

Introdução

Hoje em dia os computadores tem muito poder de processamento, porém os aplicativos não utilizam este processamento de forma contínua e sim em picos de carga.

- Isto é verdade para a maioria das pessoas: navegar pela internet, editar textos e planilhas consomem muito pouco CPU
- Mas existem casos que não se adaptam bem: programadores e jogadores utilizam CPU, memória e disco de forma mais contínua.

Introdução (2)

Os aplicativos estão cada vez maiores e conseqüentemente ocupam mais memória

- A maior parte da memória ocupada é com o código do programa, que é estático.
- A memória estática pode ser compartilhada entre as várias instâncias do aplicativo, mesmo que executadas por usuários distintos do sistema.

Introdução (3)

Atualização e manutenção são problemas freqüentes:

- se feitos de forma manual e individual, pode exigir vários profissionais para realizar a tarefa.
- se feitos de forma distribuída, precisa de maior tecnologia e conhecimento por parte dos mantenedores.
- o ideal seria atualizar em um lugar e todo o resto já estar atualizado.

Idéia: Terminal Server

Se pudéssemos usar um servidor de aplicativos, conseguiríamos:

- **melhor aproveitar o CPU:** vários usuários executariam vários processos que consomem pouco CPU, aumentando o uso total.
- **melhor aproveitar a memória:** vários usuários executando um mesmo programa compartilhariam a memória estática requerida.
- **manutenção centralizada:** a atualização no servidor de aplicativos afeta todos os usuários do servidor.

Solução: XFree86/Xorg

- Usar um servidor de terminais: soluções existem há muito tempo, dentre elas o XFree86/Xorg.
- De fato, há muito tempo atrás a moda era usar este tipo de solução em empresas e só terminou quando chegaram os fabricantes com a política “1 computador por pessoa”.

Nesta solução o aplicativo consome os recursos (CPU, memória, disco) do servidor de aplicativos e é mostrado no cliente/terminal.

CUIDADO: Servidor e Cliente X

Um cuidado que temos que tomar é com a nomenclatura usada no X:

- o **Servidor de Janelas (X)** é executado no **cliente**, onde se encontra a placa de vídeo.
- os aplicativos que executam no servidor de aplicativos e irão mostrar janelas são os **clientes** do servidor de janelas.

Servidor de Janelas (X)



saída gráfica
mostra texto digitado
em uma planilha



entrada de mouse
clique no botão "Salvar"



saída gráfica
mostra resultado gráfico
da ação salvar emitida pelo
servidor de aplicativos

Servidor de Aplicativos



+



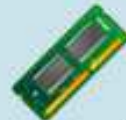
+



processa, grava entrada de
teclado e emite comandos
para saída gráfica



+



+



processa entrada, verifica
ação, salva o arquivo e
emite saída gráfica.

Os Terminais

Dado um servidor de aplicações com aplicativos clientes do X, você pode usar qualquer servidor de janelas compatível com este protocolo para servir de terminal:

- Windows com CygWin/X11 ou outra implementação
- MacOS-X com X11
- Linux, *BSD, ...

Terminais Burros

Para complementar a solução seria ideal que os terminais tivessem o mínimo de elementos possíveis:

- sem disco rígido: esquenta, barulho, consumo de energia, quebra.
- sem disquete: quebra, barulho, ocupa espaço.
- poucos aplicativos: consome memória e CPU

Solução: **Terminais Burros**

Terminais Burros (2)

Em geral tem:

- processador fraco: vai rodar apenas o kernel, o X11 e poucos aplicativos de auxílio.
- pouca memória: cerca de 16Mb de RAM, varia dependendo dos recursos gráficos desejados.
- sem disco: consegue o sistema operacional via rede (TFTP, NFS).
- ambiente de software minimalista: shell, aplicativos básicos, X11.

Boot via rede

Ao ligar o computador, a BIOS procura por um sistema em vários locais:

- primeiros setores dos discos (HD, disquete, USB storage, ...)
- placas de rede com suporte a boot via eeprom.
- nas placas mais novas, a própria bios usa a placa de rede para fazer o processo de boot via rede.

Boot via rede (2)

O processo de *boot* via rede utiliza-se de:

- BOOTP (RFC951 - *Bootstrap Protocol*): o cliente envia um pacote contendo o endereço físico da placa de rede (*MAC Address*) pedindo informações sobre o IP e onde conseguir o kernel. A placa de rede é configurada.
- TFTP (RFC1350 - *Trivial File Transfer Protocol*): o cliente pede o kernel e o recebe, armazenando em memória. Posteriormente este será iniciado.

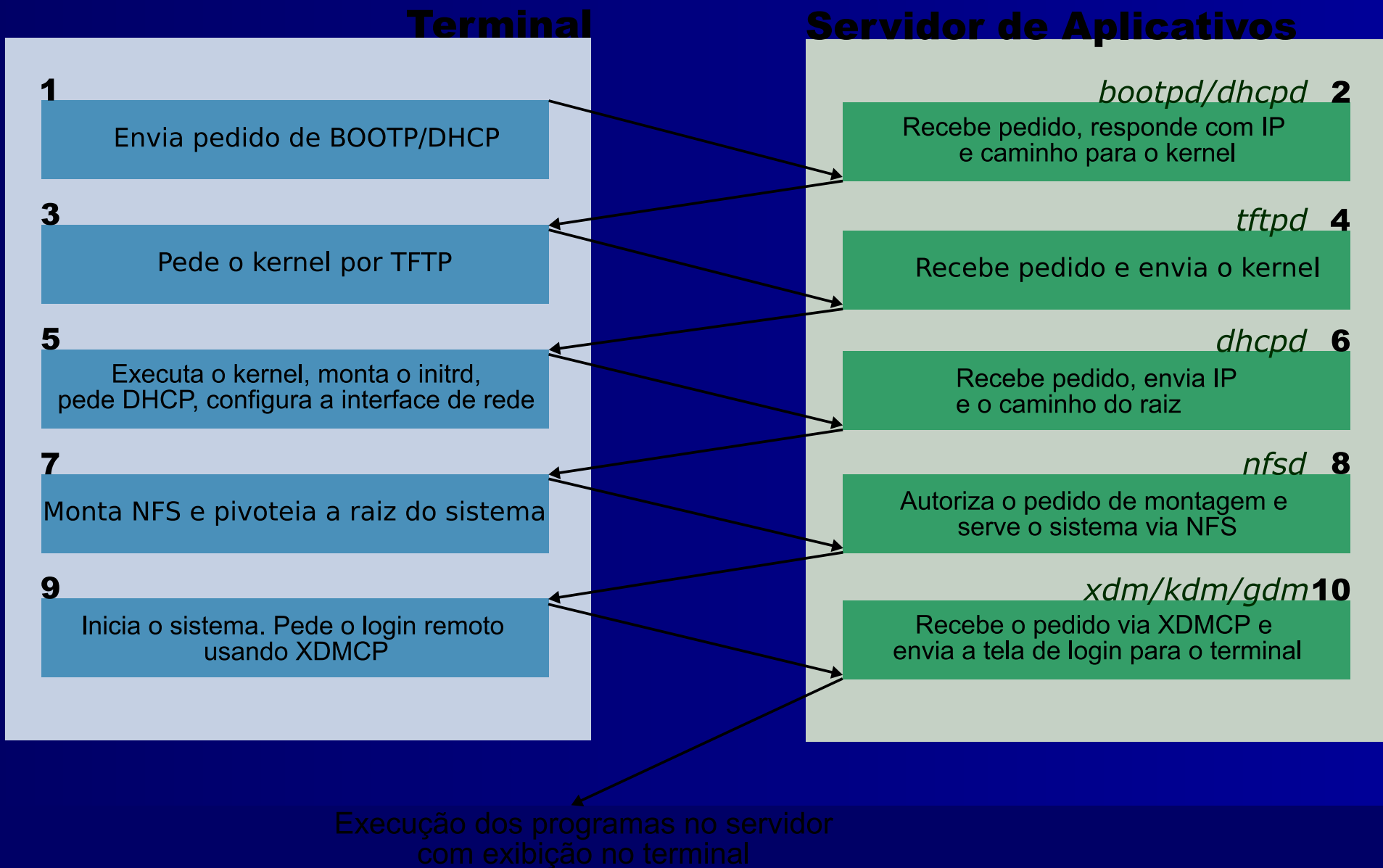
Os projetos Etherboot e Netboot nos ajudam nesta etapa.

Boot via rede (3)

A partir de então, o Sistema Operacional prossegue:

- o kernel tem um disco em memória associado (initrd, *Initial Ram Disc*), este disco conterá um sistema básico com drivers, cliente dhcp e outros.
- cliente dhcp: utiliza para configurar a placa de rede e conseguir o endereço da raiz.
- cliente nfs: utiliza para montar a raiz do sistema.

Boot via rede (4)



Obtendo os aplicativos remotamente

O servidor de janelas (X) é iniciado pelo *X Display Manager* (DM), tal como: kdm, xdm, gdm, ...

Este DM gerencia um conjunto de *displays* X, os quais podem executar local ou remotamente.

Para controlar um *display* remoto:

- inicia-se o X com um pedido de login via XDMCP (*X Display Manager Control Protocol*).
- no servidor de aplicativos o DM responde com uma tela de entrada (*login*). Esta tela é o primeiro aplicativo executado no servidor de aplicativos e mostrado no terminal.

Linux Terminal Server Project

Até agora vimos as tecnologias e fundamentos que podem ser utilizados para atingir o nosso objetivo, porém existe o projeto LTSP que junta tudo de forma a facilitar o processo de instalação e administração.

Com o LTSP temos disponíveis:

- os sistemas minimalistas para serem usados nos terminais
- utilitários para auxiliar na configuração de cada terminal individualmente ou em grupos
- documentação centralizada de como montar os sistemas

Exemplo: dhcpd.conf

```
default-lease-time      21600;
max-lease-time          21600;
option subnet-mask      255.255.255.0;
option broadcast-address 192.168.0.255;
option routers          192.168.0.254;
option domain-name-servers 192.168.0.254;
option domain-name      "ltsp";
option root-path        "192.168.0.254:/opt/ltsp/i386";
subnet 192.168.0.0 netmask 255.255.255.0 {
    use-host-decl-names  on;
    option log-servers   192.168.0.254;
    host ws001 {
        hardware ethernet 00:11:22:33:44:55;
        fixed-address     192.168.0.1;
        filename          "/lts/vmlinuz-2.4.26-ltsp-1";
    }
    host ws002 {
        hardware ethernet 00:11:22:33:44:66;
        root-path         "192.168.0.254:/opt/ltsp/ppc";
    }
}
```

Exemplo: (NFS) /etc/exports

```
/opt/ltsp/i386          192.168.0.0/255.255.255.0(ro,no_root_squash,sync)
/opt/ltsp/ppc          192.168.0.0/255.255.255.0(ro,no_root_squash,sync)
/var/opt/ltsp/swapfiles 192.168.0.0/255.255.255.0(rw,no_root_squash,async)
```

Exemplo: (DM) kdmrc, xdm-config,

- xdm-config:

```
# DisplayManager.requestPort: 0
```

- kdmrc:

```
[Xdmcp]
```

```
Enable=true
```

```
#Port=177
```

- gdm.conf:

```
[xdmcp]
```

```
Enable=true
```

```
MaxSessions=100
```

```
...
```

```
[servers]
```

```
0=/usr/bin/X11/X
```

Exemplo: Its.conf

```
[Default]
SERVER = 192.168.0.254
X_MOUSE_PROTOCOL = "IMPS/2"
X_MOUSE_DEVICE = "/dev/psaux"
SCREEN_01 = startx
```

```
[ws001]
PRINTER_0_DEVICE = "/dev/lp0"
PRINTER_0_TYPE = "P"
```

```
[ws002]
USE_NFS_SWAP = Y
SWAPFILE_SIZE = 64m
SCREEN_01 = shell
```

```
[ws003]
XSERVER = ati
X_MODE_0 = 1600x1200
```

Resolvendo Problemas

- `lspcfcg`
- `netstat -anp`
- `showmount -e`
- `/var/log/messages` (syslog),
`/var/log/everything/current` (metalog)
- `SCREEN_01 = shell`
- `rpcinfo -p localhost`
- `tcpdump` e `Ethereal`

Escalabilidade e Disponibilidade

Utilizando-se de diversas técnicas pode-se aumentar o sistema e também a disponibilidade do mesmo:

- OpenMosix: distribui processos entre vários sistemas (cluster).
- Linux Virtual Server: distribuição de carga para servidores e alta disponibilidade.
- Sistema de *Storage* usando RAID ou outro dispositivo para aumentar desempenho e evitar perda de dados.

Referências

- <http://ltsp.org/>
- <http://etherboot.org/>,
<http://netboot.sf.net/>
- <http://xfree86.org/>, <http://x.org/>
- <http://www.isc.org/sw/dhcp/>,
<http://www.faqs.org/rfcs/rfc2131.html>
- <http://www.tldp.org/HOWTO/Xterminals/>

Fim!

Dúvidas???

Contato

Gustavo Sverzut Barbieri

Email: barbieri@gmail.com

ICQ: 17249123

MSN: barbieri@gmail.com

Jabber: gsbarbieri@jabber.org

Esta palestra se encontra em

<http://gsbarbieri.sytes.net/palestras/ltsp/>