4 MINICURSO ESPECÍFICO

Gerenciamento de Clusters de Alto Desempenho

Professor

Rafael Bohrer Ávila[1]

(avila@inf.ufrgs.br)

Resumo

Os clusters de alto desempenho são, atualmente, as arquiteturas mais utilizadas para a prática do processamento paralelo e distribuído. Nos últimos anos, com a tendência de construção de clusters cada vez maiores, o gerenciamento da máquina torna-se uma atividade cada vez mais complexa e que envolve diversos aspectos de seu uso e funcionamento. Nesse contexto, este curso aborda as principais atividades envolvidas no gerenciamento de um cluster, como a instalação e configuração do sistema operacional, cadastro de usuários, configuração do software de programação, manutenção, e alocação e monitoração de recursos.

[1]Graduação em Ciências da Computação pela Universidade Federal de Santa Catarina. Mestrado em Ciência da Computação pela Universidade Federal do Rio Grande do Sul e Doutorado em Ciência da Computação pela Universidade Federal do Rio Grande do Sul, UFRGS, Brasil, e pelo *Institut National Polytechnique de Grenoble*, INPG, França. Atualmente professor na Universidade de Santa Cruz do Sul, Departamento de Informática, e na Universidade Federal do Rio Grande do Sul, Instituto de Informática, Departamento de Informática Aplicada.

4.1. Introdução

A sub-área da Computação chamada comumente de *Computação Baseada em Clusters*[1] é uma tendência que vem se desenvolvendo a passos largos nas últimas duas décadas. Tendo tido seu grande impulso no início dos anos 90 [21, 30], é uma área que acompanha diretamente o desenvolvimento tecnológico de ponta em equipamentos de processamento, comunicação e armazenamento de dados, e na qual muito se tem investido, tanto em ambientes voltados a pesquisa e desenvolvimento quanto na indústria e no comércio.

As razões para tanto interesse na Computação Baseada em Clusters vêm principalmente de uma característica marcante de grande parte dos clusters implantados no mundo inteiro: o uso de componentes "de prateleira", ou seja, equipamentos amplamente disponíveis comercialmente para o grande público, como os computadores PC, e de software de livre utilização e distribuição, como o sistema operacional GNU/Linux [16, 10, 26]. Essa característica traz dois grandes benefícios imediatos: o uso de tecnologia atual, pois os componentes de hardware podem ser facilmente substituídos por outros mais atualizados, de melhor desempenho e/ou capacidade, e uma significativa redução de custos, já que nenhuma tecnologia especializada é envolvida e o software é gratuito.

4.2. O Modelo Beowulf de Computação Baseada em Clusters

A abordagem "de prateleira" recém comentada foi utilizada primeiramente pela equipe de Thomas Sterling e Donald Becker, pesquisadores do *NASA's Goddard Space Flight Center*, nos Estados Unidos, em um sistema cujo nome transformou-se praticamente em sinônimo de computação paralela de baixo custo: o cluster *Beowulf* [30, 28, 31, 32]. Beowulf é o herói de um antigo poema inglês, no qual é narrada a história de como ele libertou o povo dos Danes do monstro Grendel; a idéia de Sterling e Becker com o cluster Beowulf era libertar os pesquisadores e cientistas da área de Processamento Paralelo e Distribuído do uso de uma classe restrita de máquinas paralelas, que os forçava a um ciclo de desenvolvimento demorado e muitas vezes oneroso financeiramente.

Em 1994, Sterling e sua equipe foram desafiados pela NASA com o projeto de construir uma máquina paralela com 10 Gigabytes de espaço de armazenamento e desempenho de 1 GFLOPS de pico, cujo custo, porém, não deveria ultrapassar um orçamento de cerca de US$50.000,00. Diversos grupos de pesquisa americanos, na época, estavam investindo em máquinas paralelas com nós fracamente acoplados, formadas

[1] Do inglês *Cluster Computing*, *Clustered Computing* ou *Cluster-based Computing*. Na língua portuguesa, as divergências ficam por conta do termo *cluster*, normalmente traduzido para *agregado* ou *aglomerado*. Ao longo deste texto, opta-se pelo uso da palavra não traduzida, cluster, sem o uso do estilo itálico, por entender-se que é um termo já estabelecido entre a comunidade de paralelismo no Brasil e, por isso, identifica mais prontamente ao leitor o tipo de máquina paralela do qual se está falando.

por redes de estações de trabalho, utilizando software de programação como PVM [11] e o recém surgido padrão MPI [18]. A outra opção, mais tradicional, seria a compra de uma máquina MPP. Em ambos os casos, o orçamento seria ultrapassado em até 10 vezes o valor máximo estipulado devido aos custos das máquinas e dos sistemas operacionais comerciais.

O grupo voltou-se, então, para a já estabelecida nova classe de computadores pessoais, os PCs. Com processadores como o 486 DX4 e o Pentium, já seria possível construir um computador paralelo baseado nesse tipo de máquina que se aproximasse do desempenho necessário. A única peça que faltava para o nascimento do primeiro cluster Beowulf veio pouco tempo depois, com o lançamento da versão 1.0 do Linux.

O primeiro cluster Beowulf era composto de 16 nós com processadores 486 e rede Ethernet 10 Mb/s. Essa máquina era capaz de atingir 42 MFLOPS de desempenho de pico. Embora ainda longe dos 1 GFLOPS solicitados pela NASA, esse desempenho já era comparável ao de máquinas paralelas comerciais da época como a Paragon e a CM-5. Apenas 3 anos depois, com 32 nós de processamento usando processadores Pentium Pro, o desempenho da máquina ultrapassou os 2 GFLOPS, e poucos meses adiante os clusters passaram a figurar na lista TOP500 [34], que periodicamente relaciona, desde 1993, os 500 computadores mais poderosos do mundo.

Percebe-se, com esse breve relato, o tamanho do impacto que a abordagem da Computação Baseada em Clusters teve na área de Processamento Paralelo e Distribuído; em um intervalo de apenas 4 anos, os clusters passaram do nível de protótipo experimental para concorrentes das máquinas de maior poder computacional do mundo. Desde então, a facilidade de construção e manutenção de clusters do tipo Beowulf vem despertando o interesse de grupos de pesquisa do mundo inteiro, permitindo que praticamente qualquer instituição tenha a disponibilidade de uma máquina paralela para o desenvolvimento de suas pesquisas e execução de aplicações. Essa facilidade, aliada ao rápido progresso tecnológico, levou à construção de clusters cada vez maiores, que hoje chegam facilmente à casa dos milhares de nós, e conseqüentemente originam novos problemas.

Dentre esses problemas, destaca-se o do *gerenciamento*. Instalar, configurar e manter um cluster passou de um problema simples a um conjunto complexo de programas e ferramentas que visam automatizar, sobre um conjunto de dezenas, centenas ou milhares de nós, as ações típicas do gerenciamento de uma máquina: criação de usuários, instalação de software, atualizações, entre outras.

A Figura 4.1 ilustra uma topologia típica de instalação de um cluster. A máquina chamada de *frontal* é a que dá acesso ao cluster. É através dela que um usuário pode acessar os nós de processamento, vindo de algum lugar da Internet. Os nós têm a função de executar as aplicações paralelas em si, trocando informações por meio de suas conexões de rede. Cabe observar que a máquina frontal serve também de divisor entre a rede externa e a rede interna do cluster, o que oferece algumas vantagens do ponto de vista da segurança.

Ao longo deste texto, serão abordadas algumas das principais atividades ligadas ao gerenciamento de um cluster, procurando-se fornecer uma visão prática do tipo de procedimentos que devem ser realizados, relacionando-os com o software usado na prática, mas ao mesmo tempo tentando não torná-lo simplesmente uma “receita de

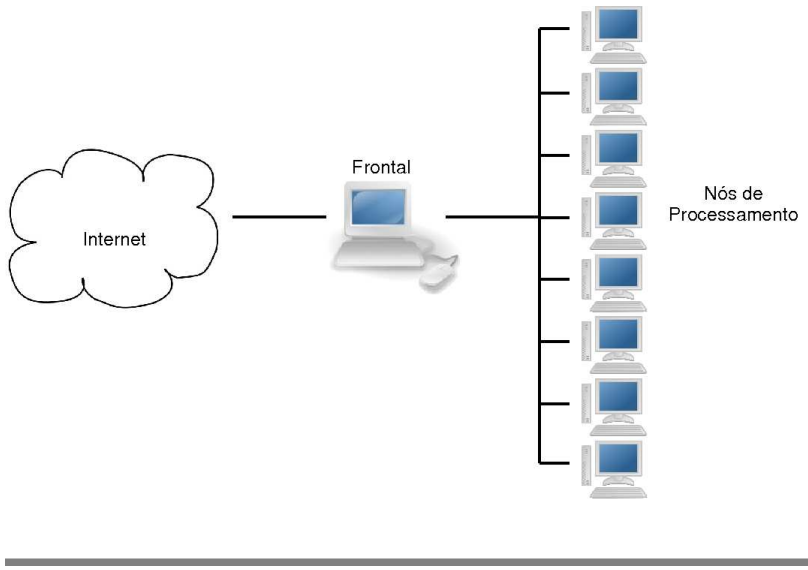

Figura 4.1: Uma topologia típica de cluster.

bolo" para construir uma máquina paralela.

As principais etapas no gerenciamento do cluster são divididas nas seções apresentadas a seguir.

4.3. Instalação e Configuração do Sistema Operacional

A instalação de um cluster começa por uma decisão simples, mas que influencia diretamente o seu uso futuro: a escolha do Sistema Operacional. Em muitos casos, na verdade, esta não é uma escolha, e sim uma conseqüência das necessidades dos usuários. Assim, uma empresa que trabalhe com aplicativos Microsoft, por exemplo, vai precisar dessa plataforma em suas máquinas. Por outro lado, se a motivação para a instalação do cluster são atividades de pesquisa em uma universidade ou laboratório, normalmente se tem a possibilidade de escolha. Atualmente, o GNU/Linux é o Sistema Operacional mais utilizado nesse tipo de ambiente [32], devido a uma série de razões:

- É um sistema eficiente, estável, com uma interface de programação madura e bem adaptada a um ambiente de rede devido à sua herança dos sistemas Unix;
- Tendo seu código aberto, pode ser analisado e livremente adaptado a necessidades específicas que possam surgir;
- Está disponível para ser baixado e instalado da Internet a qualquer momento;
- É de distribuição e utilização gratuitas.

Tabela 4.1: Distribuições GNU/Linux.

Distribuição	Sistema de Pacotes	Formato de Pacotes
Debian	APT	.deb
Ubuntu	APT	.deb
Mandriva	RPM	.rpm
Fedora Core	RPM	.rpm
Gentoo	Portage	.ebuild (código-fonte)

O GNU/Linux é atualmente disponibilizado sob os mais diferentes "sabores", chamados de *distribuições*. Uma distribuição é caracterizada principalmente por um processo específico de instalação do SO (Sistema Operacional) e um sistema de instalação/configuração de pacotes de software, normalmente definindo um formato específico para distribuição desses últimos. Ultimamente, tem sido também bastante comum o surgimento de distribuições derivadas, baseadas em distribuições já existentes, que herdam as características funcionais de sua ancestral (ex. formato dos pacotes) mas apresentando uma política ou filosofia diferente, por exemplo pela inclusão de pacotes mais voltados a máquinas *desktop*. A Tabela 4.1 lista algumas das principais distribuições GNU/Linux da atualidade.

A distribuição adotada como base para a elaboração deste texto é a *Debian GNU/Linux*, por apresentar uma excelente combinação de características de qualidade de software, segurança e procedimentos de manutenção. Cabe ressaltar, entretanto, que a maior parte do software necessário em um cluster pode ser encontrado em qualquer distribuição atual, e que portanto basta adaptar os procedimentos ao estilo da distribuição que se usa. A escolha de uma distribuição é, acima de tudo, questão de gosto pessoal do administrador.

4.3.1. Instalação Básica

O primeiro passo na instalação do cluster é a instalação do SO em todas as máquinas. Isso pode ser feito manualmente, se forem poucas (ex. cluster de 4 nós), ou então através de um processo automatizado de instalação em massa, como será visto posteriormente.

Neste momento, é importante lembrar de uma característica específica de um cluster, que freqüentemente é ignorada na fase de instalação. Um cluster é uma máquina com uma finalidade específica, a de rodar aplicações paralelas. Não faz muito sentido, em um cluster, instalar um programa para tocar MP3, ou um processador de texto. Portanto, muito do software normalmente instalado em máquina GNU/Linux pode ser descartado. Isso traz uma série de vantagens:

- Melhoria no desempenho da máquina, pois muitos pacotes de software implicam na execução de processos servidores que ocupam CPU e memória RAM
- Melhoria na segurança do cluster, pois diminui o número de serviços disponíveis e programas potencialmente com bugs

- Redução do espaço ocupado em disco

- Redução no tempo necessário para instalações e atualizações, pois menos pacotes têm que ser (re)instalados

As distribuições de GNU/Linux (e mesmo de outros SOs como o Windows XP) normalmente dão ao usuário a opção de "instalação mínima". Esta é uma boa opção para a instalação inicial, lembrando que qualquer pacote que eventualmente tenha faltado pode ser instalado depois. Com o passar do tempo, ao adquirir experiência com uma determinada distribuição, aprende-se a escolher melhor o perfil de instalação mais adequado. Muitas distribuições permitem também a instalação de perfis personalizados, o que facilita bastante a tarefa de instalações futuras.

Basicamente, o que se precisa em uma primeira instalação é acesso à rede, que normalmente vai ser usada pra instalar o restante do sistema. Nada de ambiente gráfico, servidor de email, servidor Web, suite Office, etc.

Uma parte importante da instalação inicial é a escolha da faixa de endereços IP a serem usados, bem como os nomes das máquinas. Sugere-se o uso de faixas de IPs reservados, como 192.168.0.0/16 ou 10.0.0.0/8, para não ocupar uma faixa de IPs válidos da instituição. Quanto aos nomes das máquinas, não há muita diferença entre uma ou outra escolha, mas as tarefas de administração podem ser facilitadas pelo uso de nomes como *n1*, *n2*, *n3*, e assim por diante.

4.3.2. Nós de Processamento

A função dos nós de processamento é somente executar as aplicações que são submetidas ao cluster. Portanto, é neles, especialmente, que vale a regra de não instalar pacotes adicionais.

Uma função especial que deve obrigatoriamente ser instalada nos nós é uma maneira de acessá-los remotamente. Para isso, dois pacotes são comumente usados: o RSH (*Remote Shell*) e o SSH (*Secure Shell*).

4.3.2.1. RSH

O *Remote Shell* [27], ou RSH, é o mecanismo de acesso originalmente usado nos primeiros clusters. É um serviço, herdado dos ambientes Unix, que permite tanto a abertura de um shell remoto quanto a execução remota de um comando. A Figura 4.2 mostra um exemplo de uso do serviço.

O RSH caiu em desuso nos últimos anos devido a questões de segurança, pois todo o tráfego entre as duas máquinas, inclusive a digitação de eventuais senhas necessárias para o acesso, circula de forma aberta na rede, sendo passível de interceptação por *sniffers*. No contexto dos clusters, apesar de não haver o mesmo grau de preocupação no acesso aos nós, o RSH apresenta outro problema, relacionado ao limite do número de conexões, que também o fez deixar de ser utilizado.

Pelos dois motivos, o RSH vem sendo substituído pelo SSH.

```
avila@frontal:~$ rsh n1 hostname
n1
avila@frontal:~$ rsh n1
Linux n1 2.6.12-1-686 #1 Tue Sep 6 15:10:40 UTC 2005 i686 GNU/Linux

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: Sun Dec 18 11:53:40 2005 from frontal
avila@n1:~$
```

Figura 4.2: Exemplo de utilização do RSH.

4.3.2.2. SSH

O *Secure Shell* [24], ou SSH, surgiu como substituto do RSH na função de execução remota de shell e comandos, mas incluindo mecanismos de criptografia dos dados transmitidos. Assim, é consideravelmente mais difícil interceptar uma informação ou capturar uma senha. Adicionalmente, o SSH não apresenta o mesmo problema de conexões existente no RSH.

Durante a instalação do pacote, ou quando de seu primeiro disparo, o SSH cria um par de chaves privada e pública para autenticação da própria máquina. Toda vez que um cliente acessa um servidor SSH, a chave é verificada na base de dados do cliente para conferir se corresponde à chave informada nos acessos anteriores (com exceção, é claro, do primeiro acesso, onde a chave é inicialmente armazenada). Se houver uma diferença, pode significar que alguém está interceptando a comunicação, num ataque conhecido como *Homem-do-Meio (Man-in-the-Middle)* [33, 4, 17].

Os nós do cluster devem permitir acesso por SSH a partir do nó frontal sem que seja preciso digitar uma senha. Se não fosse assim, usar o cluster seria bastante inconveniente (imagine disparar uma aplicação sobre 1000 nós dessa forma...). Existem basicamente duas formas de se fazer isso.

A primeira consiste em obrigar cada usuário a criar um par de chaves para a autenticação. Como as chaves ficam no diretório-home do usuário, e esse diretório é compartilhado por todos os nós, a autenticação pode ser feita dessa forma.

Outra maneira de atingir o mesmo resultado, mas sem exigir nenhuma ação dos usuários, é configurar a chamada *Host-based Authentication* (Autenticação Baseada em Máquina). Neste mecanismo, o SSH permite acesso de um usuário sem solicitar senha desde que ele venha de uma máquina conhecida. Como as máquinas do cluster são todas conhecidas, esse mecanismo pode ser usado.

Para habilitar esse tipo de autenticação, algumas configurações devem ser feitas em cada nó:

- No arquivo `/etc/ssh/sshd_config`, habilitar a opção HostbasedAuthentication e desabilitar IgnoreRhosts

```
avila@n1:~$ cat /etc/ssh/shosts.equiv
frontal
n1
n2
n3
n4
n5
n6
```

Figura 4.3: Exemplo de arquivo `/etc/ssh/shosts.equiv`**.**

- No `/etc/ssh/ssh_config`, habilitar as opções HostbasedAuthentication e EnableSSHKeysign, e desabilitar StrictHostKeyChecking
- Criar o arquivo `/etc/ssh/shosts.equiv`, contendo o nome da máquina frontal e os dos nós, como mostra a Figura 4.3

4.3.3. Máquina Frontal (*Front-end*)

A máquina frontal é normalmente responsável por diversas tarefas no cluster:

- Atua como a "porta de entrada" do cluster, sendo normalmente acessível de qualquer lugar da rede;
- Realiza a autenticação primária dos usuários; ou seja, para poder usar o cluster, o usuário deve possuir login na máquina frontal;
- Contém os arquivos dos usuários, oferencendo também mecanismos para que esses arquivos possam ser transferidos de e para o cluster;
- É onde os usuários vão compilar e eventualmente depurar seus programas, portanto a máquina frontal deve oferecer os compiladores, bibliotecas, depuradores, etc.;
- É também bastante comum que a máquina frontal execute serviços de gerência do cluster, como DHCP, NFS, firewall, entre outros. Embora seja possível ter máquinas dedicadas a esses serviços, vamos trabalhar sobre essa configuração por ser mais simples e normalmente atender bem às necessidades do cluster.

4.3.3.1. SSH

O serviço de SSH na máquina frontal deve ser configurado como normalmente é encontrado em ambientes de rede, ou seja, exigindo autenticação por chave pública ou por senha.

Deve-se, entretanto, fazer a mesma alteração no arquivo `/etc/ssh/ssh_config` que foi feita nos nós. Ela vai servir para o funcionamento do mecanismo de autenticação *Host-based* usado naqueles.

```
avila@frontal:~$ cat /etc/dnsmasq.conf
domain-needed
bogus-priv
interface=eth1
dhcp-range=10.0.0.1,10.0.0.10
dhcp-host=00:06:5b:29:52:55,n1
dhcp-host=00:06:5b:29:52:52,n2
[...]
```

Figura 4.4: Exemplo de arquivo `/etc/dnsmasq.conf`**.**

4.3.3.2. DHCP

O DHCP (*Dynamic Host Configuration Protocol*) [8] é um recurso amplamente usado em ambientes de rede que fornece a um conjunto de clientes uma série de informações sobre a própria rede: endereço IP da máquina, máscara de sub-rede, roteador padrão, servidores de nomes, etc. A vantagem desse mecanismo é que essas informações não precisam ser configuradas estaticamente em cada um dos clientes, facilitando eventuais alterações e centralizando o controle dos parâmetros da rede. Por exemplo, se por algum motivo o administrador resolve dividir uma determinada rede em duas sub-redes, basta ajustar a faixa de endereços IP e a máscara no servidor DHCP, e os clientes receberão os novos parâmetros automaticamente na próxima inicialização.

O servidor DHCP tradicionalmente usado em ambientes Unix é o próprio pacote `dhcp-server` [6]. Esse pacote implementa o servidor com todas as suas extensões.

Em um cluster, entretanto, pode-se usar uma alternativa mais simples, pois somente as funções básicas são necessárias. Um pacote bastante adequado é o DNSmasq [7], que implementa um servidor DHCP simples e de quebra ainda oferece a função de servidor de DNS para os nós, se necessário.

Um exemplo de configuração do DNSmasq pode ser visto na Figura 4.4. As diretivas mais relevantes são interface, dhcp-range e dhcp-host. A primeira define em que interfaces de rede o servidor vai atender requisições. Normalmente, vamos definir somente a interface interna, que dá acesso aos nós, nesta diretiva. A seguir temos a definição da faixa de IPs que serão distribuídos pelo servidor. Por último, faz-se uma relação dos endereços MAC de cada nó com o endereço IP desejado para ele (mapeados através do `/etc/hosts`).

4.3.3.3. NFS

Freqüentemente, o servidor frontal é também o servidor de arquivos do cluster. Para isso, a solução mais amplamente empregada é o NFS (*Network File System*) [1].

Assim como o RSH, o NFS é herdado de ambientes Unix. Usa-se o NFS para o compartilhamento de arquivos em uma rede local, de modo que os usuários enxergam o mesmo conjunto de arquivos não importando em qual máquina da rede estejam trabalhando. No cluster, o NFS vai ser usado para implementar essa característica no acesso aos nós de processamento.

A configuração do NFS exige, no lado do servidor, que os diretórios a serem

```
avila@frontal:~$ cat /etc/exports
# /etc/exports: the access control list for filesystems
# which may be exported to NFS clients. See exports(5).
/home/users    10.0.0.0/255.0.0.0(rw,sync)
```

Figura 4.5: Exemplo de arquivo `/etc/exports`**.**

```
avila@n1:~$ cat /etc/fstab
# /etc/fstab: static file system information.
#
[...]
frontal:/home/users /home/users nfs
    defaults,rsize=8192,wsize=8192,intr,bg 0 0
```

Figura 4.6: Exemplo de arquivo `/etc/fstab`**.**

compartilhados sejam *exportados* através de uma configuração especial. Os dados relativos à exportação são configurados no arquivo `/etc/exports`, como mostrado na Figura 4.5.

O exemplo mostra que o diretório `/home/users` está sendo exportado para toda a sub-rede do cluster (ou seja, todos os nós), e que as permissões devem ser de leitura e escrita.

Nos nós de processamento, deve ser feita a *montagem* dos diretórios compartilhados, através de uma configuração no arquivo `/etc/fstab`. A Figura 4.6 mostra um exemplo.

A linha mostrada[2] indica a montagem do diretório remoto `/home/users`, sendo do tipo NFS, com as opções-padrão de montagem, blocos de dados de 8192 bytes e montagem em *background* (útil se o servidor demora mais do que os nós para inicializar).

4.4. Cadastro de Usuários

Outra parte fundamental da administração do cluster é o cadastro de usuários. Durante o processo de instalação e configuração, deve-se decidir como o cadastro e autenticação nos nós serão feitos.

Uma etapa existente em qualquer caso é a criação do usuário localmente na máquina frontal. Para isso, as distribuições de GNU/Linux normalmente oferecem um comando "`adduser`" ou "`useradd`", ou ainda uma interface gráfica para tal.

A decisão a ser tomada se refere à maneira de fazer os nós enxergarem os usuários cadastrados. Duas soluções podem ser adotadas.

A primeira é o uso do serviço NIS (*Network Information Service*) [15]. O NIS é mais um dos serviços Unix herdados pelos clusters.

[2] A linha foi quebrada, para fins de melhor apresentação neste texto; na prática, todos os parâmetros devem ser colocados em uma linha só.

Na máquina frontal, deve ser rodado o servidor NIS, normalmente chamado de `ypserv`, e a ele deve ser atribuído um *domínio*. Este pode ser, por exemplo, o nome do próprio cluster. A maioria das distribuições GNU/Linux configura o NIS, por padrão, para atuar como cliente, portanto deve-se verificar na distribuição como fazê-lo atuar também como servidor. No Debian, deve-se setar a variável NISSERVER=true no arquivo `/etc/default/nis`. Em qualquer sistema, como última etapa da instalação, deve-se executar "`/usr/lib/yp/ypinit -m`" para inicializar a base de dados do servidor. O nome escolhido para o domínio deve ser colocado no arquivo `/etc/defaultdomain`, tanto no servidor quanto nos clientes.

Os clientes NIS devem, além de definir o domínio, incluir uma linha especial nos arquivos `/etc/passwd` e `/etc/group`. Esta linha vai indicar que a base de usuários e grupos deve obter informações também do NIS, além das informações locais. O formato dessa linha é uma sinal de mais, seguido de tantos dois-pontos quanto o número de campos usados em cada arquivo. Portanto, no `/etc/passwd` a linha é +:::::: e no `/etc/group` é +:::.

Uma vez instalado o NIS, todos os usuários podem se autenticar em qualquer máquina cliente. Se alguma alteração for feita na base de usuários (ex. troca de senha ou inclusão de um novo usuário), deve-se executar o comando "`make`" no diretório `/var/yp` para a devida atualização.

Embora o NIS seja razoavelmente simples de ser instalado, pode parecer um mecanismo um tanto “pesado” para uso em um cluster. Na verdade, uma vez que a autenticação dos usuários nos nós é feita por chave pública ou por autenticação de máquina, o que se precisa é somente ter os UID dos usuários. Isso pode ser conseguido simplesmente pela cópia do arquivo `/etc/passwd` da máquina frontal para os nós. Apenas deve-se tomar o cuidado de não eliminar nenhum usuário de sistema que tenha sido criado pela instalação de pacotes de software nos nós. Isso pode acontecer se o pacote é usado somente nos nós e não na frontal.

4.5. Configuração do Software de Programação

Como já colocado anteriormente, é na máquina frontal que os usuários vão compilar e depurar seus programas. Para tanto, é necessário que os compiladores, ferramentas e bibliotecas de programação estejam disponíveis.

Muitos desse componentes de software podem ser encontrados diretamente como pacotes das distribuições Linux, portanto sua instalação é trivial. Outros, entretanto, podem exigir instalação manual. É para esses casos que se usa o diretório `/usr/local`.

Um bom exemplo é a instalação da biblioteca MPI [18]. Mesmo disponível como pacote das distribuições, é bastante comum querer instalar versões mais recentes e variantes dessa biblioteca.

O problema de se instalar software dessa maneira é quando do uso de bibliotecas de ligação dinâmica, que devem estar presentes em tempo de execução. Se o software é instalado somente na máquina frontal, essas bibliotecas não vão ser encontradas nos nós no momento da execução das aplicações, e conseqüentemente a execução vai falhar. A solução é portanto compartilhar também o `/usr/local` entre o nó frontal e os de

```
avila@frontal:~$ cat /etc/exports
# /etc/exports: the access control list for filesystems
# which may be exported to NFS clients. See exports(5).
/home/users   10.0.0.0/255.0.0.0(rw,sync)
/usr/local    10.0.0.0/255.0.0.0(ro,sync)

avila@n1:~$ cat /etc/fstab
# /etc/fstab: static file system information.
#
[...]
frontal:/home/users /home/users nfs
defaults,rsize=8192,wsize=8192,intr,bg 0 0
frontal:/usr/local  /usr/local  nfs
defaults,rsize=8192,wsize=8192,intr,bg 0 0
```

Figura 4.7: Novas versões dos arquivos `/etc/exports` e `/etc/fstab`.

processamento. A Figura 4.7 mostra como ficam as configurações de `/etc/exports` e `/etc/fstab` com esse novo compartilhamento. Note que o `/usr/local` pode ser compartilhado como somente-leitura (opção `ro` no `/etc/exports`), pois toda instalação de software nesse diretório é feita no frontal.

Um detalhe importante com instalação de software local é tornar os executáveis disponíveis no caminho padrão (*PATH*) dos usuários. Isso pode ser conseguido pela inclusão dos diretórios corretos no arquivo `/etc/profile`. Como exemplo, a Figura 4.8 mostra a inclusão do diretório que contém os executáveis das bibliotecas MPI e Java. Novos pacotes de software podem ser facilmente incluídos, bastando adicionar o nome do pacote à lista.

```
avila@frontal:~$ cat /etc/profile
# /etc/profile: system-wide .profile file for the Bourne shell (sh(1))
# and Bourne compatible shells (bash(1), ksh(1), ash(1), ...).
[...]
for i in mpi java; do
       PATH=$PATH:/usr/local/$i/bin
done
export PATH
```

Figura 4.8: Inclusão de pacotes de software locais no PATH padrão dos usuários.

4.6. Alocação e Monitoração de Recursos

Para o uso diário do cluster, é recomendável que se utilize um software dedicado ao controle de alocação dos nós. Um dos mais utilizados é o PBS (*Portable Batch System*) [23].

A idéia fundamental é que cada usuário, quando deseja usar o cluster, solicite ao sistema o número de nós que necessita. Assim pode ser feito um controle de quais nós estão sendo usados e quais estão disponíveis. Se, no momento que um usuário submete sua solicitação, não há o número suficiente de nós livres, ele entra em uma fila.

O PBS permite dois tipos de solicitação: interativa ou em lote. Uma solicitação em lote segue exatamente o mecanismo recém descrito, através da passagem de um script que deve conter os passos da execução da aplicação. Por exemplo, se for uma aplicação MPI, o script passado ao PBS é o responsável por chamar o comando "`mpirun`", podendo fazer uso de variáveis definidas pelo PBS com parâmetros da submissão. A Figura 4.9 ilustra esse exemplo. Se a quantidade de nós disponíveis no momento da submissão atende ao que foi solicitado, o script é executado imediatamente. Senão, como descrito anteriormente, ele é colocado em uma fila e executado assim que possível.

```
#! /bin/sh
#PBS -l walltime=2:00:00
#PBS -l nodes=4
mpirun -machinefile $PBS_NODEFILE mandelbrot 0 0 2 2
```

Figura 4.9: Script para submissão em lote no PBS.

A outra forma de usar o PBS é através de uma solicitação interativa. Neste caso, não é necessário passar um script pois, no momento em que for possível atender a solicitação (em função do número de nós disponíveis), será aberto um interpretador de comandos em um dos nós alocados para este caso, via RSH ou SSH.

Um recurso não implementado diretamente pelo PBS, mas suportado por ele, é o bloqueio de acesso aos nós. Para evitar que um usuário descuidado ou mal-intencionado utilize nós que foram alocados para outro usuário, pode-se implantar um sistema de bloqueio através do arquivo `/etc/security/access.conf`. Esse arquivo contém linhas que descrevem quais usuários podem ou não podem ter acesso à máquina. O exemplo na Figura 4.10 mostra uma configuração que permite acesso aos usuários *root* e *avila*, mas bloqueia todos os demais.

O que o PBS permite fazer é associar scripts de controle que são executados no início e no final das alocações. Esses scripts podem ser usados, então, para incluir e remover usuários das listas do `access.conf`. A configuração padrão em todos os nós seria bloquear o acesso. Quando uma solicitação é atendida, os nós correspondentes a ela são liberados para o usuário que a enviou. Quando a execução termina, o bloqueio volta a ser estabelecido.

4.6.1. Monitoração de Recursos

Além do controle de acesso, uma ferramenta bastante comum em um cluster é o software de monitoração dos recursos. Freqüentemente, tanto usuários quanto administradores desejam saber em que estado de funcionamento se encontram as máquinas.

Soluções de monitoração variam desde scripts simples até ambientes gráficos bastante completos. Uma maneira simples e de rápida implementação é fazer um script

```
# Login access control table.
#
[...]
#PBSuser
+:root avila:ALL
-ALL:ALL
```

Figura 4.10: Exemplo de bloqueio de acesso aos nós usando o `access.conf`.

como mostrado na Figura 4.11. Cada nó do cluster é acessado e são listados os processos do usuário atual.

```
#! /bin/sh
NODES="n1 n2 n3 n4 n5 n6"
for n in $NODES; do
       echo "*** $n ***"
       ssh $n ps x
done
```

Figura 4.11: Script simples para listagem dos processos nos nós do cluster.

Um exemplo um pouco mais elaborado é mostrado na Figura 4.12. A carga da CPU é medida através do *loadavg* e mostrada na tela, para cada nó, com marcas de "#". Os usuários logados em cada máquina são também mostrados. A saída é reproduzida na Figura 4.13.

Soluções mais completas de monitoração permitem a análise de diversos outros parâmetros, como uso da memória RAM, atividades nos discos, vazão da rede, etc. Uma ferramenta bastante completa que oferece todas essas opções é o Ganglia. Sua tela inicial mostra um panorama do cluster como um todo, com gráficos mostrando a utilização de CPU em cada nó e a carga global do sistema. Clicando-se sobre um dos gráficos, pode-se obter informações mais detalhadas como a utilização de memória virtual, pacotes transferidos pela rede, entre outros. A Figura 4.14 apresenta parte da tela do Ganglia sendo usado na monitoração do *iCluster2*[3], do INRIA de Grenoble, França.

Um outro exemplo de ferramenta de monitoração é o Monika [22], desenvolvido no Laboratoire ID/IMAG de Grenoble. Ele atua em sintonia com o OAR [2], software de escalonamento semelhante ao PBS, desenvolvido pelo mesmo grupo. O Monika é usado para verificar quais nós estão ocupados e para quais usuários. Clicando sobre um dos nós ocupados, pode-se descobrir detalhes sobre a submissão como número de nós e tempo solicitado. A Figura 4.15 ilustra uma tela do Monika também no iCluster2.

[3]http://ita.imag.fr

```
#! /bin/sh
NODES="n1 n2 n3 n4 n5 n6"
EXCLUDE='/^root/d; /^daemon/d; /^USER/d; /ps aux/d'

print_hash()
{
    local i
    for i in `seq 0 $1`; do echo -n '#'; done
    for i in `seq $1 10`; do echo -ne '.'; done
    echo -n ' '
}

for i in $NODES; do
    printf "%5s" "$i "
    if ! ping -q -c1 $i &> /dev/null; then
        echo '(*** dead ***)'
    elif ! ssh $i true &> /dev/null; then
        echo '(bloqueado)'
    else
        l=`ssh $i cat /proc/loadavg | cut -d' ' -f1`
        print_hash `echo $l .05+ 10* 1/p | dc`
        echo `ssh $i ps aux | sed -e "$EXCLUDE" | cut -d' ' -f1 | sort -u`
    fi
done
```

Figura 4.12: Script que mostra a carga e os usuários em cada nó do cluster.

```
  n1 #..........
  n2 #####..... avila
  n3 #####..... avila
  n4 #..........
  n5 #..........
  n6 ########.... avila
```

Figura 4.13: Resultado da execução do script da Figura 4.12.

Figura 4.14: Amostra de uma tela do Ganglia.

4.7. Manutenção

Uma vez que o cluster esteja instalado e seja posto em produção, começam as atividades de manutenção. Duas tarefas principais ocupam a vida do administrador: atualizações da instalação e a segurança do cluster.

4.7.1. Atualizações

Existem vários motivos pelos quais é necessário atualizar a instalação. Pacotes de software que são solicitados, correções de items que não ficaram corretos, novos métodos, etc.

Em qualquer caso, é praticamente imprescindível contar com um método de *instalação automatizada*. Instalar, configurar e atualizar algumas máquinas manualmente é fácil. Porém, em um cluster com dezenas ou centenas (ou milhares!) de nós, é uma tarefa que exigiria um trabalho braçal muito grande.

É comum, em sistemas de instalação automatizada, a definição de *golden client*. Essa é a máquina que contém a instalação de referência, sobre a qual serão feitas as atualizações necessárias, e que servirá de base para a instalação das demais. Em geral, vai ser gerada uma *imagem* da instalação de referência, a qual será posteriormente copiada para as demais máquinas.

Um script para instalação automatizada pode ser facilmente construído usando o processo de *NFS-Root* [19]. A idéia é ter, na máquina frontal ou em um servidor à parte, uma instalação básica exportada inteiramente por NFS. Essa instalação vai ser usada como partição raiz (“/”) de cada nova instalação, sendo montada logo após a carga do kernel. Tudo o que essa instalação vai fazer é copiar a imagem do *golden client* a partir

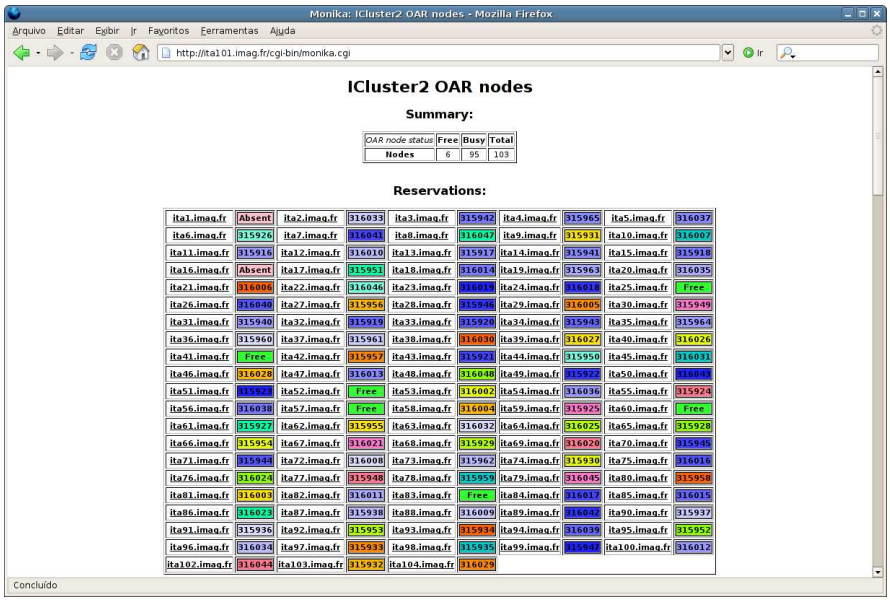

Figura 4.15: Amostra de uma tela do Monika.

do servidor, descompactá-la na nova máquina e configurar alguns parâmetros que são individuais de cada uma (ex. o *hostname*).

Existem também soluções prontas para o processo de instalação automatizada. Alguns dos pacotes de software mais utilizados são o System Installation Suite [29], que faz parte do OSCAR [25] (conjunto de pacotes de software para clusters) e o Norton Ghost [20]. Alguns pacotes são baseados em características específicas das distribuições de GNU/Linux, como o FAI [9] para o Debian. O assunto também é tema de diversas pesquisas atuais [13, 14].

4.7.2. Segurança

Embora tenhamos comentado anteriormente que a segurança internamente ao cluster não é uma das maiores preocupações, visto que é um ambiente com finalidade específica e de acesso limitado, o mesmo não se pode dizer em relação à máquina frontal. Essa sim é uma máquina normalmente exposta ao restante da rede e portanto sucetível a diversos tipos de ataque. Mesmo que o cluster não contenha dados sigilosos e que o mecanismo de instalação automatizada facilite a tarefa de reinstalação, ao menos dois cenários são preocupantes:

- Uma vez que a máquina frontal seja invadida, ela pode servir de base de apoio para outros ataques, possivelmente com conseqüências mais sérias (ex. invasão de sistemas bancários). Como a máquina frontal faz parte do caminho do invasor, o endereço IP da instituição acaba sendo envolvido, e a incomodação administrativa é certa;

- A maioria dos usuários tende a utilizar poucas senhas distintas, então é grande a possibilidade de a senha que determinado usuário usa no cluster seja a mesma

senha que ele utiliza em algum outro sistema. Se o invasor conseguir fazer a captura de senhas (ex. com algum *key logger*), ele pode ganhar acesso imediato a outras máquinas.

Algumas das mais importantes atividades de manutenção, em relação à segurança, são relacionadas a seguir.

4.7.2.1. Qualidade das Senhas

O ponto mais fraco da segurança de uma rede são os usuários. Por mais que se enfatize a importância da escolha de uma senha adequada, que combine letras maiúsculas e minúsculas, dígitos, símbolos, etc., ainda existem usuários que, por falta de experiência ou mesmo por negligência, utilizam senhas como a data de nascimento, 12345, nome da namorada, etc.

O que o administrador do cluster pode e deve fazer é executar, periodicamente (se não permanentemente, através do *cron*), algum programa de *crack* de senhas. Alguns exemplos são o *John The Ripper* [12] e o *Crack* [5]. Na medida em que as senhas fracas vão sendo quebradas, as contas correspondentes podem ser bloqueadas até que o usuário (ir)responsável faça a troca.

4.7.2.2. Atualizações do Kernel e dos Pacotes

Outra tarefa importante é estar sempre atento a eventuais vulnerabilidades descobertas no sistema e proceder imediatamente à sua atualização. A lista de difusão do *linuxsecurity.com* é uma boa fonte de informações. Outro site dedicado ao assunto é o *securityfocus.com*, que abrange também outros sistemas operacionais.

Uma vulnerabilidade grave no kernel geralmente permite a um usuário comum obter privilégios de *root*. Se o invasor conseguir capturar ou descobrir a senha de algum usuário, ele pode se beneficiar dessa falha para realizar ações privilegiadas, como instalar um *key logger* ou algum serviço “escondido” para obter acessos futuros.

Os pacotes de software usados no cluster também devem ser monitorados quanto a problemas de segurança, pois falhas em algum servidor (ex. SSH, DHCP) podem permitir acesso remoto privilegiado, podendo portanto ser ainda mais grave que falhas no kernel.

O *ChkRootKit* [3] é um script que faz uma verificação no sistema, procurando por sinais de invasão. A tarefa é semelhante à de um anti-vírus: o script procura por traços característicos de métodos de invasão conhecidos. Seu uso é uma ação recomendada no caso de suspeita de comprometimento do sistema. O ideal é que a máquina seja inicializada a partir de um *Live CD* (instalação de Linux que inicializa diretamente a partir do CD-ROM), pois o software instalado pelo invasor pode mascarar o processo de detecção.

4.7.2.3. Verificação dos Logs

Por último, é sempre recomendável uma verificação periódica dos *logs* do sistema. Explorações de vulnerabilidades ou mesmo tentativas de invasão freqüentemente

```
Sep 28 16:01:59 gppd sshd[17825]: Failed password for illegal user
    webmaster from 140.96.170.136 port 35785 ssh2
Sep 28 20:45:52 gppd sshd[372]: reverse mapping checking getaddrinfo
    for 200-180-170-188.paemt7004.dsl.brasiltelecom.net.br failed
    - POSSIBLE BREAKIN ATTEMPT!
[...]
Oct 16 03:47:55 gppd sshd[12275]: Failed password for illegal user user
    from 65.39.192.237 port 53980 ssh2
Oct 16 03:47:57 gppd sshd[12651]: Failed password for illegal user username
    from 65.39.192.237 port 54079 ssh2
Oct 16 03:47:58 gppd sshd[12653]: Failed password for illegal user ftpuser
    from 65.39.192.237 port 54189 ssh2
Oct 16 03:48:00 gppd sshd[12655]: Failed password for illegal user ftp
    from 65.39.192.237 port 54288 ssh2
Oct 16 03:48:02 gppd sshd[13058]: Failed password for illegal user linux
    from 65.39.192.237 port 54388 ssh2
Oct 16 03:48:03 gppd sshd[13060]: Failed password for www-data
    from 65.39.192.237 port 54491 ssh2
```

Figura 4.16: Tentativa de invasão em um dos clusters do GPPD da UFRGS.

causam a ocorrência de mensagens “estranhas” nos *logs*. Um exemplo de tentativa de invasão em um dos clusters do GPPD da UFRGS é mostrado na Figura 4.16. Nesse exemplo, um invasor intenciona acessar a máquina remotamente, via SSH, através de contas de usuários de sistema (*ftp, ftpuser, www-data*, *etc.*) que eventualmente estejam mal configuradas.

4.8. Bibliografia

[1] B. Callaghan, B. Pawlowski, and P. Staubach. *NFS Version 3 Protocol Specification: RFC 1831*. Internet Engineering Task Force, Network Working Group, June 1995.

[2] Nicolas Capit, Georges Da Costa, Yiannis Georgiou, Guillaume Huard, Cyrille Martin, Grégory Mounié, Pierre Neyron, and Olivier Richard. A batch scheduler with high level components. In *Proc. of the 5th IEEE/ACM International Symposium on Cluster and Grid, CCGrid*. Los Alamitos, IEEE Computer Society, May 2005.

[3] Chkrootkit - locally checks for signs of a rootkit, 2005. Disponível em: <http://www.chkrootkit.org/>. Acesso em: dez. 2005.

[4] George Coulouris. *Distributed Systems: Concepts and Design*. Addison-Wesley, Harlow, third edition, 2001.

[5] Crack, 2005. Disponível em: <http://www.crypticide.com/users/alecm/security/c50-faq.html>. Acesso em: dez. 2005.

[6] DHCP dynamic host configuration protocol, 2005. Disponível em: <http://www.isc.org/index.pl?/sw/dhcp/>. Acesso em: dez. 2005.

[7] Dnsmasq: a DNS forwarder for NAT firewalls, 2005. Disponível em: <http://thekelleys.org.uk/dnsmasq/doc.html>. Acesso em: dez. 2005.

[8] R. Droms. *DHCP Dynamic Host Configuration Protocol: RFC 2131*. Internet Engineering Task Force, Network Working Group, March 1997.

[9] FAI (Fully Automatic Installation) home page, 2005. Disponível em: <http://www.informatik.uni-koeln.de/fai>. Acesso em: dez. 2005.

[10] Free software foundation, 2005. Disponível em: <http://www.fsf.org>. Acesso em: jan. 2005.

[11] Al Geist et al. *PVM: Parallel Virtual Machine*. MIT Press, Cambridge, 1994.

[12] John the Ripper password cracker, 2005. Disponível em: <http://www.openwall.com/john/>. Acesso em: dez. 2005.

[13] Ka-deploy, 2005. Disponível em: <http://ka-tools.sourceforge.net/deploy.html>. Acesso em: dez. 2005.

[14] Rodrigo Kassick. Installation over Web Services. Trabalho de diplomação, Instituto de Informática — Universidade Federal do Rio Grande do Sul, Porto Alegre, 2005.

[15] Thorsten Kukuk. Nis, 2005. Disponível em: <http://www.tldp.org/HOWTO/NIS-HOWTO/>. Acesso em: dez. 2005.

[16] The Linux homepage, 2005. Disponível em: <http://www.linux.org>. Acesso em: jan. 2005.

[17] Alfred J. Menezes. *Handbook of Applied Cryptography*. The Crc press series on discrete mathematics and its applications. Crc, Boca Raton, 1997.

[18] MPI Forum. The MPI message passing interface standard. Technical report, University of Tennessee, Knoxville, April 1994.

[19] Nfs-Root mini-HOWTO, 2005. Disponível em: <http://www.tldp.org/HOWTO/NFS-Root.html>. Acesso em: dez. 2005.

[20] Norton ghost, 2005. Disponível em: <http://www.symantec.com/home_homeoffice/-products/backup_recovery/ghost10>. Acesso em: dez. 2005.

[21] The Berkeley NOW project, 1995. Available at: <http://now.cs.berkeley.edu>. Access in: Jan. 2005.

[22] OAR resource mangement system for high performance computing, 2005. Disponível em: <http://oar.imag.fr/>. Acesso em: dez. 2005.

[23] OpenPBS, 2005. Disponível em: <http://www.openpbs.org/>. Acesso em: dez. 2005.

[24] OpenSSH, 2005. Disponível em: <http://www.openssh.com>. Acesso em: dez. 2005.

[25] OSCAR open source cluster application resources, 2005. Disponível em: <http://oscar.openclustergroup.org/>. Acesso em: dez. 2005.

[26] Open source initiative, 2005. Disponível em: <http://www.opensource.org>. Acesso em: jan. 2005.

[27] RSH remote shell, 2005. Disponível em: <http://www.mkssoftware.com/docs/man1/rsh.1.asp>. Acesso em: dez. 2005.

[28] Daniel F. Savarese and Thomas Sterling. Beowulf. In Rajkumar Buyya, editor, *High Performance Cluster Computing: Architectures and Systems*, chapter 26, pages 625–645. Prentice Hall PTR, Upper Saddle River, 1999.

[29] SIS system installation suite, 2005. Disponível em: <http://wiki.sisuite.org/>. Acesso em: dez. 2005.

[30] Thomas Sterling, Donald J. Becker, Daniel Savarese, John E. Dorband, Udaya A. Ranawake, and Charles V. Packer. BEOWULF: a parallel workstation for scientific computation. In *Proceedings of the 24th International Conference on Parallel Processing*, pages 11–14, Oconomowoc, WI, 1995.

[31] Thomas L. Sterling, John Salmon, Donald J. Becker, and Daniel F. Savarese. *How to Build a Beowulf: a Guide to the Implementation and Application of PC Clusters*. MIT, Cambridge, 1999.

[32] Thomas Lawrence Sterling. *Beowulf Cluster Computing with Linux*. MIT Press, Cambridge, 2002.

[33] Andrew S. Tanenbaum and Maarten van Steen. *Distributed Systems: Principles and Paradigms*. Prentice Hall, Upper Saddle River, 2002.

[34] Top500, 2005. Disponível em: <http://www.top500.org>. Acesso em: jan. 2005.

Anotações dos Mini-Cursos

Anotações dos Mini-Cursos

Anotações dos Mini-Cursos

Anotações dos Mini-Cursos

Anotações dos Mini-Cursos

Anotações dos Mini-Cursos

Anotações dos Mini-Cursos

Anotações dos Mini-Cursos

Anotações dos Mini-Cursos

Anotações dos Mini-Cursos