

> HAL3: 8x cores, 32GB ram, 4x portas Ethernet 1GBASE-T, 4x portas Ethernet 10GBASE-T, 2x HDD 2TB;

Reiterando alguns dos pontos da discussão passada:

> HAL3 foi adquirido com a pretensão de ser utilizado como gateway.

Portanto deveria ficar de fora do cluster. Deveria ter um sistema de arquivos separados, etc.

-----  
Servidores de computação:

> HAL2: 80x cores, 512GB ram, 2x portas Ethernet 1GBASE-T, 2x portas SFP+ 10GBASE-T, 2x SSD 240GB, 6x HDD 8TB;

> HAL4: 80x cores, 512GB ram, 4x portas Ethernet 1GBASE-T, 2x SSD 240GB, 6x HDD 8TB, 1 GPU Nvidia V100;

As máquinas HAL2 e HAL4 deveriam ser configuradas para uso interno apenas, isto é, computação pesada pelos pesquisadores do Neuromat. Não precisam ter servidor HTTP rodando. Usuários dentro e fora do NUMEC deverão acessar via ssh, rsync, etc.

Essas duas máquinas não precisam ser integradas nem espelhadas, nem entre si nem com o servidor HTTP.

Os usuários na rede local do NUMEC tem homedir num NFS, ou só nas suas máquinas pessoais? Se tem NFS, quem é a servidora de arquivos? Uma dessas 4, ou outra?

-----  
Servidor HTTP:

> HAL1: 72x cores, 512GB ram, 2x portas Ethernet 1GBASE-T, 2x portas Ethernet 10GBASE-T, 2x SSD 240GB, 6x HDD 8TB;

A máquina HAL1 sozinha é mais que suficiente para rodar todos os serviços públicos do Neuromat. O Neuromat não é uma Google ou Amazon, nem mesmo uma Walmart, nem mesmo um IMEUSP. O acesso vai ser esporádico e não envolve computação pesada.

Esses 6x8 = 48 TB são mais que suficientes para guardar nossas bases de dados. Os dados mais volumosos (como encefalogramas) são limitados pelo custo de captura. Mesmo se quisermos guardar vídeos, 48 TB são 1000 horas com baixa compressão. Se precisarmos de mais disco, podemos remanejar fisicamente da HAL2 e HAL4

Não vejo necessidade de integrar os sistemas de arquivos dos quatro servidores. As bases de dados NES na máquina HAL1 devem ter backup e mirror em máquina separada, mas quanto mais isolada essa cópia melhor. (Mirroring/RAID protege contra falhas de hardware, mas não contra falhas de software ou fleshware. Para isso é necessário ter backups.)

> Utilizamos e pretendemos continuar a utilizar, nos nossos servidores uma arquitetura de micros serviços baseada em containers (LXC)

Containers tem suas utilidades; mas não reduzem a complexidade do sistema. Não deveriam ser usados sem necessidade real.

> construída em cima de um sistema de arquivos (ZFS) que garante a corretude, rapidez e robustez no armazenamento e gerenciamento dos dados

OK, acho...

> todos nossos servidores web são individualizados

Você ainda não me convenceu disso. Um servidor HTTP é feito para disponibilizar vários serviços diferentes, escritos em linguagens diferentes. Na reunião você disse que "#! /usr/bin/python3" pode ser versões diferentes de python; mas todas as versões de python3 são backwards-compatible, então basta instalar a mais recente.

Além disso, estou estranhando a necessidade de ter 4 versões \*do NES\* instaladas no Neuromat. Mas isso preciso discutir com a Profa. Kelly.

> necessitando um repasse das requisições HTTP através de um proxy, que fica incumbido de repassá-las a cada serviço

Como já falei, esse arranjo é ruim. A função de um proxy HTTP não é essa. Com o proxy, os servidores HTTP ficam sem saber quem acessou.

> Q: Por que utilizar micros serviços?

Q. Porque NÃO utilizar micro-serviços?

A. Na estrutura mais pedestre, você teria que gerenciar N+1 coisas: 1 servidor HTTP e N serviços web (versões do NES, website, etc).

Na sua estrutura, você vai ter que gerenciar 3N+1 coisas: N serviços, N servidores HTTP, N containers, e 1 proxy.

> Com a individualização de cada sistema, caso algum deles seja comprometido, ficará limitado a esse sistema, sem possibilidade de prejudicar os demais sistemas

Não vejo porque. No sistema pedestre, há 3 possíveis pontos centralizados de falha que podem interromper todos os serviços: o hardware, o sistema operacional, e o servidor HTTP. No seu sistema, idem -- com o proxy no lugar do servidor HTTP.

Quanto a falhas que podem interromper apenas um dos N serviços, no esquema pedestre há basicamente uma: o software desse serviço e sua base de dados.

No seu esquema, além desse mesmo risco, o container pode falhar, ou o servidor HTTP dedicado pode cair.

> O problema se limitará aos recursos específicos destinados ao container, dessa forma, se bem configurados, um sistema não tem como "derrubar" outro.

Como já falei, essa abordagem é ineficiente, especialmente numa instalação com uso esporádico como é o Numec. Se você cria 16 contêiners com 32 GB RAM cada, você praticamente garante que, mesmo durante o \*pico\* de uso de um serviço, ele só vai ter 32 GB de RAM disponíveis para cache de NFS e coisas assim, em vez de 512.

Para uma base do NES, 32 GB e 512 GB não devem fazer grande diferença; mas essa diferença é enorme para tarefas de computação pesada -- que atualmente estão no HAL1, compartilhando RAM com os demais contêiners.

> a individualização faz com que os sistemas fiquem maximamente exutos, fazendo com que o backup e a recuperação sejam rápidos

O conjunto de todos os contêiners tem pelo menos tantos arquivos e programas quanto

> além disso, cada sistema independente pode  
> ser modificado, excluído e adicionado, sem que haja influência sobre os  
> demais, e.g.

> uma instalação instável do NES que demore a subir, não fará  
> com que o site do neuromat fique fora do ar.

Isso é verdade mesmo se houver 1 servidor HTTP dando acesso a todos os NES: um problema em um website não afeta os outros websites.

> O gerenciamento pode ser complicado se houver muitas interligações entre os serviços e, para esse caso, costumamos juntar os serviços em um mesmo container, apesar de não recomendado pelos intens acima.

Ou seja, no fim das contas vocês usam contêiners sem aproveitar as supostas vantagens dos mesmos?

> Q: Por que utilizar um proxy HTTP?

Porque NÃO utilizar um proxy HTTP?

> a cada novo serviço web que desejássemos subir, fazer-se-ia necessário que requisitássemos o registro do IP do sistema que rodará o serviço

Um único servidor HTTP e um único IP, para "www.numec.prp.usp.br" ou "www.numec.ime.usp.br" , bastaria para todos os serviços \*web\* -- mesmo sem proxy.

Você quer dizer serviços \*internet\* que não são web. Digamos SSH. Tem outros?

> não possuímos controle sobre o servidor DNS da Rede IME

Não precisa ter controle. É só requisitar. Se houver uma justificativa razoável, não vão negar. Essas interações com admins do IME e da USP são parte do seu trabalho.

> A nível de gestão de operações significaria que tudo ficaria estagnado até obtermos uma resposta (burocracia) e ficamos limitados ao horário de atendimento da Seção.

Endereços de IP não são cafezinhos! Você não decide "preciso de mais um endereço IP" no meio do hacking de um script, e precisa dele "já". O Numec vai precisar de dois ou três no máximo, e pode esperar um dia ou dois...

> nesse contexto, por não dependermos de terceiros, se torna possível realizar estimativas de tempo para a configuração de serviços, podendo subir, por exemplo, um sistema em <https://abacate.numec.prp.usp.br>, sem que hajam estagnações no workflow dos desenvolvedores

Mas a evidência experimental é que o serviço \*tem\* estagnado, aparentemente porque há montes de peças no sistema que não contribuem nada a não ser mais trabalho de administração e mais chances de falha.

--

Jorge Stolfi  
Full Professor/Professor Titular  
Instituto de Computação/Institute of Computing  
UNICAMP