FMA 215

Aula 10: Uso Básico da Rede

Prof. Jorge L. deLyra

06 de Junho de 2002

1 Alguns Conceitos Relevantes

A rede é um sistema de hardware e de software através do qual sistemas computacionais diferentes podem se comunicar. O tipo de comunicação que se pode efetuar varia, indo desde a simples transferência de uma mensagem ou de um arquivo de um sistema para outro a até a integração tão próxima de vários sistemas que eles podem passar a funcionar como um só sistema do ponto de vista do usuário. Hoje em dia a conexão à rede é parte essencial do uso de sistemas de informação, tanto que costumamos denominar os sistemas de informática em rede de "sistemas de informação e comunicação". Em lugares onde existem redes locais rápidas, como em empresas e universidades, o uso simultâneo de vários sistemas integrados através da rede, de forma completamente transparente para o usuário, é rotina.

A rede Internet funciona através de um sistema de troca de parcelas de informação na forma de pacotes, conhecido como comutação de pacotes (packet switching). Neste sistema, qualquer quantidade de informação que se queira transmitir de um ponto para outro é quebrada em um determinado número de pacotes de tamanho mais ou menos fixo. Cada pacote contém vários níveis de cabeçalhos (headers) contendo as informações sobre o endereço de destino do pacote. Esta informação de destino inclui não só o endereço da máquina de destino, mas também endereçamento de conexão relativo ao processo (e portanto ao usuário) que deve receber o pacote no sistema de destino. Os pacotes são todos enviados pela rede e recebidos pelo sistema no endereço de destino, que os lê e re-monta o conteúdo completo da informação.

A rede Internet pode ser compreendida como um sistema de interconexão de redes locais, integrando-as em uma única rede de alcance global, é daí que vem o seu nome (inter-network). O conceito de rede local, em sua forma mais simples, consiste de um conjunto de máquinas interligadas fisicamente de forma direta através de um sistema de transporte de pacotes de dados, que pode ser feito com fibras óticas ou cabos de cobre. Em geral estas máquinas estão próximas umas das outras, em um mesmo prédio ou em um conjunto de prédios contíguos, como é o caso do Instituto. Dentro desta rede local não é necessário estabelecer rotas de tráfego para os pacotes, pois cada máquina da rede pode se conectar diretamente a qualquer outra máquina da mesma rede local, sem que o tráfego entre as duas tenha de passar por qualquer outra máquina.

O roteamento do pacote no caminho intermediário entre duas redes locais é feito de forma automática pela rede, nem o sistema de origem nem o de destino sabem ou precisam saber qual é a rota que vai ser usada. Tudo que o sistema de origem precisa saber é para onde enviar um pacote que não seja parte de sua rede local. Em geral isto é feito através da definição de um "portal padrão" (default gateway), que é um sistema da rede local para o qual o sistema de origem envia qualquer pacote cujo destinatário não seja um sistema na rede local à qual ele está fisicamente ligado. Este default gateway é, em geral, um roteador, um sistema computacional dedicado ao roteamento de pacotes (router). É ele que tem as ligações físicas externas à rede local e também as regras de roteamento para saber para onde enviar os pacotes, dependendo do endereço de destino.

Entretanto, os detalhes do funcionamento da rede, que só mencionamos aqui como ilustração, não são importantes do ponto de vista do usuário. Tudo que ele precisa saber sobre a rede em si são os nomes e os endereços na rede do sistema onde ele está e do sistema com o qual ele quer se conectar. Este tipo de combinação nome–endereço tem a forma geral

`<hostname>.<network-domain-name>`

onde `<hostname>` é o nome da máquina na rede local à qual ela está ligada diretamente e `<network-domain-name>` é o nome da rede local em si. Por exemplo, em nosso caso no Instituto de Física o servidor central do projeto Sócrates tem o hostname `socrates` e está ligado à rede acadêmica do Instituto, que tem como nome de domínio `if.usp.br`, de forma que o endereço completo da máquina na Internet é `socrates.if.usp.br`

Note que a situação é diferente com os sistemas da sala Pró-Aluno, pois eles estão em uma rede fechada que não é visível diretamente a partir da Internet, ou mesmo a partir do resto da rede local do Instituto. De todos os sistemas desta rede fechada apenas o servidor `socrates` faz parte da rede local do Instituto. Os micro-computadores da sala Pró-Aluno têm endereços completos dados, por exemplo, por

`pop01.if.usp.br`

Neste caso vemos que `pop01` é o hostname do micro e `if.usp.br` é o nome da rede do Instituto. Entretanto, neste caso isto é apenas uma forma de registrar que trata-se da sala Pró-Aluno do Instituto, pois em uma rede fechada o domínio é pura fantasia, talvez uma conveniência, mas não tem nenhuma função estrutural no funcionamento da rede.

Para examinar um segundo exemplo podemos apelar para um dos servidores centrais de nosso vizinho, o Instituto de Matemática e Estatística. Ele tem como hostname `mafalda` e está na rede `ime.usp.br`, de forma que o endereço completo da máquina na Internet é

`mafalda.ime.usp.br`

Como vemos, os servidores `socrates` e `mafalda` estão em redes diferentes, duas sub-redes chamadas `if` e `ime`, da rede da Universidade de São Paulo, cujo domínio é `usp.br`. Como se vê, trata-se de um sistema hierárquico mas, mais uma vez, os detalhes não são relevantes para o usuário. O ponto é que, uma vez resolvidos os problemas técnicos de rede pelos gerentes dos sistemas, as duas máquinas saberão como se comunicar uma com a outra, independentemente dos detalhes de como elas estão ligadas fisicamente ou relacionadas logicamente.

Do ponto de vista da ação direta pelo usuário a comunicação entre as máquinas pode ser de um dos seguintes tipos gerais: transferência de arquivos; execução remota de comandos não-interativos; estabelecimento de sessões interativas remotas; transferência para sistemas remotos de dados gráficos através do sistema de janelas X11, que veremos em uma aula subsequente. Nas tarefas vamos passar por cada um destes tipos de operação de rede. Além disso, a integração de sistemas em rede pode incluir outros tipos de comunicação entre eles como, por exemplo, a exportação de áreas de disco e de impressoras de um para outro e a distribuição através da rede de certos tipos de informação de sistema. Também veremos alguns exemplos disto em nossa rede local aqui no Instituto, em particular na rede fechada da sala Pró-Aluno.

2 Tarefas

A maior parte destas tarefas deve ser realizada nas máquinas da sala Pró-Aluno. Como trata-se de lidar com uma rede, que raramente está disponível em casa, esta será a melhor forma de executar as tarefas. Também é possível realizá-las em outros sistemas a partir dos quais se tenha acesso ao servidor do projeto Sócrates, mas provavelmente será menos confortável e conveniente. Quase todas as tarefas podem ser realizadas nos terminais de caracteres do console, que é como as descrevemos abaixo, mas também é possível realizá-las de dentro do sistema X11, usando dois terminais `xterm` em duas janelas separadas.

1. Em uma das máquinas da sala Pró-Aluno, vá para o primeiro console, digitando `[Ctrl]-[Alt]-[F1]`; faça um login em sua conta e vá para o diretório `/temp`[1]. Em seguida, vá para o segundo console, digitando `[Ctrl]-[Alt]-[F2]`; faça um segundo login em sua conta e, em seguida, faça um login remoto a partir desta sessão, no servidor `socrates`, executando o comando `telnet socrates.if.usp.br` e digitando em seguida, mais uma vez, o seu username (caso seja solicitado) e a sua password. Uma vez no servidor, vá para o diretório `/tmp`.

   Os dois terminais virtuais `tty1` e `tty2` associados às teclas `[F1]` e `[F2]` estão agora rodando duas sessões interativas em duas máquinas diferentes. Enquanto a sessão em `tty1` está usando o CPU e a memória RAM da máquina local em frente à qual você está sentado/a, a sessão em `tty2` está usando o CPU e a memória RAM de uma máquina remota, o servidor `socrates`. Na realidade, neste caso o processamento está sendo dividido entre as duas máquinas: a máquina local está rodando apenas um conjunto de programas que emula um terminal ASCII, enquanto qualquer comando ou aplicação que você rode dentro desta sessão será executado no servidor. Em cada um dos dois casos você está

[1] Se você não estiver fazendo isto em uma máquina da sala Pró-Aluno do Instituto, use o diretório `/tmp` em vez de `/temp`, em todas as tarefas desta aula.

num diretório local da máquina em que está processando, os diretórios /temp e /tmp.

Em cada uma das duas sessões, execute a pipeline de comandos `ps aux | less` para verificar as diferenças no estado de processamento das duas máquinas. Execute também os comandos `hostname`, `uname -a`, `uptime` e `w` em cada uma das duas sessões.

2. Na sessão do `socrates`, crie um arquivo na área `/tmp`, com o seu username no nome para caracterizá-lo como seu, usando o comando

   ```
   dd if=/dev/zero of=<username>.zeros bs=1024 count=100
   ```

   O programa `dd` copia blocos de dados de um dispositivo ou arquivo para outro, as opções `bs` e `count` determinam o tamanho e o número de blocos respectivamente. Isto irá criar um arquivo com o nome indicado, contendo 100 Kbytes de zeros binários. Verifique este fato usando o comando `ls -l`. Vá agora para a sessão do terminal `tty1` e use o comando `ls -l` lá, para verificar que ainda não existe um arquivo com este nome no diretório `/temp` na máquina local.

3. Vamos agora transferir o arquivo `<username>.zeros` de uma máquina para a outra através da rede, usando o sistema FTP (File Transfer Protocol). O comando `ftp` é um programa geral de transferência de arquivos, que pode ser usado entre quaisquer dois sistemas computacionais ligados à Internet. Ele implementa o protocolo FTP de transferência de arquivos de forma geral, sem usar propriedades específicas deste ou daquele sistema operacional.

   Para fazer isto, execute em sua sessão local, no terminal `tty1`, o comando `ftp socrates.if.usp.br`. O programa irá estabelecer uma ligação de rede com o `socrates` e em seguida solicitar a você informação de login no sistema remoto. Observe que, como o seu username é o mesmo nos dois sistemas, não será de fato necessário digitar o seu username, basta digitar `[Enter]` que o programa usará por default o seu username local. Digite em seguida a sua password no sistema remoto (que também é a mesma, mas desta vez será necessário digitar). Caso você erre a password, aparecerá uma mensagem de que o login falhou, neste caso saia do `ftp` digitando `[Ctrl]-D` e inicie o processo de novo.

4. Uma vez feito com sucesso o login no sistema remoto, você está numa sessão FTP naquele sistema. Esta á uma espécie de sessão interati-

va muito limitada, na qual existem apenas alguns poucos comandos disponíveis, o suficiente para que você possa transferir arquivos. Basicamente, você pode usar os comandos `ls` ou `dir` para listar arquivos, o comando `cd` para mudar de diretório e os comandos `get` e `put` para pegar e colocar arquivos, respectivamente. Use o comando `ls` dentro da sessão FTP para verificar que você está na raiz de sua conta no sistema remoto. Em seguida, use `cd /tmp` para ir para o diretório `/tmp` do sistema remoto e em seguida o comando `ls` mais uma vez. Deve aparecer lá o arquivo `<username>.zeros` que você criou.

Antes de pegar o arquivo, execute o comando `hash` dentro da sessão FTP. Este comando fará com que o programa reporte o progresso da transferência colocando na tela um símbolo `#` para cada bloco de 1024 bytes de dados que forem transferidos. Em seguida, pegue o arquivo do sistema remoto usando o comando `get <username>.zeros` e observe o que acontece. Verifique a velocidade com que o programa reporta que fez a transferência. Baseando-se nisto, dá para você estimar qual é a velocidade da rede que interliga as duas máquinas? Em seguida, saia do programa `ftp` usando o comando `exit` e use o comando `ls -l` em sua sessão local para verificar que o arquivo foi, de fato, transferido para o diretório `/temp` da máquina local.

5. Vamos agora criar um segundo arquivo na máquina local que você está usando, para depois transferí-lo para a máquina remota. Para criar um arquivo com 100 Kbytes de caracteres `0` execute no terminal `tty1` a sequência de comandos que segue, substituindo o seu username onde indicado.

```
popXX:/temp> cat /dev/null >! <username>.chars
popXX:/temp> set ic = 0
popXX:/temp> while ( $ic < 10240 )
while? @ ic = $ic + 1
while? echo -n 0000000000 >> <username>.chars
while? end
```

Trata-se aqui de um pequeno exemplo de programação de linha de comando. O primeiro comando cria um arquivo vazio contendo o seu username no nome. O segundo comando inicializa um contador. O comando while executa o bloco de comandos que segue, até o comando end, enquanto a condição indicada for satisfeita. O caracter @ é o operador aritmético, ele indica que nesta linha a variável ic está sendo manipulada segundo a aritmética dos números inteiros. Verifique o resultado da operação usando o comando ls -l.

Entre agora no programa ftp da mesma forma como fez antes, uma vez dentro dele vá para o diretório /tmp do servidor, execute o comando hash e transfira este arquivo para o servidor usando o comando put <username>.chars. Em seguida use o comando ls dentro do programa ftp para verificar que o arquivo foi de fato transferido.

6. Para ter uma idéia de quanta informação pode estar disponível para seu uso através do protocolo FTP, use o comando ftp para entrar nos sistemas fma.if.usp.br ou linorg.usp.br ou, no caso destes não estarem disponíveis, nos sistemas ftp.br.debian.org, ftp.debian.org ou ftp.kernel.org. Trata-se de servidores de distribuição de free software que disponibilizam serviços de FTP anônimo a qualquer usuário da Internet, ou seja, você não precisa ter contas naquelas máquinas para poder puxar delas por FTP quaisquer quantidades de free software que deseje. Em particular, todas as versões do kernel do Linux e toda a distribuição Debian estão disponíveis nos servidores mencionados acima, entre muitas outras coisas, para que você possa pegar, instalar em casa, etc. Para usar este serviço, execute o comando

   ```
   ftp fma.if.usp.br
   ```

   e, quando aparecer o prompt para o username, digite anonymous ou simplesmente ftp. Em seguida, digite o seu endereço de e-mail no lugar da password. Pronto, você estará conectado/a ao servidor, use o comando cd para passear à vontade por lá e o comando get para pegar o que quiser. Há nestes servidores centenas de GB de free software disponível, na maior parte na forma de arquivos comprimidos.

7. Existe uma outra forma de se copiar arquivos de um sistema para outro que é mais característica do sistema Linux. Trata-se do comando rcp, que só é útil entre dois sistemas Linux ou Unix entre os quais existam as estruturas de autorização apropriadas. Como este é o caso entre o servidor socrates e as máquinas da sala Pró-Aluno, você deve ser capaz de usar este comando entre o socrates e uma das máquinas da sala ou entre duas das máquinas da sala.

Para verificar o fato de que estas máquinas são todas "trusted hosts" (sistema mutuamente conhecidos e autorizados) uns dos outros, examine em cada sistema, usando um paginador ou o seu editor preferido, os arquivos `/etc/hosts.equiv` e `/etc/netgroup`, que contêm um grupo de rede chamado `rcmaccess` que consiste de uma lista dos hosts autorizados. Estas são as autorizações globais destes sistemas, mas você também pode definir uma estrutura de autorizações específica para a sua conta. Para autorizar este tipo de conexão para a sua conta, crie na raiz dela um arquivo de nome `.rhosts`, ou modifique um que já esteja lá, colocando nele um conteúdo como o que segue, ou seja, diretamente uma lista de hosts que você deseja autorizar para acesso à sua conta.

```
# JLdL 02Jun02.
# Base de dados de acesso transparente.
#
socrates.if.usp.br      <username>
pops.if.usp.br          <username>
poph.if.usp.br          <username>
pop01.if.usp.br         <username>
<...etc...>             <...etc...>
pop28.if.usp.br         <username>
```

Observe que é importante incluir o seu username na segunda coluna, bem como o nome completo das máquinas. Também é uma boa idéia deixar este tipo de arquivo legível apenas por você mesmo.

8. Tendo estabelecido as necessárias estruturas de autorização, em sua sessão do terminal `tty1` tente copiar mais uma vez para o `socrates` um dos arquivos que estão na máquina local, executando o comando:

   ```
   rcp <file>.zeros socrates.if.usp.br:/tmp/<file>.zeros.2
   ```

   Tente também uma alternativa contendo explicitamente o seu username no sistema `socrates` como parte do endereço:

   ```
   rcp <file>.chars <username>@socrates:/tmp/<file>.chars.2
   ```

   Finalmente, tente uma alternativa com dois arquivos e um diretório:

   ```
   rcp <username>.zeros <username>.chars socrates.if.usp.br:
   ```

   Neste caso, descubra onde foram parar os dois arquivos transferidos.

9. Repita as operações do ítem anterior usando o comando `scp` em vez de `rcp`. Trata-se de um comando que faz o mesmo tipo de trabalho, mas encripta todo o fluxo de dados enquanto ele trafega pela rede, de forma a garantir a inviolabilidade do conteúdo. Este comando é parte do sistema `ssh` que veremos logo adiante. Ele usa outras formas de autorização, não necessariamente o arquivo `.rhosts`. Ele se comporta de forma diferente do `rcp` no caso de não existirem as autorizações corretas entre os dois sistemas, em vez de falhar ele pede a password do usuário.

10. O comando `rcp` é parecido com um comando mais geral que nos permite executar comandos em uma máquina remota, sendo entretanto um pouco mais simples. O comando geral é a `rsh` ou "remote shell". Assim como a shell nos permite executar comandos no sistema, a shell remota nos permite executar comandos em sistemas remotos. As limitações de funcionamento, em termos do acesso ao sistema remoto, são as mesmas que para o comando `rcp`, ou seja, os dois sistemas envolvidos devem se conhecer mutuamente e devem existir entre eles as relações apropriadas de autorização. Além disso, há a limitação de que o programa a ser executado remotamente não seja um aplicativo do tipo que estabelece uma sessão interativa com o usuário. É necessário que ele seja um comando que faz alguma coisa e termina. A partir de seu terminal `tty1` descubra que usuários estão usando o `socrates` neste momento executando o comando

    ```
    rsh socrates.if.usp.br who
    ```

    Verifique a informação obtida executando o comando `who` diretamente no `socrates`, em seu terminal `tty2`. Tente fazer o mesmo com os comandos `df`, `ls` e `uptime`. No caso do `ls`, observe bem quais são os arquivos que aparecem. Se necessário, use `ls -al`.

11. Repita as operações do ítem anterior usando o comando `ssh`, a "secure shell" ou shell segura, em vez de `rsh`. Trata-se de um comando que pode fazer o mesmo tipo de trabalho que o `rsh`, entre outras coisas, mas encripta todo o fluxo de dados enquanto ele trafega pela rede, de forma a garantir a inviolabilidade do conteúdo. Ele pode usar várias outras formas de autorização, que não necessariamente o arquivo `.rhosts`. Ele se comporta de forma diferente do `rsh` no caso de não existirem as autorizações corretas entre os dois sistemas, em vez de falhar ele pede a

password do usuário. Esta nova shell remota tem também várias outras vantagens. Hoje em dia é esta a shell que usamos como default para todas as conexões dentro de nossas redes locais.

12. Já vimos, logo no início da aula, o uso do comando `telnet` para estabelecer uma sessão interativa em um sistema remoto. Assim como é o caso do comando `ftp`, trata-se de um comando de caráter bem geral, que não usa as especificidades do sistema Linux, nem depende delas. No máximo, quando ele é usado entre dois sistemas Linux apropriadamente configurados, ocorrem algumas mudanças de comportamento como, por exemplo, ele deixar de solicitar o seu username ou entrar em um modo no qual todo o tráfego de dados pela rede é encriptado.

    Existe um segundo comando que serve para este tipo de função, o qual é mais específico dos sistema Linux ou Unix. Trata-se do comando `rlogin`, que permite que façamos um login em um sistema remoto. Entre sistemas que se conhecem e entre os quais há as relações apropriadas de autorização, de forma semelhante ao que acontece no caso dos comandos `rcp` e `rsh`, é possível fazer o login no sistema remoto sem a necessidade de se digitar o username e a password. Desta forma podemos passar de forma mais transparente de um sistema para o outro dentro de um "cluster" de sistemas que estejam integrados em rede. Experimente este comando, por exemplo executando-o em seu terminal `tty1` da forma

    ```
    rlogin socrates.if.usp.br
    ```

    Verifique que você está de fato no `socrates` e em seguida saia de novo, usando o comando `exit` para terminar a shell que foi iniciada no login remoto.

13. Repita as operações do ítem anterior usando o comando `ssh` em vez de `rlogin`. A `ssh` pode fazer o mesmo tipo de trabalho que o `rlogin`, comportando-se como a `rsh` ou como o `rlogin` dependendo do número de argumentos com que a utilizamos. Na realidade a shell `rsh` também faz isto, de forma que o comando `rlogin` é de certa forma um nome redundante. A forma de se configurar a `ssh` para permitir o login sem password é um pouco mais complexa e a deixaremos para os problemas ao final da apostila. Mas a falta das autorizações apropriadas entre os sistemas não chega a representar um grande problema no caso da `ssh`, pois ela sempre pede password quando isto acontece.

14. Como mais uma forma de fazer a mesma coisa, digite simplesmente o nome do servidor `socrates` no terminal. Verifique que, mais uma vez, você acaba de fazer um login remoto no sistema `socrates`. Ou seja, esta é uma forma ainda mais transparente de ir de uma máquina para a outra dentro do "cluster" didático do Instituto. Para descobrir como isto aconteceu, localize o comando `socrates` usando o comando `which`, basta digitar `which socrates`. Uma vez que achar o diretório onde está o comando, execute no arquivo o comando `ls -l` para descobrir que tipo de coisa ele é. Sendo necessário, repita o processo até achar qual é o arquivo executável, contendo um comando, que está envolvido.

    A explicação do que acontece é uma propriedade peculiar do comando `ssh` (que também é válida para a `rsh`). Em geral o `ssh` deve ser chamado com dois argumentos, o nome do host no qual queremos executar o comando e o nome do comando a ser executado. Se dermos apenas um argumento, ele será entendido como o nome de um host no qual se deve fazer um login. Por outro lado, se chamarmos o comando `ssh` sem argumentos, mas dando a ele um nome diferente, ele tenta executar um login em um host cujo hostname seja o nome com o qual o comando foi chamado. Chamar um programa através de um link é uma forma de executá-lo com um nome diferente. Links para `ssh` são a forma universal segundo a qual permitimos o tráfego de usuários entre as máquinas de um cluster de sistemas que estão integrados em rede, formando um único sistema global do ponto de vista do usuário.

15. Existem sistemas na rede que estamos usando o tempo todo mas com os quais não interagimos de forma tão direta e explícita. Um deles é o sistema NFS (Network File System), através do qual a sua conta está visível a partir de qualquer das máquinas da sala Pró-Aluno, como se fosse uma área de disco local em cada uma delas.

    Sem este sistema você não teria um "home", ou teria um "home" diferente em cada um dos sistemas e, toda vez que fosse usar uma das máquinas, teria de transferir todos os seus arquivos para ela, usando disquetes, ou FTP a partir da última máquina em que tenha trabalhado anteriormente. Esperançosamente, você ainda se lembra de onde foi que trabalhou da última vez, não perdeu nem danificou os seus disquetes e, se aquela máquina está agora sendo usada por alguém para rodar o Windows, bem,... obviamente este esquema não seria uma idéia muito boa.

O sistema NFS não é uma forma de copiar arquivos entre duas máquinas e sim uma forma de compartilhar o mesmo arquivo entre duas ou mais máquinas. Para verificar isto, em seu terminal `tty1` volte ao sistema local, executando o comando `exit`, caso ainda esteja numa sessão remota no `socrates`. Execute um `cd` sem argumentos, o que o levará para casa (para o seu "home"). Faça o mesmo no terminal `tty2`, onde você ainda está numa sessão remota no servidor `socrates`. Voltando ao terminal `tty1` crie com o comando `touch` um arquivo chamado `trash.nfs` na raiz de sua conta. Verifique o resultado com o comando `ls`. Em seguida vá ao terminal `tty2` e use o comando `ls` para verificar que o arquivo que você acaba de criar está disponível no servidor. Não se trata de uma cópia, para verificar isto, ainda no terminal `tty2` remova o arquivo com o comando `rm trash.nfs`. Em seguida volte ao terminal `tty1` e verifique que o arquivo desapareceu de lá também.

16. O sistema NFS não é usado apenas para as áreas de usuários. Nos sistemas da sala Pró-Aluno usamos este sistema também para distribuir arquivos de sistema entre as máquinas. Execute o comando `df` no terminal `tty2` para verificar que no servidor `socrates` o disco contendo as contas dos usuários é um disco local na máquina, montado no diretório `/home` e que corresponde a algum dispositivo em `/dev`. Depois execute o mesmo comando no terminal `tty1` para verificar que no sistema local o que está montado no diretório `/home` é um filesystem exportado pelo `socrates` através da rede, com o uso do nome `poph`, pelo qual o `socrates` responde dentro da rede fechada.

    Verifique também que o que está montado no diretório `/usr` no sistema local é um disco exportado pelo servidor `socrates`, que no momento está dublando como o servidor Linux local da sala, usando desta vez o nome `pops` dentro da rede fechada. É dentro deste diretório que fica a maior parte dos programas do sistema. Observe que, de todos os filesystems de sistema (`/`, `/tmp`, `/var` e `/usr`), este último é de longe o maior. Considere a economia de espaço em disco que estamos fazendo ao usar este mesmo sistema de arquivos contendo programas para todas as 28 máquinas da sala. Os discos de cada máquina têm cerca de 10 GB cada um, que estão sendo utilizados na maior parte pelo Windows e pelo diretório temporário local `/temp`. Sem este tipo de esquema, não teria sido possível instalar em todas as máquinas um sistema Linux tão completo quanto o que temos e ainda dispor das amplas áreas temporárias locais `/temp` nos terminais.

17. Outro sistema muito importante na rede é o DNS (Domain Name Service), que serve para mapear endereços simbólicos em endereços numéricos e vice-versa. Isto é necessário porque os nomes simbólicos tais como `socrates.if.usp.br` existem apenas para nosso benefício como usuários da rede. Todo o endereçamento do tráfego de pacotes que acontece na rede é feito, na realidade, usando os correspondentes endereços numéricos. Assim, toda vez que nos referimos a alguma máquina, como por exemplo `ftp.debian.org`, o sistema precisa descobrir a que endereço numérico isto corresponde, para poder construir os pacotes com o endereçamento correto e enviá-los ao sistema de destino.

    Para descobrir quais são os endereços numéricos dos sistemas, você pode usar o comando `host <hostname>.<network-domain-name>`, tanto no `socrates` quanto nas máquinas da sala Pró-Aluno. Entretanto, note que as máquinas da sala em si estão em uma rede fechada e não estão cadastradas no sistema DNS da Internet, de forma que não vai funcionar usar este comando para as máquinas da sala. Tente executar o comando para várias máquinas do Instituto, tais como `socrates.if.usp.br`, `euclides.if.usp.br`, `latt.if.usp.br`, `gancho.if.usp.br`, `mestre.if.usp.br` e `servo.if.usp.br`. Depois disto, tente algumas outras máquinas da USP como, por exemplo, `spider.usp.br`, `linusp.usp.br` e `linorg.usp.br`. Finalmente, tente algumas máquinas de fora do país como, por exemplo, `ftp.debian.org` e `ftp.kernel.org`.

18. Outro sistema de informação importante para a rede é o sistema NIS (Network Information Service). Trata-se de um sistema de abrangência bem mais localizada, mas que distribui informações de caráter mais variado e mais específico de um sistema integrado de máquinas como o nosso cluster didático. Ele também pode ser utilizado para distribuir informação sobre os endereços de máquinas, mas é mais comum usá-lo para distribuir as informações sobre os usuários existentes nos sistemas, tais como seus usernames e as respectivas passwords. Na rede fechada da sala Pró-Aluno nós o utilizamos com estas duas funções. Sem ele, seria necessário cadastrar cada usuário e cada host da rede fechada

em cada uma das máquinas, o que obviamente significaria uma grande quantidade de trabalho repetitivo na administração dos sistemas.

Todos os usuários precisam estar cadastrados em algum lugar, que em geral é o servidor central (master server) do sistema NIS. Em nosso caso este sistema é (adivinhe só!) o servidor `socrates`. Para ver o cadastro de todos os usuários dos sistemas didáticos examine, em seu terminal `tty2`, o arquivo `/etc/passwd`, usando o seu paginador ou editor favorito. Como você vê, há nele várias colunas de dados separadas pelo caracter "`:`", que mapeiam o username do usuário em seu número de identificação no sistema, seu home, sua shell de login, etc. A única coisa que notoriamente está faltando é, curiosamente, a password! Em tempos passados ela costumava de fato ser mantida neste arquivo, em forma encriptada, mas hoje em dia a mantemos em outro, por questões de segurança: o arquivo `/etc/shadow`, que não é legível pelos usuários.

19. Assim como acontece no caso da rede, toda a operação do sistema é baseada no número de identificação do usuário e não no seu username, que só existe para conforto mental dos usuários mais humanos. Assim, toda vez que algo se refere a um usuário pelo username, o sistema tem de consultar o arquivo `/etc/passwd` para descobrir o respectivo número (o uid). Entretanto, pode-se verificar que esta informação sobre a maior parte dos usuários não está disponível nos arquivos `/etc/passwd` de cada um dos sistemas da sala Pró-Aluno. Para verificar este fato, em seu terminal `tty1` dê uma olhada no arquivo `/etc/passwd` do sistema local que você está usando. Você encontrará lá apenas dois usuários: o super-user `root` e uma conta auxiliar chamada `help`, todas as demais estão faltando.

20. Esta informação está sendo distribuída para as máquinas da sala através do sistema NIS. Você pode obter esta informação usando comandos do sistema NIS. Para começar, "descubra" qual é o servidor NIS que está servindo o seu sistema local usando, no terminal `tty1`, o comando `ypwhich`. Descubra também qual é o domínio NIS ao qual ele pertence, usando o comando `domainname`. Para obter a informação sobre os usuários, execute o comando `ypcat passwd`. Como a informação vem fora de ordem, para ver algo mais ou menos como está no arquivo do `socrates` é melhor usar a pipeline `ypcat passwd | sort | less`, na qual o comando `sort` coloca as linhas em ordem alfabética e o pagi-

nador permite que você olhe o resultado com calma. No `socrates` os usuários estão organizados no arquivo `/etc/passwd` por ordem de uid e não por ordem alfabética de usernames. Para colocar a lista nesta ordem de `uid` use a pipeline

```
ypcat passwd | sort -n -t :  +2 | less
```

Olhe as páginas de manual do comando `sort` para entender o papel de cada opção ou argumento.

21. Uma forma útil de descobrir informação sobre os usuário que existem em um sistema remoto é o comando `finger`. Este comando dá informação, não apenas sobre que usuários estão usando o sistema naquele momento, mas sobre usuários que existem cadastrados naquele sistema. Devido ao fato de que algumas vezes este tipo de informação pode ser utilizado para fins ilícitos, este comando muitas vezes está desabilitado ou fornece apenas tipos e quantidades muito limitadas de informação. Para verificar qual a informação que está disponível sobre a sua própria conta a partir do `socrates`, execute no seu terminal `tty1` o comando

    ```
    finger <username>@socrates.if.usp.br
    ```

    e, para obter alguma informação sobre quem está no sistema no momento, o comando `finger @socrates`. Tente também usar o finger com o username de alguns outros usuários do sistema.

22. Uma forma simples e eficiente de comunicação em tempo real entre usuários do sistema, ou de dois sistemas diferentes, é provido pelo programa `talk`. Para se comunicar com um usuário que esteja usando um determinado sistema remoto, basta executar o comando

    ```
    talk <username>@<hostname>.<network-domain-name>
    ```

    Se estiver de fato logado no sistema remoto, o usuário receberá uma mensagem o instruindo a responder com um comando do mesmo tipo. Entretanto, é possível que a sessão do usuário remoto não esteja permitindo conexões deste tipo. Cada usuário pode controlar se quer ou não receber este tipo de conexão com o comando `mesg`. Executando-se `mesg y` autoriza-se as conexões, com `mesg n` elas são desabilitadas. Sem argumentos, o comando mostra a situação corrente.

    Se a conexão for realizada com sucesso, a tela de seu terminal se dividirá em duas, o que você escrever aparecerá numa delas e o que o seu interlocutor digitar aparecerá na outra. Caso os dois usuários estejam logados na mesma máquina, é suficiente colocar apenas o username, omitindo os nomes da máquina e do domínio. Este seria o caso, por exemplo, de dois alunos que estejam logados no `socrates` a partir de máquinas da sala Pró-Aluno e que usem o comando dentro do servidor.

Tente usar este comando para falar com algum colega que também esteja usando o Linux na sala Pró-Aluno. Faça o mesmo dentro do `socrates`, estabelecendo uma conversação com algum outro usuário que esteja logado no servidor. Existe também um comando `ytalk`, com a mesma função, que pode ser utilizado dentro do sistema X11, no qual ele abre uma janela própria. Tente usar este comando também.

23. Existem outros comandos e sistemas que distribuem informação pela rede. Na maior parte das vezes, estes sistemas envolvem dois elementos: um comando que o usuário usa para acessar a informação e um programa que roda continuamente em cada máquina e que fornece esta informação quando é feita uma solicitação. Estes programas que rodam de forma constante são chamados de "daemons", a maior parte dos comandos que examinamos acima dependem, para a sua operação, de estarem rodando os daemons adequados nas máquinas com as quais queremos nos conectar. Por exemplo, os comandos `telnet`, `rsh`, `rlogin` e `ssh` dependem de estarem ativos os daemons `telnetd`, `rshd`, `rlogind` e `sshd`. Em alguns sistemas alguns destes daemons podem não estar habilitados e os correspondentes serviços podem não esta funcionando.

    Em nossos sistemas mantemos ativo um outro daemon, chamado `rwhod`, que permite a operação de mais dois comandos de distribuição de informação pela rede. Trata-se dos comandos `rwho` e `ruptime`, que são versões remotas dos comandos `who` e `uptime`. O primeiro dá informação sobre os usuários que estão usando os sistemas que estão rodando os daemons, o segundo dá informação sobre os próprios sistemas como, por exemplo, há quanto tempo estão funcionando desde o último boot e qual a carga em cada um.

    Experimente usar os comandos `who`, `uptime`, `rwho` e `ruptime` tanto no sistema local (terminal `tty1`) quanto no servidor remoto (terminal `tty2`). Observe que estes comandos de rede atuam apenas sobre as máquinas ligadas à mesma rede física e configuradas na mesma rede lógica, de forma que os resultados dos comandos de rede serão muito diferentes no `socrates` e nas máquinas da sala Pró-Aluno. Estes comandos e daemons atuam por meio de “broadcast” e só são efetivos dentro de uma rede local configurada logicamente como um domínio de broadcast. Usando pipelines como os comandos `grep`, `sort`, etc, você poderá filtrar e manipular a informação obtida para os mais diversos fins. O comando `ruptime` tem também algumas opções úteis de ordenamento, dê uma olhada nas suas páginas de manual.

3 Problemas e Dicas

1. Dois programas que ilustram de forma interessante o funcionamento da rede são o `ping` e o `traceroute`. O primeiro é um programa muito simples que testa a conexão até um dado host de forma parecida com o funcionamento de um sonar submarino, de onde vem o seu nome. Tente usar este comando para verificar a disponibilidade de algum host remoto como, por exemplo, em `ping www.debian.org` ou `ping ftp.kernel.org`. Observe o resultado do comando e tente interpretar o seu significado. Em seguida execute os comandos `traceroute www.debian.org` ou `traceroute ftp.kernel.org`. Este programa determina e mostra na tela a rota que está atualmente funcionando até o host remoto. Mais uma vez, tente interpretar o significado do output que aparece. Uma olhada nas páginas de manual dos dois comandos pode ser muito útil.

2. Leia as páginas de manual do comando `ssh` para descobrir como configurar este comando de tal forma que ele deixe você entrar na máquina remota sem que você tenha de digitar a sua password. Os arquivos envolvidos nisto vão estar todos dentro de um diretório chamado `.ssh` na raiz da sua conta. Procure em particular por referências aos arquivos `known_hosts` e `authorized_keys`. O comando `ssh-keygen` será importante pois é com ele que você gera as suas chaves pessoais de encriptação. O comando `ssh-copy-id` também é importante para facilitar o estabelecimento das autorizações entre hosts. Olhe as páginas de manual destes comandos.

3. Entre no `socrates` e use o comando `ruptime` para descobrir quais são as máquinas mais carregadas do Instituto, entre aquelas que rodam o

daemon `rwhod`. Em seguida, descubra como usar o comando `rwho` para obter o número de usuários diferentes que estão usando algum sistema no Instituto. Você vai ter de usar pipelines e os comandos `sort`, `grep`, `uniq`, `cut` e `wc` devem ser úteis.

4. Você também pode puxar software dos servidores `fma.if.usp.br` ou `ftp.debian.org` usando um "browser" como o `netscape` ou o `mozilla`. Denominamos o servidor `fma.if.usp.br` de "espelho", pois o que ele faz é copiar e re-distribuir software a partir de vários sites da Internet, usando para isto um programa chamado `wget`. Já o servidor `ftp.debian.org` é um site original, do qual o servidor `fma.if.usp.br` espelha parte do conteúdo. Descubra como usar o `netscape` ou o `mozilla` para acessar o espelho e puxar dele alguns pacotes de software. Considere usar como endereços, dentro do browser, as possibilidades

   `http://fma.if.usp.br` e `ftp://ftp.debian.org`.

   Você poderá navegar pela estrutura dos filesystems do espelho clicando em links que vão aparecer no seu browser. Para puxar arquivos por FTP ou HTTP e salvá-los em disco, em vez de mostrá-los na tela, clique nos links correspondentes enquanto mantém apertada a tecla `[Shift]`. Esta é uma forma muito conveniente de navegar pelo espelho e puxar free software.

   Outra forma muito útil de puxar arquivos através da rede pelos protocolos HTTP e FTP, em especial em conexões lentas e que caem muito, é o uso do programa `wget`, que é capaz de retomar as conexões pelo protocolo FTP do ponto onde elas pararam devido à interrupção. Tente usar este comando, para saber como, dê uma boa olhada no seu manual.

4 Uma Lista de Comandos Relevantes

`finger <username>@<hostname>.<network-domain-name>` : mostra alguma informação sobre o usuário no sistema remoto, independentemente dele estar ou não usando o sistema neste momento.

`ftp <hostname>.<network-domain-name>` : estabelece uma sessão interativa com capacidades limitadas numa máquina remota, através da qual se pode transferir arquivos entre os dois sistemas.

host <hostname>.<network-domain-name> : retorna o endereço numérico da Internet que corresponde ao endereço simbólico digitado como argumento.

hostname : retorna o nome na rede da máquina onde se está.

mesg [y|n] : habilita (y) ou desabilita (n) conexões de sistemas de mensagens com o seu terminal; sem argumentos, mostra a situação corrente.

rcp <argumento1> <argumento2> : copia arquivos de e para uma máquina remota, desde que a máquina local e a máquina remota tenham entre si as necessárias relações de autorização; os argumentos têm a forma geral

<username>@<hostname>.<network-domain-name>:<path>/<file>

rlogin <hostname>.<network-domain-name> : estabelece uma sessão interativa numa máquina remota; solicita password se necessário.

rsh <hostname>.<network-domain-name> <comando> : executa um comando numa máquina remota, desde que a máquina local e a máquina remota tenham entre si as necessárias relações de autorização.

ruptime : mostra o estado do sistema das máquinas da rede local.

rwho : mostra os usuários correntemente logados nas máquinas da rede local.

scp <argumento1> <argumento2> : copia arquivos de e para uma máquina remota, encriptando o tráfego de dados entre as duas; solicita password se necessário; é parte integrante do sistema ssh; os argumentos têm a forma geral

<username>@<hostname>.<network-domain-name>:<path>/<file>

ssh <hostname>.<network-domain-name> [<comando>] : estabelece uma sessão interativa numa máquina remota, encriptando todo o tráfego que passa pela rede; solicita password se necessário; também pode executar um comando numa máquina remota.

talk <username>@<hostname>.<network-domain-name> : estabelece uma sessão de conversação escrita (digitada) com um usuário que esteja logado em uma máquina remota.

telnet <hostname>.<network-domain-name> : estabelece uma sessão interativa numa máquina remota; solicita password e username, se necessário.

`uptime` : mostra o estado do sistema local.

`w` : mostra quem está logado no sistema.

`who` : mostra os usuários do sistema local.

`ytalk <username>@<hostname>.<network-domain-name>` : estabelece uma sessão de conversação escrita (digitada) com um usuário que esteja logado em uma máquina remota, criando uma janela própria no sistema de janelas X11.

5 Arquivos de Configuração

Os arquivos de configuração que são específicos do ambiente de rede já foram mencionados no texto. São os arquivos de controle de acesso na raiz de sua conta, o arquivo `.rlogin` e os arquivos dentro do diretório `.ssh/`.