04-23 15:19:18 WEST; 3min 4s ago
       Duration: 58ms
         Process: 12177 ExecStartPre=/usr/libexec/ceph/ceph-osd-prestart.sh --cluster ${CLUSTER} --id 0 (code=exited, status=0/SUCCESS)
         Process: 12181 ExecStart=/usr/bin/ceph-osd -f --cluster ${CLUSTER} --id 0 --setuser ceph --setgroup ceph (code=exited, stat...
      Main PID: 12181 (code=exited, status=1/FAILURE)
        CPU: 91ms

Apr 23 15:20:53 proxmox1 systemd[1]: Failed to start ceph-osd@0.service - Ceph object storage daemon osd.0.
Apr 23 15:21:48 proxmox1 systemd[1]: ceph-osd@0.service: Start request repeated too quickly.
Apr 23 15:21:48 proxmox1 systemd[1]: ceph-osd@0.service: Failed with result 'exit-code'.
Apr 23 15:21:48 proxmox1 systemd[1]: Failed to start ceph-osd@0.service - Ceph object storage daemon osd.0.
Apr 23 15:21:53 proxmox1 systemd[1]: ceph-osd@0.service: Start request repeated too quickly.
Apr 23 15:21:53 proxmox1 systemd[1]: ceph-osd@0.service: Failed with result 'exit-code'.
```

Figura 43: Estado do *ceph-osd@0* e mensagens de erro

5.7.4 Discussão:

Mesmo com *min_size* = 1, a falha de dois discos impacta severamente a operação do *cluster*, embora continue a ser possível realizar operações de escrita com apenas uma réplica, o que naturalmente compromete a tolerância a falhas.

Verificou-se ainda que, após a falha ou remoção de um OSD, a sua reintegração no *cluster* exige a remoção completa do OSD no Ceph e a sua posterior adição como um novo OSD, não sendo necessário reiniciar o servidor. Esta necessidade resulta da corrupção do ficheiro *keyring*, essencial para a autenticação do OSD, que ocorre aquando da sua remoção forçada.

5.8 Teste 8 – Adicionar um novo disco

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.8.1 Objetivo:

Verificar se o nó reconhece um novo disco e se é possível adicioná-lo ao Ceph sem necessidade de reiniciar o servidor.

5.8.2 Procedimento:

Para este teste, foi inserido um novo disco no nó *proxmox2*, em funcionamento (Figura 44) e adicionando-o como um OSD no ceph (*osd 3*).

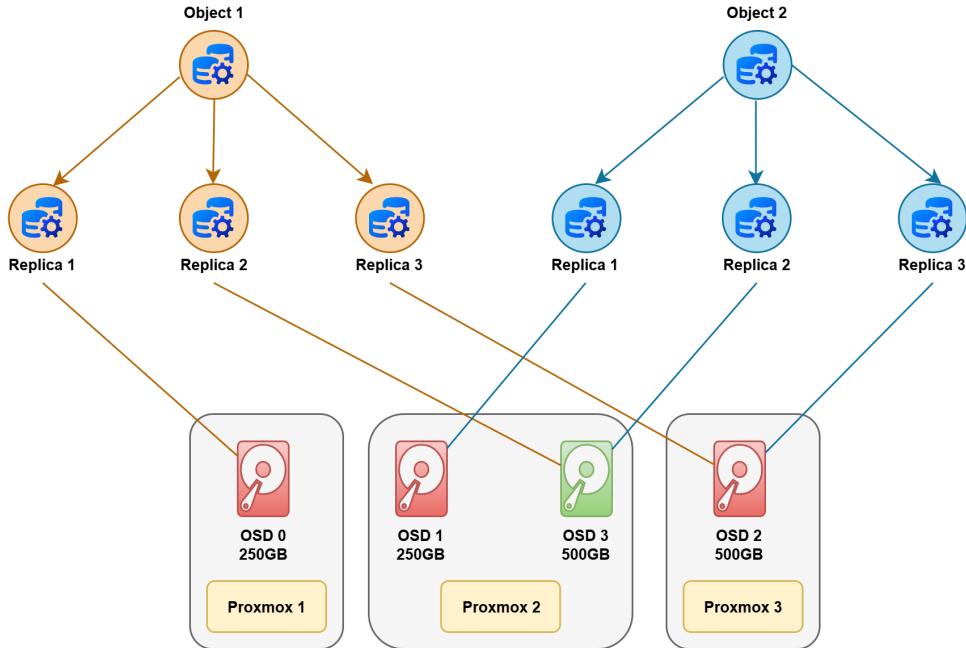


Figura 44: Exemplo da arquitetura do teste 5

5.8.3 Resultado:

Após a inserção do disco, o Proxmox reconheceu-o imediatamente, permitindo a sua adição ao cluster Ceph, como um novo OSD (*osd 3*), sem necessidade de reiniciar o servidor (Figura 45). O total de armazenamento disponível nas pools do Ceph foi atualizado automaticamente, refletindo o aumento da capacidade de armazenamento.

Name	Class	OSD Type	Status	Version	weight	reweight	Used (%)	Total	Apply/Commit Latency (ms)	PGs ↓
default										
proxmox3				17.2.8						
osd.2	hdd	bluestore	up ○ / in ●	17.2.8	0.22739	1.00	6.84	232.83 GiB	92 / 92	44
proxmox2				17.2.8						
osd.1	hdd	bluestore	up ○ / in ●	17.2.8	0.45479	1.00	6.30	465.76 GiB	38 / 38	71
osd.3	hdd	bluestore	up ○ / in ●	17.2.8	0.22739	1.00	1.34	232.88 GiB	71 / 71	11
proxmox1				17.2.8						
osd.0	hdd	bluestore	up ○ / in ●	17.2.8	0.22739	1.00	12.19	232.83 GiB	26 / 26	67

Figura 45: Novo disco adicionado como *osd 3*

5.8.4 Discussão:

Com este teste, conclui-se que o Ceph demonstra uma elevada flexibilidade, escalabilidade e capacidade de adaptação na gestão dinâmica de recursos de armazenamento, permitindo a adição

de novos OSDs de forma transparente e sem necessidade de interrupção do serviço.

5.9 Teste 9 – Migração do Sistema Operativo Proxmox VE para um Novo Disco

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.9.1 Objetivo:

Verificar a possibilidade de transferir o sistema operativo Proxmox VE de um disco com falhas para um novo disco vazio, garantindo a preservação de toda a configuração sem necessidade de reinstalação, e avaliando se o novo disco permite o arranque e funcionamento normal do sistema após a migração.

5.9.2 Procedimento:

Para este teste, foi utilizado o comando *dd* para realizar uma cópia bit a bit do disco original (*/dev/sda*), que apresentava falhas detetadas pelo SMART e não passou na verificação, para um novo disco (*/dev/sdc*), que foi posteriormente montado no lugar do disco original.

Após inserir o novo disco no mesmo nó onde o disco com falhas estava instalado, e confirmada a sua deteção pelo sistema (*/dev/sdc*), executou-se o seguinte comando:

```
dd if=/dev/sda of=/dev/sdc bs=64K conv=noerror,sync status=progress
```

5.9.3 Resultado:

Após a execução do comando *dd*, o novo disco */dev/sdc* passou a conter uma cópia integral do disco original, incluindo a tabela de partições, o sistema operativo e todos os dados do Proxmox VE (Figura 46). Posteriormente, o nó Proxmox 3 foi desligado, o disco com falhas (*/dev/sda*) foi removido e substituído pelo novo disco. Ao reiniciar o sistema, o Proxmox arrancou corretamente

a partir do novo disco, sem necessidade de qualquer intervenção adicional, confirmando o correto funcionamento do ambiente Proxmox VE.

```
root@proxmox3:~# dd if=/dev/sda of=/dev/sdc bs=64K conv=noerror,sync
132991483904 bytes (133 GB, 124 GiB) copied, 1915 s, 69.4 MB/s
dd: error reading '/dev/sda': Input/output error
2029288+1 records in
2029289+0 records out
132991483904 bytes (133 GB, 124 GiB) copied, 1919.43 s, 69.3 MB/s
293367775232 bytes (293 GB, 273 GiB) copied, 4817 s, 60.9 MB/s
dd: error reading '/dev/sda': Input/output error
4476435+2 records in
4476437+0 records out
293367775232 bytes (293 GB, 273 GiB) copied, 4821.1 s, 60.9 MB/s
293371707392 bytes (293 GB, 273 GiB) copied, 4831 s, 60.7 MB/s
dd: error reading '/dev/sda': Input/output error
4476494+3 records in
4476497+0 records out
293371707392 bytes (293 GB, 273 GiB) copied, 4835.98 s, 60.7 MB/s
320052330496 bytes (320 GB, 298 GiB) copied, 6000 s, 53.3 MB/s
4883922+4 records in
4883926+0 records out
320072974336 bytes (320 GB, 298 GiB) copied, 6007.74 s, 53.3 MB/s
```

Figura 46: Execução do comando dd no proxmox 3

5.9.4 Discussão:

A migração do sistema operativo para o novo disco ocorreu com sucesso. Todos os serviços, configurações e VMs mantiveram-se intactos e funcionais, confirmando a eficácia do procedimento. Este resultado demonstra que a substituição de discos defeituosos pode ser realizada de forma transparente, sem interrupções significativas e sem perda de dados, garantindo a continuidade operacional do ambiente Proxmox VE.

5.10 Teste 10 – Teste de adição de OSD e restabelecimento de réplicas no Ceph

Neste teste, são apresentados os objetivos, os procedimentos realizados, os resultados obtidos e a respetiva discussão.

5.10.1 Objetivo:

Verificar o armazenamento do *cluster Ceph* ao operar inicialmente com apenas dois OSDs, numa pool configurada para três réplicas com mínimo de duas. Avaliar o processo de adição de um novo OSD ao *cluster* para garantir a criação da terceira réplica em falta, analisando o rebalanceamento automático dos dados, assim como as alterações no armazenamento e na distribuição das réplicas.

5.10.2 Procedimento:

Foi adicionado um novo disco de 250 GB (Figura 48) num nó que já se encontrava em funcionamento. E adicionado como OSD no ceph.

5.10.3 Resultado:

A Figura 47 mostra o estado do *cluster* antes da operação, com apenas dois OSDs ativos e uma capacidade total de 699 GiB.

ID	CLASS	WEIGHT	REWEIGHT	SIZE	RAW USE	DATA	OMAP	META	AVAIL	%USE	VAR	PGS	STATUS
0	hdd	0.22739	1.00000	233 GiB	33 GiB	31 GiB	21 KiB	1.8 GiB	200 GiB	14.06	1.53	97	up
1	hdd	0.45479	1.00000	466 GiB	31 GiB	31 GiB	22 KiB	425 MiB	434 GiB	6.73	0.73	97	up
				TOTAL	699 GiB	64 GiB	62 GiB	43 KiB	2.2 GiB	635 GiB	9.17		

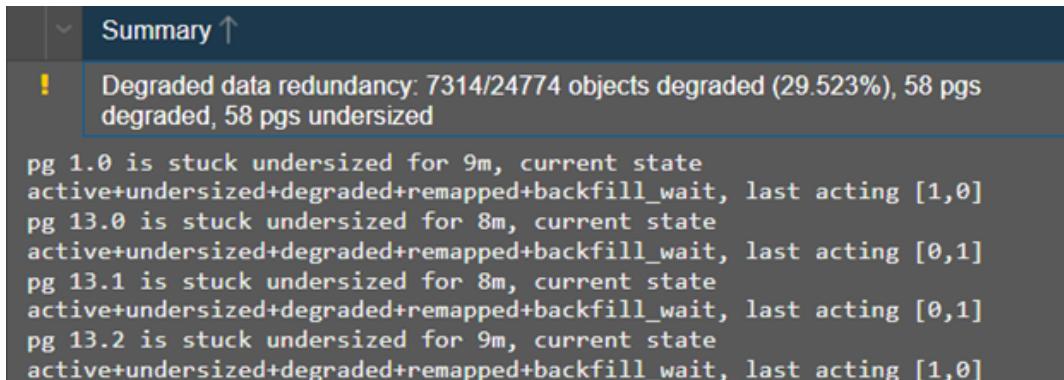
Figura 47: Estado do *cluster* antes da adição do novo OSD

Após a inserção e adição do disco como novo OSD, o *Ceph* reconheceu automaticamente o equipamento e iniciou o processo de rebalanceamento dos dados no *cluster*. A Figura 48 confirma o aumento da capacidade total de armazenamento para 931 GiB, refletindo a integração do novo disco. Contudo, observa-se que o campo *DATA* ainda apresentava valores assimétricos em relação aos demais OSDs, indicando que o processo de garantia da terceira réplica ainda não estava concluído.

ID	CLASS	WEIGHT	REWEIGHT	SIZE	RAW USE	DATA	OMAP	META	AVAIL	%USE	VAR	PGS	STATUS
0	hdd	0.22739	1.00000	233 GiB	33 GiB	31 GiB	21 KiB	1.8 GiB	200 GiB	14.06	2.03	97	up
1	hdd	0.45479	1.00000	466 GiB	31 GiB	31 GiB	22 KiB	425 MiB	434 GiB	6.73	0.97	97	up
2	hdd	0.22739	1.00000	233 GiB	407 MiB	117 MiB	1 KiB	290 MiB	232 GiB	0.17	0.02	32	up
				TOTAL	931 GiB	64 GiB	62 GiB	45 KiB	2.5 GiB	867 GiB	6.92		

Figura 48: Capacidade total do *cluster* após a adição do novo OSD

A Figura 49 apresenta o estado do *cluster* durante o rebalanceamento, evidenciado pela presença de objetos degradados, PGs marcadas como *degraded* e utilização desigual entre os OSDs, indicando que algumas réplicas ainda não foram totalmente garantidas. Este estado é esperado durante o processo de redistribuição dos dados, pois o Ceph mantém a consistência e a disponibilidade ao garantir as três réplicas configuradas na pool.



The screenshot shows a 'Summary' section with a warning message: 'Degraded data redundancy: 7314/24774 objects degraded (29.523%), 58 pgs degraded, 58 pgs undersized'. Below the message, there is a list of OSDs with their current states:

- pg 1.0 is stuck undersized for 9m, current state active+undersized+degraded+remapped+backfill_wait, last acting [1,0]
- pg 13.0 is stuck undersized for 8m, current state active+undersized+degraded+remapped+backfill_wait, last acting [0,1]
- pg 13.1 is stuck undersized for 8m, current state active+undersized+degraded+remapped+backfill_wait, last acting [0,1]
- pg 13.2 is stuck undersized for 9m, current state active+undersized+degraded+remapped+backfill_wait, last acting [1,0]

Figura 49: Estado do *cluster* durante o rebalanceamento dos dados

Os gráficos de utilização da CPU e da rede (Figura 50 e Figura 51) mostram um aumento significativo da atividade, seguido de uma queda muito grande, indicando o fim do processo de redistribuição de dados.

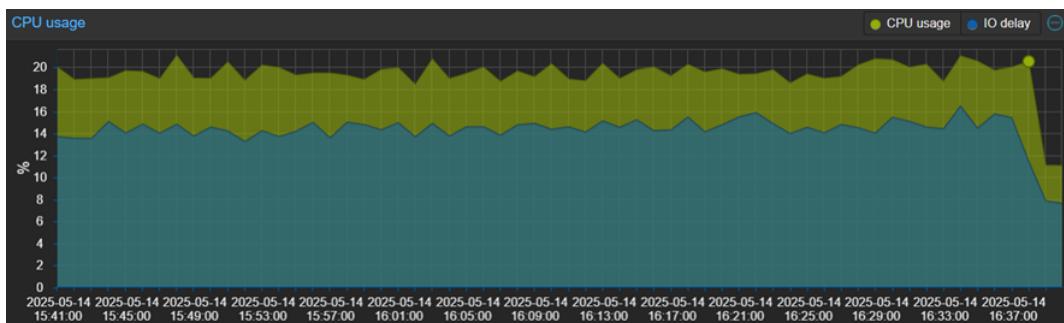


Figura 50: Utilização de CPU durante o rebalanceamento

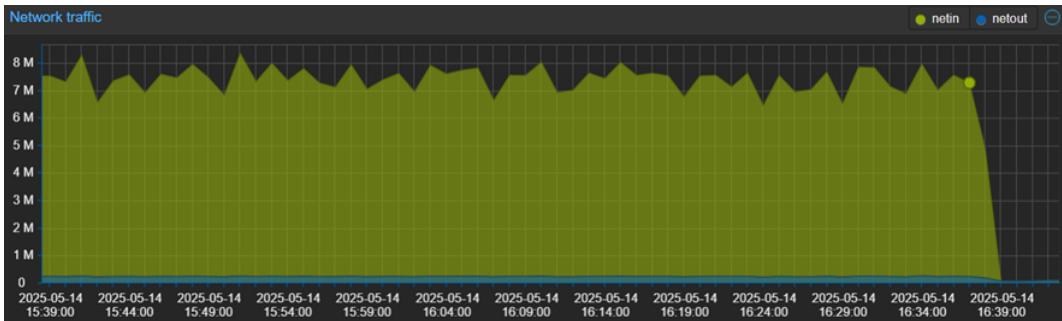


Figura 51: Tráfego de rede durante o rebalanceamento

A Figura 52 apresenta o estado final do *cluster*, já com os três OSDs ativos e integrados, e todas as PGs em estado *active+clean*. O estado geral do *cluster* foi restaurado para *HEALTH_OK*.

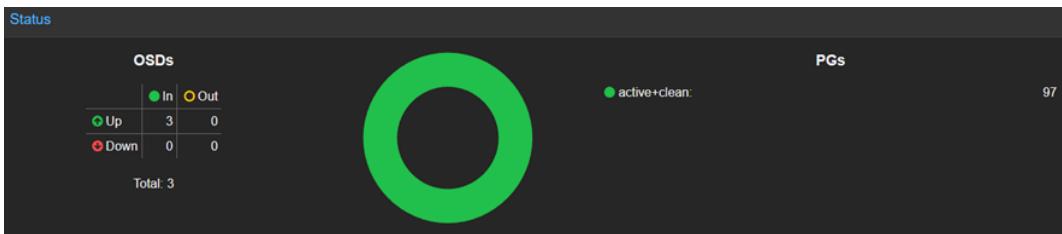


Figura 52: Estado final do *cluster* com 3 OSDs

A Figura 53 mostra o comando *ceph osd df* após a conclusão do rebalanceamento, onde se verifica que os dados foram redistribuídos de forma uniforme, com 31 GiB de *DATA* em cada OSD.

ID	CLASS	WEIGHT	REWEIGHT	SIZE	RAW USE	DATA	OMAP	META	AVAIL	%USE	VAR	PGS	STATUS
0	hdd	0.22739	1.00000	233 GiB	33 GiB	31 GiB	21 KiB	1.8 GiB	200 GiB	14.05	1.37	97	up
1	hdd	0.45479	1.00000	466 GiB	31 GiB	31 GiB	22 KiB	433 MiB	434 GiB	6.73	0.66	97	up
2	hdd	0.22739	1.00000	233 GiB	31 GiB	31 GiB	1 KiB	298 MiB	202 GiB	13.41	1.31	97	up
TOTAL				931 GiB	95 GiB	93 GiB	45 KiB	2.5 GiB	836 GiB	10.23			
MIN/MAX VAR: 0.66/1.37 STDDEV: 3.51													

Figura 53: Distribuição equilibrada dos dados nos OSDs após o rebalanceamento

5.10.4 Discussão:

A adição do novo OSD foi bem-sucedida e o rebalanceamento foi iniciado automaticamente pelo *Ceph*. O sistema aumentou a capacidade total disponível de 699, GiB para 931, GiB, redistribuiu os dados entre os três OSDs e restaurou o estado de saúde do *cluster*. A operação decorreu sem a necessidade de reiniciar e confirmou a escalabilidade e resiliência da solução.

O Ceph assegura, sempre que possível, a manutenção do número configurado de réplicas para cada objeto na pool. Caso alguma réplica esteja temporariamente indisponível, o sistema inicia automaticamente o processo de criação da réplica em falta assim que houver capacidade e condições para tal, garantindo a integridade e disponibilidade dos dados.

5.11 Teste 11 – Escrita intensiva no Ceph ao nível RADOS com 2 e 3 réplicas

Neste teste, são apresentados os objetivos, os procedimentos realizados, as análises feitas, os resultados obtidos e a respetiva discussão.

5.11.1 Objetivo:

Avaliar o desempenho do sistema de armazenamento distribuído *Ceph* ao nível mais baixo (camada RADOS), através da realização de testes de escrita intensiva com a ferramenta *rados bench*. Pretende-se analisar o impacto direto sobre os OSDs, observando métricas como a utilização da CPU, latência de escrita, tráfego de rede entre os nós e ocupação do espaço na *pool*, com vista à comparação dos cenários configurados com 2 e 3 réplicas, de forma a validar a estabilidade, o desempenho e a escalabilidade do *cluster* em situações de utilização intensiva.

5.11.2 Procedimento:

Foram realizados dois testes com o comando *rados bench*, no modo de escrita (*write*), com duração de 5 minutos (300 segundos) e 4 threads de execução (*-t 4*). O parâmetro *--no-cleanup* foi utilizado para manter os dados no final do teste.

Os testes foram efetuados em duas pools distintas, uma configurada com 2 réplicas (testpool2) e outra com 3 réplica (testpool3).

Nó de execução: proxmox2

```
# 2 réplicas
rados bench -p testpool2 300 write -t 4 --no-cleanup
# 3 réplicas
```

```
rados bench -p testpool3 300 write -t 4 --no-cleanup
```

5.11.3 Análise:

No gráfico de utilização de CPU (Figura 54), observa-se uma ligeira diferença entre os cenários de duas e três réplicas. Com três réplicas, regista-se um aumento na utilização de CPU, sobretudo no nó proxmox1, que apresenta uma carga mais elevada e consistente ao longo do teste, em comparação com o cenário de duas réplicas. Este aumento reflete o processamento adicional necessário para garantir a manutenção da terceira réplica dos dados.

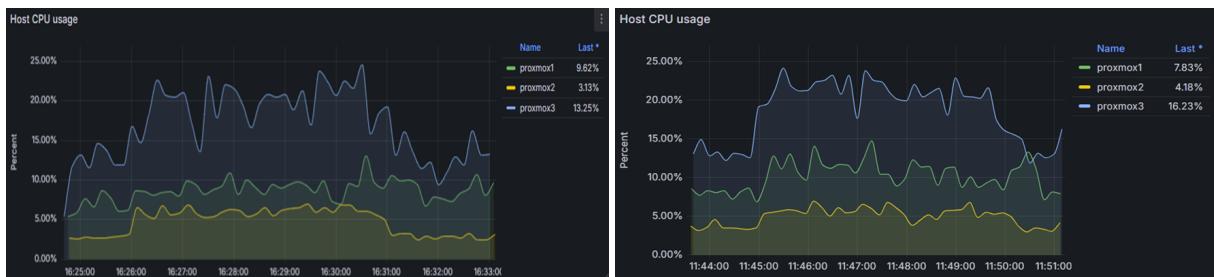


Figura 54: Utilização de CPU com 2 réplicas (esq.) e 3 réplicas (dir.)

Registaram-se picos de latência mais acentuados, no OSD 0 a ultrapassar os 300ms (Figura 55), com 2 réplicas. Isto deve-se ao facto de, com apenas 2 réplicas, a escrita ser distribuída por um número mais reduzido de OSDs, o que pode sobrecarregar temporariamente um dos discos (que acaba por ser selecionado com maior frequência). Com 3 réplicas, a escrita é distribuída por mais OSDs, o que resulta numa latência mais estável, com valores médios entre 100–150ms e menor variação ao longo do tempo.

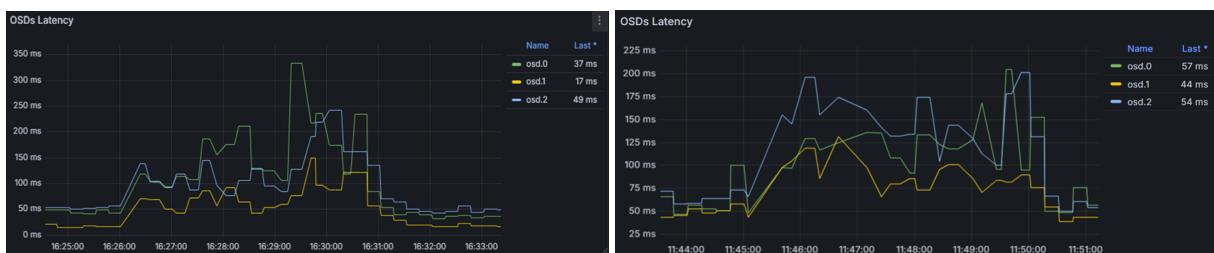


Figura 55: Latência dos OSDs com 2 réplicas (esq.) e 3 réplicas (dir.)

Verificou-se um tráfego de saída (Figura 56) significativamente mais elevado no *proxmox2*, ultrapassando os 12MB/s, refletindo o facto de ser o nó onde o teste foi executado e, consequentemente, o principal responsável pela escrita de dados para os restantes nós do *cluster*. O *proxmox1* e o *proxmox3* também apresentaram tráfego relevante, por receberem os dados destinados à replicação. No cenário com 3 réplicas, observa-se uma distribuição mais equilibrada do tráfego entre os três nós, como seria expectável devido à replicação adicional, o que confirma o impacto direto do fator de replicação na carga de rede entre os nós.

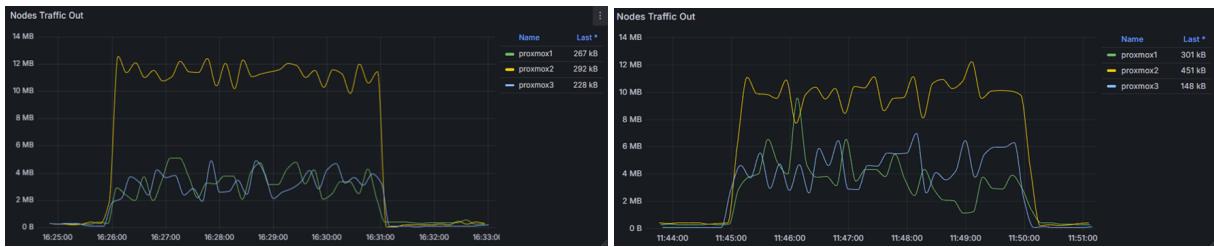


Figura 56: Tráfego de saída com 2 réplicas (esq.) e 3 réplicas (dir.)

Os gráficos de tráfego de entrada (Figura 57) evidenciam diferenças relevantes entre as configurações com 2 e 3 réplicas. No teste com 2 réplicas, observou-se um tráfego de entrada mais elevado no nó *proxmox1*, o que indica que este teve um papel mais ativo como destino primário dos dados. Os nós *proxmox2* e *proxmox3* também participaram na receção dos dados, mas com valores ligeiramente inferiores.

No teste com 3 réplicas, apesar de o número de réplicas ser superior, o tráfego de entrada no *proxmox2* foi inferior ao observado no teste com 2 réplicas. Esta aparente contradição explica-se pelo facto de o Ceph distribuir dinamicamente os OSDs primários responsáveis por armazenar os dados recebidos, com base no algoritmo CRUSH. Assim, no teste com 3 réplicas, a carga foi mais uniforme repartida entre os três nós, sendo *proxmox2* menos solicitado diretamente, o que justifica os valores inferiores de tráfego nesse nó.

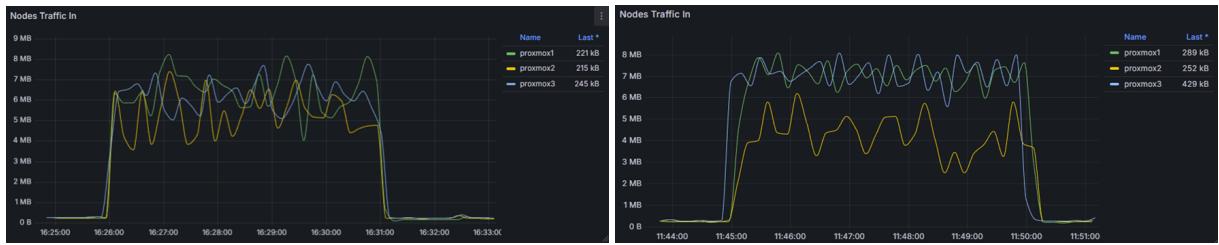


Figura 57: Tráfego de entrada com 2 réplicas (esq.) e 3 réplicas (dir.)

Os gráficos apresentados na Figura 58 correspondem às estatísticas de writes do Ceph, refletindo a taxa de escrita no sistema durante os testes com 2 e 3 réplicas. No teste com 2 réplicas, observa-se uma atividade de escrita mais frequente e com picos acentuados, atingindo até 11MB/s, com oscilações regulares entre 2 e 4MB/s. Já no teste com 3 réplicas, os picos são mais baixos e espaçados, registando-se apenas um valor mais elevado próximo dos 9MB/s, mantendo-se, no geral, abaixo dos 2MB/s.

Esta diferença justifica-se pelo custo adicional associado ao mecanismo de replicação. Com 3 réplicas, cada operação de escrita exige confirmação em três OSDs, o que introduz maior latência e reduz o *throughput* (taxa de transferência) global de escrita. Com apenas 2 réplicas, esse impacto é inferior, permitindo uma maior fluidez e volumes de escrita mais elevados em curtos períodos de tempo.

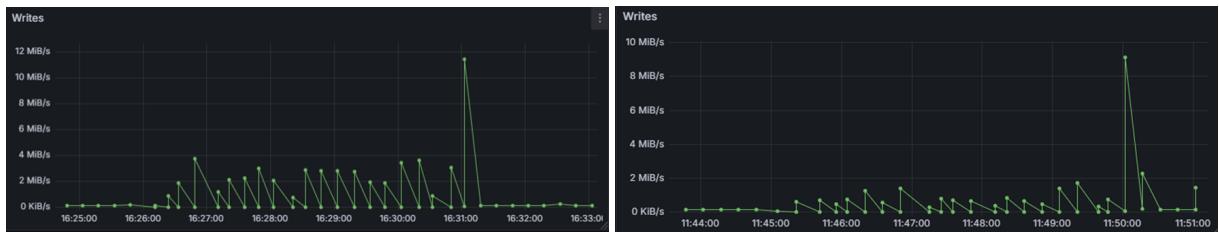


Figura 58: Escritas no Ceph com 2 réplicas (esq.) e 3 réplicas (dir.)

5.11.4 Resultado:

Com os valores presentes na Tabela 12 confirma-se o comportamento esperado do sistema de replicação do Ceph: o volume real de dados armazenados é multiplicado pelo número de réplicas, sendo este fator determinante na utilização do espaço. Além disso, evidencia-se que, mesmo com

menos dados efetivamente escritos no teste com 3 réplicas, o espaço ocupado aproximou-se ao do teste com 2 réplicas, devido ao maior fator de replicação.

Parâmetro	Replicação 2/2 (Figura 59)	Replicação 3/3 (Figura 60)
Número de escritas	766	469
Tamanho de cada escrita	4.194.304 bytes (4 MiB)	4.194.304 bytes (4 MiB)
Dados originais escritos	2,99 GiB	1,83 GiB
Fator de replicação	2	3
Espaço total ocupado	5,98 GiB	5,50 GiB

Tabela 12: Resumo das escritas com replicação 2/2 e 3/3

Create	Edit	Destroy	Pool #	Name	Size/min	# of Placem...	Optimal # of ...	Autoscaler...	CRUSH Rule (ID)	Used (%)
			1	.mgr	3/2	1	1	on	replicated_rule (0)	56.64 MiB (0.01%)
			13	cephfs_data	3/2	32	32	on	replicated_rule (0)	47.01 GiB (6.24%)
			14	cephfs_metadata	3/2	32	16	on	replicated_rule (0)	8.61 MiB (0.00%)
			15	Storage1	3/2	32	32	on	replicated_rule (0)	75.76 GiB (9.69%)
			23	testpool2	2/2	32	32	on	replicated_rule (0)	5.98 GiB (0.84%)
										128.81 GiB

Figura 59: Espaço total ocupado na pool *testpool2* com 2 réplicas

Create	Edit	Destroy	Pool #	Name	Size/min	# of Placem...	Optimal # of ...	Autoscaler...	CRUSH Rule (ID)	Used (%)
			1	.mgr	3/2	1	1	on	replicated_rule (0)	56.64 MiB (0.01%)
			13	cephfs_data	3/2	32	32	on	replicated_rule (0)	47.01 GiB (6.28%)
			14	cephfs_metadata	3/2	32	16	on	replicated_rule (0)	8.61 MiB (0.00%)
			15	Storage1	3/2	32	32	on	replicated_rule (0)	75.76 GiB (9.75%)
			23	testpool2	2/2	32	32	on	replicated_rule (0)	5.98 GiB (0.85%)
			26	testpool3	3/3	32	32	on	replicated_rule (0)	5.50 GiB (0.78%)
										134.31 GiB

Figura 60: Espaço total ocupado na pool *testpool3* com 3 réplicas

5.11.5 Discussão:

Durante os testes de escrita, verificou-se que a configuração com 2 réplicas resultou num volume total de dados originais escritos superior ao da configuração com 3 réplicas. A diferença ultrapassou 1 GiB, o que se justifica pelo facto de, com 3 réplicas, o desempenho de escrita ser inferior, uma vez que o sistema necessita garantir a consistência entre um maior número de OSDs, reduzindo assim a taxa de escrita ao longo do tempo.

5.12 Teste 12 – Leitura intensiva no Ceph ao nível RADOS com 2 e 3 réplicas

Neste teste, são apresentados os objetivos, os procedimentos realizados, as análises feitas, os resultados obtidos e a respetiva discussão.

5.12.1 Objetivo:

Avaliar o desempenho do sistema de armazenamento distribuído *Ceph* durante uma operação de leitura intensiva. Pretende-se observar o impacto desta operação nas métricas de utilização de CPU, latência dos OSDs, tráfego de rede e comportamento dos nós, comparando os resultados obtidos com 2 réplicas e com 3 réplicas.

5.12.2 Procedimento:

Foram realizados dois testes com o comando rados bench, no modo de escrita (read), com duração de 5 minutos (300 segundos) e 4 threads de execução (-t 4). O parâmetro --no-cleanup foi utilizado para manter os dados no final do teste. Para garantir a medição real do desempenho de leitura nos OSDs, a cache do sistema operacional foi previamente limpa.

Os testes foram efetuados em duas pools distintas: uma configurada com 2 réplicas (testpool2) e outra com 3 réplicas (testpool3).

Nó de execução: proxmox2

```
# 2 réplicas
rados bench -p testpool2 300 read -t 4 --no-cleanup
# 3 réplicas
rados bench -p testpool3 300 read -t 4 --no-cleanup
```

5.12.3 Análise:

A análise dos gráficos de utilização da CPU (Figura 61) revelou comportamentos bastante semelhantes entre os testes com 2 e 3 réplicas. O consumo de CPU manteve-se relativamente estável.

No entanto, observou-se um ligeiro aumento pontual no proxmox2 no início da execução do teste, coincidindo com o momento de arranque da operação de leitura, o que é esperado dado que foi o nó onde o comando foi executado. No entanto, com 3 réplicas, o consumo manteve-se com valores mais estáveis até ao final da execução.

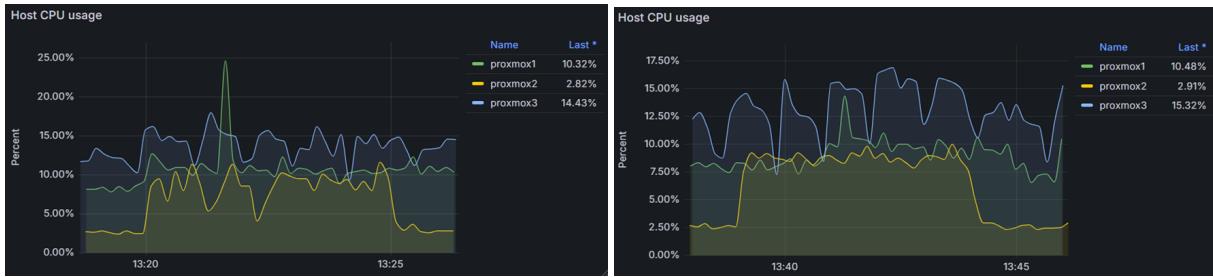


Figura 61: Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)

Os gráficos de latência dos OSDs apresentam comportamentos similares (Figura 62) mantendo-se maioritariamente abaixo dos 100ms. Observam-se, no entanto, picos pontuais de latência superiores, nomeadamente no *osd.2* no teste com 2 réplicas (acima dos 300ms) e no *osd.0* no teste com 3 réplicas (cerca de 225ms). No geral, a latência manteve-se controlada em ambos os testes, com pequenas variações que podem estar associadas tanto à distribuição dos pedidos de leitura pelo algoritmo CRUSH, como às diferenças de desempenho entre os próprios discos físicos.

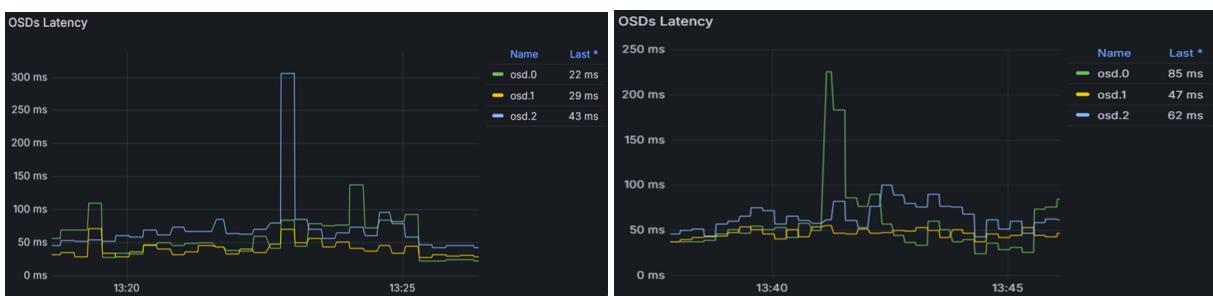


Figura 62: Latência dos OSDs com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de tráfego de entrada (Figura 63) dos nós durante os testes de leitura, observa-se que o padrão de utilização se mantém praticamente idêntico tanto com 2 como com 3 réplicas. O tráfego no nó proxmox2 destaca-se, atingindo valores constantes superiores a 12,MB/s em ambos

os cenários, o que confirma que a leitura foi efetuada a partir deste nó, sendo ele o responsável principal por disponibilizar os dados ao utilizador.

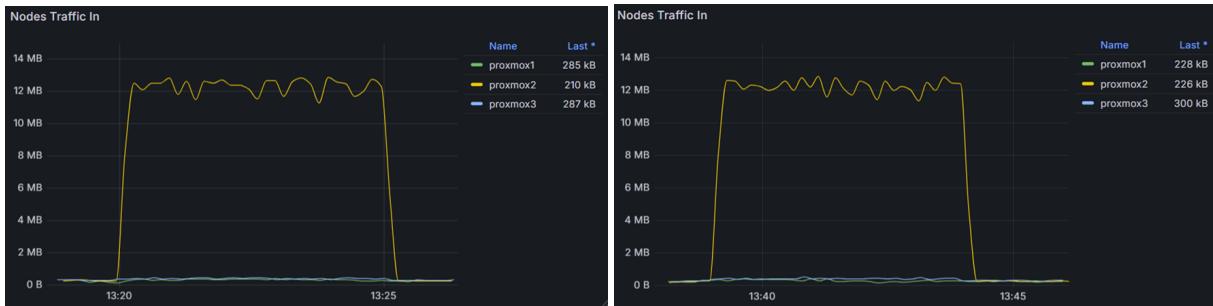


Figura 63: Tráfego de entrada com 2 réplicas (esq.) e 3 réplicas (dir.)

No teste com 2 réplicas, o tráfego de saída (Figura 64) foi equilibrado entre os nós, indicando uma distribuição uniforme dos OSDs primários. No teste com 3 réplicas, as taxas de saída foram, em média, mais baixas do que no cenário com 2 réplicas, sugerindo que, em vários momentos, a leitura foi satisfeita localmente, sem necessidade de recorrer aos outros nós. No entanto, já na fase final do teste, observa-se que o proxmox3 apresenta um tráfego de saída superior, indicando que foi mais frequentemente selecionado como OSD primário nessa fase. De forma geral, com 2 réplicas o gráfico apresenta um perfil mais uniforme ao longo do tempo.

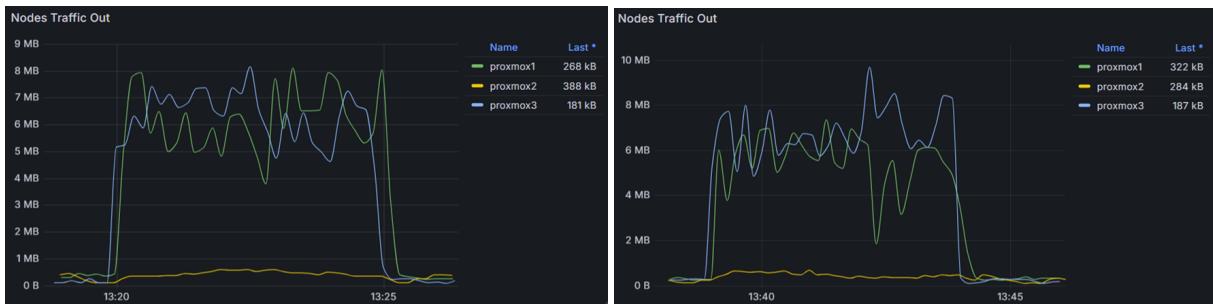


Figura 64: Tráfego de saída com 2 réplicas (esq.) e 3 réplicas (dir.)

Os gráficos apresentados na Figura 65 correspondem às estatísticas de leituras do Ceph durante os testes com 2 e 3 réplicas. Em ambos os casos, observa-se uma taxa de leitura semelhante, com picos entre 25 e 32MB/s e uma variação ligeira entre intervalos. Não se registaram diferenças significativas no desempenho das leituras em função do número de réplicas.

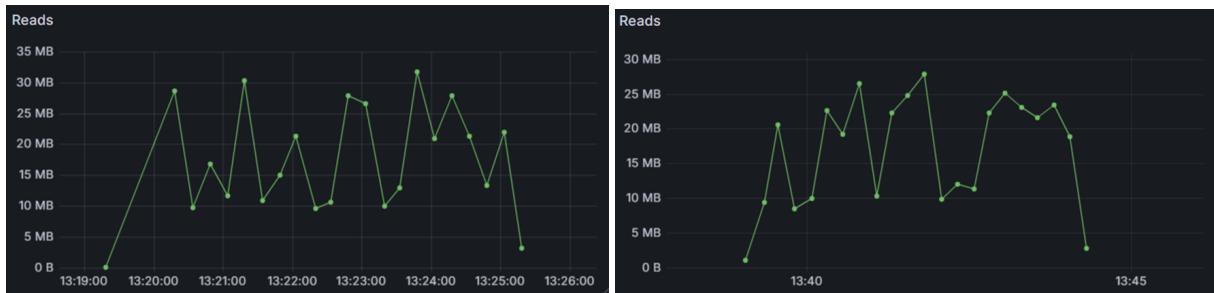


Figura 65: Leituras do Ceph com 2 réplicas (esq.) e 3 réplicas (dir.)

5.12.4 Resultado:

Com os valores presentes na Tabela 13, verifica-se que o número total de leituras foi superior no teste com 2 réplicas, resultando num volume de dados lido mais elevado em comparação com o teste com 3 réplicas. Esta diferença está associada à menor sobrecarga do sistema na configuração com 2 réplicas, permitindo maior taxa de leitura durante o tempo de teste.

Parâmetro	Replicação 2/2	Replicação 3/3
Número de leituras realizadas	1.496	1.355
Tamanho de cada leitura	4.194.304 bytes (4 MiB)	4.194.304 bytes (4 MiB)
Dados totais lidos	5,84 GiB	5,29 GiB

Tabela 13: Resumo das leituras com replicação 2/2 e 3/3

5.12.5 Discussão:

Verificou-se que o desempenho geral se manteve estável e eficiente em ambos os cenários, com latências reduzidas, tráfego de rede equilibrado e utilização moderada de CPU. No entanto, a configuração com 2 réplicas permitiu um número superior de leituras e um volume total de dados lidos maior, o que pode ser explicado pelo facto de que, com menos réplicas, o sistema exige menor coordenação e comunicação entre os OSDs. Essa redução na sobrecarga operacional resulta em menor consumo de recursos, melhorando a eficiência das operações de leitura.

5.13 Teste 13 – Escrita sequencial no Ceph RBD com 2 e 3 réplicas

Neste teste, são apresentados os objetivos, os procedimentos realizados, as análises feitas, os resultados obtidos e a respetiva discussão.

5.13.1 Objetivo:

Avaliar o desempenho de escrita sequencial num volume RBD do Ceph com múltiplas tarefas em paralelo e sem utilização de cache do sistema. O objetivo é simular uma operação prolongada de escrita direta, com o intuito de analisar o Ceph RBD, observando a utilização de CPU, latência dos OSDs, tráfego de rede e volume efetivo de dados escritos. O teste é realizado com fatores de 2 réplicas e 3 réplicas para comparar o impacto da replicação no desempenho.

5.13.2 Procedimento:

Foram realizados dois testes com o comando *fio*, no modo de escrita sequencial (write), com duração de 5 minutos (300 segundos) e 4 tarefas paralelas (*-numjobs=4*). O acesso ao volume RBD foi feito em modo direto (*-direct=1*) e blocos de 4MiB (*-bs=4M*). O parâmetro *-time_based* foi incluído para garantir que o teste decorresse durante o tempo especificado, independentemente do volume de dados.

Os testes foram realizados a partir de pools com 2 (pool2RBD) e 3 réplicas (pool3RBD), sendo criada uma imagem RBD para cada pool.

Nó de execução: proxmox2

```
# Teste com 2 e 3 réplicas
# Criar imagem RBD:
rbd create pool2RBD/testimg2x --size 10240
rbd create pool3RBD/testimg3x --size 10240
# Mapear imagem RBD:
rbd map pool2RBD/testimg2x
rbd map pool3RBD/testimg3x
```

```
# Executar teste de escrita:
fio --name=seqwrite --filename=/dev/rbd{x} --rw=write --bs=4M --iodepth=1 --numjobs=4 --
direct=1 --runtime=300 --time_based --group_reporting
```

5.13.3 Análise:

Os gráficos apresentados na Figura 66 mostram a utilização da CPU durante os testes com as imagens RBD. Comparativamente, observa-se que o cenário com 2 réplicas apresenta oscilações maiores na utilização do CPU, embora os níveis médios sejam semelhantes em ambos os casos.

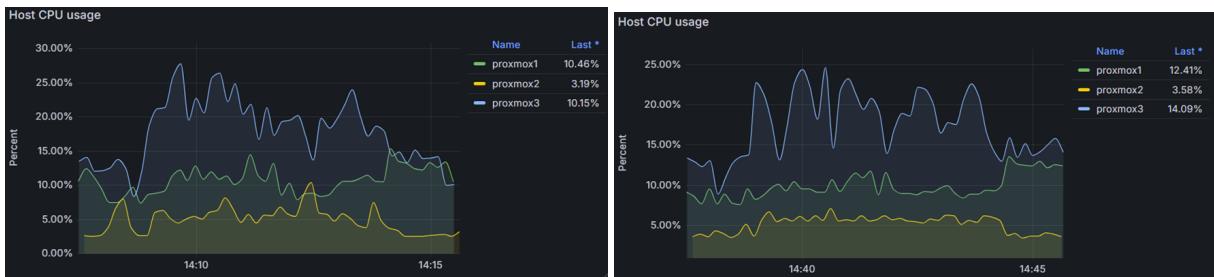


Figura 66: Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)

Observa-se uma diferença nos gráficos de tráfego de entrada (Figura 67) entre os testes com 2 e 3 réplicas. No cenário com 2 réplicas, o nó proxmox3 registou maior carga, indicando um papel mais ativo na receção dos dados primários. Já no teste com 3 réplicas, o tráfego distribuiu-se de forma mais equilibrada entre os nós, com menor envolvimento do proxmox2, resultado da sua escolha mais frequente como OSD primário pelo algoritmo CRUSH, em comparação com o cenário de 2 réplicas.

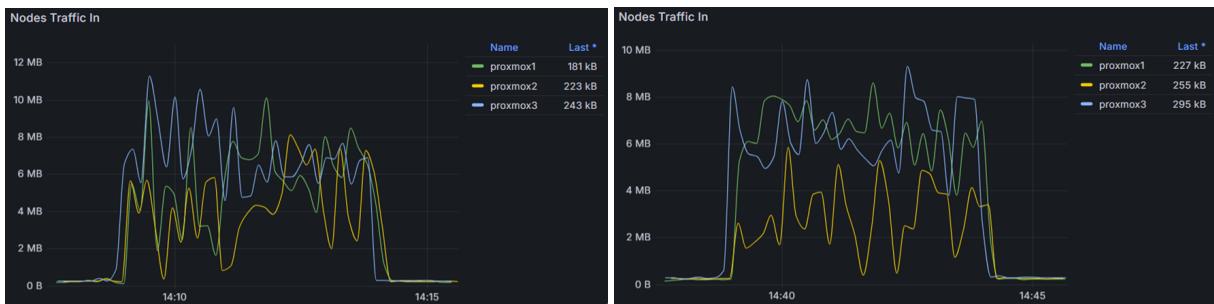


Figura 67: Tráfego de Entrada com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de tráfego de saída (Figura 68), verifica-se que o nó proxmox2 apresenta consistentemente o maior volume de dados enviados em ambos os testes. Isto deve-se ao facto de ser o nó responsável pela execução do comando *fio*, que gera a carga de escrita. Assim, é natural que este nó envie mais dados para os restantes, independentemente do número de réplicas. A diferença entre os dois testes é pouco significativa, embora se observe uma maior variação na taxa de saída no teste com 3 réplicas.

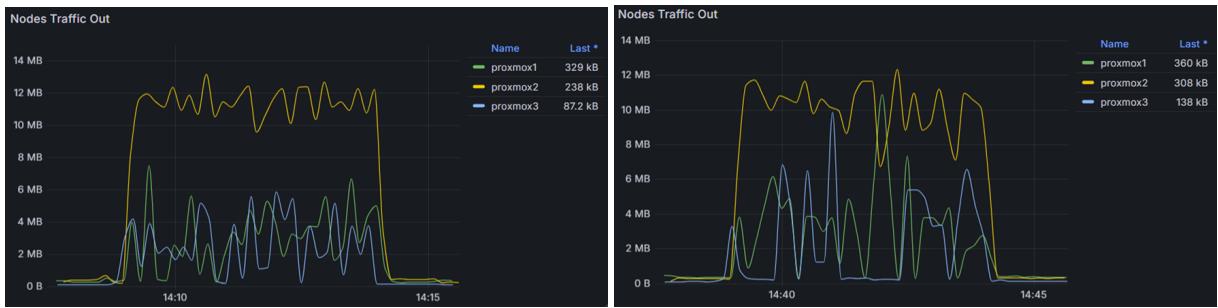


Figura 68: Tráfego de Saída com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de latência dos OSDs (Figura 69), verifica-se que, no teste com 2 réplicas, os valores de latência foram globalmente mais baixos e mais estáveis, com a maioria das medições a manter-se abaixo dos 120ms. Os OSDs apresentaram um comportamento relativamente uniforme, sem picos acentuados. No teste com 3 réplicas, observou-se um aumento das latências, com vários picos a ultrapassarem os 200ms, em particular nos OSDs 0 e 2. Esta maior variabilidade está associada à coordenação entre três réplicas, exigindo mais tempo para confirmar cada operação de escrita.

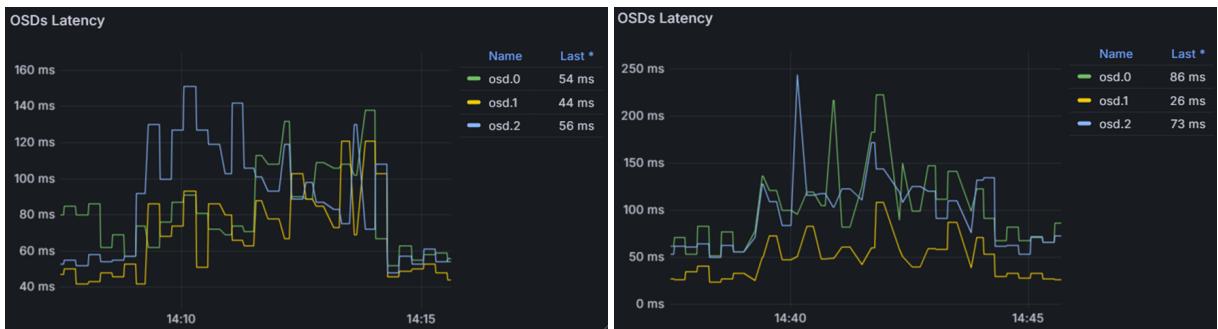


Figura 69: Latência dos OSDs com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de escritas do Ceph (Figura 70), verifica-se que, no teste com 2 réplicas, a atividade de escrita apresenta picos frequentes e de maior intensidade, chegando a ultrapassar os 11 MiB/s. Isto indica uma taxa de escrita mais elevada e rápida, embora menos estável. No caso das 3 réplicas, os picos são mais baixos e espaçados, mantendo-se abaixo dos 5 MiB/s, refletindo um impacto adicional da replicação, que aumenta a latência e reduz o *throughput*, resultando numa escrita mais lenta.

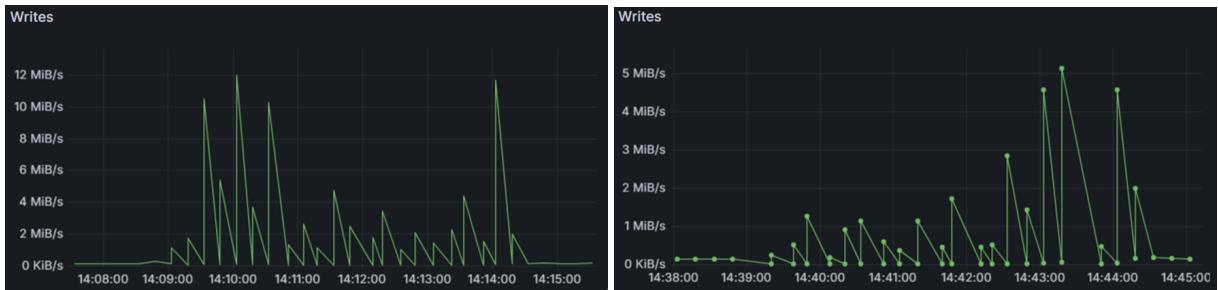


Figura 70: Escritas do Ceph com 2 réplicas (esq.) e 3 réplicas (dir.)

5.13.4 Resultado:

A Tabela 14 resume os dados dos testes realizados, evidenciando que, embora a replicação 3x implique um maior fator de cópia, o volume de dados escrito foi menor (0,36 GiB), resultando num total ocupado inferior (1,08 GiB) face à replicação 2x (1,36 GiB).

Métrica	Replicação 2 (Figura 71)	Replicação 3 (Figura 72)
Volume escrito	0,68 GiB	0,36 GiB
Número de réplicas	2	3
Total escrito no <i>cluster</i>	1,36 GiB	1,08 GiB

Tabela 14: Resumo do espaço ocupado nas pools com diferentes fatores de replicação

Create	Edit	Destroy	Pool #	Name	Size/min	# of Place...	Optimal # ...	Autoscale...	CRUSH Rule (ID)	Used (%)
			1	mgr	3/2	1	1	on	replicated_rule (0)	56.64 MiB (0.01%)
			13	cephfs_data	3/2	32	32	on	replicated_rule (0)	47.01 GiB (6.19%)
			14	cephfs_metadata	3/2	32	16	on	replicated_rule (0)	8.61 MiB (0.00%)
			15	Storage1	3/2	32	32	on	replicated_rule (0)	75.76 GiB (9.61%)
			27	pool2RDB	2/2	32	32	on	replicated_rule (0)	1.36 GiB (0.19%)
										124.19 GiB

Figura 71: Espaço total ocupado na pool testpool2 com 2 réplicas

Pool #	Name	Size/min	# of Place...	Optimal # ...	Autoscale...	CRUSH Rule (ID)	Used (%)
1	mgr	3/2	1	1	on	replicated_rule (0)	56.64 MiB (0.01%)
13	cephfs_data	3/2	32	32	on	replicated_rule (0)	47.01 GiB (6.19%)
14	cephfs_metadata	3/2	32	16	on	replicated_rule (0)	8.61 MiB (0.00%)
15	Storage1	3/2	32	32	on	replicated_rule (0)	75.76 GiB (9.62%)
27	pool2RDB	2/2	32	32	on	replicated_rule (0)	1.36 GiB (0.19%)
28	pool3RBD	3/3	32	32	on	replicated_rule (0)	1.08 GiB (0.15%)
							125.27 GiB

Figura 72: Espaço total ocupado na pool testpool3 com 3 réplicas

5.13.5 Discussão:

A análise dos resultados do teste confirma que a replicação em Ceph impacta diretamente o espaço ocupado no armazenamento e o desempenho das operações. Embora o teste com 3 réplicas apresente um fator de replicação superior, o espaço ocupado foi ligeiramente inferior devido à menor quantidade de dados escritos. Por outro lado, a replicação em 2 nós permitiu maior volume de dados escritos e ocupação proporcionalmente maior do espaço, uma vez que o sistema necessita de realizar menos operações de replicação em cada escrita, permitindo alcançar taxas de escrita mais elevadas durante o período de teste.

Comparativamente ao teste de escrita direta sobre a camada RADOS do Ceph na subseção 5.11 verificou-se que a escrita através da camada RBD foi significativamente mais lenta. Este resultado reflete a sobrecarga adicional introduzida pelas camadas superiores de abstração, como a gestão de blocos, metadata e a coordenação dos clients RBD, que acrescentam complexidade e latência ao processo de escrita.

5.14 Teste 14 – Leitura sequencial no Ceph RBD com 2 e 3 réplicas

Neste teste, são apresentados os objetivos, os procedimentos realizados, as análises feitas, os resultados obtidos e a respetiva discussão.

5.14.1 Objetivo

Avaliar o desempenho de leitura sequencial num volume RBD do Ceph, utilizando múltiplos fluxos em paralelo, para compreender o impacto do fator de replicação na latência, uso de CPU, tráfego de rede e *throughput*.

5.14.2 Procedimento:

Foram realizados seis testes com o comando *rbd bench*, no modo de leitura (*read*), com duração de 5 minutos (300 segundos) e 4 threads de execução (*--io-threads=4*). Para garantir a medição real do desempenho de leitura nos OSDs, a cache do sistema operacional foi previamente limpa.

Os testes foram executados individualmente em cada um dos 3 nós, abrangendo duas *pools* distintas: uma com 2 réplicas (pool2RBD) e outra com 3 réplicas (pool3RBD), utilizando o seguinte comando:

Nó de execução: proxmox1, proxmox2, proxmox3

```
rbd bench pool2RDB/testimg2x --io-type read --io-size 4M --io-threads 4  
rbd bench pool3RDB/testimg3x --io-type read --io-size 4M --io-threads 4
```

5.14.3 Análise:

Durante o teste, verifica-se um aumento na utilização da CPU em todos os nós do *cluster*, refletindo a carga gerada pela operação de leitura. O aumento é especialmente notório no nó onde o comando foi executado: no Proxmox1, a utilização de CPU sobe para valores próximos dos 15%, no Proxmox2 mantém-se geralmente abaixo dos 10%, e no Proxmox3 os picos atingem cerca de 30%. Esta variação indica que a carga é maior no nó onde o teste está a ser realizado, sendo claramente visível nos gráficos apresentados na Figura 73.

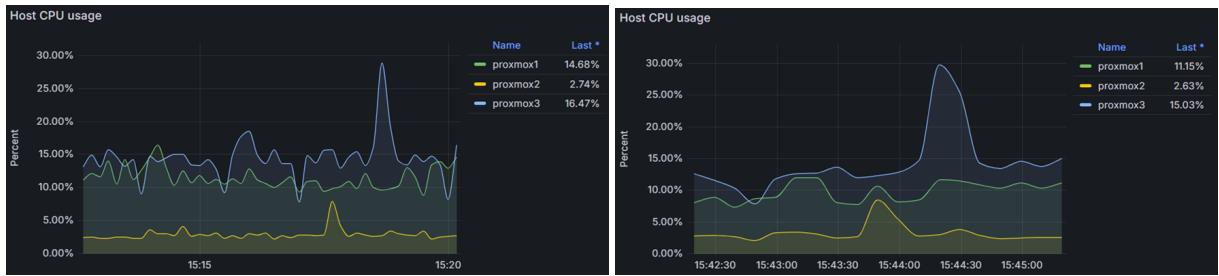


Figura 73: Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de tráfego de entrada (Figura 74), observa-se que, no teste com 2 réplicas, os valores máximos de tráfego são superiores aos registados no teste com 3 réplicas. Esta diferença resulta do facto de, no cenário com 3 réplicas, a carga ser distribuída de forma mais equilibrada entre os nós, originando picos individuais mais baixos. No teste com 2 réplicas, quando executado no nó proxmox2, o tráfego de entrada é menor, uma vez que o OSD do proxmox2 foi escolhido mais vezes como OSD primário, reduzindo a necessidade de comunicação com os restantes OSDs. Em ambos os casos, cada nó apresenta períodos distintos de maior atividade, correspondentes à execução local do teste.

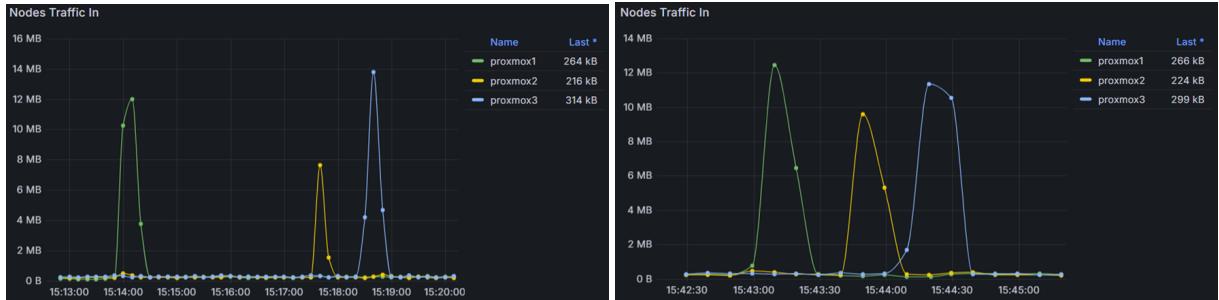


Figura 74: Tráfego de entrada com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de tráfego de saída (Figura 75), observa-se que, em ambos os testes, o nó que está a realizar a leitura não apresenta aumento significativo no tráfego de saída, enquanto os outros OSDs mostram picos claros, refletindo a redistribuição dos dados. Verifica-se ainda que os valores de tráfego de saída são ligeiramente superiores no teste com 2 réplicas.

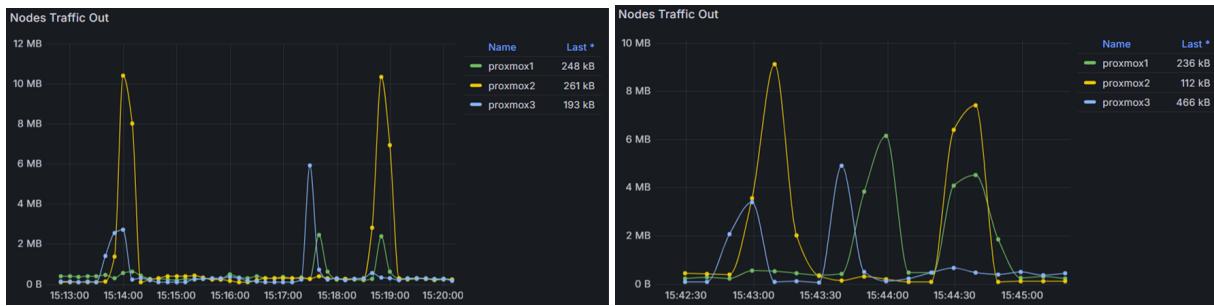


Figura 75: Tráfego de saída com 2 réplicas (esq.) e 3 réplicas (dir.)

5.14.4 Resultado:

A Tabela 15 compara o desempenho dos três nós nos testes com 2 réplicas e 3 réplicas. No teste com 2 réplicas, o proxmox2 foi mais rápido e apresentou maior *throughput*, pois acedeu à maioria dos dados localmente, sem necessidade de recorrer à rede para obter dados de outros OSDs. No teste com 3 réplicas, o tempo e o *throughput* do proxmox2 degradaram-se ligeiramente, uma vez que, com mais réplicas, a distribuição dos dados pelos nós é mais dispersa, aumentando a probabilidade de ter de aceder a réplicas remotas através da rede. Nos nós proxmox1 e proxmox3, as diferenças foram menos expressivas, dependendo essencialmente da localização das réplicas e da carga de leitura atribuída em cada execução.

Métrica	Proxmox1	Proxmox2	Proxmox3
Replicação 2/2			
Tempo total	21 s	7 s	18 s
Operações realizadas (ops)	256	256	256
Operações por segundo	12,07 ops/s	34,80 ops/s	13,70 ops/s
Taxa média de leitura (<i>throughput</i>)	48 MiB/s	138 MiB/s	55 MiB/s
Replicação 3/3			
Tempo total	15 s	12 s	19 s
Operações realizadas (ops)	256	256	256
Operações por segundo	16,31 ops/s	20,80 ops/s	13,47 ops/s
Taxa média de leitura (<i>throughput</i>)	65 MiB/s	83 MiB/s	54 MiB/s

Tabela 15: Comparação das métricas nos testes com replicação 2/2 e 3/3

5.14.5 Discussão:

Os valores de leitura variam consoante a localização original dos dados, pois o sistema tenta sempre ler a réplica primária, e também pelo overhead de sincronização, que é maior no caso de 3 réplicas. No entanto, não se registaram diferenças significativas no desempenho entre as leituras com 2 e 3 réplicas.

O **overhead** refere-se ao trabalho adicional que o sistema tem de realizar para copiar os dados para várias réplicas.

5.15 Teste 15 – Escrita Sequencial no CephFS com 2 e 3 réplicas

Neste teste, são apresentados os objetivos, os procedimentos realizados, as análises feitas, os resultados obtidos e a respetiva discussão.

5.15.1 Objetivo:

Avaliar o desempenho de escrita sequencial no sistema de ficheiros CephFS sob carga prolongada, simulando operações de escrita direta com múltiplas tarefas paralelas. O teste compara as configurações com 2 réplicas e 3 réplicas, com o objetivo de observar o impacto da replicação no desempenho.

5.15.2 Procedimento:

Foram realizados dois testes com o comando *fio*, no modo de escrita sequencial (write), com duração de 5 minutos (300 segundos) e 4 tarefas paralelas (*-numjobs=4*). Foi utilizado acesso direto à diretoria (*-direct=1*) e blocos de 4MiB (*-bs=4M*). O parâmetro *-time_based* garantiu que o teste decorresse durante o tempo especificado, independentemente do volume de dados.

Os testes foram realizados em duas pools distintas: uma com 2 réplicas (*testcephfs*) e outra com 3 réplicas (*cephfs*).

Nó de execução: proxmox2

```
# Teste com 2 réplicas
fio --name=cephfs-write --directory=/mnt/pve/testcephfs --rw=write --bs=4M --numjobs=4 --
    runtime=300 --time_based --ioengine=libaio --iodepth=4 --group_reporting --direct=1

# Teste com 3 réplicas
fio --name=cephfs-write --directory=/mnt/pve/cephfs --rw=write --bs=4M --numjobs=4 --
    runtime=300 --time_based --ioengine=libaio --iodepth=4 --group_reporting --direct=1
```

5.15.3 Análise:

Os gráficos da Figura 76 mostram a utilização da CPU nos três nós durante o teste de escrita sequencial no CephFS, com configurações de 2 réplicas e 3 réplicas. Observa-se que não houve grandes variações na utilização da CPU entre os dois testes.

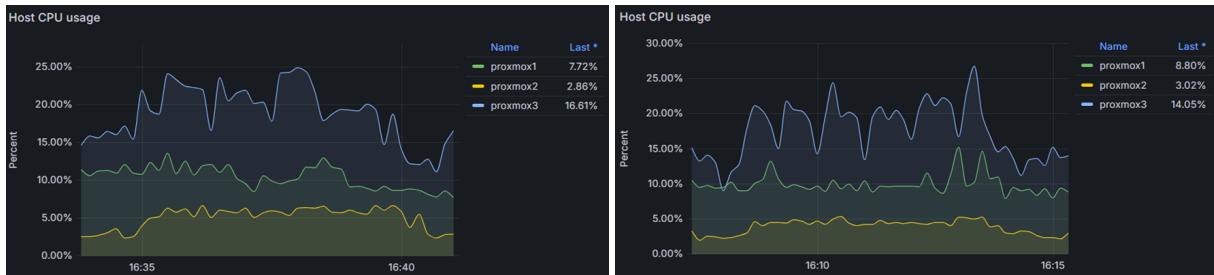


Figura 76: Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de tráfego de saída (Figura 77), observa-se que o nó proxmox2 apresenta um volume de tráfego significativamente superior em relação aos outros nós. Isto deve-se ao facto de ser o nó onde o teste foi executado. Os valores nos outros nós são consideravelmente inferiores, refletindo o papel passivo na replicação dos dados. Esta tendência mantém-se tanto para a configuração com 2 réplicas como para a de 3 réplicas, embora haja ligeiras variações na intensidade do tráfego entre os dois casos.



Figura 77: Tráfego de Saída com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de tráfego de entrada (Figura 78), observa-se uma variação significativa no comportamento do proxmox2, nó onde o comando do teste foi executado. No cenário com 2 réplicas, o proxmox2 apresenta um tráfego de entrada mais elevado, pois foi mais vezes escolhido para armazenar réplicas dos dados. Já no teste com 3 réplicas, esse valor é menor, devido à sua escolha mais frequente como OSD primário, o que reduz o tráfego de entrada necessário, uma vez que o comando foi executado neste nó.



Figura 78: Tráfego de Entrada com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de latência dos OSDs (Figura 79), observa-se que, no teste com 2 réplicas, a latência é geralmente menor e mais estável em comparação com o teste com 3 réplicas, onde surgem picos mais elevados. Esta diferença evidencia o impacto do overhead adicional decorrente da replicação em 3 réplicas, que pode aumentar a latência durante as operações de escrita.

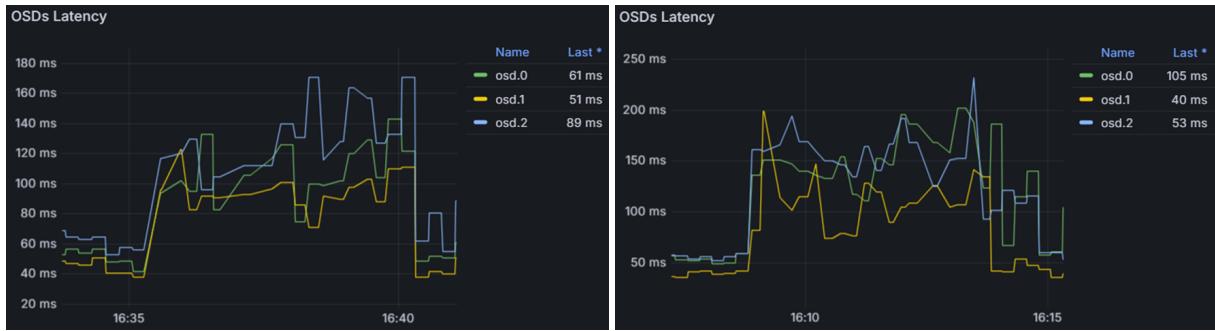


Figura 79: Latência dos OSDs com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de escritas do Ceph (Figura 80), observa-se que a taxa de escrita no teste com 2 réplicas atinge picos superiores, chegando a aproximadamente 11MiB/s e 6MiB/s em diferentes momentos. Já no teste com 3 réplicas, os picos são mais baixos, atingindo apenas cerca de 3MiB/s. Esta diferença indica que a replicação a dois nós permite maior velocidade de escrita, enquanto a replicação a três nós implica maior overhead, resultando em latência acrescida e redução no *throughput*.

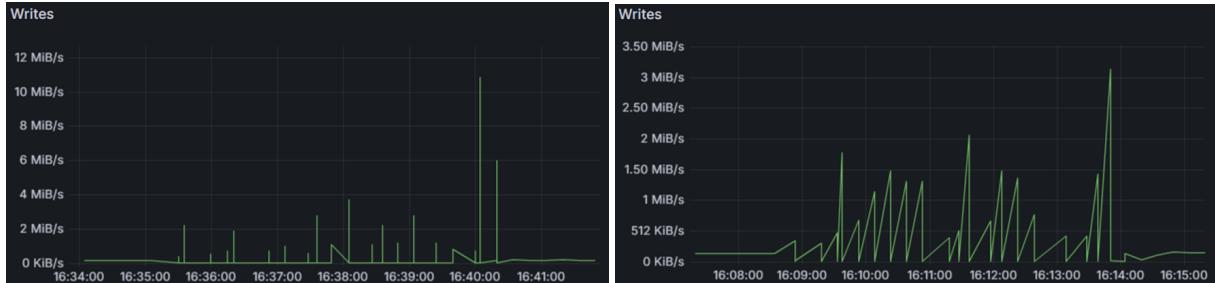


Figura 80: Escritas do Ceph com 2 réplicas (esq.) e 3 réplicas (dir.)

5.15.4 Resultado:

A Tabela 16 mostra que, com 2 réplicas, o sistema atingiu uma taxa de escrita superior (9,6MiB/s) e escreveu um volume de dados úteis significativamente maior (2,92GiB) do que com 3 réplicas (6,3MiB/s e 1,89GiB). A diferença de dados escritos ultrapassa 1GiB, evidenciando o impacto do aumento do número de réplicas no desempenho. Estes resultados confirmam que a replicação com 3 réplicas introduz um overhead adicional que reduz a eficiência da escrita.

Métrica	Replicação 2/2	Replicação 3/3
Duração do teste	305 s	305,5 s
Taxa média de escrita (bw)	9,6 MiB/s	6,3 MiB/s
Volume de dados escrito	2,92 GiB	1,89 GiB
Fator de replicação	2	3
Total escrito no <i>cluster</i>	5,84 GiB	5,67 GiB

Tabela 16: Resumo das métricas dos testes de escrita no CephFS

5.15.5 Discussão:

O sistema de ficheiros CephFS apresenta um desempenho de escrita ligeiramente superior com 2 réplicas, tanto em termos de taxa média de escrita como no volume total de dados escritos. Com 3 réplicas, verificou-se uma redução no desempenho, justificada pelo aumento do tráfego de rede, maior latência dos OSDs e o *overhead* associado à escrita em múltiplos nós. No entanto, apesar da degradação ligeira do desempenho, o sistema manteve uma taxa de escrita bastante próxima, demonstrando a robustez do CephFS mesmo com um fator de replicação superior.

5.16 Teste 16 – Leitura Sequencial no CephFS com 2 e 3 réplicas

Neste teste, são apresentados os objetivos, os procedimentos realizados, as análises feitas, os resultados obtidos e a respetiva discussão.

5.16.1 Objetivo:

Avaliar o desempenho de leitura sequencial no sistema de ficheiros CephFS sob carga prolongada, simulando operações de leitura direta com múltiplas tarefas paralelas. O teste compara as configurações com 2 réplicas e 3 réplicas, de forma a observar o impacto da replicação no desempenho.

5.16.2 Procedimento:

Foram realizados dois testes com o comando *fio*, no modo de leitura sequencial (read), com duração de 5 minutos (300 segundos) e 4 tarefas paralelas (*--numjobs=4*). Foi utilizado acesso direto á diretoria (*--direct=1*) e blocos de 4MiB (*--bs=4M*). O parâmetro *--time_based* garantiu que o teste decorresse durante o tempo especificado, independentemente do volume de dados. Para garantir a medição real do desempenho de leitura nos OSDs, a cache do sistema operacional foi previamente limpa.

Os testes foram realizados em duas pools distintas: uma com 2 réplicas (*testcephfs*) e outra com 3 réplicas (*cephfs*).

Nó de execução: proxmox2

```
# Teste com 2 réplicas
fio --name=read-testcephfs --directory=/mnt/pve/testcephfs --rw=read --bs=4M --numjobs=4
--runtimes=300 --time_based --ioengine=libaio --iodepth=4 --group_reporting --direct=1

# Teste com 3 réplicas
fio --name=read-cephfs --directory=/mnt/pve/cephfs --rw=read --bs=4M --numjobs=4 --
runtimes=300 --time_based --ioengine=libaio --iodepth=4 --group_reporting --direct=1
```

5.16.3 Análise:

Nos gráficos da Figura 81, observa-se uma utilização de CPU ligeiramente superior no teste com 3 réplicas, em comparação com o teste com 2 réplicas. Esta diferença é expectável, dado que a leitura com mais réplicas implica uma maior coordenação entre os OSDs.

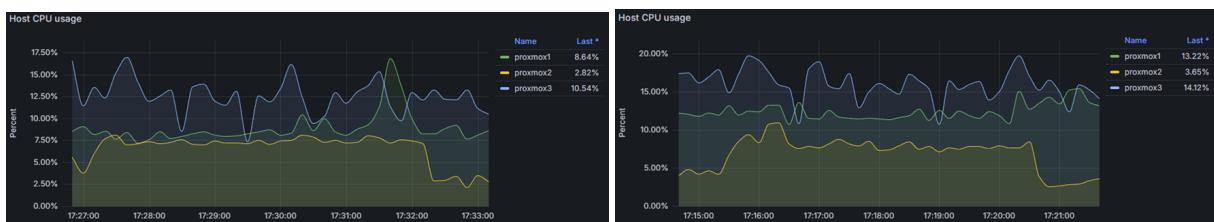


Figura 81: Utilização do CPU com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de tráfego de entrada Figura 82, observa-se que apenas o nó que executa o comando de leitura apresenta um tráfego de entrada significativamente elevado. Os restantes nós mantêm valores residuais e constantes, independentemente do número de réplicas.

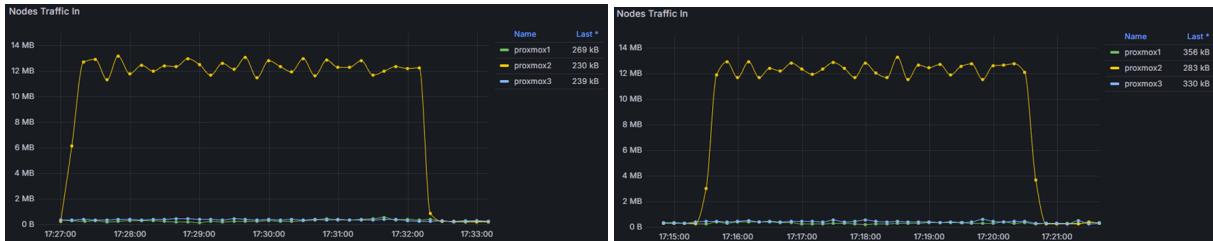


Figura 82: Tráfego de Entrada com 2 réplicas (esq.) e 3 réplicas (dir.)

Nos gráficos de tráfego de saída (Figura 83), observa-se que, no teste com 2 réplicas, os dados são obtidos de forma equilibrada entre os OSDs disponíveis, com picos regulares de atividade nos nós proxmox1 e proxmox3. No teste com 3 réplicas, embora o padrão geral de distribuição se mantenha, verifica-se um período em que o proxmox3 apresenta maior tráfego de saída, resultado de ter sido mais frequentemente selecionado como OSD primário, contendo a cópia principal dos dados.

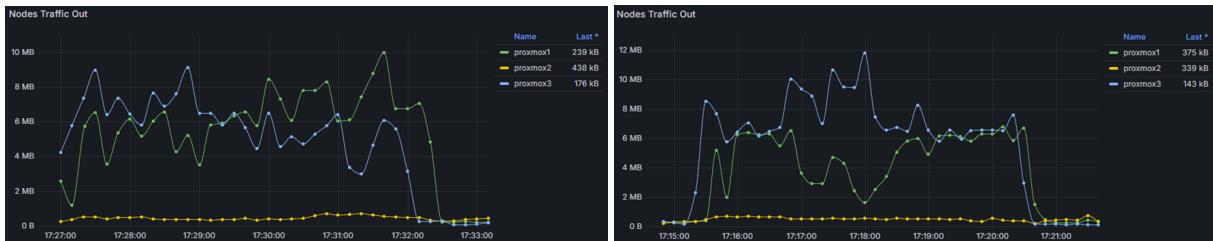


Figura 83: Tráfego de Saída com 2 réplicas (esq.) e 3 réplicas (dir.)

5.16.4 Resultado:

A Tabela 17 mostra que, com 3 réplicas, o sistema atingiu uma taxa de leitura média superior (24,5MiB/s) e leu um volume total de dados maior (7,30GiB) do que com 2 réplicas (20,4MiB/s e 5,84GiB). Esta diferença de mais de 1GiB evidencia que o aumento do número de réplicas, neste caso, não prejudicou o desempenho de leitura. Pelo contrário, poderá ter beneficiado da maior disponibilidade de réplicas para distribuir a carga entre os OSDs.

Métrica	Replicação 2/2	Replicação 3/3
Duração do teste	305,3 s	305,2 s
Volume total lido	6232 MiB \approx 5,84 GiB	7472 MiB \approx 7,30 GiB
Taxa média de leitura (Throughput)	20,4 MiB/s	24,5 MiB/s

Tabela 17: Comparação dos testes de leitura no CephFS com replicação 2/2 e 3/3

5.16.5 Discussão:

Conclui-se que a leitura sequencial no CephFS não foi negativamente afetada pelo aumento do número de réplicas. Pelo contrário, a configuração com 3 réplicas apresentou desempenho ligeiramente superior. Embora o Ceph realize as leituras preferencialmente no OSD primário (que contém os dados originais), uma maior redundância permite uma melhor distribuição da carga e reduz a sobrecarga, uma vez que os OSDs que armazenam os dados primários ficam mais distribuídos. Adicionalmente, fatores externos, como variações momentâneas na carga dos nós, latências de rede e largura de banda disponível, podem também ter influenciado as diferenças observadas. Estes resultados demonstram a robustez do Ceph, capaz de manter desempenho estável e eficiente mesmo com níveis mais elevados de replicação e resiliência.

5.17 Síntese

Este capítulo apresentou, de forma detalhada, os testes realizados sobre a infraestrutura implementada com Proxmox VE e Ceph, com o objetivo de avaliar a sua robustez, desempenho e tolerância a falhas em diferentes cenários operacionais.

Foram realizados 16 testes, organizados em duas grandes categorias: testes funcionais (casos de falha, migração e recuperação) e testes de desempenho (operações intensivas de leitura e escrita com 2 e 3 réplicas). Os resultados demonstraram que o Ceph garante elevada resiliência, permitindo a recuperação automática de falhas de discos, a integração dinâmica de novos OSDs e a manutenção de serviços mesmo em situações críticas. A migração de VMs e CTs mostrou-se significativamente

mais eficiente com armazenamento distribuído do que com armazenamento local.

Verificou-se ainda que a replicação influencia diretamente o desempenho: com 2 réplicas obtêm-se melhores resultados em termos de latência e *throughput* de escrita, enquanto 3 réplicas asseguram maior fiabilidade e distribuição de carga nas leituras. O sistema demonstrou-se equilibrado e adaptável, sendo capaz de manter um funcionamento estável sob cargas elevadas e em cenários de falha.

No capítulo seguinte, será apresentado um estudo orientado para a implementação real de uma infraestrutura de *Private Cloud*, tendo por base um orçamento previamente definido e requisitos técnicos específicos.

6 Estudo e Especificação de uma Private cloud

Este capítulo apresenta um estudo orientado para a implementação real de uma infraestrutura de *Private Cloud*, tendo por base um orçamento previamente definido. Ao longo do trabalho, foram analisados diversos cenários possíveis, com o objetivo de encontrar a solução mais adequada às necessidades específicas de quem pretende realizar a implementação.

As opções consideradas variam consoante fatores como capacidade de armazenamento, desempenho, redundância, escalabilidade e custo. Cada cenário foi avaliado de forma crítica, tendo em conta diferentes perfis de utilização e objetivos operacionais. A proposta final resultou desta análise comparativa, procurando um equilíbrio entre viabilidade técnica, eficiência económica e alinhamento com os requisitos reais de um *datacenter*.

6.1 Análise Comparativa de Opções de Implementação

Tendo em conta diferentes cenários de utilização e necessidades específicas de datacenters, foi realizado um estudo comparativo de quatro opções distintas para a implementação de uma infraestrutura de *Private Cloud*. Este estudo teve como base um orçamento limite de 50.000€, procurando identificar soluções equilibradas entre custo, desempenho, escalabilidade e fiabilidade.

As opções analisadas variam na quantidade e função dos servidores, na separação entre nós de armazenamento e nós de computação para inteligência artificial, bem como no impacto destas decisões no quorum e estabilidade do sistema distribuído Ceph. A Tabela 18 sintetiza as vantagens e desvantagens de cada proposta, permitindo uma análise crítica consoante os requisitos operacionais de quem pretende realizar a implementação.

Opção	Equipamentos	Vantagens	Desvantagens
A	3 servidores: - 1 Storage - 2 AI + Storage	- Quorum Ceph garantido - Menor consumo e espaço - Aproveitamento máximo dos servidores	- Risco elevado na instabilidade no Ceph (2 nós mistos) - AI pode afetar a performance do armazenamento - Sem separação de funções críticas
B	4 servidores: - 3 Storage - 1 AI	- Quorum Ceph com nós dedicados - Ceph mais estável - Separação total AI/Storage	- Apenas 1 servidor AI (sem redundância) - Limitado para múltiplas tarefas de AI intensivas
C	4 servidores: - 2 Storage - 1 AI + Storage - 1 AI	- Quorum Ceph garantido - Redundância de AI	- Um nó misto (risco para Ceph) - Mais difícil de gerir performance sob carga elevada
D	5 servidores: - 3 Storage - 2 AI	- Melhor estabilidade Ceph - Redundância total no AI (failover possível) - Separação total de funções	- Custo mais elevado - Maior consumo energético e gestão - Requer mais espaço físico no datacenter

Tabela 18: Comparaçao entre diferentes opções para implementação da Private Cloud

6.2 Propostas de Implementação

Com base na análise comparativa anteriormente apresentada, foram desenvolvidas quatro propostas distintas para a implementação de uma infraestrutura de *Private Cloud*, ajustadas a diferentes prioridades e requisitos operacionais.

Importa referir que o servidor representado a laranja nas arquiteturas, corresponde a um servidor dedicado a tarefas de Inteligência Artificial (AI). Este servidor inclui uma GPU NVIDIA A100 com 80 GB, uma RTX 4000 com 16 GB e 12 TB de armazenamento local. Por se tratar de *hardware* previamente adquirido, não foi contabilizado no orçamento final das propostas.

Adicionalmente, importa salientar que os servidores indicados nas propostas como servidores

de armazenamento não se encontram restritos a essa função. Sempre que necessário, poderão ser utilizados para executar serviços que não exijam GPU, promovendo assim uma utilização mais eficiente desses servidores, uma vez que apresentam especificações de *hardware* bastante superiores às recomendadas pelo Ceph. Contudo, esta utilização adicional deve ser cuidadosamente planeada, de forma a evitar a sobrecarga desses servidores e garantir que o desempenho e a estabilidade do Ceph não sejam comprometidos.

Nas subsecções seguintes, apresentamos individualmente cada proposta, detalhando a sua composição, principais vantagens e eventuais limitações. A proposta que consideramos ideal será apresentada em último lugar.

6.2.1 Opção A

A Opção A apresenta uma configuração composta por três servidores: um dedicado ao armazenamento e dois servidores híbridos com funções combinadas de AI e *Storage*, como ilustrado na Figura 84. Esta arquitetura permite garantir o quorum mínimo necessário para o funcionamento do sistema Ceph, ao mesmo tempo que maximiza a utilização dos recursos disponíveis.

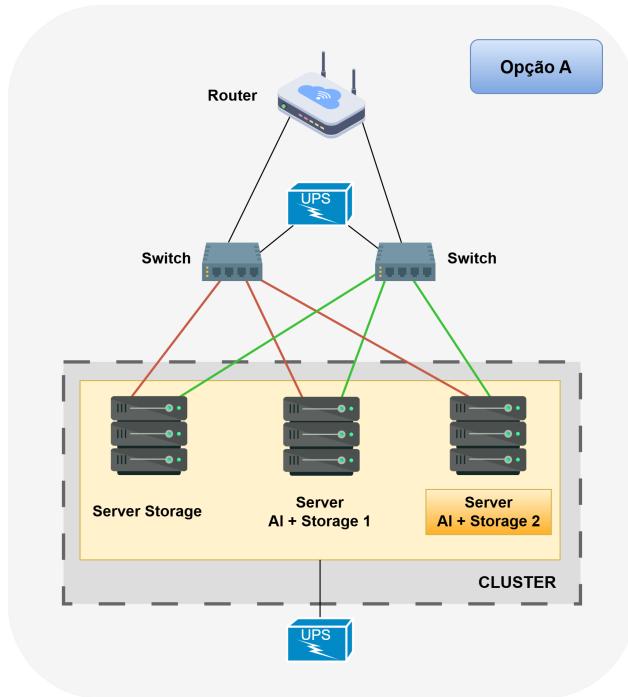


Figura 84: Arquitetura da Opção A

Entre os principais pontos fortes desta proposta destacam-se o menor consumo energético, a redução do espaço físico necessário e o aproveitamento total das capacidades computacionais dos servidores. Esta solução é especialmente atrativa em ambientes com limitações físicas.

Contudo, esta configuração implica também algumas limitações. A presença de dois nós mistos (armazenamento e computação) representa um risco acrescido para a estabilidade do Ceph, sobretudo em cenários de carga elevada. A ausência de separação entre funções críticas pode ainda comprometer o desempenho, especialmente quando são realizadas operações simultâneas intensivas de AI e de armazenamento.

Equipamento	Qtd.	Preço Unitário	Preço Total
Servidor Storage (80 TB)	1	7.820,80 €	7.820,80 €
Servidor AI + Storage (RTX 6000 48GB, 20 TB)	1	30.185,35 €	30.185,35 €
Servidor existente (A100 80 GB + RTX 4000 16 GB, 12 TB)	1	0,00 €	0,00 €

Equipamento	Qtd.	Preço Unitário	Preço Total
UPS para Storage + AI	2	1.094,00 €	2.188,00 €
Switch	2	679,00 €	1.358,00 €
UPS para Switch	1	467,99 €	467,99 €
Total			40.926,14 €

Tabela 19: Resumo de equipamentos e custos da Opção A

A Opção A foca-se no aproveitamento máximo dos recursos através de servidores mistos, reduzindo o número de equipamentos e otimizando espaço e consumo. Contudo, esta eficiência implica riscos para a estabilidade do sistema Ceph, sendo mais adequada a cenários onde a prioridade é minimizar a infraestrutura, mesmo com algum compromisso na fiabilidade.

6.2.2 Opção B

A Opção B adota uma abordagem mais conservadora e robusta, com a separação completa entre os nós de armazenamento e o nó de computação para AI, como ilustrado na Figura 85. Esta proposta inclui três servidores dedicados ao armazenamento, garantindo o quorum do sistema Ceph com elevada estabilidade e fiabilidade. Adicionalmente, um servidor separado é responsável pelas tarefas de Inteligência Artificial, evitando qualquer interferência direta entre cargas computacionais e operações de I/O.

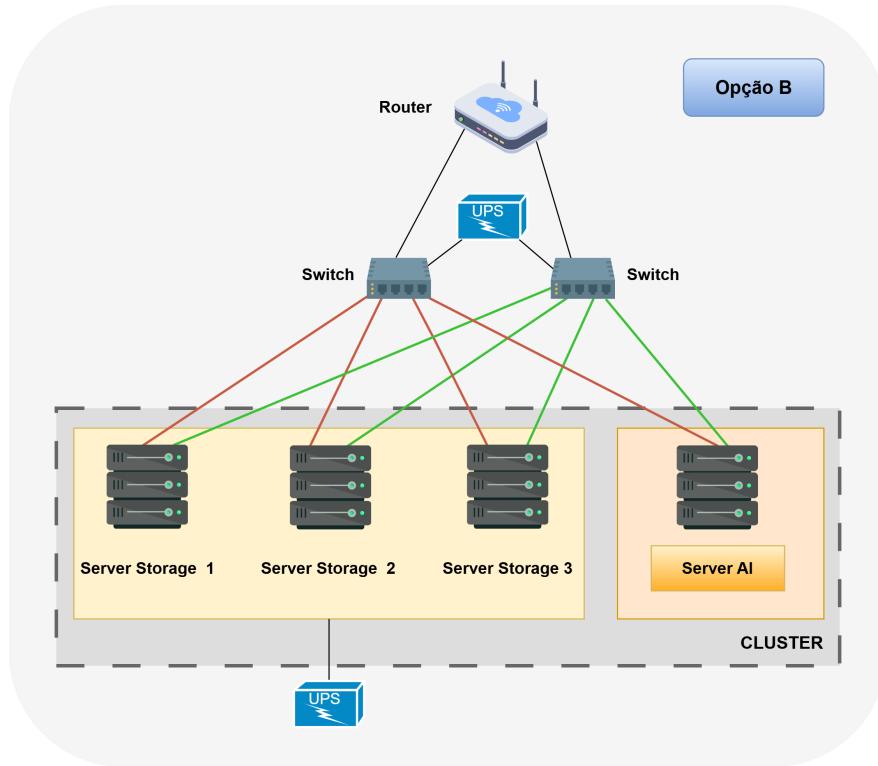


Figura 85: Arquitetura da Opção B

A principal vantagem desta arquitetura reside na separação total de funções, o que contribui para uma gestão mais simples e para uma maior previsibilidade de desempenho. O sistema Ceph opera de forma mais estável, uma vez que todos os seus nós são dedicados e não partilham recursos com processos exigentes como a computação de AI. Esta solução é especialmente atrativa em ambientes com limitações orçamentais.

Contudo, esta configuração apresenta como principal limitação o facto de contar apenas com um servidor de AI, sem qualquer redundância. Em cenários que exigem múltiplos processos de treino ou inferência simultâneos, este ponto poderá constituir um estrangulamento.

Equipamento	Qtd.	Preço Unitário	Preço Total
Servidor Storage (80 TB cada)	3	7.820,80 €	23.462,40 €
Servidor existente (A100 80 GB + RTX 4000 16 GB, 12 TB)	1	0,00 €	0,00 €
UPS para Storage	1	1.094,00 €	1.094,00 €

Equipamento	Qty.	Preço Unitário	Preço Total
Switch	2	679,00 €	1.358,00 €
UPS para Switch	1	467,99 €	467,99 €
Total			26.382,39 €

Tabela 20: Resumo de equipamentos e custos da Opção B

A proposta foi intencionalmente mantida muito abaixo do limite orçamental como apresentado na Tabela 20, permitindo margem para uma eventual evolução futura, como a adição de um segundo servidor de AI ou o aumento da capacidade de armazenamento dos servidores de *storage*. A Opção B representa, assim, uma solução estável e equilibrada, com potencial de crescimento conforme as necessidades aumentem.

6.2.3 Opção C

A Opção C combina redundância e eficiência numa configuração com quatro servidores: dois dedicados ao armazenamento, um servidor híbrido com funções de AI e Storage, e um servidor exclusivamente dedicado à AI (já existente), como ilustrado na Figura 86. Esta arquitetura garante o quorum do sistema Ceph com dois nós dedicados e um nó misto, ao mesmo tempo que assegura redundância na capacidade de processamento para tarefas de Inteligência Artificial.

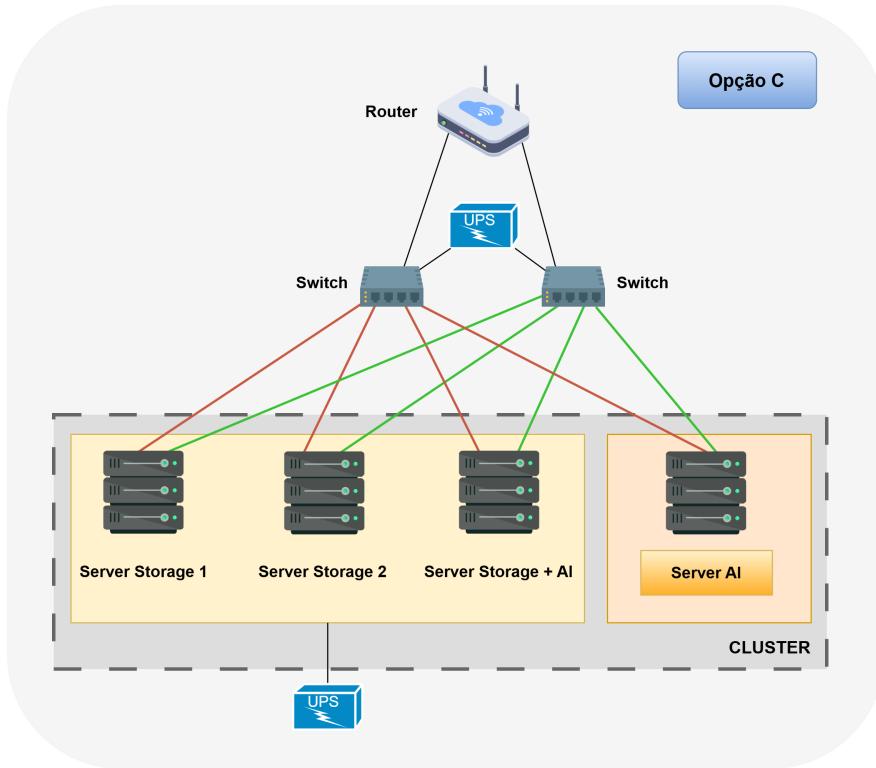


Figura 86: Arquitetura da Opção C

A grande vantagem desta proposta é a redundância ao nível da computação de AI, possibilitando o funcionamento contínuo em caso de falha de um dos servidores. Além disso, o Ceph mantém o quorum necessário, e a estrutura oferece flexibilidade para a execução simultânea de tarefas de AI e operações de armazenamento. Esta opção representa um equilíbrio entre desempenho, disponibilidade e tolerância a falhas.

Contudo, a presença de um nó misto (armazenamento + AI) pode introduzir riscos adicionais para a estabilidade do Ceph sob cargas elevadas, exigindo maior atenção à gestão dos recursos e da performance.

Para garantir a escalabilidade e estabilidade a longo prazo, é recomendável que eventuais novos nós sejam dedicados a funções específicas (armazenamento ou computação de AI), evitando a sobrecarga de recursos e facilitando a gestão do *cluster*.

Equipamento	Qty.	Preço Unitário	Preço Total
Servidor Storage (80 TB cada)	2	7.820,80 €	15.641,60 €
Servidor AI + Storage (RTX 6000 48 GB, 20 TB)	1	30.185,35 €	30.185,35 €
Servidor existente (A100 80 GB + RTX 4000 16 GB, 12 TB)	1	0,00 €	0,00 €
UPS para Storage + AI	2	1.094,00 €	2.188,00 €
Switch	2	679,00 €	1.358,00 €
UPS para Switch	1	467,99 €	467,99 €
Total			49.840,94 €

Tabela 21: Resumo de equipamentos e custos da Opção C

A Opção C aproxima-se do limite orçamental definido como apresentado na Tabela 21, apresentando uma solução robusta e com redundância ao nível da computação de AI. Apesar de incluir um nó misto, a proposta garante estabilidade para o Ceph e oferece flexibilidade para a execução simultânea de tarefas intensivas. É uma opção equilibrada para contextos que valorizam alta disponibilidade e desempenho, sem exceder o orçamento estipulado.

6.2.4 Opção D

A Opção D representa a proposta mais completa e robusta entre todas as analisadas. A arquitetura é composta por cinco servidores: três dedicados ao armazenamento e dois servidores dedicados à computação de Inteligência Artificial, garantindo redundância total tanto no *cluster* Ceph como na camada de AI, como ilustrado na Figura 87.

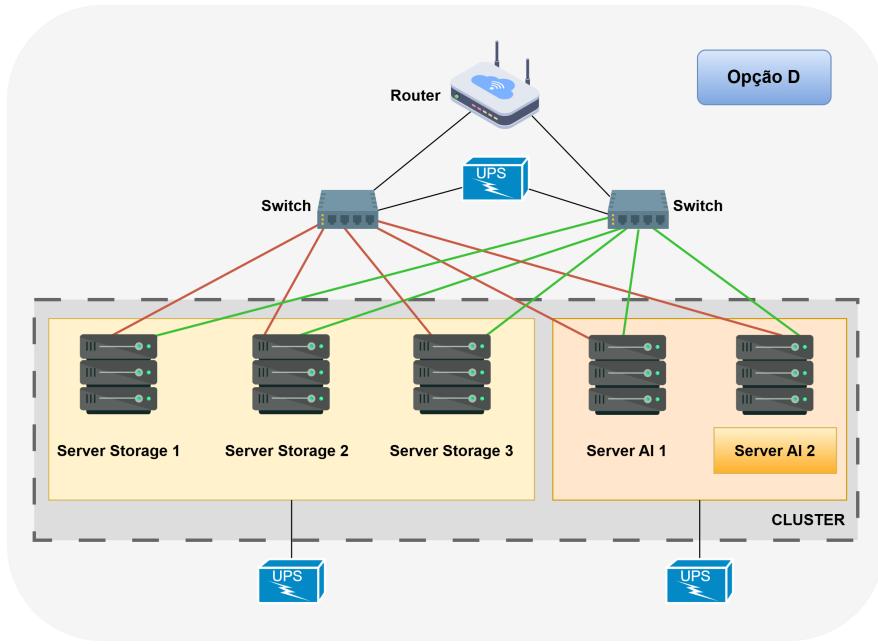


Figura 87: Arquitetura da Opção D

Esta configuração oferece a melhor estabilidade para o Ceph, ao utilizar apenas nós dedicados, e garante tolerância a falhas no processamento de AI, com dois servidores independentes. A separação total entre funções e a capacidade de executar múltiplos *jobs* de forma paralela tornam esta opção adequada para ambientes com cargas intensivas e requisitos de alta disponibilidade.

Contudo, estes benefícios vêm acompanhados de um custo mais elevado, um maior consumo energético e a necessidade de mais espaço físico no datacenter. Esta opção ultrapassa ligeiramente o orçamento de referência, mas apresenta vantagens significativas em fiabilidade e escalabilidade, justificando o investimento adicional.

Equipamento	Qtd.	Preço Unitário	Preço Total
Servidor Storage (80 TB cada)	3	7.820,80 €	23.462,40 €
Servidor AI (RTX 6000 48 GB)	1	23.368,90 €	23.368,90 €
Servidor existente (A100 80 GB + RTX 4000 16 GB, 12 TB)	1	0,00 €	0,00 €
UPS para Storage + AI	2	1.094,00 €	2.188,00 €
Switch	2	679,00 €	1.358,00 €

Equipamento	Qty.	Preço Unitário	Preço Total
UPS para Switch	1	467,99 €	467,99 €
Total			50.845,29 €

Tabela 22: Resumo de equipamentos e custos da Opção D

Apesar de ultrapassar ligeiramente o orçamento definido como apresentado na Tabela 22, a Opção D foi considerada a mais adequada para uma implementação robusta, fiável e escalável. A redundância total e a separação de funções tornam esta solução ideal para contextos exigentes, com foco na continuidade de serviço e crescimento futuro.

6.3 Análise Comparativa Final e Justificação da Escolha

Após a apresentação e análise detalhada das quatro opções de implementação, procede-se agora à sua comparação direta. A Tabela 23 resume os principais parâmetros técnicos e económicos de cada proposta.

Opção	Tipo de Servidores	Armazenamento Total	Armazenamento Ceph útil (3 réplicas)	Capacidade de AI (nova + existente)	Preço Total
A	1 Storage + 2 AI+Storage	112 TB	37 TB	1 nova + 1 existente	40.926,14€
B	3 Storage + 1 AI	240 TB	80 TB	1 existente	26.382,39€
C	2 Storage + 1 AI+Storage + 1 AI	180 TB	60 TB	1 nova + 1 existente	49.840,94€
D	3 Storage + 2 AI	240 TB	80 TB	1 nova + 1 existente	50.845,29€

Tabela 23: Comparação final entre as opções de implementação

Com base na comparação efetuada, a Opção D foi considerada a mais adequada para uma implementação real e de longo prazo. Embora ultrapasse ligeiramente o orçamento definido inicialmente, é a única que assegura redundância total em todos os níveis (armazenamento e computação) sem comprometer a separação de funções nem a estabilidade do sistema. A distribuição clara entre os servidores de armazenamento e os de AI, aliada à capacidade de executar múltiplas tarefas intensivas de forma paralela, torna esta proposta a mais robusta, escalável e preparada para crescimento futuro.

As especificações técnicas detalhadas de todos os equipamentos utilizados nas diferentes opções de implementação encontram-se disponíveis no Apêndice A, para consulta complementar.

6.4 Síntese

Este capítulo apresentou um estudo detalhado e orientado à implementação prática de uma infraestrutura de *Private Cloud*, com base num orçamento limite de 50.000€. Foram analisadas quatro opções distintas, cada uma com diferentes níveis de desempenho, escalabilidade, fiabilidade e custo. As propostas variaram na separação entre funções de armazenamento e computação para AI, número de servidores e redundância operacional.

Através de uma análise comparativa técnica e económica, foi possível identificar as vantagens e limitações de cada configuração. A Opção D destacou-se como a mais robusta, assegurando separação total de funções, redundância completa e elevada escalabilidade, sendo considerada a solução mais adequada para uma implementação real, mesmo com um ligeiro ultrapassar do orçamento definido.

No capítulo seguinte, será apresentada a conclusão do relatório, sintetizando os principais resultados obtidos ao longo do projeto e refletindo sobre a sua relevância prática e contributo para a adoção de soluções de *Private Cloud*.

7 Conclusão

Neste projeto pretendeu-se avaliar a viabilidade da criação de uma infraestrutura de *Private Cloud* com recurso a tecnologias, nomeadamente o Proxmox VE e o Ceph, com o objetivo de disponibilizar serviços a um conjunto restrito de utilizadores, garantindo maior controlo sobre os dados, segurança e eficiência no acesso aos recursos.

Ao longo das várias fases do projeto, foi possível demonstrar que os objetivos do projeto foram plenamente alcançados. Estes objetivos passaram pela conceção, implementação e avaliação de uma infraestrutura de *Private Cloud* capaz de disponibilizar serviços a um conjunto restrito de utilizadores, assegurando uma gestão centralizada, segura e eficiente da infraestrutura. A solução desenvolvida revelou-se funcional, robusta e adaptável a diferentes contextos, validando-se a sua viabilidade técnica e operacional através dos testes de desempenho, escalabilidade e tolerância a falhas.

O projeto permitiu consolidar conhecimentos na área da virtualização, do armazenamento distribuído e da gestão de infraestruturas, através da aplicação prática de ferramentas como o Proxmox VE, o Ceph e sistemas de monitorização. A integração eficaz destes componentes confirmou o potencial das tecnologias *open-source* na construção de soluções de computação em nuvem privadas, eficientes e sem custos de software.

Os testes realizados permitiram validar a viabilidade técnica e operacional da solução, evidenciando resultados relevantes em várias vertentes. O sistema demonstrou tolerância a falhas, assegurando a continuidade dos serviços mesmo perante a falha de nós ou discos, graças à replicação automática dos dados proporcionada pelo Ceph. A migração de VM revelou-se eficaz e rápida, confirmando a capacidade de resposta da infraestrutura em cenários de alta disponibilidade. Verificou-se também que a escalabilidade foi garantida, sendo possível adicionar dinamicamente nós e discos sem causar interrupções nos serviços. Por fim, o desempenho manteve-se estável, mesmo em contextos com diferentes fatores de replicação, comprovando a robustez e eficiência da arquitetura implementada.

As soluções apresentadas ao longo do projeto poderão revelar-se particularmente relevantes para organizações de pequena e média dimensão que procurem maior autonomia, segurança e controlo sobre a sua infraestrutura de IT, constituindo uma alternativa viável face a soluções comerciais.

Este projeto contribui para demonstrar a viabilidade prática da implementação de uma infraestrutura de *Private Cloud* com recursos limitados, apresentando-se como uma alternativa mais controlada, segura e personalizável face às soluções de *cloud* pública. A sua avaliação em contexto real reforça o conhecimento aplicado na área da computação em nuvem e apoia organizações que procurem uma maior autonomia na gestão dos serviços e dados.

Como trabalho futuro, considera-se relevante o desenvolvimento de uma interface dedicada à gestão de utilizadores, que permita simplificar tarefas como a criação de máquinas virtuais e a sua associação a utilizadores específicos, promovendo uma utilização mais intuitiva e segura da infraestrutura. Adicionalmente, propõe-se a exploração de soluções que colmatem algumas limitações do Proxmox VE, nomeadamente no que diz respeito à tolerância a falhas (*fault tolerance*), de forma a aumentar a resiliência e a disponibilidade do sistema.

Bibliografia

- PROXMOX VE ADMINISTRATION GUIDE, 2025. URL: <https://pve.proxmox.com/pve-docs/pve-admin-guide.pdf>
- Ceph. Welcome to ceph, 2016. URL: <https://docs.ceph.com/en/latest/>
- Ceph Benchmark, 2018. URL: https://support.alcadis.nl/Support_files/Proxmox/Proxmox%20VE//Benchmark/Proxmox-VE_Ceph-Benchmark-201802.pdf
- Supermicro, BMC IPMI, 2020. URL: https://www.supermicro.com/manuals/other/IPMI_Users_Guide.pdf
- Performance Analysis of Storage Media Cluster Using Ceph Platform, 2021. URL: <https://ieeexplore.ieee.org/document/9624251>
- File system performance comparison between Linux-based hypervisors ESXi and Proxmox, 2025. URL: <https://ieeexplore.ieee.org/document/10819089>
- ChatGPT, 2025. URL: <https://chatgpt.com/>
- Logical Volume Manager (LVM), 2023. URL: [https://pve.proxmox.com/wiki/Logical_Volume_Manager_\(LVM\)](https://pve.proxmox.com/wiki/Logical_Volume_Manager_(LVM))

Referências

- [1] Ceph. Welcome to ceph, 2016. URL: <https://docs.ceph.com/en/latest/>.
- [2] IBM. What is a data center?, 2025. URL: <https://www.ibm.com/think/topics/data-centers>.
- [3] IBM. What is virtualization?, 2025. URL: <https://www.ibm.com/think/topics/virtualization>.
- [4] VMware. What is a hypervisor?, 2025. URL: <https://www.vmware.com/topics/hypervisor>.
- [5] RedHat. What is a hypervisor?, 2023. URL: <https://www.redhat.com/en/topics/virtualization/what-is-a-hypervisor>.

- [6] VMware. What is a virtual machine?, 2025. URL: <https://www.vmware.com/topics/virtual-machine>.
- [7] IBM. What are containers?, 2025. URL: <https://www.ibm.com/think/topics/containers>.
- [8] IBM. Introduction: Clusters, 2024. URL: <https://www.ibm.com/docs/en/was-nd/9.0.5?topic=servers-introduction-clusters>.
- [9] NIST. The nist definition of cloud computing, 2011. URL: <https://nvlpubs.nist.gov/nistpubs/legacy/sp/nistspecialpublication800-145.pdf>.
- [10] IBM. Object vs. file vs. block storage: What's the difference?, 2021. URL: <https://www.ibm.com/think/topics/object-vs-file-vs-block-storage>.
- [11] TechTarget. XaaS (anything as a service), 2024. URL: <https://www.techtarget.com/searchcloudcomputing/definition/XaaS-anything-as-a-service>.
- [12] Sios. High availability, 2024. URL: <https://us.sios.com/resource/high-availability/>.
- [13] Barracuda. Failover, 2024. URL: <https://www.barracuda.com/support/glossary/failover>.
- [14] VMware. Fault tolerance definition, 2025. URL: <https://www.vmware.com/topics/fault-tolerance>.
- [15] IBM. High availability versus fault tolerance, 2025. URL: <https://www.ibm.com/docs/en/powerha-aix/7.2.x?topic=aix-high-availability-versus-fault-tolerance>.
- [16] IBM. What is backup and restore?, 2025. URL: <https://www.ibm.com/think/topics/backup-and-restore>.
- [17] Pair. Snapshots and backups: What is the difference?, 2023. URL: <https://www.pair.com/support/kb/snapshots-and-backups-what-is-the-difference/>.
- [18] IBM. What is data migration?, 2024. URL: <https://www.ibm.com/think/topics/data-migration>.

- [19] Stonefly. Understanding thin provisioning and how it works. URL: <https://stonefly.com/blog/understanding-thin-provisioning-and-its-working/>.
- [20] VMware. vsphere, 2025. URL: <https://www.vmware.com/products/cloud-infrastructure/vsphere>.
- [21] StorMagic. All about hypervisors: Esxi vs hyper-v, xenserver, proxmox, kvm, and ahv, 2024. URL: <https://stormagic.com/company/blog/all-about-hypervisors-esxi-vs-hyper-v-xenserver-proxmox-kvm-ahv/>.
- [22] Proxmox. Proxmox virtual environment, 2025. URL: <https://www.proxmox.com/en/products/proxmox-virtual-environment/overview>.
- [23] Microsoft. Hyperv technology overview, 2025. URL: <https://learn.microsoft.com/en-us/windows-server/virtualization/hyper-v/hyper-v-overview?pivots=windows>.
- [24] XenServer. More power and increased efficiency with xenserver 8.4, 2024. URL: <https://www.xenserver.com/>.
- [25] KVM. Kernel virtual machine, 2025. URL: https://linux-kvm.org/page/Main_Page.
- [26] man7. lvm(8), 2014. URL: <https://www.man7.org/linux/man-pages/man8/lvm.8.html>.
- [27] Wiki.archlinux. Lvm, 2024. URL: <https://wiki.archlinux.org/title/LVM>.
- [28] man7. lvmthin(7), 2014. URL: <https://man7.org/linux/man-pages/man7/lvmthin.7.html>.
- [29] Proxmox. Storage: Lvm thin, 2025. URL: https://pve.proxmox.com/wiki/Storage%3A_LVM_Thin.
- [30] Docs.kernel. Thin provisioning, 2024. URL: <https://docs.kernel.org/admin-guide/device-mapper/thin-provisioning.html>.
- [31] Pve Proxmox. Directory backend, 2025. URL: https://pve.proxmox.com/pve-docs/pve-admin-guide.html#storage_directory.

- [32] Proxmox. Zfs on linux, 2024. URL: https://pve.proxmox.com/wiki/ZFS_on_Linux.
- [33] TechTarget. iSCSI (internet small computer system interface), 2024. URL: <https://www.techtarget.com/searchstorage/definition/iSCSI>.
- [34] IBM. Network file system, 2024. URL: <https://www.ibm.com/docs/en/aix/7.2.0?topic=management-network-file-system>.
- [35] PvE Proxmox. Nfs backend, 2025. URL: https://pve.proxmox.com/pve-docs/pve-admin-guide.html#storage_nfs.
- [36] Anthony D'Atri, Vaibhav Bhembre, and Karan Singh. Learning ceph, 2017. URL: <https://books.google.pt/books?id=xRhKDwAAQBAJ>.
- [37] Centro Paula Souza. Ceph como object storage, 2020. URL: <https://ric.cps.sp.gov.br/bitstream/123456789/7050/1/GOMES%20-%20CEPH%20como%20object.pdf>.
- [38] Dong-Yun Lee, Kisik Jeong, Sang-Hoon Han, Jin-Soo Kim, Joo-Young Hwang, and Sangyeun Cho. Understanding write behaviors of storage backends in ceph object store, 2016. URL: <http://csl.skku.edu/papers/msst17.pdf>.
- [39] V. P. Oleksiuk and O. R. Oleksiuk. The practice of developing the academic cloud using the proxmox ve platform. *Educational Technology Quarterly*, 2021(4):605–616, 2021. URL: <https://acnsci.org/journal/index.php/etq/article/view/36>.
- [40] L. Zhao, G. Hu, and Y. Xu. Educational resource private cloud platform based on openstack. *Computers*, 13(9):241, 2024. URL: <https://www.mdpi.com/2073-431X/13/9/241>.
- [41] Harsh Oswal, Shivani Bhalsakle, and Minal Swami. Implementation - surplus resources to private cloud. In *2022 International Conference on Emerging Smart Computing and Informatics (ESCI)*, pages 1–5, 2022. URL: <https://ieeexplore.ieee.org/document/9758209>.
- [42] Henning Titi Ciptaningtyas, Ridho Rahman Hariadi, Muchammad Husni, Khakim Ghozali, Rizka Wakhidatus Sholikah, and I Made Dindra Setyadharma. Openstack implementation

using multinode deployment method for private cloud computing infrastructure. In *2023 International Seminar on Intelligent Technology and Its Applications (ISITIA)*, pages 12–17, 2023. [doi:10.1109/ISITIA59021.2023.10221042](https://doi.org/10.1109/ISITIA59021.2023.10221042).

- [43] JFE Steel Corporation. J-oscloud: Implementation of a private cloud infrastructure using openstack kilo. Technical report, JFE Steel Corporation, 2023. URL: <https://www.jfe-steel.co.jp/en/research/report/030/pdf/030-03.pdf>.
- [44] Proxmox. Proxmox ve administration guide, 2025. URL: <https://pve.proxmox.com/pve-docs/pve-admin-guide.html>.

Apêndice A

Componente	Especificações	Quantidade
CPU	Intel® Xeon® Gold 6526Y Processor 16-Core 2.80GHz 37.5MB Cache (195W)	2
Memória	64GB DDR5 5600MHz ECC RDIMM Server Memory (2Rx4) (total 128GB)	2
Disco Rígido (HDD)	16TB 3.5" HC550 7200 RPM SATA3 6Gb/s 256MB Cache 512e/4KN Hard Drive	5
Armazenamento Total (HDD)	16TB x 5 = 80TB	_____
SSD	960GB 2.5" 5400 PRO SATA 6Gb/s 3D TLC Solid State Drive (1.5W DPW)	1
Placa de Rede (Network)	Supermicro 10GbE 57416 (2x RJ45) Ethernet Network Adapter	1
Garantia	3 Anos de Garantia + 2 Anos de Garantia de Troca (Cross Shipment)	1
TPM (Trusted Platform Module)	AOM-TPM-9672V-O - Trusted Platform Module (TPM 2.0)	1
Kit de Trilho (Rail Kit)	MCP-2900-00353-00 - Supermicro 2U 3U Rail Kit (Incluído)	1
Portas On-board	2x 1GbE RJ45 LAN Ports	1
Preço Unitário	€7.820,80 (excl. IVA)	
Preço Total	€23.462,4 (excl. IVA)	

Componente	Especificações	Quantidade
CPU	AMD EPYC™ Genoa 9124 Processor 16-Core 3.00GHz 64MB Cache (200W)	2
Memória	64GB DDR5 5600MHz ECC RDIMM Server Memory (2Rx4) (total 256GB)	4
NVMe	1.9TB U.3 7mm 7450 PRO NVMe PCIe 4.0 3D TLC Solid State Drive (1.5W DWPD) (total 3.8TB)	2
SSD	960GB 2.5" 5400 PRO SATA 6Gb/s 3D TLC Solid State Drive (1.5W DWPD)	1
GPU	NVIDIA® Quadro RTX 6000 ADA 48GB GDDR6 PCIe 4.0 Graphics Card (300W)	2
Garantia	3 Anos de Garantia + 2 Anos de Garantia de Troca (Cross Shipment)	1
TPM (Trusted Platform Module)	AOM-TPM-9670V-O - Trusted Platform Module (TPM 2.0)	1
Kit de Trilho (Rail Kit)	MCP-290-00057-00 - Supermicro 4U Rail Kit (Incluído)	1
Portas On-board	2x 10GbE RJ45 LAN Ports	1
Preço Unitário	€23.368,90 (excl. IVA)	
Preço Total	€23.368,90 (excl. IVA)	