

**LIGA DE ENSINO DO RIO GRANDE DO NORTE  
CENTRO UNIVERSITÁRIO DO RIO GRANDE DO NORTE  
ESPECIALIZAÇÃO EM REDES DE COMPUTADORES**

**RAFAEL MELO VERAS**

**THIN CLIENTS COM SOFTWARE LIVRE**

**NATAL/RN**

**2014**

**RAFAEL MELO VERAS**

**THIN CLIENTS COM SOFTWARE LIVRE**

Trabalho apresentado ao curso de Especialização em Redes de Computadores do Centro Universitário do Rio Grande do Norte como requisito final para a obtenção do título de Especialista em Redes de Computadores.

**Orientador:** Prof. M.Sc. Aluizio  
Ferreira da Rocha Neto

**NATAL/RN**

**2014**

**RAFAEL MELO VERAS**

**THIN CLIENTS COM SOFTWARE LIVRE**

Trabalho apresentado ao Curso de Especialização em Redes de Computadores do Centro Universitário do Rio Grande do Norte – UNI-RN como requisito final para a obtenção do título de Especialista em Redes de Computadores.

**Aprovado em: \_\_\_\_/\_\_\_\_/\_\_\_\_**

**BANCA EXAMINADORA**

---

Aluízio F. Rocha Neto, M. Sc.  
Orientador

---

Membro Examinador

---

Membro Examinador

## RESUMO

O *thin client* é uma solução que otimiza o funcionamento de um computador de forma que diminua o tempo de ociosidade dele. Por ser um equipamento simples e que utiliza recursos de outra máquina, seu uso possibilita a redução de gastos com equipamentos e softwares. Com ele é possível ter uma rede com várias estações de trabalho utilizando apenas um computador convencional. Esta monografia visa mostrar um modo simples de implementar um ambiente computacional *thin client* com software livre, através do projeto Linux Terminal Server Project – LTSP, baseado no sistema operacional Linux e por sua vez explicar dês das tecnologias envolvidas, como elas funcionam, requisitos de hardware, instalação do servidor, inicialização dos *thin clients* ao ambiente em pleno funcionamento. Além disso, também são expostos os principais motivos que se tem ao adotar este excelente ambiente computacional, suas principais vantagens em cima dos ambientes computacionais convencionais, suas desvantagens e o que é preciso pra se ter este ambiente.

**Palavras-Chave:** *Thin Client*. Software Livre. Linux. LTSP.

## **ABSTRACT**

The thin client is a solution that optimizes the operation of a computer in ways that minimize the time of his idleness. Because it is a simple device that uses resources from another machine, its use allows the reduction of spending on equipment and software. It's possible to have a network with several workstations using only a conventional computer. This monograph aims to show a simple way to implement a thin client computing environment with free software, through the Linux Terminal Server Project - LTSP, based on the Linux operating system, and explain since the technologies involved, how they work, hardware requirements, server installation, booting the thin client, to the running environment. Moreover, also are exposed the main reasons that we have to adopt this excellent computational environment, its main advantages over conventional computational environments, downsides and what is necessary to have this environment.

**Keywords:** Thin Client. Free Software. Linux. LTSP.

## LISTA DE FIGURAS

|                                                                                   |    |
|-----------------------------------------------------------------------------------|----|
| <b>Figura 1:</b> Instalador do Ubuntu 8.04 Hardy Heron ( <i>alternate</i> ) ..... | 30 |
| <b>Figura 2:</b> Construindo o ambiente do cliente .....                          | 31 |
| <b>Figura 3:</b> Criando imagem do <i>thin client</i> .....                       | 32 |
| <b>Figura 4:</b> Configuração do boot via rede.....                               | 33 |
| <b>Figura 5:</b> Inicialização do <i>thin client</i> via PXE .....                | 34 |
| <b>Figura 6:</b> Carregando o <i>kernel</i> .....                                 | 35 |
| <b>Figura 7:</b> Carregando o ambiente .....                                      | 36 |
| <b>Figura 8:</b> Terminal pronto para uso .....                                   | 37 |

## LISTA DE TABELAS

|                                                                                    |    |
|------------------------------------------------------------------------------------|----|
| <b>Tabela 1:</b> Informações de hardware do servidor LTSP de nome LTSPS.....       | 29 |
| <b>Tabela 2:</b> Informações de hardware do <i>thin client</i> de nome LTSP01..... | 29 |
| <b>Tabela 3:</b> Informações de hardware do <i>thin client</i> de nome LTSP02..... | 29 |

## SUMÁRIO

|                                                         |    |
|---------------------------------------------------------|----|
| <b>1 INTRODUÇÃO</b>                                     | 9  |
| 1.1 PROBLEMA DE PESQUISA                                | 10 |
| 1.2 OBJETIVOS                                           | 11 |
| <b>1.2.1 Objetivos Específicos</b>                      | 11 |
| 1.3 ORGANIZAÇÃO DO TRABALHO                             | 11 |
| <b>2 REFERENCIAL TEÓRICO</b>                            | 12 |
| 2.1 THIN CLIENT                                         | 12 |
| 2.2 LINUX                                               | 12 |
| 2.3 DHCP ( <i>Dynamic Host Configuration Protocol</i> ) | 14 |
| 2.4 TFTP ( <i>Trivial File Transfer Protocol</i> )      | 15 |
| <b>2.4.1 Visão Geral do Protocolo</b>                   | 15 |
| 2.5 PXE ( <i>Preboot eXecutation Environment</i> )      | 16 |
| 2.6 LTSP ( <i>Linux Terminal Server Project</i> )       | 17 |
| <b>2.6.1 Funcionamento</b>                              | 18 |
| <b>2.6.2 Requisitos de Hardware</b>                     | 22 |
| <b>3 DESENVOLVIMENTO</b>                                | 28 |
| 3.1 METODOLOGIA                                         | 28 |
| 3.2 MONTANDO UM AMBIENTE THIN CLIENT                    | 28 |
| 3.3 PREPARAÇÃO DO AMBIENTE                              | 28 |
| 3.4 INSTALADO O SERVIDOR LTSP                           | 30 |
| 3.5 INICIALIZANDO OS <i>THIN CLIENTS</i>                | 32 |
| <b>4 CONCLUSÃO</b>                                      | 38 |
| <b>REFERENCIAL BIBLIOGRÁFICO</b>                        | 40 |

## 1 INTRODUÇÃO

Um *thin client* ("cliente magro") é um computador cliente em uma rede de modelo cliente-servidor de duas camadas o qual tem poucos ou nenhum aplicativo instalados, de modo que depende primariamente de um servidor central para o processamento de atividades.

O conceito que envolve os *thin clients* não é nenhuma novidade. A idéia de usar máquinas com pouco ou nenhum processamento é anterior ao surgimento dos computadores pessoais e era amplamente utilizada até que os computadores pessoais se tornaram populares. Na época, este tipo de equipamento era denominado "terminal burro", fazendo oposição aos equipamentos "inteligentes", que seriam os com capacidade computacional. Hoje, com um nome mais elegante, esse tipo de produto volta à ativa.

Nos últimos anos, ocorreram muitos avanços tecnológicos, avanços esses que tornaram mais eficiente a prática de centralizar o processamento em alguns poucos computadores de grande porte. Entre esses avanços o principal é a evolução das redes de acesso. A ligação entre os terminais e os servidores tem de ser confiável, veloz e barata. A própria internet torna a aceitação desta tecnologia mais fácil. Quando alguém acessa um serviço de webmail, está utilizando um software que esta em algum servidor bem distante do seu PC.

Seguindo este conceito, surgiu a computação nas nuvens que se tornou tendência, neste modelo de computação os aplicativos dos usuários, assim como seus arquivos e dados relacionados, não precisam mais estar instalados ou armazenados em seu computador. Elas ficam disponíveis na "nuvem", isto é, na rede. Ao fornecedor da aplicação cabe todas as tarefas de desenvolvimento, armazenamento, manutenção, atualização, backup, escalonamento, etc. O usuário não precisa se preocupar com nada disso, apenas com acessar e usar. Com os *thin clients*, acontece algo bem parecido.

Como dispositivo, um *thin client* é projetado para fornecer apenas aquelas funções que são úteis para programas de interface de usuário. Frequentemente tais dispositivos não incluem HDs – o chamado padrão *diskless*, ou sem disco, os quais podem ter dados corrompidos pela instalação de software malicioso, dados perdidos pela formatação acidental ou não, falhas mecânicas, dentre outros; em vez disso,

em nome de baixos custos de manutenção e do incremento do tempo médio entre falhas, um *thin client* armazena todas as informações pertinentes em um servidor central de arquivos.

Tal modelo de processamento e armazenamento de dados, aliados a economia de energia elétrica, baixa necessidade de manutenção e possibilidade de se acabar com vírus de rede e estratégias de backup centralizado podem fazer muito bem ao bolso. Além disso, em redes profissionais, pode-se alterar paradigmas de contínua troca de hardware pelo reaproveitamento de máquinas obsoletas, ou pela compra de hardware dedicado para o uso de *thin clients*. O hardware de *thin client* não custa tão mais barato do que um PC, às vezes pode custar mais caro, mas com a facilidade que se tem de implementar tal ambiente utilizando software livre, eliminando os custos com licenças, as vantagens em com baixa manutenção e durabilidade, é mais que suficiente para se pensar em adotar esta solução.

## 1.1 PROBLEMA DE PESQUISA

Um grande problema que as empresas enfrentam hoje é o trabalho depreendido com backup, configuração e manutenção das estações de trabalho, conseqüentemente elevando os custos com funcionários, hardware e energia para manter o parque de máquinas funcionando plenamente. Com o uso da computação *thin client* o trabalho depreendido e os custos são reduzidos drasticamente. Nesse modelo de computação o processamento é feito no servidor e o armazenamento dos dados também, eliminando a necessidade de configuração (instalação do sistema operacional e aplicativos) em caso de substituição de máquinas, diminuindo com isso as requisições da equipe de suporte. O backup também é reduzido, sendo necessário apenas ser feito do servidor. Além disso, os equipamentos *thin client* geralmente são mais baratos, consomem menos energia e possuem maior durabilidade. Apesar deste modelo computacional ser relativamente antigo ainda é pouco conhecido e pouco utilizado.

## 1.2 OBJETIVOS

Destacar as principais vantagens da adoção do ambiente computacional *thin client*, bem como mostrar seu funcionamento e implementar esta solução de modo simples, descomplicado e de forma rápida utilizando software livre.

### 1.2.1 Objetivos Específicos

A seguir, as etapas para o desenvolvimento deste trabalho:

- Estudo das tecnologias envolvidas para a implementação do ambiente *thin client*.
- Análise dos requerimentos para a implementação.
- Descrever o seu funcionamento.
- Passo a passo para implementação.
- Análise geral desta solução

## 1.3 ORGANIZAÇÃO DO TRABALHO

No Capítulo 2 será abordado o Referencial Teórico, com uma rápida introdução ao sistema operacional Linux, e a descrição dos protocolos utilizados, detalhando o funcionamento, os requisitos de hardware/software para implementação do ambiente, o capítulo 3 apresenta a metodologia usada no desenvolvimento desta monografia, onde o ambiente é implementado, depois da análise feita do resultado da implementação é chegada à conclusão no capítulo 4.

## 2 REFERENCIAL TEÓRICO

### 2.1 THIN CLIENT

*Thin clients* são aparelhos de pequeno porte que gastam pouca energia e possuem menor capacidade de hardware, porque não há necessidade de processamento local, sendo tudo processado no Servidor de Terminais. Por exemplo, o usuário que estiver trabalhando em um *thin client* com um processador de texto, estará, na verdade, rodando o aplicativo no servidor.

A solução *Thin Client* é econômica porque reduz custos no investimento inicial (compra de equipamentos) e, principalmente, no dia-a-dia de funcionamento do equipamento (redução do consumo de energia elétrica, atualização de software, segurança e manutenção, entre outros). Além disso, proporcionam maior controle sobre o gerenciamento das redes, já que tudo é centralizado no servidor.

O *thin client* pode ser conectado ao servidor via rede local ou via Internet, permitindo o acesso de qualquer lugar do mundo. É possível, por exemplo, manter o servidor em uma cidade enquanto funcionários de todo o país conectam-se via web. Desta forma, facilita-se o gerenciamento das informações, reduz-se o risco de perda de dados e o custo de manutenção de hardware. Os *thin clients* são equipamentos que raramente apresentam defeito, e se for o caso, basta substituir o aparelho que o ambiente de trabalho do usuário permanecerá inalterado.

### 2.2 LINUX

O Linux é um sistema operacional criado em 1991 por *Linus Torvalds* na universidade de Helsinki na Finlândia. É um sistema Operacional de código aberto distribuído gratuitamente pela Internet. Seu código fonte é liberado como *Free Software* (software livre), sob licença GPL, o aviso de copyright do kernel feito por

Linus descreve detalhadamente isto e mesmo ele não pode fechar o sistema para que seja usado apenas comercialmente.

Isto quer dizer que não é preciso pagar nada para usar o Linux, e não é crime fazer cópias para instalar em outros computadores, inclusive existe incentivo para que isto seja feito. Ser um sistema de código aberto pode explicar a performance, estabilidade e velocidade em que novos recursos são adicionados ao sistema.

Para rodar o Linux é preciso, no mínimo, de um computador 386 SX com 2 MB de memória (para um kernel até a série 2.2.x) ou 4MB (para kernels 2.4 e 2.6) e 100MB disponíveis em seu disco rígido para uma instalação básica e funcional.

Segundo (Gleydson Mazioli da Silva, 2007):

O sistema segue o padrão *POSIX* que é o mesmo usado por sistemas *UNIX* e suas variantes. Assim, aprendendo o Linux, não se terá muita dificuldade em operar um sistema do tipo *UNIX*, *FreeBSD*, *HPUX*, *SunOS*, etc., bastando apenas aprender alguns detalhes encontrados em cada sistema.

O código fonte aberto permite que qualquer pessoa veja como o sistema funciona (útil para aprendizado), corrija alguma problema ou faça alguma sugestão sobre sua melhoria, esse é um dos motivos de seu rápido crescimento, do aumento da compatibilidade de periféricos (como novas placas sendo suportadas logo após seu lançamento) e de sua estabilidade.

Outro ponto em que ele se destaca é o suporte que oferece a placas, CD-Roms e outros tipos de dispositivos de última geração e mais antigos (a maioria deles já ultrapassados e sendo completamente suportados pelo sistema operacional). Este é um ponto forte para empresas que desejam manter seus micros em funcionamento e pretendem investir em avanços tecnológicos com as máquinas que possui.

Hoje o Linux é desenvolvido por milhares de pessoas espalhadas pelo mundo, cada uma fazendo sua contribuição ou mantendo alguma parte do kernel gratuitamente. *Linus Torvalds* ainda trabalha em seu desenvolvimento e na coordenação dos grupos de trabalho do kernel.

O suporte ao sistema também se destaca como sendo o mais eficiente e rápido do que qualquer programa comercial disponível no mercado. Existem centenas de consultores especializados espalhados ao redor do mundo. É possível se inscrever em uma lista de discussão e relatar dúvidas ou alguma falha, e este relato será visto por centenas de usuários na Internet e algum irá ajudar ou avisará as pessoas responsáveis sobre a falha encontrada para devida correção.

## 2.3 DHCP (*Dynamic Host Configuration Protocol*)

DHCP é a sigla para Dynamic Host Configuration Protocol. Trata-se de um protocolo utilizado em redes de computadores que permite a estes obterem um endereço IP automaticamente.

Ao administrar uma rede pequena - por exemplo, com 5 computadores – não será difícil atribuir um número IP a cada máquina. E se a rede possuir 300 computadores? Ou mil? Certamente, o trabalho vai ser imenso e, neste caso, é mais fácil cometer o erro de dar o mesmo número IP a duas máquinas diferentes, fazendo com que estas entrem em conflito e não consigam utilizar a rede.

Segundo (ALECRIM, Emerson, 2005):

O protocolo DHCP é uma eficiente solução para esse problema, já que, por meio dele, um servidor distribui endereços IP na medida em que as máquinas solicitam conexão à rede. Quando um computador desconecta, seu IP fica livre para uso de outra máquina. Para isso, o servidor geralmente é configurado para fazer uma checagem da rede em intervalos pré-definidos.

É importante frisar que, além do endereço IP, também é necessário atribuir outros parâmetros a cada computador (host) que passa a fazer parte da rede. Com o DHCP isso também é possível. Pode-se passar à máquina-cliente máscara de rede, endereços de servidores DNS (Domain Name Server), nome que o computador deverá assumir na rede (por exemplo, infowester, infowester1 e assim por diante), rotas, etc.

Um exemplo importante sobre o uso de DHCP é o caso dos provedores de internet. Na maioria dos casos, a máquina do usuário recebe um endereço IP diferente para cada conexão à internet. Isso é possível graças à combinação do DHCP com outros protocolos, o PPP (Point to Point Protocol), por exemplo.

## 2.4 TFTP (*Trivial File Transfer Protocol*)

O Trivial File Transfer Protocol (ou apenas TFTP) é um protocolo de transferência de arquivos, muito simples, semelhante ao FTP.

O TFTP é usualmente utilizado para transferir pequenos arquivos entre "hosts" numa rede, tal como quando um terminal remoto ou um cliente inicia o seu funcionamento, a partir do servidor.

### 2.4.1 Visão Geral do Protocolo

De acordo com (Wikipedia- Trivial\_File\_Transfer\_Protocol, 2008), toda a transferência começa com um pedido de leitura ou escrita de um arquivo, que sirva também para pedir uma conexão. Se o servidor conceder o pedido, a conexão está aberta e o arquivo é emitido em blocos fixos do comprimento de 512 bytes. Cada pacote contém um bloco dos dados, e deve ser reconhecido por um pacote do reconhecimento antes que o pacote seguinte possa ser emitido.

Um pacote dos dados de uma terminação de menos de 512 sinais dos bytes de transferência. Se um pacote começar perdido na rede, o intervalo de parada pretendido da vontade do receptor e pode retransmitir seu último pacote (que pode ser dados ou um reconhecimento), assim fazendo com que o remetente do pacote perdido retransmitir esse pacote perdido.

O remetente tem que manter apenas um pacote na mão para o retransmissor, desde que o reconhecimento da etapa do fechamento garante que todos os pacotes mais antigos foram recebidos. Observe que ambas as máquinas envolvidas em transferência estão consideradas remetentes e receptores. Um emite dados e recebe reconhecimentos, o outro emite reconhecimentos e recebe dados. A maioria dos erros causa a terminação da conexão. Um erro é sinalizado emitindo um pacote do erro. Este pacote não é reconhecido, e não retransmitido (isto é, um servidor ou o usuário de TFTP podem terminar após ter emitido uma mensagem de erro), assim que a outra extremidade da conexão não pode começá-la.

Conseqüentemente os intervalos de parada estão usados detectar tal terminação quando o pacote do erro foi perdido. Os erros são causados por três tipos de eventos: o TFTP reconhece somente uma condição de erro que não causa a terminação, a porta da fonte de um pacote recebido que está incorreto. Neste caso, um pacote do erro é emitido ao anfitrião originar. Este protocolo é muito restritivo, a fim simplificar a execução. Para o exemplo, os blocos fixos do comprimento fazem a alocação para diante reto, e o reconhecimento da etapa do fechamento fornece o controle de fluxo e elimina a necessidade requisitar novamente pacotes entrantes dos dados.

## **2.5 PXE (*Preboot eXecutation Environment*)**

O *Preboot eXecutation Environment* é um ambiente para inicializar computadores usando a interface de rede independentemente da disponibilidade de dispositivos de armazenamento de dados (como discos rígidos) ou sistemas operacionais instalados.

O PXE foi introduzido como parte do framework Wired for Management da Intel. Ele faz uso de vários protocolos como IP, UDP, DHCP e TFTP e conceitos como GUID/UUID.

Segundo (Wikipedia-Preboot\_Execution\_Environment, 2008):

O termo PXE cliente, se refere somente ao papel que a máquina tem no processo de inicialização. Um cliente PXE pode ser um servidor, desktop, notebook ou qualquer outra máquina que está equipada com um código de inicialização PXE.

O firmware no cliente tenta localizar um serviço de redirecionamento PXE na rede (Proxy DHCP), a fim de receber as informações disponíveis sobre a inicialização PXE servidores. Depois de analisar a resposta, o firmware irá pedir um servidor de inicialização adequado com o caminho para o network bootstrap program (NBP), transferi-lo para a memória RAM do computador usando TFTP, possivelmente confirmá-lo, e finalmente executá-lo. Se

apenas um NBP é usado entre todos os clientes PXE que pode ser especificado utilizando BOOTP, sem qualquer necessidade de um proxy DHCP, mas um servidor de inicialização TFTP ainda é necessário.

O protocolo PXE é uma combinação de aproximadamente do DHCP e TFTP, embora com alterações sutis para ambos. DHCP é utilizado para localizar o servidor ou servidores de inicialização adequado, e o TFTP utilizado para fazer o download do programa inicial bootstrap e arquivos adicionais.

Para iniciar uma sessão PXE *bootstrap* o PXE firmware envia um pacote DHCPDISCOVER, estendido com opções PXE específicas (DHCPDISCOVER estendido), para a porta 67/UDP (porta do servidor DHCP). As opções PXE identificam o firmware como capaz de PXE, mas elas serão ignoradas pelos servidores DHCP padrão. Se o firmware recebe DHCPOFFERS a partir de tais servidores, pode configurar-se solicitando uma das configurações oferecidas.

## 2.6 LTSP (*Linux Terminal Server Project*)

Segundo (Linux Terminal Server Project Administrator's Reference – Linux Terminal Server Project, 2008), “uma das tecnologias chave na maioria dos sistemas operacionais Linux é o Linux Terminal Server Project (LTSP) que permite iniciar *thin clients* de um servidor LTSP. Para ambientes educacionais, LTSP diminui o custo em hardware, possibilitando o uso de máquinas antigas ou menos potentes como *thin clients*, bem como diminuindo o trabalho com administração tendo apenas que instalar e manter o software no servidor. Quando uma estação de trabalho falha, ela pode ser simplesmente substituída sem perda de dados ou reinstalação do sistema operacional.”

A computação *thin client* faz parte do mundo Unix a muito tempo, apesar da implementação ter evoluído um pouco, o conceito permanece o mesmo:

1. O *thin client* cuida apenas de funções como display, teclado, mouse e som.

2. O servidor faz o trabalho pesado. Todas as aplicações são executadas no servidor e são mostradas simplesmente no *thin client*.

Pelo fato do *thin client* ter um número limitado de tarefas para gerenciar, o hardware pode ser pequeno e barato. Os *thin clients* são livres de manutenção basicamente. Eles duram mais, pois não precisam armazenar... Se eles quebrarem, os dados não são perdidos, pois nada é armazenado nele. Simplesmente troca-se o cliente por outro e volta-se ao trabalho. Se o *thin client* for roubado, nada parará em mãos não autorizadas.

O servidor de terminais executa todas as aplicações e armazena todos os dados. Toda manutenção regular (atualização de softwares, administração) são feitas no servidor. O número de *thin clients* que o servidor pode suportar é proporcional a capacidade do servidor. Como o GNU/Linux faz uso eficiente dos recursos, é possível suportar um número surpreendente de *thin clients* a partir de uma máquina que só poderia ser considerada potente para um único usuário rodando outro sistema operacional.

Em um ambiente computacional de clientes magros, a estabilidade do servidor é importante. É necessário ter certeza de que o servidor possua fontes alternativas de alimentação, dependendo do nível de disponibilidade requerido, fazer uso de fontes redundantes e múltiplos discos para suporte a RAID.

### **2.6.1 Funcionamento**

LTSP é uma coleção de software que transforma um GNU/Linux em um servidor de terminal. LTSP é único a partir de outros sistemas em *thin client* que é considerado por muitos como o mais fácil de manter. Outros sistemas *thin client*, como o *Microsoft Windows Terminal Services*, exigem que cada cliente tenha um software que inicializa o sistema para um ponto que deve ser capaz de se conectar ao servidor. Este poderia ser um sistema operacional mínimo que simplesmente fornece uma interface para se conectar ao servidor. Sistemas como este geralmente requerem mais manutenção e administração, como o software local que inicializa

*thin client*, pode corromper ou conter bugs que exijam atenção. LTSP, por outro lado, não requer nenhum software no lado do cliente. Exige apenas uma interface de rede capaz de inicializar via PXE, o que muitos *thin clients* e PCs já possuem. Isto significa que não é preciso, absolutamente, de mídias físicas de armazenamento (disco rígido, flash compacto, etc) para inicializar o *thin client*. Isto reduz significativamente a quantidade de administração necessária para manter a rede em funcionamento. O processo de iniciar um *thin client* através de um servidor LTSP é o seguinte:

- *Thin clients* através de um protocolo chamado inicialização PXE (*Pre-execution Environment*).
- PXE solicita um endereço IP a partir de um local servidor DHCP.
- O servidor DHCP passa parâmetros adicionais para o *thin client* e transfere uma imagem do GNU/Linux via TFTP para a RAM no cliente.
- O *thin client* então inicializa a imagem do GNU/Linux transferida, detecta o hardware, e se conecta a sessão X do servidor LTSP (normalmente tratados pelo LDM).

A partir daí, todas as operações, como a autenticação de usuário e sua senha, lançar aplicações, e ver sites são realmente tratadas no servidor LTSP e não no *thin client*. O servidor LTSP transfere todas as informações gráficas para o *thin client* através da rede.

### **O processo de inicialização:**

1. Carregar o kernel do Linux para a memória do *thin client*. Isto pode ser feito várias maneiras diferentes, usando o PXE, gPXE, Etherboot, ou yaboot.
2. Uma vez que o kernel tenha sido carregado na memória, ele vai começar a execução.
3. O kernel irá inicializar quaisquer dispositivos de baixo nível que necessita, tais como discos RAM. Quando o kernel é baixado, ele também baixa um

"*initramfs*", ou "sistema de arquivos inicial para a RAM", que contém alguns utilitários e shell scripts que o *thin client* necessita para começar.

4. O kernel começa o seu trabalho, executando um pequeno shell script que irá dar o pontapé de saída do sistema. Este script é chamado `/init`, e fica na raiz do `initramfs`.
5. O script `/init` começa pela montagem do `/proc` e `/sys`, inicia o `udev` para descobrir e inicializar o hardware, especialmente a placa de rede, que é necessário para cada aspecto da inicialização daqui para frente. Assim, ele cria um pequeno disco de RAM, onde qualquer local de armazenagem, onde todo armazenamento local é colocado.
6. A interface de rede loopback é configurada. Esta é a interface de rede que tem como seu endereço IP 127.0.0.1.
7. Um pequeno cliente DHCP chamado `ipconfig` irá então ser executado, para fazer outra consulta a partir do servidor DHCP.
8. Quando o `ipconfig` recebe uma resposta do servidor, a informação que recebe é utilizada para configurar a interface Ethernet, e determinar o servidor para montar a raiz do sistema.
9. Até este ponto, o sistema de arquivo raiz, foi um disco RAM. Agora, o script `/init` irá montar um novo sistema de arquivos raiz, quer através NBD ou NFS. No caso de NBD, a imagem que é normalmente carregada é `/opt/ltsp/images/i386.img`. Se a raiz é montada via NFS, então o diretório que é exportado a partir do servidor é tipicamente `/opt/ltsp/i386`. Ela não pode simplesmente montar os novos arquivos como raiz. É necessário primeiro montá-lo para um diretório separado. Em seguida, ele fará um `run-init`, que irá trocar o atual sistema de arquivos raiz para um novo sistema de arquivos. Quando se completa, o sistema de arquivos será montado em `/`. Neste ponto, qualquer diretório que precisa ter permissão de escrita para ocorrer uma inicialização normal, como `/tmp` ou `/var`, são montados a partir do disco RAM

no momento.

10. Após a montagem do novo sistema de arquivos raiz estar completa, o script `/init` já fez o seu papel e agora precisamos invocar o real `init` em `/sbin/init`.
11. O programa `init` irá iniciar a configuração do ambiente *thin client*, e lendo os comandos de inicialização no arquivo `/etc/rcS.d`.
12. Ele irá executar o comando `ltsp-client-setup` que irá configurar muitos aspectos do ambiente *thin client*, tais como habilitação do NBD swap, a criação do *syslogging* remoto, carregamento de quaisquer módulos do kernel especificados, etc.
13. Em seguida, o programa `init` começará a executar os comandos no diretório `/etc/rc2.d`.
14. Um dos itens no `/etc/rc2.d` diretório é o `ltsp-client-core` comando que será executado enquanto o *thin client* esta iniciando.
15. O arquivo `lts.conf` será analisado, e de todos os parâmetros em que o arquivo que pertencem a este *thin client* será definido como variáveis do ambiente para o script `S20ltsp-client-core` usar.
16. Se o som for configurado neste momento, o *daemon* do *pulseaudio* é iniciado, para permitir conexões remotas de áudio a partir do servidor para se conectar e tocar no *thin client*.
17. Se o *thin client* tiver o suporte a dispositivos locais ativado, o programa `ltspfsd` é iniciado para permitir que o servidor leia a partir desses dispositivos como um memory sticks ou CD-ROM anexada ao *thin client*.
18. Neste ponto, nenhuma das sessões de tela que definidas no `lts.conf` será executado.

Sessões de tela são o que pretendes lançar, em todas as telas virtuais no seu terminal. Estas são as telas virtuais padrão que todas as distribuições GNU/Linux distribuições possuem, ou seja, Alt + F1, até Alt + F10.

Por padrão, um padrão de caracteres baseado *getty* será executado na tela 1 (SCREEN\_01 no lts.conf arquivo).

Bem como, se nada mais é especificado no arquivo lts.conf, um script de tela ldm é executado no SCREEN\_07. O LTSP Display Manager (LDM) é o gestor de logins padrão para o LTSP.

19. Se o SCREEN\_07 está definido para um valor de LDM, ou startx, então o *X Windows System*, serão iniciado, dando a interface gráfica do usuário.

Por padrão, o servidor Xorg irá tentar descobrir automaticamente a placa de vídeo, criará uma configuração padrão /etc/X11/xorg.conf no disco-ram do terminal, e iniciará o xorg com essa configuração personalizada.

20. O servidor X irá iniciar também um túnel encriptado via ssh para o servidor, no caso do LDM, ou uma consulta XDMCP para o servidor LTSP, no caso do *startx*. De qualquer forma, uma caixa de login aparecerá no terminal.
21. Neste ponto, o usuário pode logar-se. Eles estarão sentados em um *thin client*, mas terão uma sessão no servidor. Todos os comandos que forem executados, serão executados no servidor, mas o retorno será mostrada no *thin client*.

## 2.6.2 Requisitos de Hardware

- **Servidor**

Uma rede *thin client* LTSP é bastante escalável; uma máquina moderadamente poderosa pode servir vários *thin clients*, e se for preciso adicionar

mais *thin clients*, pode-se expandir a capacidade do servidor existente, ou, simplesmente adicionar mais servidores.

Dimensionamento de servidores em uma rede LTSP é mais arte do que ciência. O quão grande servidor será, depende largamente do que se está planejando fazer com a rede *thin client*. Os requisitos necessários para um servidor de rede onde o uso será somente um pouco de navegação na web, sem Java ou Flash, será muito diferente de uma rede onde se deseja trabalhar com gráficos pesados, jogos interativos, e animações em Flash.

### **Exemplo de configuração mediana:**

Memória: 256MB + 192 MB \* número de terminais.

Exemplo com 20 terminais:  $256 + 192 * 20 = 4096$  MB

Então quatro gigabytes de memória é uma estimativa justa para o servidor. Ter certeza de que possui memória suficiente é a coisa mais importante a fazer para ajudar no desempenho de um servidor *thin client* LTSP. Se não tem memória suficiente no servidor, o mesmo terá que usar o disco rígido como memória "virtual". Discos rígidos são muito mais lento do que a memória RAM, sendo assim ocorrerá uma certa lentidão, se isso acontecer.

#### **○ Processadores**

O quão rápido deve ser o processador que se precisa é totalmente dependente dos programas que se planeja usar. Jogos interativos requerem um pouco mais do que um editor de textos. Se o intuito é utilizar plug-ins Java e Flash no navegador web, eles podem consumir uma grande quantidade de processamento. Para um modelo misto, ou seja, algumas pessoas jogando, poucas pessoas navegando na web, e poucas pessoas escrevendo em um editor de textos, um processador de 2GHz ou melhor deverá ser capaz de lidar adequadamente com cerca de 20 pessoas com poucos atrasos, porém um processador de 3GHz seria melhor.

Para redes maiores, migrar para um SMP (Symmetric Multi Processing), ou um servidor com múltiplos processadores pode ser vantajoso.

- **Discos**

É aconselhável usar algum tipo de RAID no servidor de terminais. Além de salvar os dados quando um único disco falha, melhora o desempenho (especialmente o desempenho de leitura, que é tipo de acesso mais comum). Para as pessoas com um orçamento restrito, a criação de RAID 1 via software, com 2 discos SATA com Comando Nativo de Enfileiramento proporcionará bons resultados. Se um orçamento maior estiver disponível, e uma grande rede, a criação do servidor com RAID 10, juntamente com discos de 10.000 RPM proporcionará a melhor velocidade possível. Isto dará desempenho e confiabilidade.

- ***Thin Clients***

Grandes quantidades de implantações LTSP estão em salas de aula, e normalmente, nestas situações, o objetivo primordial é a reutilização de hardware existente de que a escola já possui. No entanto, *thin clients* de fato podem também ser utilizados.

- **Processador**

Para usar o padrão, o modo seguro de LTSP, é necessário ter um processador um pouco mais rápido. Qualquer CPU de 533 MHz, ou melhor, deve prover um desempenho aceitável.

- **Rede**

Um *thin client* é carregado através da rede, usando um pequeno programa chamado *network boot loader*. Este programa é localizado na própria placa, ou, para as mais velhas, que não possuem este recurso, o usuário pode fornecer um em um disquete ou CD Rom, que podem ser utilizados para iniciar o *thin client*. Os *network boot loaders* mais comuns que podem ser utilizados são:

## **PXE**

Esta é uma das mais comuns, e muitas placas de rede suportam.

### **gPXE**

O gPXE é uma implementação Livre do *PXE boot loader*. É o sucessor da Etherboot, e é mantido pelo projeto Etherboot.

### **Etherboot**

Este é uma excelente alternativa para o PXE, pode ser iniciado a partir de um disquete, cartão de memória ou CD-ROM, ou, ser gravado em uma EPROM, se a sua placa tem suporte para uma.

- **RAM**

O mínimo para um *thin client* trabalhar é de cerca de 48MB, mas será inviável pela lentidão, portanto é recomendada a instalação de pelo menos 128MB.

- **Placa de Vídeo**

Basicamente qualquer placa de vídeo que use o barramento PCI e possua 32MB ou mais de memória.

- **Rede**

Em uma rede com mais de 20 usuários, é recomendado o uso de uma rede Gigabit Ethernet. Embora o uso normal varie de 0,5 a 2Mbit, os clientes podem ter picos bastante elevados (70mbit), especialmente quando se vê conteúdo multimídia.

## **2.6.4 MÉTODOS DE INICIALIZAÇÃO**

A inicialização do *thin client* via rede pode ser feita de várias formas.

- **Boot ROM**

Dependendo da sua placa de rede, ela já pode ter uma boot ROM.

- **Etherboot e gPXE**

Iniciar com Etherboot requer que o kernel do Linux seja marcado com o *mknbi-linux*, o que permite que o kernel seja inicializado, prefixando o kernel com um código adicional, e anexando o *initrd* ao fim do kernel. Os kernels que são oferecidos com LTSP já estão marcados, e prontos para inicializar com o Etherboot.

- **PXE**

Parte da *Wired for Management* especificação do final da década de 1990 incluiu uma especificação para a tecnologia boot ROM conhecida como o *Pre-boot Execution Environment* comumente abreviada como PXE.

Um PXE boot ROM pode carregar um arquivo de no máximo 32kb. Um kernel Linux é um pouco maior do que isso. Portanto, o PXE é configurado para carregar uma 2ª etapa da inicialização chamada *pxelinux*, que é pequeno o suficiente para ser carregado. Ele sabe como carregar arquivos muito maiores, como um kernel Linux.

Imagens do kernel LTSP já estão configuradas para fazer isso por padrão.

- **Mídia local**

Se a placa de rede do *thin client* não tem uma Boot ROM integrada, e não se tem acesso a um gravador de EPROM, existem outras possibilidades, se a antiga máquina possuir uma unidade de disquete ou CD-ROM. Se assim for, então pode-se utilizar meios de comunicação locais para iniciar o *thin client*.

- Disquete

Inicializar o Etherboot a partir de um disquete é uma excelente forma de iniciar um *thin client* LTSP que não tem uma Boot ROM. Etherboot é carregado no setor de inicialização do disquete. Em seguida, ele irá funcionar apenas como um Boot ROM. O código de inicialização será executado, a placa de rede será inicializada, e o kernel será carregado a partir do servidor de rede.

- Disco rígido

O disco rígido pode ser usado com um gerenciador de boot, para

carregar o kernel e o *initrd*. Existe a possibilidade de carregar a imagem Boot ROM do Etherboot a partir do disco rígido, e que irá agir como um Boot ROM.

- CD-ROM

Uma CD-ROM inicializável pode ser carregado com um kernel Linux, ou uma imagem Etherboot.

- Dispositivo USB

Assim como um CD-ROM, disquetes e disco rígido, também é possível fazer uso de um pendrive para inicializar um módulo Etherboot.

### **3 DESENVOLVIMENTO**

#### **3.1 METODOLOGIA**

Esta monografia é uma pesquisa exploratória e descritiva quando elaborada a partir de material já publicado, constituído principalmente de livros, artigos de periódicos e materiais da Internet, baseado nesse material foi realizado um estudo sobre o ambiente computacional *thin client*, com foco no projeto livre, LTSP e a implementação deste ambiente. Realizando também um estudo de caso com o objetivo de realizar uma pesquisa sólida e detalhada. Para a simulação do ambiente foram utilizadas três máquinas virtuais, uma atuando como servidor, onde foi instalado o LTSP e as demais, sem disco rígido, atuando como clientes.

#### **3.2 MONTANDO UM AMBIENTE THIN CLIENT**

Neste capítulo será mostrado como é simples e rápido montar um ambiente computacional *thin client* utilizando a distribuição Linux. Ubuntu versão 8.04 - Hardy Heron (disponível em <http://releases.ubuntu.com/releases/8.04/ubuntu-8.04.1-alternate-i386.iso>), que já possui um instalador integrado do LTSP (Linux Terminal Server Project).

#### **3.3 PREPARAÇÃO DO AMBIENTE**

Para montar o ambiente foi utilizado o VirtualBox, que é um software de virtualização para a empresas, bem como para usuários domésticos, disponível gratuitamente como Open Source Software, nos termos da GNU General Public License (GPL). Atualmente, VirtualBox roda em Windows, Linux, Macintosh e OpenSolaris hospeda e suporta um grande número de sistemas operacionais hóspedes, incluindo o Windows (NT 4.0, 2000, XP, 2003 Server, Vista), DOS / Windows 3.x, Linux (2.4 e 2.6), Solaris, OpenSolaris, FreeBSD, OpenBSD, NetBSD, entre outros.

Foram criadas três máquinas virtuais, onde uma será o servidor LTSP e as outras sem disco rígido, disquete ou CD-ROM, somente com a placa de rede, serão os *thin clients*.

Configuração de hardware das máquinas virtuais:

- **LTSPS (*Servidor*)**

**Tabela 1:** Informações de hardware do servidor LTSP de nome LTSPS

|                          |               |
|--------------------------|---------------|
| <b>Memória:</b>          | <b>640 MB</b> |
| <b>Disco Rígido:</b>     | <b>4 GB</b>   |
| <b>Memória de Vídeo:</b> | <b>16 MB</b>  |
| <b>Placas de Rede:</b>   | <b>2</b>      |

- **LTSPC01 (*Thin Client*)**

**Tabela 2:** Informações de hardware do *thin client* de nome LTSP01

|                          |               |
|--------------------------|---------------|
| <b>Memória:</b>          | <b>128 MB</b> |
| <b>Disco Rígido:</b>     | <b>Nenhum</b> |
| <b>Memória de Vídeo:</b> | <b>32 MB</b>  |
| <b>Placas de Rede:</b>   | <b>1</b>      |

- **LTSPC02 (*Thin Client*)**

**Tabela 3:** Informações de hardware do *thin client* de nome LTSP02

|                          |               |
|--------------------------|---------------|
| <b>Memória:</b>          | <b>128 MB</b> |
| <b>Disco Rígido:</b>     | <b>Nenhum</b> |
| <b>Memória de Vídeo:</b> | <b>32 MB</b>  |
| <b>Placas de Rede:</b>   | <b>1</b>      |

### 3.4 INSTALADO O SERVIDOR LTSP

Para a instalação do servidor LTSP foi feita como base no documento *LTSP Quick Install*, disponível no wiki comunitário sobre LTSP no Ubuntu (<https://help.ubuntu.com/community/UbuntuLTSP>) , sendo uma ótima opção para quem deseja se aprofundar, o documento citado esta disponível em: <https://help.ubuntu.com/community/UbuntuLTSP/LTSPQuickInstall>

O processo de instalação é bem simples e rápido com quase nenhuma interação. O instalador irá configurar tudo automaticamente se o servidor possuir duas placas de rede. Se não for esse o caso, será informado o que se deve modificar para funcionar com apenas uma placa de rede.

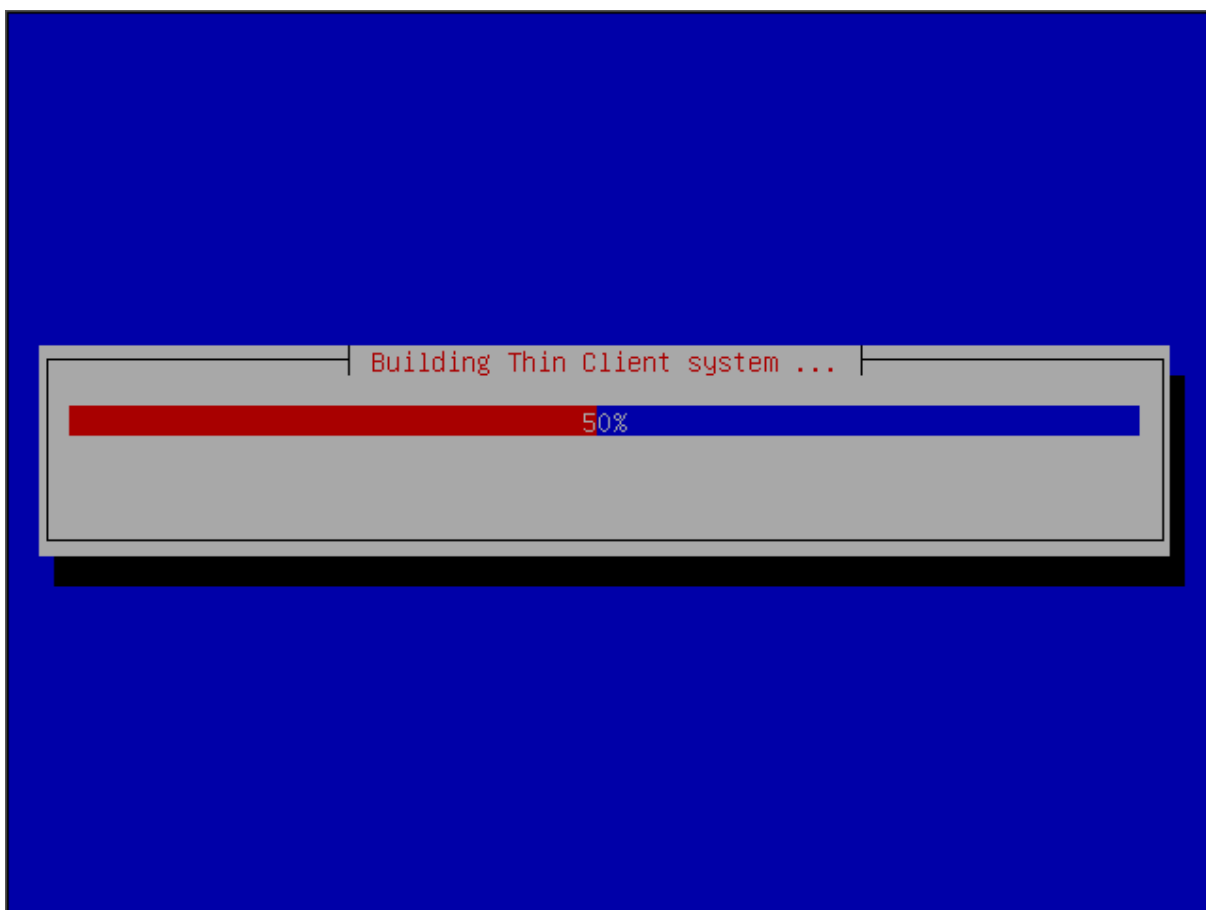
Depois de inicializar com o CD, pressione F4. Um menu com os possíveis modos de instalação irá aparecer. Selecione "Install na LTSP Server", como mostrado na figura 1.

**Figura 1:** Instalador do Ubuntu 8.04 Hardy Heron (*alternate*)



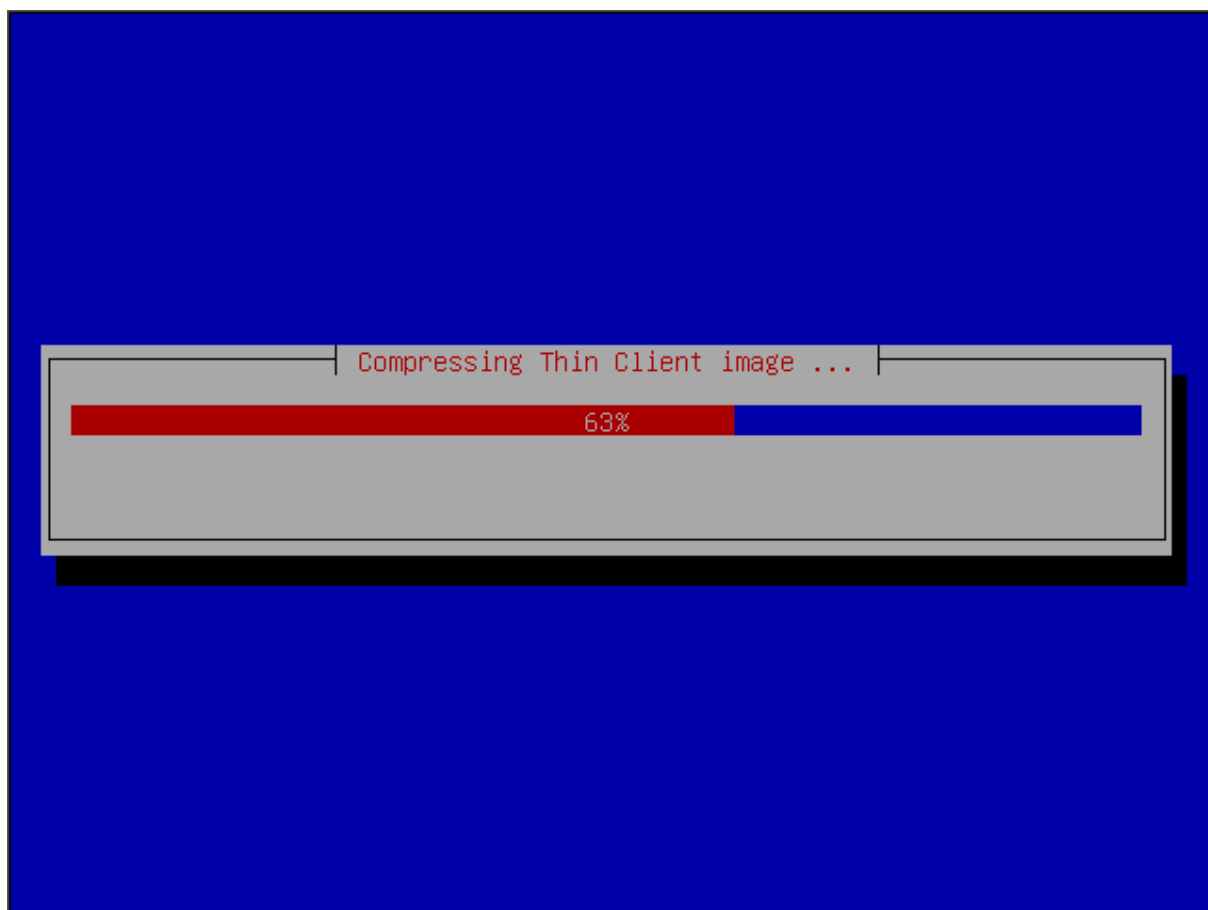
Agora basta seguir em frente com a instalação. Rumo ao final da instalação, conforme figura 2, o instalador irá começar a construir o ambiente do cliente a partir dos pacotes no CD.

**Figura 2:** Construindo o ambiente do cliente



A figura 3 ilustra a compressão deste ambiente em uma imagem que posteriormente será usada para inicializar os clientes.

**Figura 3:** Criando imagem do *thin client*



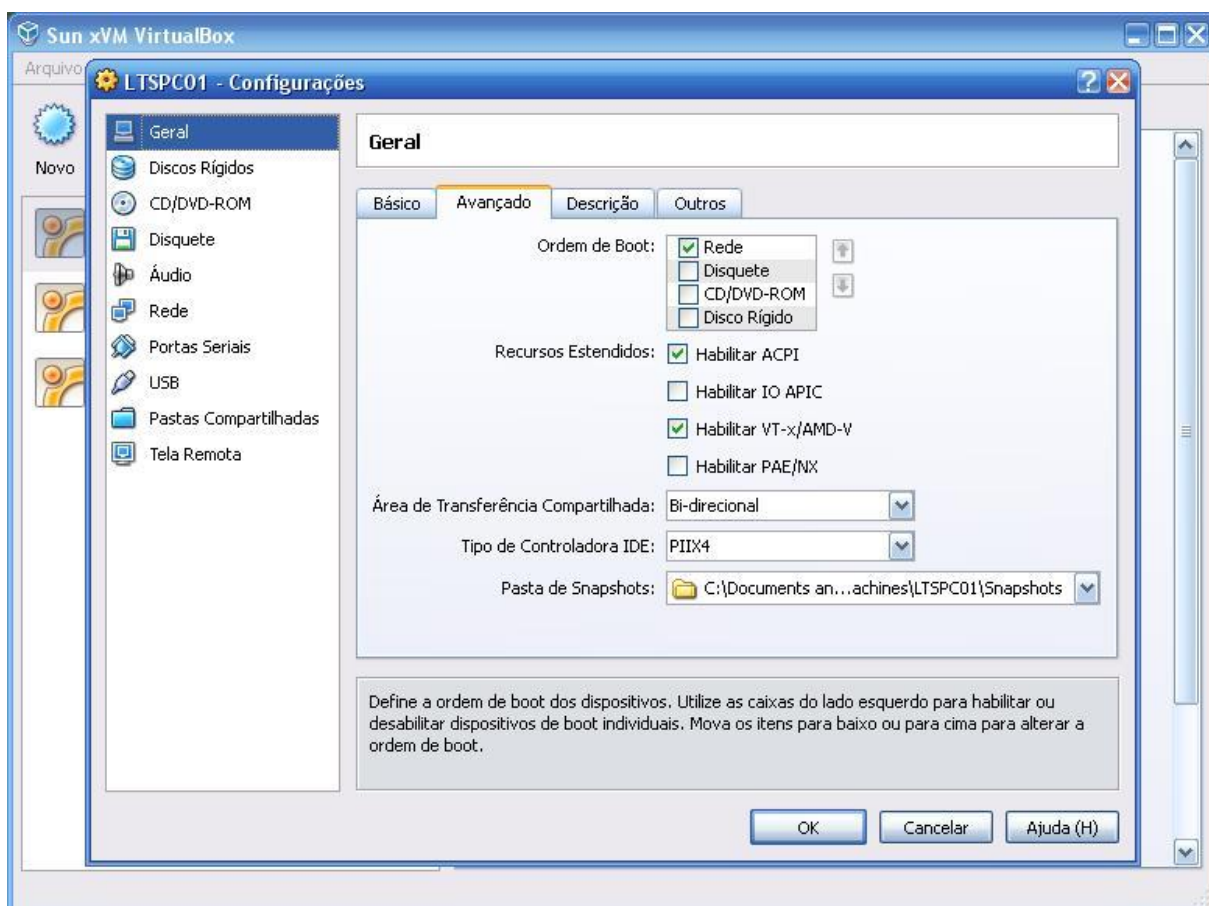
Após o instalador concluir e reiniciar no novo sistema, este já será capaz de inicializar os *Thin Clients*.

### 3.5 INICIALIZANDO OS *THIN CLIENTS*

A parte mais importante e “trabalhosa” já esta concluída que foi a instalação do servidor LTSP. Agora a parte fácil que é inicializar os *thin clients* utilizando placas de rede com suporte a PXE.

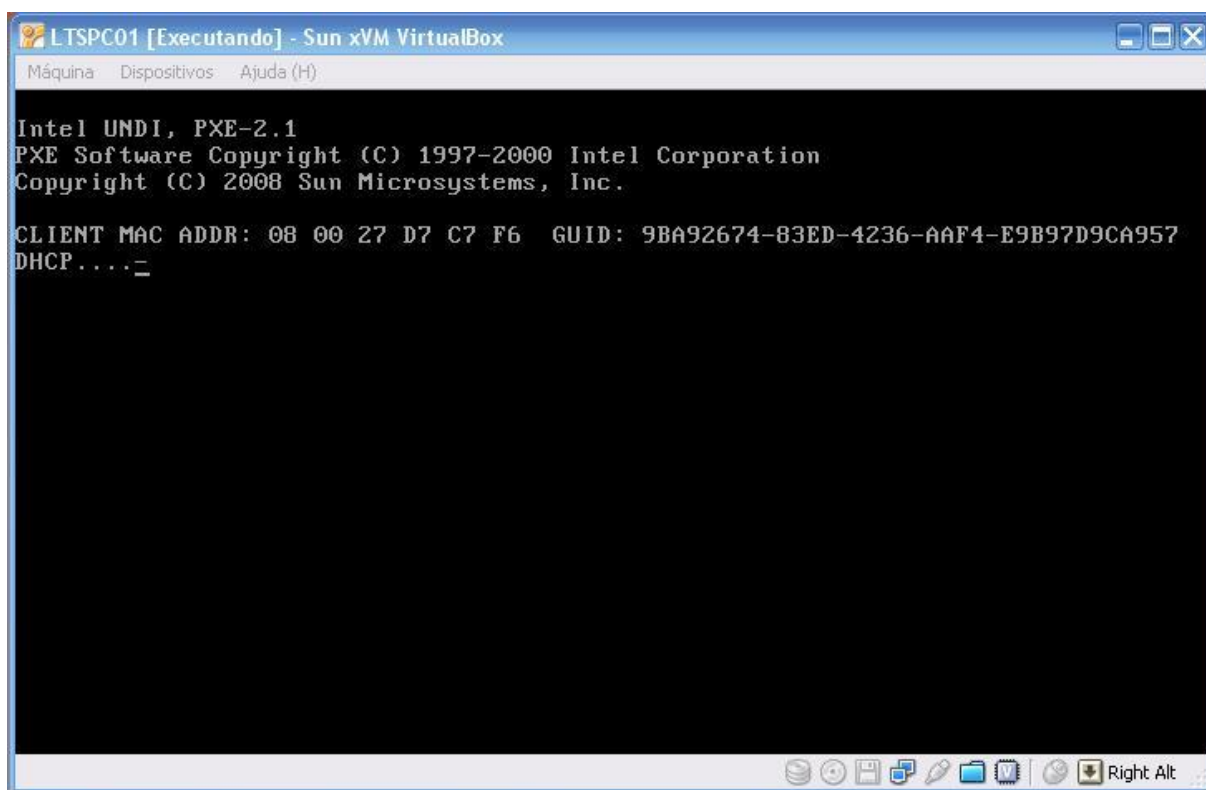
Supondo que o servidor e os *thin clients* estão na mesma rede, tudo que precisa ser feito é seleccionar a inicialização via placa de rede e iniciar o *thin client*, esta configuração esta ilustrada na figura 4.

Figura 4: Configuração do boot via rede



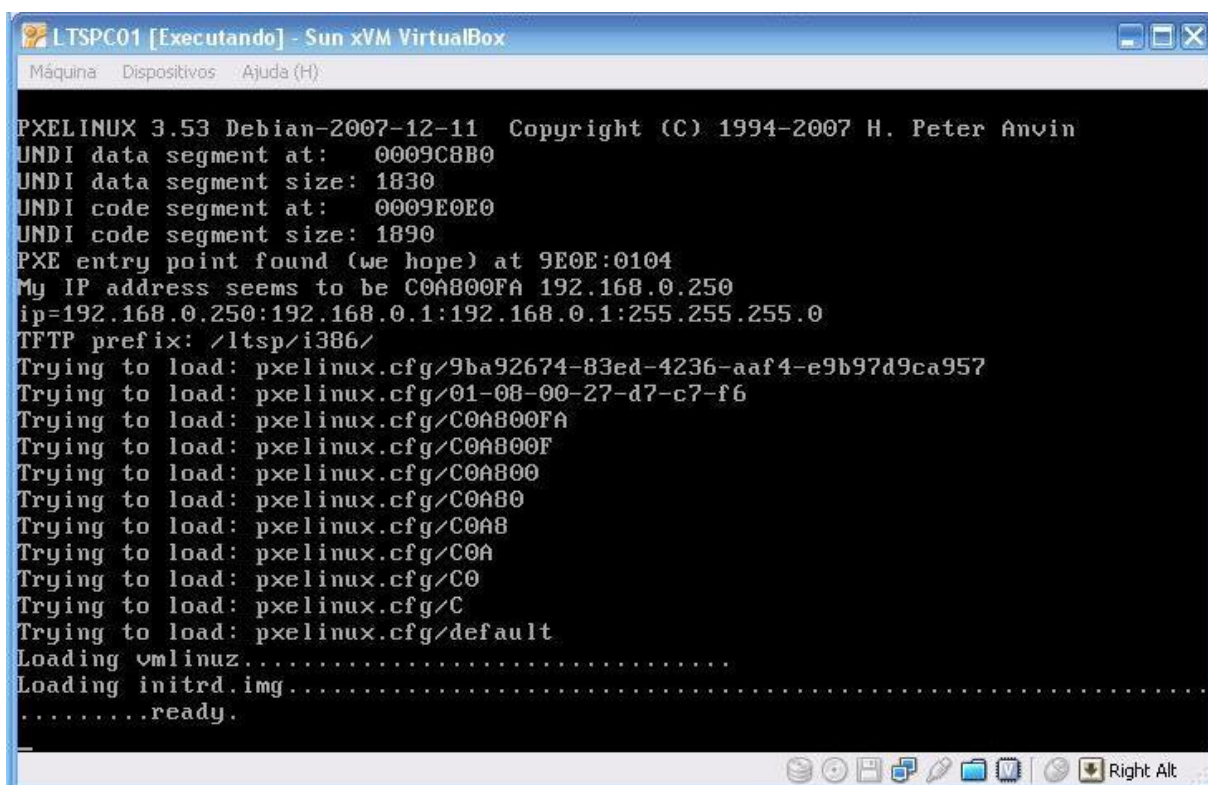
Depois de configurada a inicialização via rede, o *thin client* poderá ser iniciado, como mostrado na figura 5.

**Figura 5:** Inicialização do *thin client* via PXE



Como descrito (mais detalhadamente) no capítulo 2.6.2, após encontrar o servidor o processo de inicialização, ilustrado na figura 6, começará.

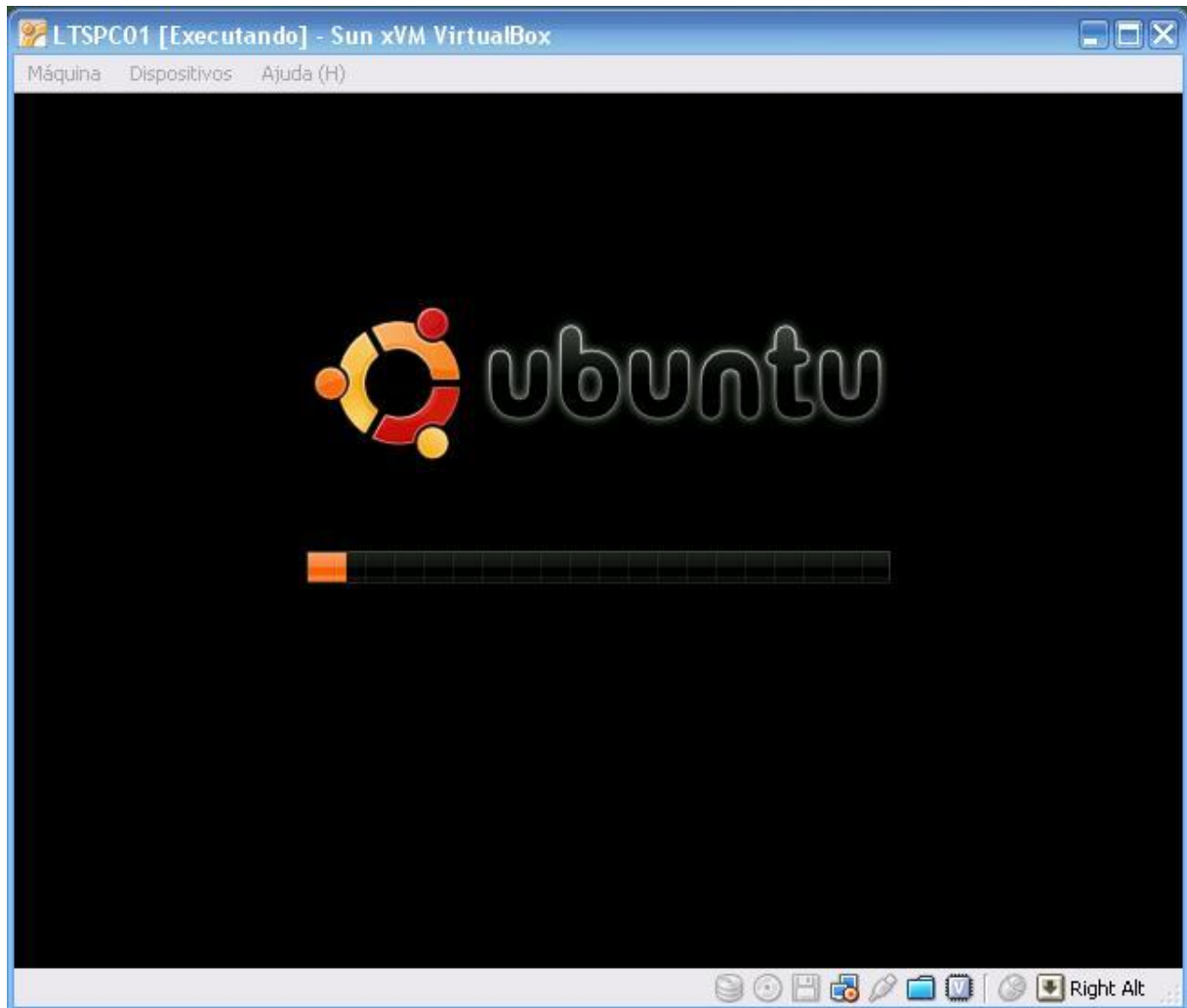
Figura 6: Carregando o kernel



```
LTSPC01 [Executando] - Sun xVM VirtualBox
Máquina  Dispositivos  Ajuda (H)

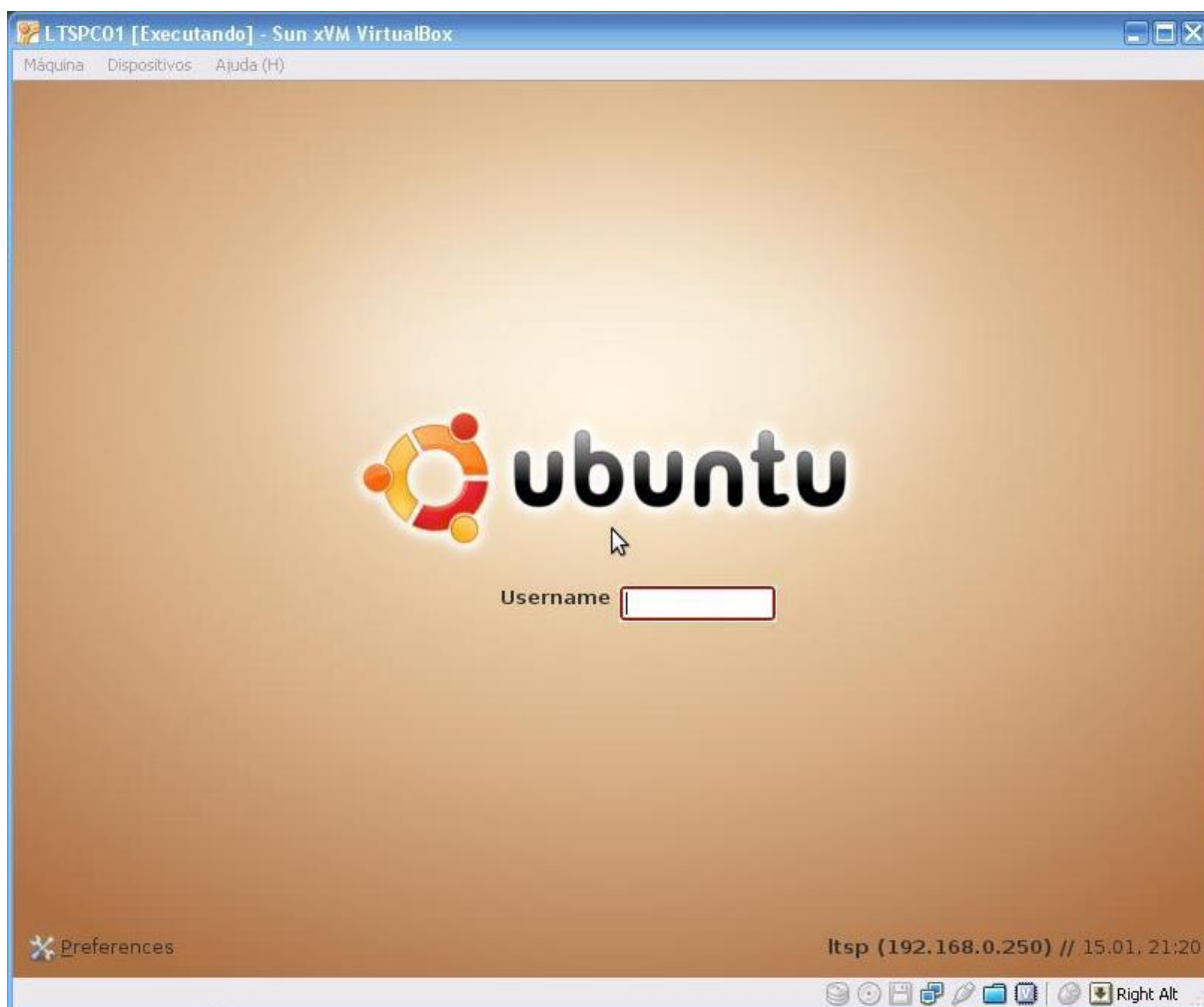
PXELINUX 3.53 Debian-2007-12-11 Copyright (C) 1994-2007 H. Peter Anvin
UNDI data segment at: 0009C8B0
UNDI data segment size: 1830
UNDI code segment at: 0009E0E0
UNDI code segment size: 1890
PXE entry point found (we hope) at 9E0E:0104
My IP address seems to be C0A800FA 192.168.0.250
ip=192.168.0.250:192.168.0.1:192.168.0.1:255.255.255.0
TFTP prefix: /ltsp/i386/
Trying to load: pxelinux.cfg/9ba92674-83ed-4236-aaf4-e9b97d9ca957
Trying to load: pxelinux.cfg/01-08-00-27-d7-c7-f6
Trying to load: pxelinux.cfg/C0A800FA
Trying to load: pxelinux.cfg/C0A800F
Trying to load: pxelinux.cfg/C0A800
Trying to load: pxelinux.cfg/C0A80
Trying to load: pxelinux.cfg/C0A8
Trying to load: pxelinux.cfg/C0A
Trying to load: pxelinux.cfg/C0
Trying to load: pxelinux.cfg/C
Trying to load: pxelinux.cfg/default
Loading vmlinuz.....
Loading initrd.img.....ready.
```

Na etapa seguinte, após o kernel ter sido carregado os passos, do 3 ao 19, serão executados e o ambiente carregado, figura 7.

**Figura 7:** Carregando o ambiente

E por fim os passos 19, 20 e 21 são executados, onde o servidor X será iniciado e uma tela de login, ilustrada na figura 8, surgirá indicando que o *thin client* está pronto para ser utilizado.

**Figura 8:** Terminal pronto para uso



## 4 CONCLUSÃO

Como podemos ver a implementação do ambiente computacional *thin client*, utilizando software livre, é bem simples, rápida e fácil. Sendo possível montar um ambiente completo gastando apenas poucas horas de trabalho e com um custo muito baixo em vários âmbitos.

A economia começa na implantação e gestão deste ambiente, que é rápida e fácil. Um único servidor é tudo o que é necessário para criar, gerenciar e administrar uma rede inteira. A manutenção também se torna mais ágil e menos trabalhosa, como os dados do usuário ficam todos armazenados no servidor, o backup precisará ser realizado apenas nele. Em caso de problemas no *thin client*, o mesmo pode ser rapidamente substituído sem que seja necessária a reinstalação de softwares e restauração de backups.

Com a crescente preocupação com o consumo de energia, migrando-se de desktops convencionais para *thin clients*, este consumo pode ser reduzido à metade. A reciclagem de velhas máquinas também ajuda o ambiente, tornando uma solução verde para o ambiente e poupar energia é importante para muitos gestores de TI hoje em dia.

Com todas essas vantagens e simplicidade fica difícil falar em desvantagens deste ambiente, porem apesar de poucas, elas existem. Ter apenas um servidor para gerir, o que por um lado é bom, pode se tornar uma grande desvantagem em caso de falhas; Se não houver a devida preocupação com a redundância do hardware, e o servidor ficar inoperante, todos os *thin clients* também ficarão. Sendo assim é fundamental investir na redundância de fontes de alimentação, discos, memórias, etc.

Outro ponto que se deve dar uma atenção maior é na qualidade e capacidade da Rede. Fazendo uso de uma rede com uma pequena largura de banda e/ou que tenha problemas constantes de latência ou perda de pacotes, certamente isto afetará significativamente os *thin clients*. Podendo até mesmo tornar os *thin clients*

inutilizáveis, pois o processamento não será fluentemente transmitido pelo servidor. Esses casos fazem com que o *thin client* fique muito difícil de usar, porque a resposta do servidor afetará tanto no visual quanto no desempenho de processamento do cliente.

Avaliando as vantagens e desvantagens do ambiente computacional *thin client* e a sua utilização com software livre, através do LTSP, é possível concluir que se trata de uma solução bastante eficaz, simples, fácil de gerir, facilmente escalável, de manutenção rápida, totalmente centralizada, com baixo consumo de energia, sem custo com licenças de sistema operacional, pois o GNU/Linux é, e sempre será a livre para adquirir, usar e modificar, bem como o LTSP, sem custo com licenças de software, pois também se tratam de software livres e com baixo investimento em hardware, contando também com a possibilidade da reutilização dos hardwares obsoletos. O LTSP pode ser usado não só dentro de grandes instituições, mas também dentro de qualquer outra instituição, independente do seu porte, seja ela pública ou privada, ajudando a oferecer uma solução com ótimo desempenho e sem custos elevados.

## REFERENCIAL BIBLIOGRÁFICO

- ALECRIM, Emerson – Protocolo DHCP. Disponível em: <http://www.infowester.com/dhcp.php>. Acesso em: 07/11/2008.
- Linux Terminal Server Project Administrator's Reference, - LTSP. Disponível em: <http://www.ltsp.org/~sbalneav/LTSPManual.html>. Acesso em: 12/10/2008.
- SILVA, Gleydson Mazioli da – Linux. Disponível em: <http://focalinux.cipsga.org.br/guia/iniciante/ch-intro.html#s-introducao-linux>. Acesso em: 05/11/2008.
- WIKIPEDIA – PXE. Disponível em: [http://en.wikipedia.org/wiki/Preboot\\_Execution\\_Environment](http://en.wikipedia.org/wiki/Preboot_Execution_Environment). Acesso em: 16/11/2008.
- WIKIPEDIA – Trivial File Transfer Protocol. Disponível em: [http://pt.wikipedia.org/wiki/Trivial\\_File\\_Transfer\\_Protocol](http://pt.wikipedia.org/wiki/Trivial_File_Transfer_Protocol). Acesso em: 08/11/2008.