Curso de Ciência da Computação
Universidade Estadual de Mato Grosso do Sul

Thin client Raspberry PI

Felipe Lima Morais

Prof. Dr. Fabrício Sérgio de Paula

Dourados, MS
2015

Curso de Ciência da Computação
Universidade Estadual de Mato Grosso do Sul

Thin client Raspberry PI

Felipe Lima Morais

Novembro de 2015

Banca Examinadora:

Prof. Dr. Fabrício Sérgio de Paula (Orientador)
Área de Computação - UEMS

Prof. Dr. Ricardo Luís Lachi
Área de Computação - UEMS

Prof. Dr. Nilton Cézar de Paula
Área de Computação - UEMS

Thin client Raspberry PI

Felipe Lima Morais

Este exemplar corresponde à redação final da monografia da disciplina Projeto Final de Curso devidamente corrigida e defendida por Felipe Lima Morais e aprovada pela Banca Examinadora, como parte dos requisitos para a obtenção do título de Bacharel em Ciência da Computação.

Dourados, 26 de novembro de 2015

Prof. Dr. Fabrício Sérgio de Paula

AGRADECIMENTOS

A Deus por ter me dado saúde e força para superar as dificuldades.

Obrigado aos meus pais, pelo amor, incentivo e apoio incondicional, sempre me fazendo entender que o futuro é feito a partir da constante dedicação no presente!

A esta universidade, seu corpo docente, direção e administração que oportunizaram a janela que hoje vislumbro um horizonte superior, eivado pela acendrada confiança no mérito e ética aqui presentes.

Meu agradecimento ao Fabrício Sérgio de Paula, por ser meu orientador, me estigando a obter conhecimento e por me auxiliar nas dificuldades.

Meus agradecimentos aos amigos Calebe Paes, Gabriel de Biasi, Larissa Mendes, Rodolpho Pivetta Sabino e todos companheiros do SweetRice, irmão na amizade que fizeram parte da minha formação e que vão continuar presentes em minha vida com certeza.

Meu agradecimento ao Jean Barbosa Siqueira, pela grande ajuda no desenvolvimento, auxiliando em todas as dificuldades encontradas pelo caminho.

RESUMO

O trabalho consiste no estudo de conceitos sobre o uso do *Raspberry PI* como um *thin client* e na criação de um ambiente *thin client*, sendo possível verificar as vantagens e desvantagens da utilização do *Raspberry PI*. O estudo tem a intenção de auxiliar na decisão de criar, ou não, um ambiente *thin client* que utiliza em seus clientes um *Raspberry PI* e também mostrar a viabilidade da substituição dos *thin clients* de um ambiente já implantado, sendo esses *thin clients*, computadores *desktop*. São descritos todos os passos para a criação de um ambiente *thin client* e também os passos para a utilização do *Raspberry PI* como um *thin client*, com essas informações foi implantado um ambiente *thin client*, para a execução de testes tornando possível a captura de dados que possibilitaram a obtenção de resultados práticos. Sendo esses resultados, claros e objetivos sobre a utilização do *Raspberry PI* como um *thin client*, mostrando os problemas de usabilidade e as vantagens econômica dessa utilização. São vantagens como o custo de manutenção e de implantação de um ambiente com *Raspberry PI*. E além de apresentar o desempenho do *Raspberry PI* no consumo dos recursos de *hardware* disponíveis no servidor.

Palavras-chave: *Thin client*; *Raspberry PI*; LTSP.

ABSTRACT

The work consists of studying concepts on using the Raspberry PI as a thin client and to create a thin client environment, it is possible to check the advantages and disadvantages of using the Raspberry PI. The study is intended to assist in the decision to create, or not, a thin client environment we use for our customers Raspberry PI and also show the feasibility of replacing the thin clients to an already deployed environment, and these thin clients, desktop computers. It is described all the steps to creating a thin client environment and also the steps for using the Raspberry PI with a thin client with this information was deployed a thin-client environment for running tests making it possible to capture data They allowed the construction of results. Clear and objective results on the use of the Raspberry PI as a thin client, showing the usability issues and the economic advantages of such use. Advantage as the cost of maintenance and deployment of an environment with Raspberry PI. And besides presenting the performance of Raspberry PI in hardware resources of consumption available on the server.

Keywords: Thin client; Raspberry PI; LTSP.

SUMÁRIO

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

A	Ampere
ARM	Advanced RISC Machine
BIOS	Basic Input/Output System
CAL	Client Access License
CD	Compact Disc
CD-ROM	Compact Disc Read-Only Memory
CPU	Central Processing Unit
DC	Direct Current
DDR3	Double Data Rate tipo 3
DSLR	Digital Single-Lens Reflex
DHCP	Dynamic Host Configuration Protocol
DVI	Digital Visual Interface
DRBL	Diskless Remote Boot in Linux
GB	GigaByte
GIF	Graphics Interchange Format
GHZ	GigaHertZ
GNU	GNU is Not Unix
GPIO	General Purpose Input/Output
GPU	Graphics Processing Unit
HD	Hard Disc
HD	High Definition
HDMI	High-Definition Multimedia Interface
IP	Internet Protocol

I/O	Input/Onput
LTSP	Linux Terminal Server Project
LTS	Long Term Support
MB	MegaByte
MHZ	MegaHertZ
NFS	Network File System
RAM	Random Access Memory
NIS	Network Information Service
PAL	Phase Alternating Line
PC	Personal Computer
PXE	Preboot eXecution Environment
RCA	Radio Corporation of America
RJ-45	Registered Jack tipo 45
SO	Sistema Operacional
TI	Tecnologia da Informação
USB	Universal Serial Bus
V	Volts
VGA	Video Graphics Array

1 INTRODUÇÃO

Thin client, ou cliente magro, é um computador cliente totalmente dependente de um servidor. O servidor compartilha seus recursos de *hardware* como disco rígido (HD), Memória RAM (*Random-Access Memory*) e CPU (*Central Processing Unit*) a todos os *thin clients*. Um *thin client* funciona através da rede, carregando um sistema operacional (SO) do servidor, permitindo o acesso aos programas existentes no servidor, facilitando assim o *backup* e a atualização dos programas utilizados e a interação com o usuário é análoga a um computador que opera de maneira convencional.

O *Raspberry PI* é um computador de baixo custo, que permite um leque de opções na sua utilização. Ele possui tamanho de um cartão de crédito, contendo o processador, GPU (*Graphics Processing Unit*) e a memória RAM em um circuito integrado. É alimentado com energia de um 1 ampere (A) e 5 Volts (V) e pesa em torno de 45 gramas. Entre as várias utilizações do *Raspberry PI*, existe a utilização dele como um *thin client*.

Este trabalho envolve o estudo de conceitos sobre o uso do *Raspberry PI* como um *thin client* e também descreve as configurações necessárias, tanto no *Raspberry PI*, quanto no servidor. E com base nesse ambiente de teste, apresenta e discute os resultados obtidos a partir de um *Raspberry PI* como um *thin client* acessando o servidor.

Nos capítulos 2.1 e 2.2 referente ao referencial teórico, será introduzido o conhecimento necessário para um bom entendimento sobre *thin client* e *Raspberry PI*, explicando cada um, mostrando os principais produtos no mercado, e suas aplicações bem sucedidas.

No capítulo 3.1 é apresentado os equipamentos usados nos testes, além da descrição de como foi executado os testes. Em seguida nos capítulos 3.2 e 3.3 é descrito os passos para a configuração do ambiente, afim de coletar os dados e medir o desempenho do *Raspberry PI* na utilização da interface gráfica e na utilização dos recursos do servidor mostrados no capítulo 4.1, gerando resultados que são apresentados em gráficos.

2 REFERENCIAL TEÓRICO

2.1 thin client

2.1.1 O que é?

Thin client, ou cliente magro (tradução literal), é um computador cliente em uma rede com paradigma de computação centralizada (cliente-servidor), esse cliente possui poucos ou nenhum aplicativo instalado, sendo totalmente dependente do servidor. O servidor executa programas e armazena dados para seus clientes/*thin clients*, facilitando o *backup* e a atualização desses programas, além de compartilhar os recursos de *hardware* como disco rígido, memória RAM, CPU, entre outros (THINCLIENTBRASIL, 2015b; TANENBAUM, 2010).

Existem aparelhos de *thin client* que são equipamentos similares a computadores pessoais (PCs). O que difere esses equipamentos de um PC comum é sua estrutura interna, onde não necessariamente o aparelho de *thin client* possui um disco rígido. Cabe ressaltar que a palavra "*thin*" se refere a uma pequena imagem de *boot* que é o processo de inicialização de qualquer sistema computacional (THINCLIENTBRASIL, 2015b).

O ambiente de *thin client* consiste em um servidor ligado a um ou mais *thin clients*. Cada *thin client* necessita de uma quantidade mínima de recurso de *hardware* para sua utilização (IBM, 2014). Um *thin client* carrega um sistema operacional do servidor, permitindo o acesso aos programas existentes no mesmo. A utilização desse cliente é análoga a um computador que funciona de maneira convencional (THINCLIENTBRASIL, 2015b; MORIMOTO, 2013).

O *hardware* disponível no servidor possibilita a estes dispositivos realizarem as operações/tarefas tais como a execução de programas, sem possuir as especificações mínimas exigidas pelos *softwares*, algo que em um computador que processa seus dados de maneira convencional não seria possível (TANENBAUM, 2010).

O ambiente funciona com computadores clientes que podem ser desprovidos de leitores de CD-ROM (*Compact Disc Read-Only Memory*), unidades de disquetes e HD. Tornado o gerenciamento dos recursos centralizado no ambiente *thin client*, uma vez que os arquivos e aplicações são inseridos no servidor, pois é o único que necessita de disco rígido para seu funcionamento (TANENBAUM, 2010; THINCLIENTBRASIL, 2015b).

Os *thin clients* possui uma comunicação com o servidor através de uma rede local sendo uma das características principais o fato do sistema operacional utilizado não estar na máquina acessada pelos usuários, e sim no servidor. Mas os dispositivos

I/O (*input/output*) como monitor, teclado e mouse, são componentes locais usados para a comunicação do usuário com o sistema, o que faz parecer que eles estão utilizando computadores independentes (RICHARDS, 2007; THINCLIENTBRASIL, 2015b).

2.1.2 Softwares

A implantação de uma infraestrutura com *thin clients* necessita de *softwares* e serviços específicos para que o servidor disponibilize todas as funcionalidades necessárias aos clientes. Atualmente existem algumas implementações no mercado que oferecem tal suporte para que um computador transforme-se em servidor. As principais implementações são descritas a seguir.

2.1.2.1 LTSP

O LTSP ou *Linux Terminal Server Project*, foi fundado por Jim McQuillan e Ron Colcernian em 1999. A ideia era fornecer um terminal gráfico ou em modo texto de um servidor GNU/*Linux*, com uma distribuição *Linux* compartilhada na rede e acessada pelos terminais dos clientes via NFS[1] (*Network File System*) (FARIA, 2009).

Para um cliente LTSP funcionar é necessário estar na mesma rede do servidor e ser inicializado via rede. Já no servidor é instalado o LTSP, que contém as configurações do ambiente *thin client*, permitindo assim, que os clientes LTSP executem os aplicativos instalados no servidor e acessem todos os recursos disponibilizados por ele (FARIA, 2009).

O LTSP funciona em servidores *Linux*, sendo uma solução flexível, de baixo custo e eficiente, que vem sendo utilizado em escolas, empresas e organizações de todo o mundo. Muitos ambientes *thin client* usam a tecnologia LTSP para implementação de sistemas *diskless* (sem disco), sendo utilizados PCs antigos para navegar na *web*, enviar Email, criar documentos e executar outros aplicativos de *desktop*, proporcionando maior vida útil a estes equipamentos (FARIA, 2009; LTSP.ORG, 2014).

2.1.2.2 DRBL

DRBL ou *Diskless Remote Boot in Linux*, fornece um ambiente semelhante ao LTSP. Esse *software* funciona em distribuições Debian, Ubuntu, Red Hat, Fedora, CentOS e SuSE. O DRBL utiliza recursos de *hardware* distribuídos, o que torna possível aos clientes terem acesso total ao *hardware* local (SOURCEFORGE.NET, 1999).

O DRBL usa PXE/*etherboot*[2], NFS e NIS[3] (*Network Information Service*) em

[1] É um sistema de compartilhamento de arquivos entre máquinas de uma rede (XAVIER, 1996).

[2] É um *boot* remoto desenvolvido pela Intel, gravado na ROM da placa de rede que permite que sejam inicializados através da rede, carregando todo o *software* necessário a partir de um servidor previamente configurado (MORIMOTO, 2005c).

[3] É um serviço desenvolvido para distribuir informação pela rede (LINUX SUPPORT, 2007).

seu funcionamento, não sendo necessário instalar o sistema operacional GNU/*Linux* diretamente no disco rígido de cada cliente na rede, pois o disco rígido é opcional. Se um disco rígido estiver presente, o DRBL pode fazer uso dele como uma memória de SWAP[4] (SOURCEFORGE.NET, 1999; FARIA, 2009; TUZI, 2006).

No DRBL outros sistemas operacionais instalados (localmente) nos clientes DRBL, não serão afetados. Isto pode ser útil, por exemplo, durante uma implementação do ambiente *thin client*, onde os usuários ainda possuem a opção de iniciar o sistema local e executar algumas aplicações disponíveis dentro do sistema local. O DRBL permite essa grande flexibilidade em sua implantação (SOURCEFORGE.NET, 1999).

2.1.2.3 Thinstation

O Thinstation é um *software* para ambientes que utilizam *thin clients*. Seu desenvolvimento foi iniciado em 2003 por iniciativa de Miles Roper e hoje é desenvolvido por vários colaboradores (THINSTATION, 2014; FARIA, 2009).

O Thinstation é baseado em *Linux*, mas é possível se conectar diretamente a um servidor Microsoft Windows ou Unix. Como a maioria das aplicações exige um servidor gráfico, o cliente terá um terminal que irá se conectar a um servidor para trabalhar em um ambiente gráfico, essa tecnologia é usada principalmente em salas de aulas, escritórios, empresas ou departamentos (THINSTATION, 2014).

Após a instalação do Thinstation, ele irá gerar uma imagem personalizada do sistema, onde podem funcionar como clientes de um servidor ou trabalhar como terminais autônomos, executando um ambiente gráfico local (THINSTATION, 2014).

O Thinstation roda em *hardware* PC comum (classe i686 32/64 bits) e pode também se utilizar computadores antigos como clientes de uma rede cliente-servidor. O cliente não necessita possuir um disco rígido, pois ele é inicializado pela rede. E dispositivos como disquete, HD, CD-ROM, USB (*Universal Serial Bus*) e impressoras ligadas diretamente aos clientes não são suportados. A última versão estável é a versão 5.4 (THINSTATION, 2014; FARIA, 2009).

2.1.3 Produtos

No Brasil existe a empresa Thin Client Brasil, uma revendedora licenciada para a venda de aparelho de *thin client*. O *site* possui alguns modelos, todos sem a descrição do valor, mas foi feito contato com a empresa e com isso forneceram uma lista com todos os preços e vários arquivos com a descrição de cada produto. As **Tabelas de 1** a **5** mostram informações sobre os produtos.

[4] É a memória virtual que funciona como uma extensão da memória RAM, que fica armazenada no disco.

Tabela 1 – Especificação do modelo TCBR200

Especificação	
Processador	ARM-A9 Dual Core 1GHz
RAM	512MB
Chip Gráfico	Graphics Card Type MALI400 1080P
Saída de vídeo	VGA e HDMI
Dimensões	11,3cm x 11,3cm x 2,4cm
Peso	155g
Portas USB	3
Saída de áudio	1 P2
Porta de rede	RJ-45
Power	DC 5v/2A
Valor	R$ 650,00

Fonte: (THIN CLIENT BRASIL, 2015)

Tabela 2 – Especificação do modelo TCBR200W

Especificação	
Processador	ARM-A9 Dual Core 1GHz
RAM	512MB
Chip Gráfico	Graphics Card Type MALI400 1080P
Saída de vídeo	VGA e HDMI
Dimensões	11,3cm x 11,3cm x 2,4cm
Peso	155g
Portas USB	3
Saída de áudio	1 P2
Porta de rede	RJ-45 , Wireless(3dbi)
Power	DC 5v/2A
Valor	R$ 699,00

Fonte: (THIN CLIENT BRASIL, 2015)

Tabela 3 – Especificação do modelo NC630

Especificação	
Processador	ARM11 800MHz
RAM	128MB
Chip Gráfico	–
Saída de vídeo	–
Dimensões	12cm x 17cm x 3cm
Peso	200g
Portas USB	3
Saída de áudio	2 P2 (input e output)
Porta de rede	RJ-45
Valor	R$ 440,00

Fonte: (THIN CLIENT BRASIL, 2015)

Tabela 4 – Especificação do modelo NC630W

Especificação	
Processador	ARM11 800MHz
RAM	128MB
Chip Gráfico	–
Saída de vídeo	–
Dimensões	12cm x 17cm x 3cm
Peso	200g
Portas USB	3
Saída de áudio	2 P2 (input e output)
Porta de rede	RJ-45 , Wireless(3dbi)
Valor	R$ 510,00

Fonte: (THIN CLIENT BRASIL, 2015)

Tabela 5 – Especificação do modelo TCBR100

Especificação	
Processador	–
RAM	–
Chip Gráfico	–
Saída de vídeo	VGA
Dimensões	9,8cm x 9,8cm x 2,1cm
Peso	200g
Portas USB	4 + 1(mini USB)
Saída de áudio	2 P2 (input e output)
Porta de rede	RJ-45
Valor	R$ 510,00

Fonte: (THIN CLIENT BRASIL, 2015)

Os modelos apresentados, possuem características semelhantes, apesar do preço variar mais que as características. É importante ressaltar que esses valores foram obtidos de apenas uma loja, onde facilita a comparação entre os preços e as especificações.

Na busca por mais modelos, foram encontradas algumas lojas que revendem esse tipo de aparelho individualmente no Brasil. Existem outras que vendem produtos em forma de pacote, mas foge do foco do trabalho. Os produtos encontrados estão descritos nas **Tabelas 6**, **7** e **8**.

Tabela 6 – Especificação do modelo Wyse D10DP

Especificação	
Processador	AMD G-Series T48E de 1,4 GHz e 2 núcleos
RAM	DDR3 2 GB
Chip Gráfico	Radeon HD 6250
Saída de vídeo	DisplayPort, DVI-I
Resolução de vídeo	2560 x 1600, 1920 x 1200
Dimensões	6,7cm x 1,6cm x 7,3cm
Peso	0,93kg
Protocolo	–
Portas USB	2
Saída de áudio	Mini de 1/8 polegadas, Alto-falante mono interno
Porta de rede	RJ-45, wireless
Power	–
Valor	R$ 2.030,00

Fonte: (DELL, 2015)

Tabela 7 – Especificação do modelo ENTC-1000

Especificação	
Processador	Cirrus Logic EP9307 ARM, dual-core de 200 MHz
RAM	64 MB
Chip Gráfico	Não possui
Saída de vídeo	VGA
Resolução de vídeo	2560 x 1600, 1920 x 1200
Dimensões	9,5cm x 15cm x 3cm
Peso	0,93kg
Protocolo	RDP 2.4.1
Portas USB	2
Saída de áudio	1 mini-jack 3,5 mm
Porta de rede	RJ-45
Power	5 VDC
Valor	R$ 326,00

Fonte: (ATERA, 2015)

Tabela 8 – Especificação do modelo Nc600

Especificação	
Processador	800 mhz
RAM	128 MB
Chip Gráfico	Não possui
Saída de vídeo	VGA
Resolução de vídeo	2560 x 1600, 1920 x 1200
Dimensões	11.5cm x 11.5cm x 2.5cm
Portas USB	3
Saída de áudio	1 P2
Porta de rede	RJ-45
Power	5.0V 2.4A
Valor	R$ 420,00

Fonte: (LOJA WT, 2015)

Nas **Tabelas 6**, **7** e **8**, é possível analisar uma discrepância muito grande em relação aos valores, como no caso da **Tabela 6**, que apresenta o aparelho *thin client* com uma configuração equivalente a um desktop de bom sdesempenho, enquanto os demais possui apenas configurações essenciais para seu funcionamento.

2.1.4 Aplicações

A maioria dos recursos computacionais em sistemas *desktop* não é plenamente aproveitada. A tecnologia *thin client* é uma solução que otimiza o funcionamento do servidor, diminuindo o tempo que o computador permanece ocioso, minimizando a subutilização de seus recursos (THIN CLIENT BRASIL, 2015).

Com a utilização de *thin clients*, torna-se possível a implantação de vários clientes na rede, permitindo o acesso aos recursos disponibilizados pelo servidor. Os requisitos de *hardware* do servidor irá determinar o número de terminais, o que deve ser bem pensado antes de elaborar e formatar o ambiente utilizado (THIN CLIENT BRASIL, 2015).

Cabe ressaltar que existem estabelecimentos em que a tecnologia *thin client* foi homologada pela Thin Client Brasil (2015), dentre estes estabelecimentos podemos destacar:

a) Escritório de contabilidade;

b) Escritório de arquitetura;

c) Empresa de *marketing*/design;

d) Escritório de advocacia;

e) Empresa de engenharia;

f) Laboratório de Informática;

g) Empresa de comunicação;

h) Escolas e Universidades;

i) Prefeituras;

j) *Stand* de vendas;

k) Balcão de atendimento;

l) Concessionárias;

m) Farmácias;

n) Bibliotecas;

o) Fábricas em geral;

p) Pizzarias;

q) Lojas de material de construção;

r) Supermercados.

Além de ser bastante econômico, os aparelhos de *thin client* são ecologicamente corretos. O consumo de energia é bem menor, comparado às soluções convencionais e também gera menos lixo eletrônico no meio ambiente, como sendo apenas alguns dos benefícios que podem ser aproveitados, através da implantação desta estrutura. Logo abaixo há uma lista de benefícios da utilização do *thin client* de acordo com a Thin Client Brasil (2015).

a) Baixo investimento inicial;

b) Baixo custo de administração de TI;

c) Facilidade de proteção e gerenciamento de rede;

d) Baixo custo de *hardware*;

e) Menor custo para licenciamento de *softwares*;

f) Baixo consumo de energia;

g) Não desperta interesse dos ladrões, diminuindo risco de furto;

h) Resistência a ambientes hostis;

i) Menor dissipação de calor para o ambiente (economia com ar condicionado);

j) Não possui ruídos (ao contrário dos PCs);

k) Manutenção muito baixa;

l) Possui maior vida útil, gerando menos lixo eletrônico.

As principais vantagens em utilizar uma infraestrutura formada por *thin clients* estão relacionadas à economia de energia, *software* e *hardware*, dados que são apresentados nas **Tabelas 9** e **10**. Na parte referente à economia de energia, é calculado o consumo direto e indireto, pois um *thin client* consome cerca de 5% da energia de um computador convencional. Além do consumo próprio deve se levar em conta a economia de energia elétrica relacionada a utilização de arcondicionado, já que os aparelhos de *thin client* dissipam menos calor para o ambiente (THINCLIENTBRASIL, 2015a).

Tabela 9 – Tabela informando o cálculo para o consumo de energia elétrica anual.

Solução Convencional	Gasto anual em Energia para 30 computadores	R$ 11.484,00
Solução *thin client*	Gasto anual em Energia para 01 Servidor + 30 Terminais *thin clients*	R$ 1.041,22
Economia		R$ 10.442,78

Fonte: (THINCLIENTBRASIL, 2015a)

A economia referente ao *software*, diz respeito as licenças de aplicativos e sistemas, devido a centralização das aplicações, sendo suficiente a compra de apenas uma licença

do produto. Mas isso depende do sistema operacional, no caso da plataforma Microsoft é necessário comprar licença de Terminal Server, além de possuir licenças de CAL (Client Access License) para todos os clientes (MICROSOFT, 2011). Já para o *hardware*, quando o administrador da rede pensa em atualizar o *hardware*, é necessário apenas a mudança em seu servidor. Com isso, todos os clientes serão beneficiados com este upgrade (THINCLIENTBRASIL, 2015a).

Tabela 10 – Tabela informando o cálculo para manutenção anual (*hardware/software*).

Solução Convencional	Gasto anual com manutenção para 30 computadores	R$ 10.800,00
Solução *thin client*	Gasto anual com manutenção para 01 Servidor + 30 Terminais *thin clients*	R$ 3.600,00
Economia		R$ 7.200,00

Fonte: (THINCLIENTBRASIL, 2015a)

2.2 Raspberry PI

2.2.1 O que é?

O *Raspberry PI* é um computador de baixo custo financeiro, baseado na arquitetura ARM[5] (*Advanced RISC Machine*) e que consome 1A e 5V, fornecida pela sua porta micro-USB. Esse baixo consumo torna dispensável o dissipador de calor no dispositivo. Ele possui uma unidade central, processador gráfico e também um *hardware* de áudio e de comunicações, todos inseridos em um único componente, um chip de circuito impresso (UPTON; HALFACREE; PASCHOA, 2013; RASPBERRY PI FOUNDATION, 2015).

Upton, Halfacree e PASCHOA (2013) afirmam que a arquitetura ARM é algo incomum no mundo dos *desktops*, o que faz *Raspberry PI* não seja compatível com o *software* tradicional de PCs por não possuir o mesmo conjunto de instruções. Apesar disso, existe uma porção de *softwares* disponíveis que utilizam instruções ARM e está em crescimento com a popularidade do *Raspberry PI*.

2.2.2 Surgimento

O projeto surgiu originalmente como uma ferramenta para o aprendizado de linguagens de programação, especialmente em países de terceiro mundo, onde queriam oferecer um computador barato o suficiente para que cada estudante pudesse praticar em casa (MORIMOTO, 2012).

Em 2006, Eben Upton e sua equipe contendo professores, acadêmicos e entusiastas da computação em torno dele, desenvolveram os primeiros conceitos para o *Raspberry PI*, sendo este baseado no microcontrolador Atmel ATmega. Em 2009, foi criada oficialmente a *Raspberry PI Foundation* (UPTON; HALFACREE; PASCHOA, 2013).

Em agosto de 2011, foi produzida a primeira série com aproximadamente 50 placas, elas serviram principalmente como uma plataforma para desenvolvedores, para depuração e demonstração das capacidades do produto. Em dezembro de 2011, foi a vez da série beta onde foi produzido 25 placas, já baseada no *layout* de produção (HEIN, 2013).

Após os criadores eliminarem as últimas falhas, ocorreu um leilão online em janeiro de 2012, onde 10 placas desta série beta foram vendidas por um total de £ 16.336,00. Em 29 de fevereiro de 2012, os servidores web da fundação e duas distribuidoras oficiais foram sobrecarregados em poucos minutos com a quantidade de pedidos. Muitos clientes esperaram durante horas para submeter uma pré-encomenda (HEIN, 2013).

[5] Arquitetura de processador de 32 bits, que visa a simplificação das instruções, para atingir a máxima eficiência por ciclo, gerando um baixo consumo de energia (ROBO LIVRE, 2011).

2.2.3 Tipos/Produtos

O *Raspberry PI* destaca-se por ter o tamanho de um cartão de crédito e mesmo assim é capaz de reproduzir vídeos em definição HD[6] (*High Definition*), editores de texto e jogos. No final de 2012, os usuários ganharam uma loja de aplicativos própria para *Raspberry PI* onde é possível baixar e disponibilizar programas desenvolvidos (REDAÇÂO GALILEU, 2015). Abaixo segue a lista de componentes contidos no *Raspberry PI*:

Processador: Em alguns modelos, o *Raspberry PI* possui um processador de *system-on-chip* de 32 bits com 700 MHz, mas no modelo B2 possui um *quad-core* com 900MHz. Todos os modelos são construídos sobre a arquitetura ARM. O chip ARM apresenta em uma variedade de arquiteturas com diferentes núcleos configurados. O modelo B+ tem 512MB de memória RAM, já o modelo B2 vem com 1GB de RAM, enquanto os outro continham apenas 256 MB.

***Slot* para cartão de memória:** O armazenado é feito em um cartão de memória, pois não possui um disco rígido. O *slot* varia de modelo a modelo, podendo ser SD ou MicroSD.

Porta USB: Todos os modelos possuem entrada USB mas a quantidade varia de modelo a modelo, como no caso modelo A que possui apenas uma. Já no modelo B, há duas portas USB 2.0, e no modelo B+ em diante possui quatro portas USB 2.0. Algumas das primeiras placas do *Raspberry PI* foram limitadas quanto à quantidade de corrente que elas poderiam fornecer. Alguns dispositivos USB podem chegar a 500mA. A placa original do *Raspberry PI* suportava 100mA ou menos, mas as revisões mais recentes alcançam até a especificação completa das portas USB 2.0 que é 500mA.

Porta *Ethernet*: O modelo B em diante possui porta *Ethernet* padrão RJ45. O modelo A não tem, mas pode ser conectado a uma rede com fios por meio de um adaptador de rede *Ethernet* USB. A conectividade *Wi-Fi* é feita apenas por meio de um adaptador USB externo.

Conector HDMI (*High-Definition Multimidia Interface*): A porta HDMI está presente em todos os modelos, e oferece saída de áudio e vídeo digital.

Saída de áudio analógico: Conector de áudio analógico padrão de 3,5 mm que é destinado a conduzir cargas de alta impedância. A qualidade é muito inferior à saída de áudio HDMI.

[6] É o sistemas de resolução da tela corresponde ao número de pixels presentes. O HD faz referência à resolução de 1280 x 720 pixels, que por sua vez combina com telas *widescreen* (16:9).

Saída de vídeo composto(*Composite*): Um conector tipo RCA (*Radio Corporation of America*) fornece sinais de vídeo NTSC (*National Television System Committee*) ou PAL (*Phase Alternating Line*) e existe em apenas no modelo A e B. O sinal de vídeo é um conector de 4 polos TRRS, sendo assim necessário um cabo apropriado. Esses conectores produzem um formato de vídeo de resolução baixa, uma segunda opção caso não haja tela com entrada HDMI.

Entrada de energia: Conector microUSB é usado para fornecer energia, a porta microUSB foi escolhida porque é um conector barato, e fontes de alimentação microUSB são fáceis de encontrar.

GPIO *hearder*: Possui o propósito de entrada e saída de corrente (GPIO). Eles são um conjunto de conexões que têm várias funções, mas a principal delas é conectar o *Raspberry PI* a um circuito eletrônico, usando normalmente programas para controlar o circuito.

Existem cinco modelos de *Raspberry PI*, e a cada modelo lançado, o produto é aprimorado, sendo que a última versão lançada pode executar quase todas distribuições ARM GNU/*Linux*, incluindo Ubuntu Snappy Core, e também o Microsoft Windows 10 (RASPBERRY PI FOUNDATION, 2015). Nas **Tabelas de 11** a **15** apresentam os modelos produzidos comercialmente:

Raspberry PI 1 Modelo A

Tabela 11 – Especificação do *Raspberry PI* Geração 1 Modelo A

Especificações	
Processador	700 MHz ARM1176JZF-S core
RAM	256 MB
Chip Gráfico	Broadcom VideoCore IV @ 250 MHz
Saída de vídeo	RCA Composto e HDMI
Cartão de memória	SD cards
Portas USB	1
Saída de áudio	Conector RCA, HDMI
Porta de rede	não possui
Power	5V via MicroUSB ou header GPIO
Valor	Não encontrado no mercado

Fonte: Raspberry Pi portugal (2013)

Raspberry PI 1 Modelo A+

Tabela 12 – Especificação do *Raspberry PI* Geração 1 Modelo A+

Especificações	
Processador	700 MHz ARM1176JZF-S core
RAM	256 MB
Chip Gráfico	Broadcom VideoCore IV @ 250 MHz
Saída de vídeo	Composite, HDMI e Raw LCD
Cartão de memória	SD cards
Portas USB	1
Saída de áudio	Conector de 3.5 mm (Composite), HDMI
Porta de rede	não possui
Power	5V via MicroUSB ou header GPIO
Valor	R$ 189,90

Fonte: Filipelop (2015)

Raspberry PI 1 Modelo B

Tabela 13 – Especificação do *Raspberry PI* Geração 1 Modelo B

Especificações	
Processador	700 MHz ARM1176JZF-S core
RAM	256 MB ou 512 MB (compartilhada com GPU)
Chip Gráfico	Broadcom VideoCore IV @ 250 MHz
Saída de vídeo	RCA Composto e HDMI
Cartão de memória	SD cards
Portas USB	2
Saída de áudio	Conector RCA, HDMI
Porta de rede	RJ-45
Power	5V via MicroUSB ou header GPIO
Valor	Não encontrado no mercado

Fonte: multilogica (2014)

Raspberry PI 1 Modelo B+

Tabela 14 – Especificação do *Raspberry PI* Geração 1 Modelo B+

Especificações	
Processador	700 MHz ARM1176JZF-S core
RAM	256 MB ou 512 MB (compartilhada com GPU)
Chip Gráfico	Broadcom VideoCore IV @ 250 MHz
Saída de vídeo	Composite, HDMI e Raw LCD
Cartão de memória	MicroSD card
Portas USB	4
Saída de áudio	Conector de 3.5 mm (Composite), HDMI
Porta de rede	RJ-45
Power	5V via MicroUSB ou header GPIO
Valor	R$ 229,90

Fonte: Filipelop (2015)

Raspberry PI 2 Modelo B

Tabela 15 – Especificação do *Raspberry PI* Geração 2 Modelo B

Especificações	
Processador	900 MHz Quad-core ARM Cortex-7
RAM	1GB
Chip Gráfico	VideoCore IV 3D graphics core
Saída de vídeo	Composite, HDMI e Raw LCD
Cartão de memória	MicroSD card
Portas USB	4
Saída de áudio	Conector de 3.5 mm (Composite), HDMI
Porta de rede	RJ-45
Power	5V via MicroUSB ou header GPIO
Valor	R$ 279,90

Fonte: Filipelop (2015)

2.2.4 Aplicações

As utilidades do *Raspberry PI* têm se multiplicado, possibilitando desenvolver funcionalidade para facilitar de alguma maneira o dia a dia. Segue uma lista com algumas aplicações em que o *Raspberry PI* é utilizado:

Otto A câmera Otto funciona como uma máquina fotográfica que cria GIFs (*Graphics Interchange Format*) ao invés de fotos. A câmera sincroniza com o smartphone de modo que você pode facilmente compartilhar os GIFs (RAUCHWERK GUSTAVO HUBER, 2014).

Professor de braile O projeto MUDRA visa ensinar crianças com deficiência visual a ler em braile, usando um *Raspberry PI*. O sistema indica sonoramente através de um "teclado" qual a letra (DAWLE, 2014).

Automação de casas Possibilita ligar e desligar as luzes remotamente ou qualquer outro objeto ligado a casa por meio do *Raspberry PI*, que também dá o acesso para outros dispositivos o controlarem também (TREACY, 2013).

Servidor web pessoal O *Raspberry PI* pode ser ligado à internet, e usado como um servidor web, com um consumo de energia bem menor.

Câmera Pi - Câmera DSLR com computador embutido Incorpora um *Raspberry PI* em uma câmera DSLR (*Digital Single-Lens Reflex*), permitindo que um fotógrafo transmita para um PC ou *tablet*, as imagens capturas, sendo que o controle remoto da câmera pode ser através de um *smartphone*, de qualquer lugar do mundo. Também permite programar a câmera para tirar fotos em intervalos precisos (HUNT, 2012).

Mesa de fliperama Este projeto envolve um pouco de trabalho com madeira, mas o resultado é uma mesa de arcada controlada pelo *Raspberry PI* que lhe permite jogar seus antigos jogos favoritos de mesa, utilizando uma tela (TREACY, 2013).

BerryTerminal É uma distribuição *Linux* minimalista que deixa o *Raspberry PI* apto a acessar uma rede *thin client*. Ele permite aos usuários acessar um servidor LTSP (BOS, 2014).

3 INSTALAÇÃO E CONFIGURAÇÃO

3.1 Ambiente de desenvolvimento

Para o desenvolvimento da monografia, foi feito um levantamento das tecnologias utilizadas, afim de conhecer todos os *softwares* e suas configurações necessários para a implantação de um ambiente *thin client* e assim verificar a usabilidade[1] do *Raspberry PI* como um *thin client*. Para verificar a usabilidade, foi desenvolvido uma bateria de testes, onde foi utilizado um servidor, um *Raspberry PI* e também um computador *desktop* funcionando como *thin client*.

No servidor foi utilizado o Ubuntu 12.04 LTS que é um sistema operacional de código aberto, construído a partir do núcleo *Linux* e baseado no Debian (SILBER, 2006). Foi utilizado o *software* LTSP (Linux Terminal Server Project) para implantação do ambiente *thin client*, devido a sua ampla comunidade para auxiliar na sua implantação.

Outro motivo para a utilização do LTSP é a existência de distribuição *Linux* para *Raspberry PI* que permite aos usuários acessar qualquer servidor LTSP. Essa distribuição é chamada de BerryTerminal. O BerryTerminal é uma pequena distribuição *Linux* projetada para transformar o *Raspberry PI* em um *thin client* (BOS, 2014).

Os testes foram executados utilizando um servidor com processador Intel i5-4210U com 1,7Ghz com 4 GB de memória RAM DDR3 com velocidade de1600 Mhz, uma placa gráfica NVIDIA GeForce GT 740M de 2GB e 250GB de HD SSD (*Solid-State Drive*) com escrita de sequencial de até 520 MB/s.

O Rapsberry PI utilizado é o modelo B com 512 MB de memória RAM. E o computador utilizado para comparação possui processador Intel i5-2400 com 3,1Ghz, memória RAM de 8GB DDR3, placa de vídeo *off-board* 1GB DDR3.

A bateria de testes possui quatro testes, todos possuem o servidor e um *thin client* ligado por um cabo RJ-45 de um metro. O primeiro teste é feito com o *Raspberry PI* como *thin client* usado por um usuário. O segundo teste utiliza um computador como *thin client* também usado por um usuário. Os dois primeiros testes são feitos para comparar a usabilidade do *Raspberry PI* em relação ao computador.

O terceiro teste é feito com o *Raspberry PI* como *thin client* executando um *script* simulando a ações de um usuário. O quarto teste é igual ao terceiro mas substituindo o *Raspberry PI* por um computador. O *script* utilizando, simula um usuário abrindo vários programas já inclusos na instalação do sistema operacional, como aplicações do LibreOffice

[1] É utilizado no texto, para definir quantitativamente a relação do usuário com o computador.

(Writer, Calc e o Draw). Também foi iniciado o navegador Firefox e dois games, Sudoku e o Paciência. O *script* utilizado inicializou as aplicações em tempos predeterminados, que pode ser visto no **Apêndice C**.

Os dois últimos testes dessa bateria serviram para mostrar uma comparação entre o *Raspberry PI* e um computador relacionado ao consumo de memória e transferência de dados pela rede. A partir do terceiro teste foram executados *scripts* no servidor para coletar dados sobre o uso dos recursos. Cada teste foi executado três vezes, afim de gerar uma média de todos os valores obtidos.

3.2 Servidor

Para que um computador venha a se tornar um servidor LTSP, é necessário instalar um pacote contido em um dos repositórios padrões do Ubuntu. O pacote se chama *ltsp-server-standalone*. No ato da instalação ele menciona outros pacotes que são essenciais para fornecer o serviço aos seus *thin clients*.

Para a instalação dos pacotes foi utilizado o recurso *apt-get*, que permite a instalação, a atualização de pacotes e a solução das dependências dos pacotes (UBUNTU, 2014). O comando usado para a instalação do pacote principal LTSP no Ubuntu foi:

```
sudo apt-get install ltsp-server-standalone
```

Após a instalação do pacote *ltsp-server-standalone* com suas dependências o computador já possuía todos os serviços e configurações base para executar a função de servidor. Mas foi necessário definir configurações que variam de cada computador usado e de cada rede local.

As alterações foram feitas em arquivos de configuração, para definir qual interface de rede foi disponibilizada para os *thin clients* acessarem o servidor, configurar o DHCP[2] (*Dynamic Host Configuration Protocol*) para distribuir os IPs (*Internet Protocol*) aos computadores. Para determinar qual interface de rede foi usada pelos *thin clients*, foi alterado o seguinte arquivo:

```
/etc/default/isc-dhcp-server
```

Dentro desse arquivo deve possui uma variável com o nome de *INTERFACES*. Foi inserida na variável a interface que se comunica com os *thin clients*. No computador usado para teste foi utilizada a interface “eth0”.

[2] É um protocolo que permite todos os computadores da rede recebam suas configurações de rede automaticamente a partir de um servidor central (MORIMOTO, 2005b).

```
INTERFACES = "eth0"
```

O Próximo passo é configurar o serviço de DHCP. A placa de rede ligada aos *thin clients* deve ter um endereço IP fixo e distribuir para cada cliente um endereço IP, informando a máscara da rede e o *gateway* padrão. Para configurar o serviço foi acessado o arquivo:

```
/etc/network/interfaces
```

Nesse arquivo foi adicionado o trecho apresentado a seguir, caso não exista linhas de configurações para a interface desejada. A linha iniciada com "auto ..." lista as interfaces que devem ser ativadas automaticamente e as demais linhas abaixo possui a configuração da sua respectiva interface (MORIMOTO, 2012; BARBOSA, 2006). Foram adicionados novas configurações no final do arquivo.

```
...
auto eth0
iface eth0 inet static
address 192.168.0.1
netmask 255.255.255.0
network 192.168.0.0
broadcast 192.168.0.255
```

A configuração supracitada foi usada para aproveitar a configuração padrão do pacote *ltsp-server-standalone*. Caso seja necessário alterar alguma das configurações, é necessário modificar o arquivo *dhcpd.conf* que está localizado no diretório:

```
/etc/ltsp/dhcpd.conf
```

Nesse arquivo são encontradas as configurações relativas ao serviço de DHCP e também da distribuição de arquivos do servidor aos *thin clients*. Este arquivo de configuração é próprio do ambiente *thin client*. A alteração nesse arquivo afetará como o serviço de DHCP do LTSP irá disponibilizar os recursos iniciais aos clientes.

O próximo passo na configuração do servidor LTSP, é a criação da imagem do sistema utilizada pelos clientes LTSP. No entanto, os clientes que acessaram o servidor utilizando o *Raspberry PI*, tiveram problemas com a interface Unity utilizada por padrão no Ubuntu a partir da versão 11.04. Para solucionar este problema foi alterada a interface padrão do Ubuntu para uma interface *Gnome Classic* chamada de *gnome-session-fallback*, baixada e instalada com o seguinte comando:

```
sudo apt-get install gnome-session-fallback
```

Com a interface já instalada foi utilizado o seguinte comando para definir a interface do *Gnome Classic* como a interface padrão:

```
sudo /usr/lib/lightdm/lightdm-set-defaults -s gnome-fallback
```

Com essa alteração já é possível iniciar a criação da imagem do sistema utilizado pelos clientes LTSP. Para a criação da imagem é usado um comando que pode variar de acordo com arquitetura do servidor, e principalmente com a arquitetura dos clientes. O comando para a criação da imagem é:

```
ltsp-build-client
```

Esse comando gerará uma imagem com a mesma arquitetura do SO usado no servidor, mas como o servidor atende clientes que utilizam *Raspberry PI* com a arquitetura ARM de 32 bits, sendo obrigatório que a imagem gerada seja de 32 bits. Para isso foi utilizado o seguinte comando:

```
ltsp-build-client --arch="i386"
```

O comando *ltsp-build-client* é responsável por gerar o SO a ser usado pelo *thin client*. Essa criação faz o SO do *thin client* ser independente do sistema operacional do servidor, deste modo, é possível instalar e remover programas no servidor sem alterar o SO do cliente e vice-versa. Após a criação do SO, é gerada a imagem do mesmo a ser utilizada pelos clientes *thin client*. O caminho para esse sistema gerado e para a imagem de disco criada, ficam armazenadas em

```
/opt/ltsp/
```

O comando *ltsp-build-client* pode ser dividido em quatro passos. No primeiro passo ele busca todos os pacotes do SO do servidor. No passo dois, já com todos os pacotes necessários ao funcionamento do sistema, é iniciado o processo de descompactação dos pacotes.

No passo três ele irá baixar da Internet todos os programas e aplicativos existentes no SO do servidor. Esses programas são instalados e configurados no diretório pertencente ao sistema operacional do *thin client*. Em seguida, no último passo, ele monta uma imagem de disco que é armazenada em um local que todos os *thin clients* acessam na hora de sua inicialização.

Com a finalização desse comando, o servidor está apto a iniciar clientes LTSP pela interface de rede selecionada. E as alterações realizadas no SO do servidor, como instalação

ou remoção de algum programa, não serão repassadas aos clientes, pois são dois sistemas distintos.

Como são dois sistemas operacionais distintos, um para o servidor e outro para os clientes, é possível acessar o SO do cliente para instalar e remover programas usando o terminal. Para efetuar essas modificações no SO do cliente, utiliza-se o seguinte comando:

```
sudo chroot /opt/ltsp/i386/ <comando desejado>
```

Os comandos existentes são limitados devido ao sistema não estar em funcionamento. Um exemplo é acessar a máquina cliente executando o comando de *ifconfig*. Será acusado um erro por não existir interface de rede para o SO informar.

Na execução de comando que gere alguma alteração no SO do cliente, a mesma só poderá ser visualizada após uma atualização na imagem de disco gerada no final do comando *ltsp-build-client*, em um arquivo com a extensão .img usado como SO pelos *thin clients*. Para atualizar esse arquivo, é necessário executar esse comando:

```
sudo ltsp-update-image
```

Esse comando monta a imagem do SO do cliente da mesma maneira do que ocorre no quarto passo da execução do comando *ltsp-build-client*.

Com o conhecimento de todos os passos para a construção de um servidor LTSP, foi desenvolvido um *script* (disponivel no **Apêndice A)** para a executação de todos os passos, deixando o computador/servidor configurado e pronto para ser usado como servidor LTSP. O *script* reconhece todas as interfaces de rede e permite escolher a interface que será usada pelos clientes LTSP para o acesso ao servidor.

3.3 Raspberry PI

A tarefa de utilizar um computador como *thin client* é simples. Basta ir nas configurações de *boot*, e escolher a opção de inicializá-lo pela rede, algo que existe na maioria dos computadores. Mas alguns computadores antigos não possuem essa opção. Então utiliza-se outro método, que é iniciar o computador através de um CD que possui um programa especifico para permitir a inicialização do computador pela rede.

O *Raspberry PI* não possui nenhum tipo de *software* embutido como a BIOS[3] (*Basic Input/Output System*), ou seja, diferente de um computador que possui a BIOS, o *Raspberry PI* de modo geral não pode iniciar *boot* sem acessar seu modo de armazenamento

[3] É um programa de computador pré-gravado em memória permanente (*firmware*) executado por um computador quando ligado. Ele é o suporte básico de acesso ao *hardware* (MORIMOTO, 2005a).

principal, que é o cartão de memória. Por isso não é possível utilizar o *Raspberry PI* sem um cartão de memória.

Então, para utilizar o *Raspberry PI* como *thin client*, foi usado um SO básico no cartão de memória para que pudesse ser inicializado pela rede, uma solução semelhante à usada em computadores antigos. Sendo assim utilizamos um projeto de chamado de BerryTerminal.

Para isso é necessário baixar o projeto na página principal do BerryTerminal disponível em <http://berryterminal.com/doku.php>. Com o download da versão comprimida, o processo e simples. Basta descompactar o arquivo dentro do um cartão de memória vazio. Caso use o projeto existente no GitHub, é necessário compilar o projeto executando o *build.sh*, e transferindo os arquivos de dentro pasta chamada "*output*", criada após a compilação, para o cartão de memória vazio.

Com isso, já é possível utilizar o *Raspberry PI* como um *thin client*. Isso se a saída de vídeo utilizada for com um monitor com entrada HDMI ou a saída de vídeo secundária, pois para utilizar o *Raspberry PI* é necessário que uma das saídas de vídeo estejam conectadas.

Como os testes foram feitos em um laboratório onde os monitores só possuam entradas VGA, foi adquirido um adaptador que transforma a saída HDMI em VGA. Mas a tecnologia usada na saída do HDMI do *Raspberry PI* não reconhece a conexão feita por um adaptador. Mediante ao problema, foi alterado o arquivo de configuração dentro do cartão de memória com o nome:

```
config.txt
```

As configurações adicionadas dentro do arquivo têm o intuito de forçar a o reconhecimento do adaptador da maneira correta. Para esse feito foi adicionado três comandos no final do arquivo, cada um com seu respectivo valor.

```
...
hdmi_force_hotplug=1
hdmi_group=2
hdmi_drive=2
```

A utilização do *hdmi force hotplug* com o valor 1 informado que a saída de vídeo será o HDMI, mesmo que o *Raspberry PI* não tenha detectado. Isso é necessário porque o *Raspberry PI* não identifica o conversor VGA. Com isso é cancelada a porta de saída de vídeo hdmi (VILLA, 2013).

Já o *hdmi group*, controla a resolução e frequência de saída. O valor 1 é usado com monitores de alta definição. E o valor 2 é usado para conectar monitores de computador convencionais (VILLA, 2013).

O *hdmi drive*, altera a tensão da porta de saída, que tem os valores 1 ou 2. O 1 refere-se a tensão para DVI (*Digital Visual Interface*) sem som, enquanto 2 funciona com som o que é padrão do HDMI (VILLA, 2013).

4 RESULTADOS E CONCLUSÃO

4.1 Resultados obtidos

Durante a implantação do servidor LTSP foi utilizado vários conteúdos existentes em páginas *web*, comunidades e fóruns relacionados ao assunto, como as informações disponibilizadas por Junior (2011), Almeida (2013), Lavender (2014), Morimoto (2005), Xavier (2015) e Dias (2014).

O motivo de usar o Ubuntu 12.04 LTS (*Long Term Support*) no servidor LTSP ao invés da versão 14.04 LTS, ocorreu pelo fato de na versão 14.04 LTS conter configurações no pacote do *gnome-session* que tornam obrigatória a aceleração de *hardware* em computadores que forem inicializados pela rede. Esse é o caso dos *thin clients* de um servidor LTSP, algo que o *Raspberry PI* não possui, tornando a versão do Ubuntu inutilizável para os testes.

Na realização dos testes, os resultados de usabilidade não foram favoráveis ao *Raspberry PI* utilizando o *software* BerryTerminal. Um problema encontrado no teste de usabilidade, foi impossibilidade de acessar um dispositivo de armazenamento USB, pois o próprio *software* BerryTerminal não reconhece o dispositivo.

Mas o problema que conexão o USB vai além do *software*, o modelo do *Raspberry PI* utilizado possui um problema de alimentação, onde dependendo da energia solicitada pelo dispositivo conectado via USB, causa uma queda de energia deligando o *Raspberry PI* por instantes o que na aplicação atual desliga a conexão com o servidor.

Na utilização do computador como *thin client*, foi possível acessar o dispositivo de armazenamento conectado na entrada USB, tornando possível a transferência de dados entre o servidor e o dispositivo de armazenamento conectado no *thin client*.

Durante os testes para comparar a usabilidade do computador e *Raspberry PI*, o computador desempenhou melhor o seu papel, sem nenhum tipo de *delay* na renderização da tela. Na utilização do *Raspberry PI* como *thin client*, foi notado alguns *delays* na renderização da tela durante a execução de alguns programas como firefox, Sudoku e o gerenciador de arquivos. O delay pode ser notado durante a inicialização dos programas.

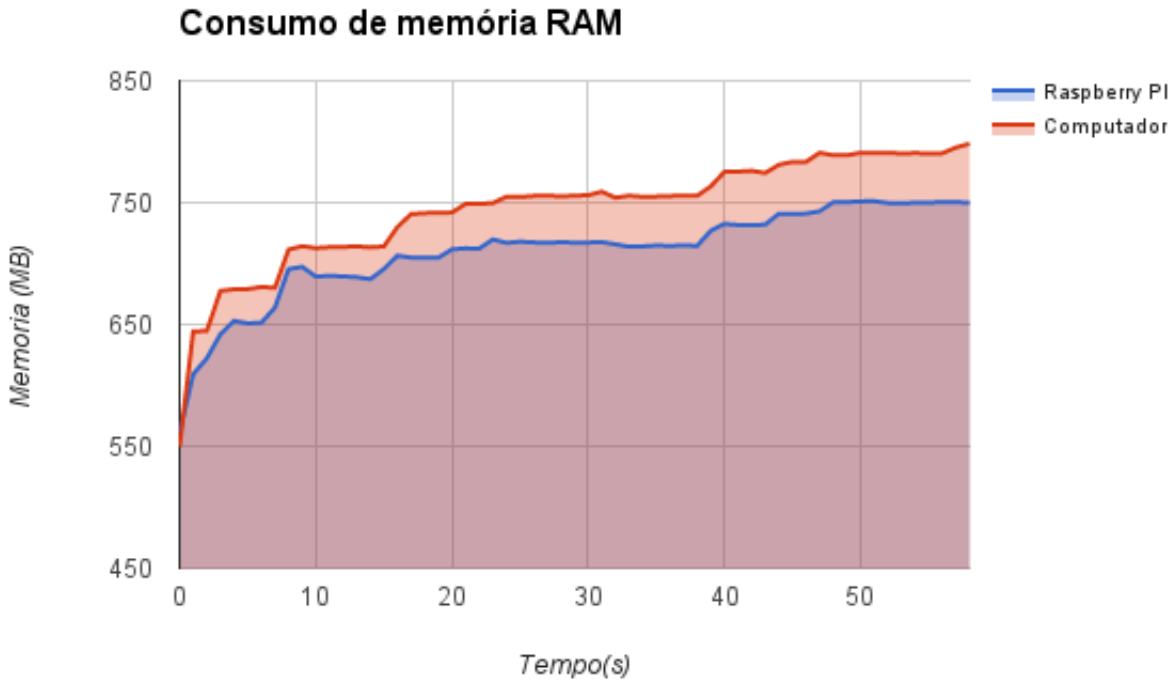
Um ponto marcante na usabilidade, foi a execução de vídeos em *stream* utilizando o navegador, onde o *Raspberry PI* deixou a desejar, executado vídeos com poucos *frames* por segundo, piorando o problema quando se reproduzia vídeos em tela cheia. Enquanto o computador desempenhou seu papel corretamente.

No restante dos testes foi utilizo *scripts* tanto no servidor para coletar as informações do sistema (que estão disponíveis no **Apêndice B**), quanto nos clientes para a execução de

programas do sistema para simular a execução de um usuário. Cada teste só é efetuado após o reinicio de todas as máquinas para que testes anteriores não interfiram nos resultados obtidos.

Já nos testes onde ocorreu a coleta de informações do sistema, existe uma grande semelhança na utilização dos recursos do servidor, em ambos casos. A **Figura 1** ilustra a quantidade de memória utilizada do servidor no momento dos testes do *Raspberry PI* e do computador, ambos sendo utilizados como *thin client*.

Figura 1 – Quantidade de memória usada no servidor durante o uso do *Raspberry PI* com *thin client* e do computador como *thin client*.

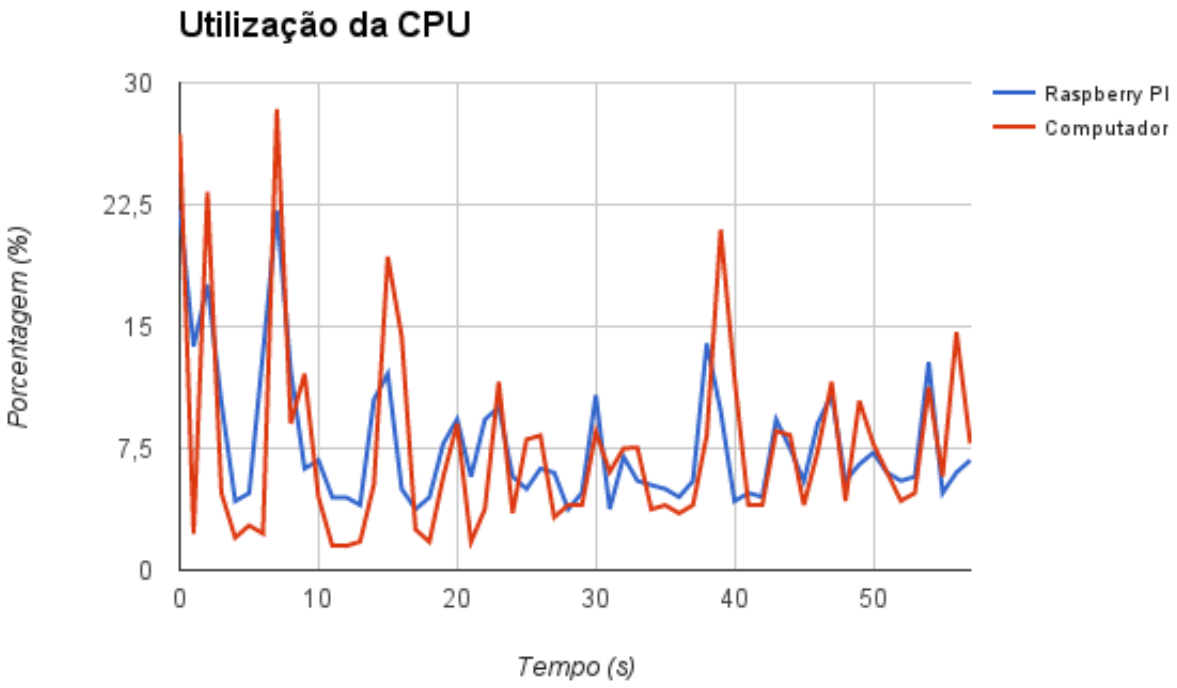


Fonte: Elaborado pelo autor.

Os dados apresentados na **Figura 1** indicam que o *Raspberry PI* utilizou em quase todo o período do teste uma quantidade menor de memória RAM que o computador, ambos executados nas mesmas condições. Durante a coleta desses dados utilizando o *Raspberry PI*, o pico de memória foi de 751,2109 MB, enquanto o computador utilizou o teto de 798,3632 MB. Uma pequena diferença, levando em conta que o servidor antes de iniciar os testes utilizava entre 561,1641 MB e 549,8554 MB.

A **Figura 2** ilustra a utilização da CPU durante o período dos testes três e quatro, onde é notado uma alternância entre o computador e o *Raspberry PI*, no maior uso da CPU. O que indica uma equivalência entre as ambas partes, já que a diferença é pequena do uso da CPU do servidor solicitada pelos *thin clients* no mesmo instante do teste.

Figura 2 – Porcentagem da utilização da CPU do servidor durante o uso do *Raspberry PI* com *thin client* e do computador como *thin client*.

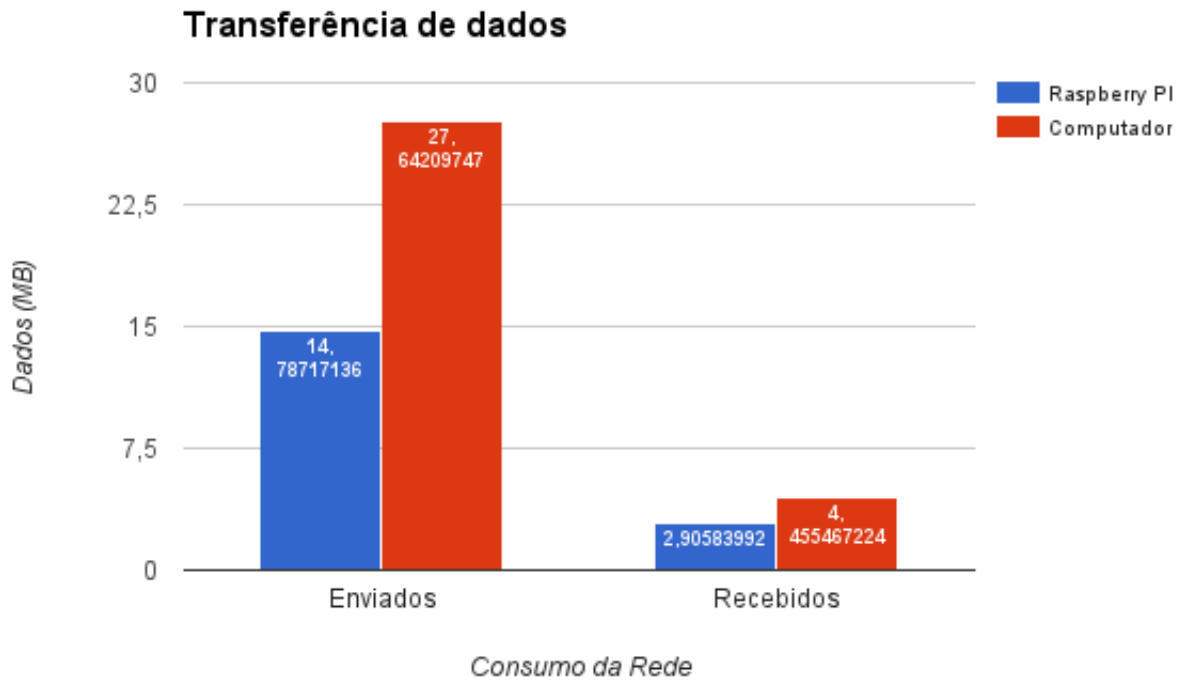


Fonte: Elaborado pelo autor.

A última informação coletada dos testes está relacionada a transferência de dados pela rede, tendo em vista que para um melhor desempenho no ambiente *thin client*, a rede local não pode ficar sobrecarregada. Foi coletada a quantidade de dados transferidos pela interface do servidor no instante de iniciou os testes e outra coleta após o tempo limite dos testes. Sendo assim possível verificar a quantidade de dados enviados e recebidos da interface do servidor usada para a comunicação com os *thin clients*.

A **Figura 3** informa a quantidade de dados enviados e recebidos durante os testes. Lembrando que esses dados foram coletados do servidor.

Figura 3 – Quantidade de dados transferidos pela rede durante o uso do *Raspberry PI* com *thin client* e do computador como *thin client*.



Fonte: Elaborado pelo autor.

A **Figura 3** apresenta os resultados na utilização do *Raspberry PI* como *thin client*, onde necessitou de 14,78717136 MB de dados enviados do servidor para executar os testes, isso é aproximadamente 53,47% da quantidade solicitada pelo computador no modo *thin client*. Os resultados continuam favoráveis ao *Raspberry PI* nos dados que são enviados para o servidor, pois o ele enviou aproximadamente 65,21% de dados que o computador.

As informações apresentadas na **Figura 3** indicam que a utilização de *Raspberry PI* no ambiente *thin client* torna a rede local menos congestionada, transferindo menos dados e consequentemente ocupando menos largura de banda. Mas isso provavelmente ocorre pelo fato do *Raspberry PI* ser um pouco mais lento durante os testes, processando menos frames de vídeo transferidos pela rede.

4.2 Conclusão

O *Raspberry PI* modelo B usando o BerryTerminal, traz benefícios ao ambiente *thin client*, como a diminuição da utilização dos recursos do servidor, a economia de energia utilizada e a economia para criação de infraestrutura de computadores. A sua utilização se enquadra no conceito de TI verde, ou *Green IT*, pois envolvendo uma nova estrutura de

refrigeração e um melhor aproveitamento dos espaços físicos.

O *Raspberry PI* como um *thin client*, nos teste consumiu cerca de 6,27% menos memória RAM que um computador com *thin client*. Essa diferença em um ambiente com várias maquinas, talvez possibilita talvez a inserção de mais *Raspberrys PI* do que computadores. Além de utilizar uma quantidade menor da banda em sua rede local devido ao déficit na usabilidade.

Nos testes executados não foi possível acessar dispositivos de armazenamento conectados via USB no *Raspberry PI*, impossibilitando a transferência dos dados utilizados no ambiente *thin client*. Uma alternativa que viabiliza essa transferência é a conexão do dispositivo no servidor, algo incomum pois na maioria das vezes não é acessível aos usuários.

A usabilidade do *Raspberry PI* como *thin client* não gerou resultados favoráveis, já que ele teve um mal desempenho na execução de vídeos e dificuldades na exibição da tela como a perda de algumas transições.

No geral, os resultados da comparação entre o computador e o *Raspberry PI*, ambos como *thin client*, foram neutros já que o *Raspberry PI* teve um melhor desempenho na utilização dos recursos e o computador desempenhou um ótimo papel usabilidade e na fluidez da utilização. Mas o *Raspberry PI* possui outras vantagens em relação ao computador sendo um *thin client*, como preço e a economia de energia, tanto diretamente quanto indiretamente.

Com esses dados é possivel dizer que o *Raspberry PI* é uma opção de *thin client*, mas vale ressaltar que é preciso verificar qual a finalidade do ambiente *thin client*, pois o *Raspberry PI* com o BerryTerminal não é aconselhado em ambientes onde ocorre edição de vídeos ou imagens. Mas recomentado em ambientes onde se utiliza nuvens de armazenamento de dados.

Para a obtenção de resultados mais precisos, é necessário avançar com a pesquisa produzindo outros testes com diferentes modelos de *Raspberry PI* para verificar se os problemas encontrados, estão presentes apenas no modelo usado, ou não. Também buscar outros *softwares* que permitam que o *Raspberry PI* seja um *thin client*, semelhante ao que o BerryTerminal faz.

É possivel também verificar a quantidade de *thin clients* suportados por uma rede local. Ou realização de testes para determinar o desempenho de uma thin client a grande distância, determinando até que distância o *Raspberry PI* como *thin client* traz bons resultados.

REFERÊNCIAS

ALMEIDA, L. F. de. *MONTE SEU PRÓPRIO LTSP (LINUX TERMINAL SERVER PROJECT)*. 2013. Disponível em: <http://sejalivre.org/monte-seu-proprio-ltsp-linux-terminal-server-project/>. Acesso em: 28-07-2015.

ATERA. *Thin Client Encore ENTC-1000 p/ terminal Windows/Linux*. 2015. Disponível em: <http://www.atera.com.br/produto/ENTC-1000/>. Acesso em: 18-04-2015.

BARBOSA, T. C. R. Ltsp5. Universidade Estadual de Goiás. 2006. Disponível em: <http://arquivos.suporte.ueg.br/linux_ltsp.pdf>.

BOS, F. *BerryTerminal*. 2014. Disponível em: <http://www.berryterminal.com/doku.php>. Acesso em: 18-03-2015.

DAWLE, A. S. S. *Raspberry Pi-Based dispositivo Braille Aprendizagem: Projeto Mudra*. 2014. Disponível em: <http://technabob.com/blog/2014/04/27/raspberry-pi-braille-learning-device/>. Acesso em: 30-05-2015.

DELL. *Thin client Dell Wyse D10DP*. 2015. Disponível em: <http://www.dell.com/br/p/wyse-d10dp/pd>. Acesso em: 18-04-2015.

DIAS, P. *Como montar um servidor LTSP no Ubuntu*. 2014. Disponível em: <http://www.prminformatica.com.br/2014/02/montar-servidor-ltsp-no-ubuntu.html>. Acesso em: 27-07-2015.

FARIA, C. P. de S. *Thin Clients: Soluções, Implementação e Desempenho*. Dissertação (Mestrado) — UNIVERSIDADE FEDERAL DO PIAUÍ - UFPI, 2009. Disponível em: <http://www.ufpi.br/subsiteFiles/eml/arquivos/files/tcc/clistenes.pdf>.

FILIPELOP. *EMBARCADOS*. 2015. Disponível em: <http://www.filipeflop.com/embarcados-ct-93348?pageNum=1&sortBy=1>. Acesso em: 30-05-2015.

HEIN, W. *Raspberry Pi aplicado a projetos do mundo real*. Linux Magazine, 2013. Disponível em: <http://www.linuxnewmedia.com.br/images/uploads/pdf_aberto/LM_100_60_65_06_tut_raspberry_pi.pdf>. Acesso em: 28-05-2015.

HUNT, D. *Câmara Pi - DSLR Camera com computador embutido*. 2012. Disponível em: <http://www.davidhunt.ie/raspberry-pi-in-a-dslr-camera/>. Acesso em: 30-05-2015.

IBM. *Visão Geral sobre a Topologia de Cliente Thin (Windows)*. 2014. Disponível em: <http://www-01.ibm.com/support/knowledgecenter/SSEPGG_9.5.0/com.ibm.db2.luw.qb.client.doc/doc/c0007236.html?lang=pt-br>. Acesso em: 18-07-2015.

JUNIOR, H. N. P. *FUNCIONAMENTO DO LTSP E INSTALAÇÃO DO SERVIDOR*. 2011. Disponível em: <http://www.vivaolinux.com.br/artigo/Instalacao-e-configuracao-do-LTSP-5-no-Ubuntu-11.04>. Acesso em: 27-07-2015.

LAVENDER, H. *Como montar um servidor LTSP no Ubuntu*. 2014. Disponível em: <http://www.uzerp.com/blog/running-raspberry-pis-as-thin-clients-with-ubuntu-14-04-lts/>. Acesso em: 27-07-2015.

LINUX SUPPORT. *Configuração do NIS*. 2007. Disponível em: <http://www.linuxsupport.com.br/configuration/nis.html>. Acesso em: 27-10-2015.

LOJA WT. *Thin Client Nc600*. 2015. Disponível em: <http://www.lojawt.com.br/informatica/thin-client-nc600.html>. Acesso em: 18-04-2015.

LTSP.ORG. *Introduction to LTSP*. 2014. Disponível em: <http://www.ltsp.org/>. Acesso em: 20-05-2015.

MICROSOFT. *Desktop Virtualization*. [S.l.], 2011.

MORIMOTO, C. *Redes e servidores Linux: guia prático*. [S.l.]: Sul Editores, 2005. ISBN 9788520504048.

MORIMOTO, C. *A revolução do Raspberry Pi*. 2012. Disponível em: <http://www.hardware.com.br/artigos/raspberrypi/>. Acesso em: 28-05-2015.

MORIMOTO, C. *SERVIDORES LINUX - GUIA PRATICO*. [S.l.]: SULINA, 2013. ISBN 9788599593134.

MORIMOTO, C. E. *BIOS*. 2005. Disponível em: <http://www.hardware.com.br/termos/bios>. Acesso em: 27-10-2015.

MORIMOTO, C. E. *DHCP*. 2005. Disponível em: <http://www.hardware.com.br/termos/dhcp>. Acesso em: 27-10-2015.

MORIMOTO, C. E. *PXE*. 2005. Disponível em: <http://www.hardware.com.br/termos/pxe>. Acesso em: 27-10-2015.

MULTILOGICA. *Raspberry Pi modelo B - 512 Mb (descontinuado)*. 2014. Disponível em: <https://multilogica-shop.com/raspberry-pi-modelo-b>. Acesso em: 08-05-2015.

RASPBERRY PI FOUNDATION. *WHAT IS A RASPBERRY PI?* 2015. Disponível em: <http://www.raspberrypi.org/help/what-is-a-raspberry-pi/>. Acesso em: 25-03-2015.

RASPBERRY PI PORTUGAL. *RASPBERRY PI MODELO A*. 2013. Disponível em: <http://raspberrypiportugal.com/modelos/raspberry-pi-modelo-a/>. Acesso em: 30-05-2015.

RAUCHWERK GUSTAVO HUBER, T. D. D. *Meet OTTO - The Hackable GIF Camera*. 2014. Disponível em: <https://www.kickstarter.com/projects/1598272670/meet-otto-the-hackable-gif-camera>. Acesso em: 30-05-2015.

REDAÇÂO GALILEU. *Raspberry Pi – o computador que custa 25 dólares*. 2015. Disponível em: <http://revistagalileu.globo.com/Revista/Common/0,,EMI330119-17770,00-RASPBERRY+PI+O+COMPUTADOR+QUE+CUSTA+DOLARES.html>. Acesso em: 18-07-2015.

RICHARDS, D. *Linux Thin Client Networks Design and Deployment*. [S.l.]: Packt Publishing, 2007. (From technologies to solutions). ISBN 9781847192059.

ROBO LIVRE. *Arquitetura ARM*. 2011. Disponível em: <http://robolivre.org/uploads/documentos/arquiteturas-de-computadores/ARM-2011-08-08_1_1343264119.pdf>. Acesso em: 27-10-2015.

SILBER, J. *Q&A: Canonical's Jane Silber says upcoming Ubuntu Linux to be enterprise-ready*. 2006. Disponível em: <http://www.computerworld.com/article/2554755/linux/q-a--canonical-s-jane-silber-says-upcoming-ubuntu-linux-to-be-enterprise-ready.html>. Acesso em: 27-10-2015.

SOURCEFORGE.NET. *About DRBL*. 1999. Disponível em: <http://drbl.org/>. Acesso em: 20-05-2015.

TANENBAUM, A. *Sistemas operacionais modernos*. [S.l.]: Prentice-Hall do Brasil, 2010. ISBN 9788576052371.

THIN CLIENT BRASIL. *O que é thin client?* 2015. Disponível em: <http://www.thinclientbrasil.com/thin-client.php>. Acesso em: 18-05-2015.

THINCLIENTBRASIL. Apresentação tcbr soluções. Mensagem Pessoal. 2015.

THINCLIENTBRASIL. *Como funciona o thin client?* 2015. Disponível em: <http://www.thinclientbrasil.com/thin-client/como-funciona-thin-client.php>. Acesso em: 18-07-2015.

THINSTATION. *FAQ*. 2014. Disponível em: <https://github.com/Thinstation/thinstation/wiki/FAQ>. Acesso em: 26-05-2015.

TREACY, M. *20 Awesome Projects for Raspberry Pi Microcomputers*. 2013. Disponível em: <http://www.treehugger.com/slideshows/gadgets/20-awesome-projects-raspberry-pi-microcomputers/page/20/>. Acesso em: 28-05-2015.

TUZI, F. *DRBL: A quick and easy thin client server*. 2006. Disponível em: <http://archive09.linux.com/feature/57273>. Acesso em: 25-05-2015.

UBUNTU. *Package management with APT*. 2014. Disponível em: <https://help.ubuntu.com/community/AptGet/Howto>. Acesso em: 07-10-2015.

UPTON, E.; HALFACREE, G.; PASCHOA, C. *Raspberry Pi – Manual do Usuário*. [S.l.]: NOVATEC, 2013. ISBN 9788575223512.

VILLA, L. *TUTORIAL RASPBERRY PI – USO DE CONVERTIDORES HDMI-VGA*. 2013. Disponível em: <https://geekytheory.com/tutorial-raspberry-pi-uso-de-convertidores-hdmi-vga/>. Acesso em: 10-06-2015.

XAVIER, D. V. *Utilização e Operação do NFS*. 1996. Disponível em: <http://www.di.ufpe.br/~flash/resultados/cursos/taais/1996-1/danise/paper3.html>. Acesso em: 27-10-2015.

XAVIER, K. S. *Configurando a Rede e o Servidor DHCP - LTSP (Terminais Leves) - Parte 2*. 2015. Disponível em: <http://www.devmedia.com.br/configurando-a-rede-e-o-servidor-dhcp-ltsp-terminais-leves-parte-2/22052>. Acesso em: 27-07-2015.

Apêndices

APÊNDICE A – SCRIPT PARA CRIAR SERVIDOR LTSP

Script também está acessível no repositório público do trabalho disponível em <https://github.com/FelipeLimaM/thin-client-raspberry-pi-tcc/tree/master/thin_client/build_server>

build_server.sh

```
#!/bin/bash

# author Felipe Lima Morais - Graduando de Ciência da Computação
CHOICE_INTERFACE_DIR=/etc/default/isc-dhcp-server
CONF_INTERFACE_DIR=/etc/network/interfaces

NEW_CONF=ConfInterface
NEW_CONF_BAK=ConfInterface.backup

#verifica se é root
if [[ $EUID -ne 0 ]]; then
    echo "This script must be run as root" #1>&2
else
    INSTALL_PACKAGE=ltsp-server-standalone

    clear
    echo " update packages"
    sleep 1
    sudo apt-get install update

    clear
    echo " upgrade of packages"
    sleep 1
    sudo apt-get upgrade -y

    clear
    echo " download/install package ltsp-server-standalone install"
```

```
    sleep 2
    sudo apt-get install $INSTALL_PACKAGE -y

# instala e configura o gnome-fallback
sudo apt-get install gnome-session-fallback -y
sudo/usr/lib/lightdm/lightdm-set-defaults -s gnome-fallback

#verifica se o sistema é 32 ou 64 bits
    if ((1<<32)); then
        echo your 64bits system
        ltsp-build-client --arch="i386"
    else
        echo you 32bits system
        ltsp-build-client
    fi
clear

# encontra todas as interfaces do sistema
# ALTER  /etc/default/isc-dhcp-server
connect=$(ifconfig |  grep -E encap | awk -F 'Link' '{print $1 }' \
| awk -F '$' '{print $1 }')
echo "choose the interface to the LTSP environment?"
echo ""
select result in $connect
do
if [ -n "$result" ]; then

INTERFACE=$result
sed -i 's/INTERFACES\=.*/INTERFACES\="'$result'\"/g' $CHOICE_INTERFACE_DIR
break
fi
done

#cria um backup do arquivos alterados para restaurar (segurança apenas)
# BACKUP / ALTER  /etc/network/interfaces

(cat $NEW_CONF) > $NEW_CONF_BAK

#substitui os dados para inserir a interface selecionada anteriormente
```

```
sed -i 's/???/'$result'/g' $NEW_CONF

(cat $NEW_CONF) >> $CONF_INTERFACE_DIR

(cat $NEW_CONF_BAK) > $NEW_CONF

rm $NEW_CONF_BAK

#reinicia o computador apos um ENTER
clear
echo "click ENTER to restart the system"
read
sudo reboot

fi
--

ConfInterface

auto ???
iface ??? inet static
        address 192.168.0.1
        network 192.168.0.0
        netmask 255.255.255.0
        broadcast 192.168.0.255
--
```

APÊNDICE B – SCRIPTS PARA COLETAR DADOS DO SERVIDOR

Os Scripts também são acessíveis no repositório público do trabalho disponível em <https://github.com/FelipeLimaM/thin-client-raspberry-pi-tcc/tree/master/thin_client/test>

get_data_server.sh

```
#!/bin/bash

file_memory='memory'
file_cpu='cpu'

#data/hora do inicio e fim dos testes
start_time=`date -d "10/19/2015 23:50" +%s`
end_time=`date -d "10/19/2015 23:51" +%s`

#verifica se é root
if [[ $EUID -ne 0 ]]; then
    echo "This script must be run as root" #1>&2
else
#inicializa a coleta do consumo de memória
echo "wait.."
bash collect_data_memory.sh & #start collection
aux=$(cat exec.wa)
pid_memory=`expr $aux - 2`
echo "pid_memory $pid_memory"

#inicializa a coleta do consumo da CPU
bash collect_data_cpu.sh & #start collection
aux=$(cat exec.wa)
pid_cpu=`expr $aux - 2`
echo "pid_cpu $pid_cpu"

#aguarda até a data/hora selecionada para o inicio
```

```
sleep $(expr $start_time - ‘date +%s‘) #start time to script
echo "go go $(date)"

#coleta os dados iniciais do transferencia de dados
start_RX=$(ifconfig eth0 | grep ’bytes’|\
 awk -F "(:|\()" ’{print $2}’) #get RX initial
start_TX=$(ifconfig eth0 | grep ’bytes’|\
 awk -F "(:|\()" ’{print $4}’) #get TX initial
start_memory=$(free -kt | grep Mem |\
 awk ’{print expr ($3 - ($6 + $7))}’) #get memory initial
start_cpu=$(cat <(grep ’cpu ’ /proc/stat) <(sleep 1 && grep ’cpu ’ /proc/stat) |\
 awk -v RS="" ’{print ($13-$2+$15-$4)*100/($13-$2+$15-$4+$16-$5)}’) #get cpu initial

rm $file_memory
rm $file_cpu

#loop
while [ true ]; do
echo "loop"
sleep 1
if [ $(date +%s) -ge $end_time ]; then
break;
fi
done

#separa os dados do servidor em arquivos separados (CPU, MEMORIA)
if [ -a test_cpu ]; then rm test_cpu; fi
if [ -a test_memory ]; then rm test_memory; fi
(cat $file_memory)>test_memory
(cat $file_cpu)>test_cpu

#coleta os dados finais do transferencia de dados
end_RX=$(ifconfig eth0 | grep ’bytes’| awk -F "(:|\()" ’{print $2}’) #transfer net
end_TX=$(ifconfig eth0 | grep ’bytes’| awk -F "(:|\()" ’{print $4}’) #transfer net

#solution data (net)
net_RX=‘expr $end_RX - $start_RX‘
net_TX=‘expr $end_TX - $start_TX‘
```

```
#solution data (memory)
memory_used=0
$start_memory
memory_used=(‘python -c "print (max([float(line.rstrip(’\n’))\
 for line in open(’test_memory’)]) - float($start_memory))/1024"‘)

#solution data (CPU)
cpu_used=0
cpu_max=(‘python -c "print max([float(line.rstrip(’\n’))\
 for line in open(’test_cpu’)])"‘)

date
echo -e "memory used \t $memory_used"
echo -e "trafic send \t $net_RX"
echo -e "trafic reciv \t $net_TX"
echo -e "cpu upper \t $cpu_max"
fi
--

collect_data_cpu.sh

#!/bin/bash

file=’cpu’

if [ -a $file ]; then rm $file; fi
(echo $BASHPID)>exec.wa

while (cat <(grep ’cpu ’ /proc/stat) <(sleep 1 && grep ’cpu ’ /proc/stat) |\
 awk -v RS="" ’{print ($13-$2+$15-$4)*100/($13-$2+$15-$4+$16-$5)}’)>>$file; do
    i=0;
done
--

collect_data_memory.sh

#!/bin/bash

file=’memory’
```

```
if [ -a $file ]; then rm $file; fi
(echo $BASHPID)>exec.wa

while (free -kt | grep Mem | awk '{print expr ($3 - ($6 + $7))}')>>$file; do
    sleep 1
done
--
```

APÊNDICE C – SCRIPT QUE SIMULAR A UTILIZAÇÃO DE UM CLIENTE

Script também está acessível no repositório público do trabalho disponível em <https://github.com/FelipeLimaM/thin-client-raspberry-pi-tcc/blob/master/thin_client/test/ltsp_client.sh>

ltsp_client.sh

```
#!/bin/bash

arr=("firefox" "libreoffice --writer" "rhythmbox" "libreoffice --calc"\
 "/usr/games/gnome-sudoku" "libreoffice --draw" "libreoffice --calc"\
 "/usr/games/sol");

temp=(3 5 2 6 1 0 4 3 2 8 4 1 4 5 1 6 3 1 6 7 3 1 5 2 5);

n=‘expr ${#arr[@]} - 1‘

#data/hora do inicio e fim dos testes
start_time=‘date -d "10/19/2015 23:50" +%s‘
end_time=‘date -d "10/19/2015 23:51" +%s‘

#aguarda até a data/hora selecionada para o inicio
sleep $(expr $start_time - ‘date +%s‘) #start time to script

while [ true ]; do

for i in ‘seq 0 $n‘; do
${arr[$i]} &
sleep ${temp[$i]}
done
#verifca se já deve terminar
if [ $(date +%s) -ge $end_time ]; then
break;
fi
```

```
done
--
```