Proposta de um Cluster para Aplicações de HPC com o uso de Docker e Infiniband

Weslley da C. Silva [1], **Guilherme A. Geronimo**[1] **e Odorico M. Mendizabal**[2]

[1]Superintendência de Governança Eletrônica e T.I. e Comunicação – SeTIC

[2]Departamento de Informática e Estatística
Universidade Federal de Santa Catarina – UFSC
Florianópolis – SC – Brasil

{weslley.silva,guilherme.geronimo,odorico.mendizabal}@ufsc.br

Resumo. *O custos elevados para a aquisição e manutenção de infraestrutura para HPC tem sido alvo de preocupação por parte dos pesquisadores, de forma a buscarem alternativas como a de clusters compartilhados entre grupos de pesquisa, utilização de virtualização e o uso de contêineres. Desta forma, diante da criação de novos desafios enfrentados pelas equipes, este trabalho apresenta uma proposta de um cluster HPC com o uso de contêineres Docker.*

1. Introdução

A *High Performance Computing* (HPC) é uma área crucial para aplicações científicas e de engenharia, em que o poder computacional é fundamental para o sucesso dos projetos. Embora existam serviços de HPC em nuvem disponíveis para o público em geral, muitos grupos de pesquisa acabam por preferir investir em infraestrutura própria para garantir uma maior privacidade e controle sobre os recursos computacionais disponíveis.

No entanto, a aquisição e manutenção de uma infraestrutura própria pode ser um grande desafio, tanto do lado financeiro quanto do lado de recursos humanos para gerenciar e manter toda a estrutura necessária. Alternativamente, *clusters* compartilhados entre grupos de pesquisas podem reduzir o alto custo de aquisição de hardware e a mão de obra para o gerenciamento dos sistemas, porém cria novos problemas dado o crescimento do seu uso como: compartilhamento de recursos, complexidade dos ambientes, problemas de usabilidade, indisponibilidade de recursos, conflitos de dependências e aplicações com diferentes requisitos.

Com a modernização de tecnologias como a de contêineres de software, tornou-se possível flexibilizar a execução de diversas aplicações em único nó, cada uma em um contêiner isolado. Como consequência dessa modernização, surgiram estudos como [Yu and Huang 2015], [Ermakov and Vasyukov 2017] e [Zhou et al. 2021] que tratam da utilização de contêineres em ambientes de HPC tornando-se possível manter e gerenciar diversos ambientes em uma mesma infraestrutura com isolamento de recursos e sem os conflitos de dependências que marcam os ambientes compartilhados.

Devidos os estudos citados anteriormente, foi percebido que é possível sanar diversas demandas de HPC da universidade através de uma solução de *cluster* compartilhado entre os grupos de pesquisa com a utilização de contêineres. Portanto esse trabalho propõem a construção de um *cluster* HPC por meio de contêineres *Docker* [Docker b], interconexão com *Infiniband* [Buyya et al. 2002] e o software SLURM [Yoo et al. 2003] para gerenciamento de filas de processamento.

2. Abordagem Adotada

Tendo sido feito o levantamento dos requisitos funcionais e não-funcionais da solução, foi idealizado um serviço simples e funcional. A Figura 1 apresenta a interação entre usuário e serviço, conforme os passos descritos a seguir.

A Usuário entra com os dados básicos para submissão do *job*, como nome da imagem do contêiner e comando a ser executado;
B Orquestrador/ adiciona o *job* na fila do Gerenciador de Filas;
C Usuário é notificado da entrada (ou falha) da submissão;
D Mudanças de status do *job* são registradas pelo Gerenciador de Filas (e.g. evolução na fila, ínicio, fim, falha, cancelamento etc);
E Orquestrador notifica o usuário sobre a evolução do seu *job*.

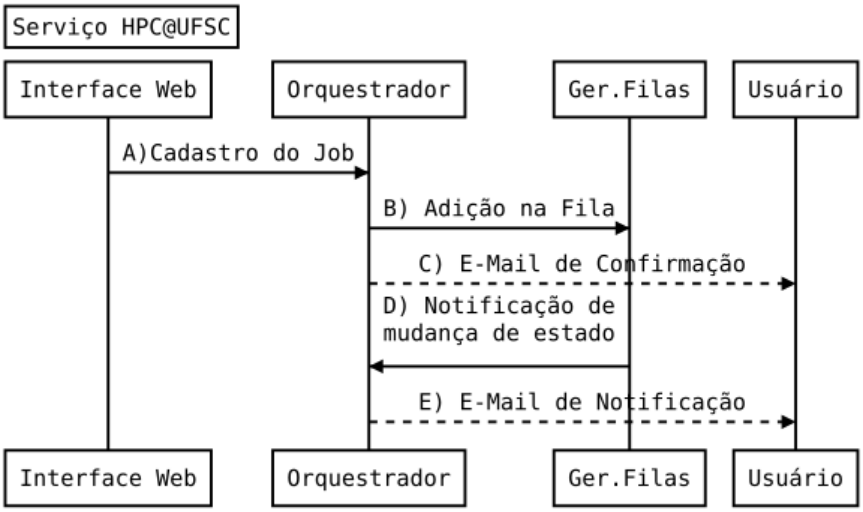

Figura 1. Diagrama de interação.

Para a redução de complexidade do projeto foi escolhido o orquestrador Docker Swarm [Docker a], já que o mesmo é uma ferramenta nativa da plataforma Docker para gerenciamento de *clusters* de contêineres em larga escala.

Tabela 1. Hardware do Cluster OCN

Modelo	2x SGI C2108-G9	SGI H2106-G7	SGI C2112-4G10 4-bay
CPU	2x AMD Opteron 6344	2x AMD Opteron 6272	2xAMD Opteron 6376
Memória	64GB	270GB	64GB
Rede	2x1GbE	2x1GbE	2x1GbE
Infiniband	Mellanox MT25408A0	Mellanox MT25408A0	Mellanox MT25408A0

Como hardware utilizado inicialmente no projeto, o *cluster* OCN compreende 4 servidores apresentados na Tabela 1, sendo que o modelo SGI C2112-4G10 tem 4 baias, ou seja, o mesmo comporta 4 nós com a configura descrita na Tabela 1. Ao todo soma-se 7 nodos, 248 cores de processamento e aproximadamente 675 GB de memória *RAM*. Para a interconexão entre os nós e *uplink*, é utilizado um *switch* Edgecore ECS4610-26T de 26

portas e para a interconexão entre os nodos do *cluster* através de *Infiniband* um Mellanox IS-5030.

Na camada de softwares, está sendo utilizando como S.O. o Ubuntu 20.04, o *docker engine* na versão v20.10.20 e o Portainer [Portainer] para gerenciamento das stacks.

3. Implementação

Inicialmente para manter simples a submissão de *jobs* para os usuários, foi decidido pela utilização de um *Google Forms* para o envio das tarefas a serem executadas no *cluster*. Os dados são processados por um *cluster Docker Swarm* que contém ao menos um *manager* e N *workers*.

No *cluster* há 4 tipos de contêineres: (i) uma instância do gerenciador do fluxo de trabalho (*workflow*/orquestrador), (ii) uma instância do *SlurmDBD*, (iii) uma instância do *SlurmCTLD* e (iv) N instâncias do *SlurmD*. De forma a simplificar a solução, foram concentrados em uma única imagem os serviços do SLURM (*SlurmDBD, SlurmCTLD e SlurmD*).

Para a orquestração dos serviços da solução, foi criado um arquivo *docker-compose.yml* que descreve a estrutura é o mesmo pode ser encontrado em: [SeTIC b] e [SeTIC a]. Também de forma a prover uma visualização da fila de processamento, foi desenvolvido utilizando a ferramenta Metabase [Metabase] um *dashboard* [HPC@UFSC] (Figura 2) com os dados do Banco do SLURM.

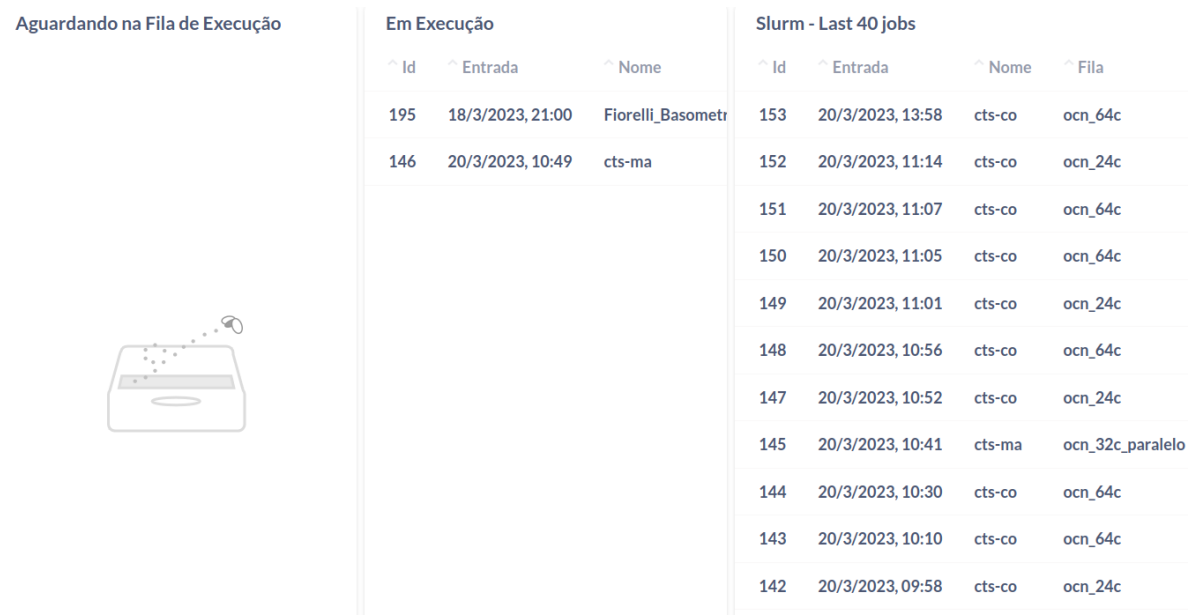

Figura 2. Dashboard para a visualização de Jobs

4. Considerações Finais e Próximos passos

Atendendo a comunidade da UFSC com aproximadamente 40 nós, 7TB de *RAM*, 1200 núcleos de processamento e 40.000 núcleos de processamento gráfico, os *clusters* para HPC representam uma boa carga de trabalho e também hardware, demonstrando o quão importante é a atualização e manutenção dessa infraestrutura existente. Com a

implantação de um novo serviço de *cluster* HPC mais flexível e melhor gerenciado, já houve crescimento no uso por parte dos grupos de pesquisa. No período compreendido de setembro de 2022 à fevereiro de 2023 (6 meses) o autosserviço desenvolvido atendeu a pelo menos 6 grupos de pequisa, provendo mais de 23 softwares distintos, totalizando mais de 1400 horas de processamento e mais de 420 *jobs* submetidos.

Os próximos passos vislumbrados são: (i) Implementação de uma solução para softwares que necessitem de processamento distribuído (e.g. OpenMPI), (ii) funcionalidade de pré-requisitos entre os *jobs* submetidos (e.g. fluxo de trabalho), (iii) evolução da interface web disponibilizada ao usuários, inclusive possibilitar a utilização de soluções como o Galaxy [Galaxy], (iv) melhorias na configuração e utilização da rede *infiniband* como a melhora na utilização de banda entre os contêineres nos *clusters*.

Agradecimentos

Agradecemos à Superintendência de Governança Eletrônica e Tecnologia da Informação e Comunicação (SeTIC) – UFSC, por proporcionar a infraestrutura necessária para que esse projeto pudesse ser realizado, e aos laboratórios que diariamente utilizam e testam o ambiente montado.

Referências

Buyya, R., Cortes, T., and Jin, H. (2002). *An Introduction to the InfiniBand Architecture*, pages 616–632.

Docker. Swarm mode overview. https://docs.docker.com/engine/swarm/. Acessado: 16-03-2023.

Docker. What docker? https://www.docker.com/what-docker/. Acessado: 16-03-2023.

Ermakov, A. and Vasyukov, A. (2017). Testing docker performance for HPC applications. *CoRR*, abs/1704.05592.

Galaxy. Projeto. https://usegalaxy.org. Acessado: 16-03-2023.

HPC@UFSC. Dashboard de filas. https://dashboards.setic.ufsc.br/public/dashboard/2eb0b36c-40f2-4b88-a031-19cdd62cc3f2. Acessado: 16-03-2023.

Metabase. Projeto. https://www.metabase.com/. Acessado: 16-03-2023.

Portainer. Container management. https://portainer.io. Acessado: 16-03-2023.

SeTIC, D. Imagem slurm. https://codigos.ufsc.br/setic-hpc/slurm. Acessado: 16-03-2023.

SeTIC, D. Slurm workflow. https://codigos.ufsc.br/setic-hpc/workflow. Acessado: 16-03-2023.

Yoo, A. B., Jette, M. A., and Grondona, M. (2003). Slurm: Simple linux utility for resource management. In Feitelson, D., Rudolph, L., and Schwiegelshohn, U., editors, *Job Scheduling Strategies for Parallel Processing*, pages 44–60, Berlin, Heidelberg. Springer Berlin Heidelberg.

Yu, H. and Huang, W. (2015). Building a virtual HPC cluster with auto scaling by the docker. *CoRR*, abs/1509.08231.

Zhou, N., Georgiou, Y., Pospieszny, M., Zhong, L., Zhou, H., Niethammer, C., Pejak, B., Marko, O., and Hoppe, D. (2021). Container orchestration on hpc systems through kubernetes. *Journal of Cloud Computing*, 10(1):16.