PONTIFÍCIA UNIVERSIDADE CATÓLICA DECAMPINAS

CENTRO DE CIÊNCIAS EXATAS, AMBIENTAIS E DE TECNOLOGIAS - CEATEC

MESTRADO PROFISSIONAL EM GESTÃO DE REDES DE TELECOMUNICAÇÕES

LEANDRO FILIAGI MACHADO

PROXY IP DE BAIXO CUSTO E MÚLTIPLOS SENSORES PARA CIDADES INTELIGENTES

CAMPINAS
2015

LEANDRO FILIAGI MACHADO

PROXY IP DE BAIXO CUSTO E MÚLTIPLOS SENSORES PARA CIDADES INTELIGENTES

Dissertação apresentada ao Programa de Pós Graduação Stricto Sensu em Engenharia Elétrica do Centro de Ciências Exatas, Ambientais e de Tecnologias da Pontifícia Universidade Católica de Campinas como requisito para obtenção do título de Mestre em Engenharia Elétrica.

Orientador: Prof. Dr. Alexandre de Assis Mota

PUC-CAMPINAS
2015

Ficha Catalográfica
Elaborada pelo Sistema de Bibliotecas e
Informação - SBI - PUC-Campinas

t001.64404 Machado, Leandro Filiagi.
M149p Proxy IP de baixo custo e múltiplos sensores para cidades inteli-gentes / Leandro Filiagi Machado. - Campinas: PUC-Campinas, 2015.
107p.

Orientador: Alexandre de Assis Mota.

Dissertação (mestrado) – Pontifícia Universidade Católica de Campinas, Centro de Ciências Exatas, Ambientais e de Tecnologias, Faculdade de Engenharia Elétrica.
Inclui bibliografia.

1. Redes de computadores.2. Detectores. 3.Sistemas de comunicação sem fio. 4. Engenharia elétrica. 5. Interconexões de rede. I. Mota,Alexandrede Assis. II. Pontifícia Universidade Católica de Campinas.
Centro de Ciências Exatas, Ambientais e de Tecnologias. Faculdade de Engenharia Elétrica. III. Título.
18.ed. CDD – t001.64404

LEANDRO FILIAGI MACHADO

PROXY IP DE BAIXO CUSTO E MÚLTIPLOS SENSORES PARA CIDADES INTELIGENTES

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica do Centro de Ciências Exatas, Ambientais e de Tecnologias da Pontifícia Universidade Católica de Campinas como requisito parcial para obtenção do título de Mestre em Gestão de Redes de Telecomunicações.
Área de Concentração: Gerência de Redes de Teleinformática. Orientador: Prof. Dr. Alexandre de Assis Mota.

Dissertação defendida e aprovada em 18 de dezembro de 2015 pela Comissão Examinadora constituída dos seguintes professores:

Prof. Dr. Alexandre de Assis Mota
Orientador da Dissertação e Presidente da Comissão Examinadora
Pontifícia Universidade Católica de Campinas

Profa. Dra. Lia Toledo Moreira Mota
Pontifícia Universidade Católica de Campinas

Prof. Dr. Armando Villares Ferrer
INDRA Brasil

Dedico este trabalho aos meus pais, Heloisa e Leonardo, pelo incentivo e apoio em todas as minhas escolhas e decisões. Também à minha esposa, Patrícia Marques Creace, minha filha, Beatriz, e meu filho Lucas que acabou de nascer, pelo apoio, incentivo e compreensão para transpor todos os obstáculos.

A Vitória desta conquista dedico com todo meu amor, unicamente, a vocês!

AGRADECIMENTOS

Agradeço a todos os colegas mestrandos e professores pela dedicação, companheirismo e troca de experiências durante o curso.

Ao Prof. Dr. Alexandre de Assis Mota,
Orientador e incentivadornas pesquisas e trabalhos que realizei no Programa de Pós-Graduação em Engenharia Elétrica da Pontifícia Universidade Católica de Campinas, muito obrigado pelo apoio, atenção, dedicação e amizade;

ÀProfa. Dra. Lia Toledo Moreira Mota,
Por todos os esforços e por toda dedicação oferecida aos alunos do programa, também pelo exemplo de condução objetiva, altiva e otimista na criação de valores intelectuais;

Ao Prof. Dr. Omar Carvalho Branquinho,
Pelas importantes sugestões e incentivos durante o curso;

Ao Daniel e Ricardo, técnicos do Laboratório de Eletrônica,
Pela atenção, paciência, ajuda e amizade.

À Pontifícia Universidade Católica de Campinas,
Pela bolsa concedida durante o período do curso de Mestrado Profissional em Gestão de Redes de Telecomunicações.

Por fim, mas não menos importante, um agradecimento especial atodos os meus familiares, em particular à minha Mãe, que nunca deixoude acreditar em mim, e sempre me incentivou a continuar mesmo nassituações mais adversas.

“Talvez não tenha conseguido fazer o melhor, mas lutei para que o melhor fosse feito. Não sou o que deveria ser, mas Graças a Deus, não sou o que era antes”

(Martin Luther King)

RESUMO

MACHADO, Leandro Filiagi.**Proxy IP de baixo custo e múltiplos sensores para cidades inteligentes**. 2015. Dissertação (Mestrado em Engenharia Elétrica) – Programa de Pós-Graduação em Gestão de Redes de Telecomunicações, Pontifícia Universidade Católica de Campinas, Campinas, 2015.

Com o avanço da inteligência Urbana e as Cidades Inteligentes, a importância do monitoramento remoto e coleta de dados aumentou. Em relação ao monitoramento de dados, o ProxyIP é umaplataforma importante para a realizaçãoda interconexão entre os sensores e a Internet. Na literatura recente, várias arquiteturas de gateway para sensores de rede têm sido propostas para integrar Rede de Sensores sem Fio (RSSF) e Internet. A maioria das implementações de gateways de RSSF recuperam dados de sensores e exibem os resultados para os clientes via web. A desvantagem dessas soluções é que eles usam protocolos específicos para ligar os sensores, proibindo assim a interação direta entre clientes e nós sensores. Como opção, a adoção do protocolo SNMP para gerenciamento de sensores tem o potencial para reduzir a complexidade do gateway. Por esta razão, este trabalho apresenta a concepção e implementação de uma plataforma aberta e de baixo custo com o uso e modificação de um roteador sem fio comercial, com conexão serial para se comunicar com os sensores, que são ligado a um microcontrolador Arduino Nano. Os resultados das aplicações mostraram que o Proxy IP pode interligar com êxito multiplos sensores e a Internet, onde os dados podem ser transmitidos em todo o mundo através da Ethernet ou WLAN.

Palavras-chave:SNMP; Proxy;IP; Cidades Inteligentes; Sensores; Baixo Custo

ABSTRACT

MACHADO, Leandro Filiagi. **Proxy IP low cost and multiple sensors to smart cities**. 2015. Dissertação (Mestrado em Engenharia Elétrica) – Programa de Pós-Graduação em Gestão de Redes de Telecomunicações, Pontifícia Universidade Católica de Campinas, Campinas, 2015.

With the advancement of Urban Intelligence and Smart Cities, the importance of remote monitoring and data collection increased. Regarding the monitoring data, the IP Proxy is an important platform for the realization of interconnection between the sensors and the Internet. In recent literature, several gateway architectures for network sensors have been proposed to integrate Wireless Sensor Network (WSN) and Internet. Most WSN gateways implementations retrieve sensor data and display the results to customers via the web. The disadvantage of these solutions is that they use specific protocols to connect the sensors, thereby prohibiting direct interaction between clients and sensor nodes. Optionally, the adoption of sensors for SNMP management has the potential to reduce the complexity of the gateway. For this reason, this paper presents the design and implementation of an open platform and low cost with the use and modification of a non-commercial wireless router with serial connection to communicate with the sensors, which are connected to an Arduino Nano microcontroller. The results showed that applications of the IP Proxy can successfully connect multiple sensors and the Internet, where data can be transmitted all over the world via the Ethernet or WLAN

Keywords: SNMP; Proxy; IP; Smart Cities; Sensors; Low Cost

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

3G	=	Tecnologia Móvel de Terceira Geração
4G	=	Tecnologia Móvel de Quarta Geração
AP	=	*Acess Point*
CPU	=	*Central Processing Unit*
DashBoard	=	Painel de Instrumentos/Monitoramento
DHCP	=	*Dynamic Host Configuration Protocol*
DNS	=	*Domain Name Service*
EEPROM	=	*Electrically-Erasable Programmable Read-Only Memory*
FLASH	=	Memória Não Volátil, do tipo EEPROM
GPL	=	*General Public License*
HTTP	=	*Hypertext Transfer Protocol*
IDE	=	*Integrated Development Environment*
IEEE	=	*Institute of Electrical and Electronics Engineers*
IETF	=	*Internet Engineering Task Force*
IoT	=	Internet *of Things / Internet das Coisas*
IP	=	Internet *Protocol*
IPv6	=	Internet *Protocol* versão 6
LAN	=	*Local Area Network*
LDR	=	*Light Dependent Resistor*
MAC	=	*Medium Access Control*
MB	=	Mega Bytes
MBPS	=	Mega Bytes por Segundo
MIPS	=	*Microprocessor without Interlocked Pipeline Stages*
OID	=	*Object Identification*
OpenWrt	=	Linux *Open Source* para dispositivos embarcados
PC	=	*Personal Computer*
PCB	=	*Printed Circuit Board*
PUC-Campinas	=	Pontifícia Universidade Católica de Campinas
PWM	=	*Pulse Width Modulation*
QoS	=	*Quality of Service*

RAM	=	*Random Access Memory*
RFC	=	*Request for Comment*
RSSF	=	Rede de Sensores Sem Fio
RX	=	Recepção
SNMP	=	*Single Network Management Protocol*
SO	=	Sistema Operacional
SSH	=	*Secure Shell*
SWITCH	=	Comutador, Ponte MultiPortas
TCP	=	*Transmission Control Protocol*
TCP/IP	=	*Transmission Control Protocol /* Internet *Protocol*
TIC	=	Tecnologia da Informação e Comunicação
TX	=	Transmissão
UART	=	*Universal Asynchronous Receiver/Transmitter*
UDP	=	*User Datagram Protocol*
UPnP	=	*Universal Plug and Play*
USB	=	*Universal Serial Bus*
VIM	=	VI *Improved* (Editor Modal)
WAN	=	*Wide Area Network*
WEP	=	*Wireless Encryption Protocol*
WLAN	=	*Wireless Local Area Network*
WPA	=	*Wi-Fi Protected Acess*
WPA 2	=	*Wi-Fi Protected Acess* 2
WSN	=	*Wireless Sensor Network*

SUMÁRIO

1. INTRODUÇÃO

Com a rápida evolução da tecnologia, um dos efeitos é a mudança no ambiente urbano. A chamada inteligência urbana é fruto desta mudança, com a utilização de sensores interconectados em rede que fornecem dados diversos sobre o ambiente urbano viabilizando a concepção de Cidades Inteligentes. O conceito de Cidade Inteligente aparece com frequência no contexto do desenvolvimento urbano. Não existe uma definição consensual para o conceito de Cidade Inteligente [1-2].Um conceito que é considerado o mais abrangente, afirma que as Cidades Inteligentes são aquelas que reconhecem a importância e se utilizam da tecnologia da informação e comunicação(TIC) para alavancar competitividade econômica, promover suporte as ações de gestão ambiental e proporcionar melhoria da qualidade de vida dos cidadãos [3-6]. Com tantas definições é relevante a importância das TIC nas Cidades Inteligentes. No cenário da prestação de serviços aos cidadãos, o uso da internet tem estreitado a relação cidadão-governo, criando uma nova forma de relacionamento, evitando o afluxo de pessoas aos postos de atendimento ao público, fornecendo aos cidadãos todas as informações das quais eles precisam para ter uma vida mais inteligente, através de sites ou portais municipais [3, 5, 7, 8].

Uma importante ferramenta para a implementação de uma Cidade Inteligente é a Internet das Coisas (IoT), que tem como característica a presença de um conjunto de objetos(Coisas), como sensores, atuadores, entre outros objetos, que por meio de conexão (Internet), é capaz de interagir e cooperar uns com os outros para alcançar objetivos em comum. Apresentado por [9] como a nova evolução da Internet. Ainda conforme [9], a Internet das Coisas teve um marco importante em algum momento entre 2008 e 2009, que foi quando pela primeira vez, o número de dispositivos conectados à Internet ultrapassou o número de pessoas no mundo. Em 2020, a estimativa é que se tenha 50 bilhões de dispositivos conectados contra aproximadamente 7,6 bilhões de pessoas em todo o mundo. A partir dos problemas comumente enfrentado pelas cidades, surge o desafio de combinar diferentes informações com TIC, a fim de mitigar estes problemas e promover melhores condições de vida aos cidadãos. Neste sentido, as tecnologias IoT também podem ser integradas como ferramentas para

monitorar os eventos de um ambiente, atuar a fim de conter uma situação emergencial ou, até mesmo, divulgar uma informação relevante.

1.1 Motivação

Com o avanço da inteligência urbana e as cidades inteligentes, a importância de monitoramento e coleta de dados aumentou.

Com a possibilidade de uso em diversas aplicações, a plataforma ProxyIP criada neste trabalho pode ser utilizada em várias aplicações onde é necessário o monitoramento utilizando sensores.

Deste modo, este trabalho busca ajudar a fazer o monitoramento de dispositivos/ambientes por meio da plataforma ProxyIP.

1.2 Objetivo

O objetivo deste trabalho é o desenvolvimento de um *gateway*, utilizando um *hardware* comercial de baixo custo, que após a troca do *firmware* por um novo baseado em Linux e o desenvolvimento de scripts, ganhe novas funcionalidades. Instalado em uma cidade inteligente, pode monitorar diversos fatores utilizando os sensores e repassar as informações para uma central de comando.

Sua função é fazer a interconexão entre múltiplos sensores e a Internet. A visualização dos dados obtidos pode ser feita via protocolo SNMP ou via painel de monitoramento *DashBoard*.

Como objetivos específicos, pode-se destacar:

1. O desenvolvimento de scripts para a coleta de dados dos sensores. Os dados podem ser visualizados via painel de monitoramento Web ou via SNMP.

2. O desenvolvimento de script para a utilização de múltiplos sensores utilizando SNMP, onde cada sensor tenha um identificador de objeto (OID) associado a ele.

3. O desenvolvimento do painel de monitoramento web – *DashBoard*, para visualização dos sensores e *download* do arquivo de coleta dos dados.

4. A modificação de *hardware* e a modificação de *software* para a utilização de um pendrive para aumento da capacidade de armazenamento de dadose utilização do sistema.

1.3 Organização

Este trabalho está organizado da seguinte forma.

O Capítulo 2 apresenta o *hardware*, o *firmware* e o *software* utilizados neste trabalho.O Capítulo 3 apresenta o protocolo SNMP, suas versões e particularidades.O Capítulo 4 apresenta a implementação do Proxy IP, a modificação do *hardware*,a instalação do novo *firmware* e demais configurações.O Capítulo 5 apresenta as aplicações e resultados obtidos com o uso da plataforma ProxyIP proposta.Por fim, o Capítulo 6 apresenta as conclusões do trabalho desenvolvido.

2. PROXY IP – HARDWARE, FIRMWARE E SOFTWARE

O Proxy IP é a plataforma de interconexão entre os sensores e a rede TCP/IP. Ele possui diversas interfaces de comunicação, que são: Ethernet, Wi-Fi e Serial. O hardware utilizado neste trabalho consiste na utilização de um microcontrolador, onde o mesmo disponibiliza diversas portas analógicas e digitais, para que os sensores sejam conectados e um roteador wireless comercial. A seguir sãodetalhados os dois elementos de hardware. A Figura 1 ilustra a proposta de funcionamento do Proxy IP.

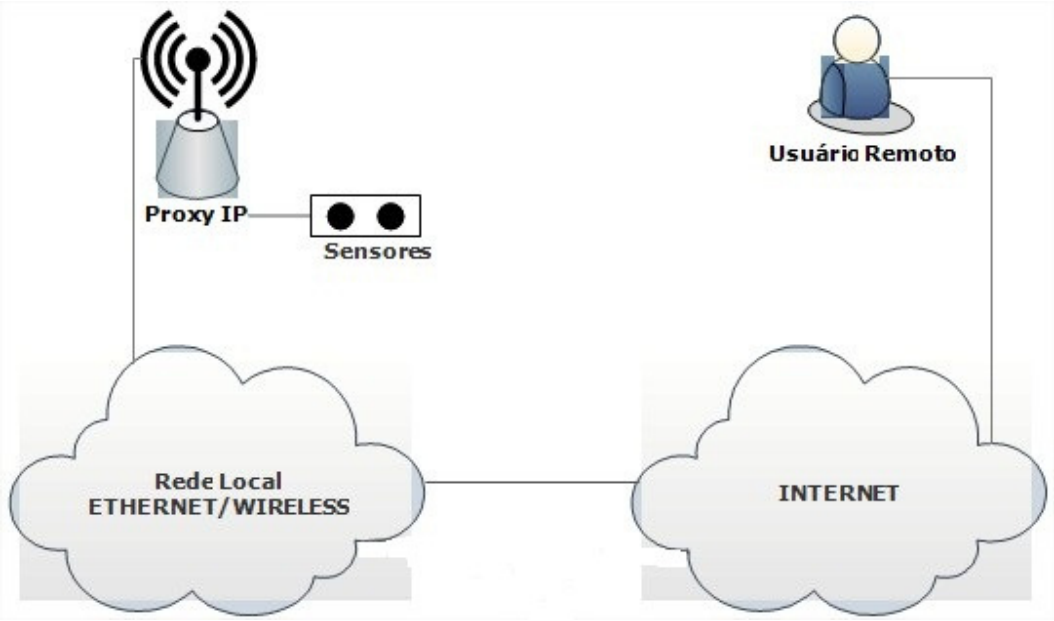

Figura1: Proposta de funcionamento do Proxy IP.

A tabela 1 ilustra o valor total para montagem do Proxy IP, lembrando que não está incluso os sensores.

Tabela 1: Equipamentos e Valores para montagem da Plataforma Proxy IP

EQUIPAMENTO	VALOR
Microcontrolador Arduino Nano 3.0	R$ 30,00
Roteador Wireless TP-LINK MR3020	R$ 90,00
Pendrive 8GB	R$ 15,00
TOTAL	**R$ 135,00**

A tabela 2 ilustra os equipamentos similares existentes no mercado. Fonte [10].

Tabela 2: Equipamentos Similares Existentes no Mercado [10]

EQUIPAMENTO	VALOR
HWg-STE – Termômetro IP Gerenciável com 2 entradas Digitais	R$ 1.839,00
Netrs2321p 1port Rs232 Serial Over Ip - Startech.com	R$ 1.075,00
Coletor De Dados E Gateway Ip/serial - Netbuffer V	R$ 1.600,00

2.1 Plataforma Arduino

O Arduino [2] é uma plataforma de prototipagem eletrônica e desenvolvimento de controle de sistemas interativos, de baixo custo e com design portátil.Com um projeto de hardware e software livres, pode ser facilmente customizado, baseado em entradas/saídas.

2.1.1 Arduino – Hardware

A placa Arduino é uma placa que possui um microcrontrolador, o ATmega328 da Atmel[12]. A Figura 2 ilustra o microcontrolador citado e suas portas.

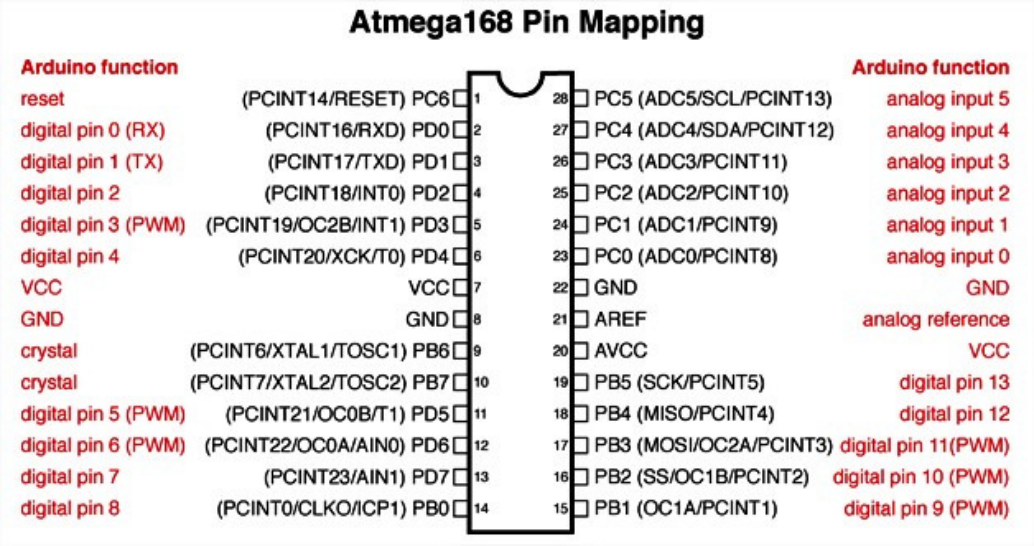

Figura 2: Layout da Pinagem do ATmega328 [13].

Como possui licença aberta, é possível construir sua própria placa Arduino compatível, utilizando o microcrontrolador ATmega.Porém o nome Arduino é marca registrada, e não é possível chamar a placa compatível de

Arduino.Existem diversas versões da placa Arduino, com diferentes tamanhos e funcionalidades.Alguns exemplos de placas Arduino são [14]: Arduino UNO, Arduino Lilypad, Arduino Duemilanove, Arduino Diecimila, Arduino Extreme, Arduino Mega, Arduino Nano e Arduino Mini.

As características do Arduino utilizado neste trabalho são:

- **Arduino Nano 3.0, baseado no microcontrolador ATmega328;**
    - Microcontrolador Atmel ATmega328;
    - I/O: 14 Pinos Digitais e 8 Analógicos;
    - Porta USB 2.0;
    - 32KB de memória Flash;
    - Peso: 5 g;
    - Baixo custo;
    - Design portátil.

A Figura 3 mostra o hardware do Arduino Nano.

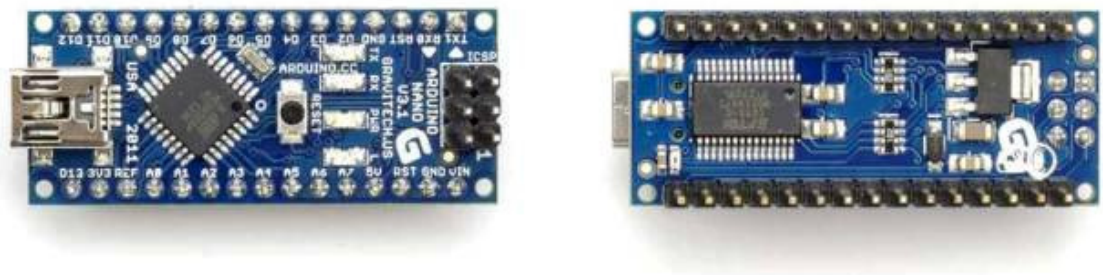

Figura3: Hardware do Arduino Nano [15].

2.1.2 Arduino – Software

Outro componente da plataforma Arduino é o editor de códigos Arduino IDE. Ela contém o software onde poderá ser feita a programação e comunicação com a placa Arduino. Na IDE podem ser programados os sketches (nome dos programas Arduino), utilizando a linguagem Processing e o código é transmitido ao Arduino via conexão USB. A Figura 4 mostra o ambiente da IDE Arduino.

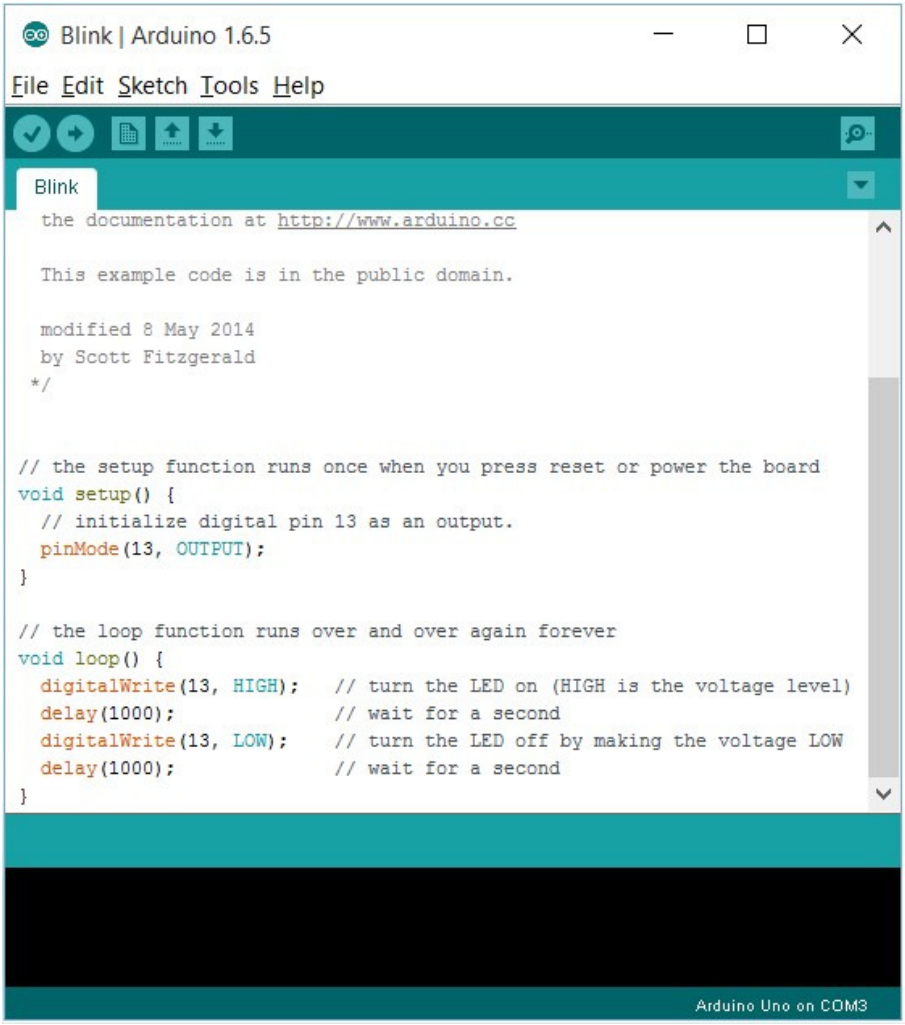

Figura 4: IDE de Programação Arduino [16].

2.2 Roteador *Wireless*TP-LINK

Neste trabalho, foram utilizados dois modelos de roteadores *wireless*:

- **TP-LINK TL-WR740N / 741ND - Versão 4.23 / 4.3**

O TL-WR741ND [17] é um dispositivo de conexão de rede *wireless*/cabeada de baixo custo (em torno de R$ 70,00), integrado com 4 portas LAN 10/100Mbps, 1 porta WAN de 10/100Mbps, um roteador de compartilhamento de Internet e função de *switch*.

É compatível com os padrões IEEE 802.11b/g/n, e oferece desempenho de até 150Mbps utilizando a tecnologia IEEE 802.11n.

O roteador possui um processador Atheros AR9331 com *clock* de 400 MHz, possui 32MB de memória RAM e 4MB de memória FLASH. Além disso, possui 1 porta WAN de 10/100Mbps e 1 porta serial. Este modelo possui antena removível. A Figura 5 ilustra o modelo TL-WR741ND com e sem *case*.

Figura5: Visão Externa e Interna do TL-WR741ND [17].

O modelo TL-WR740N se diferencia do modelo TL-WR741ND apenas por não possuir antena removível.

- **TP-LINK TL-MR3020- Versão 1.8**

O TL-MR3020 [18] é um dispositivo de conexão de rede *wireless*/cabeada de baixo custo (em torno de R$ 90,00), integrado com 1 porta LAN/WAN de 10/100Mbps, 1 porta USB 2.0 para modem 3G/4G, 1 porta mini USB para alimentação de energia e um roteador de compartilhamento de Internet. Compatível com os padrões IEEE 802.11b/g/n, oferece desempenho de até 150Mbps utilizando a tecnologia IEEE 802.11n.

Por ser um roteador pequeno e leve, é também chamado de roteador 3G de viagem.

O roteador possui um processador Atheros AR9331 com *clock* de 400 MHz, possui 32MB de memória RAM e 4MB de memória FLASH. Além disso, ele possui 1 porta LAN/WAN de 10/100Mbps e 1 porta serial.A Figura 6 ilustra o modelo TL-MR3020 com e sem *case*.

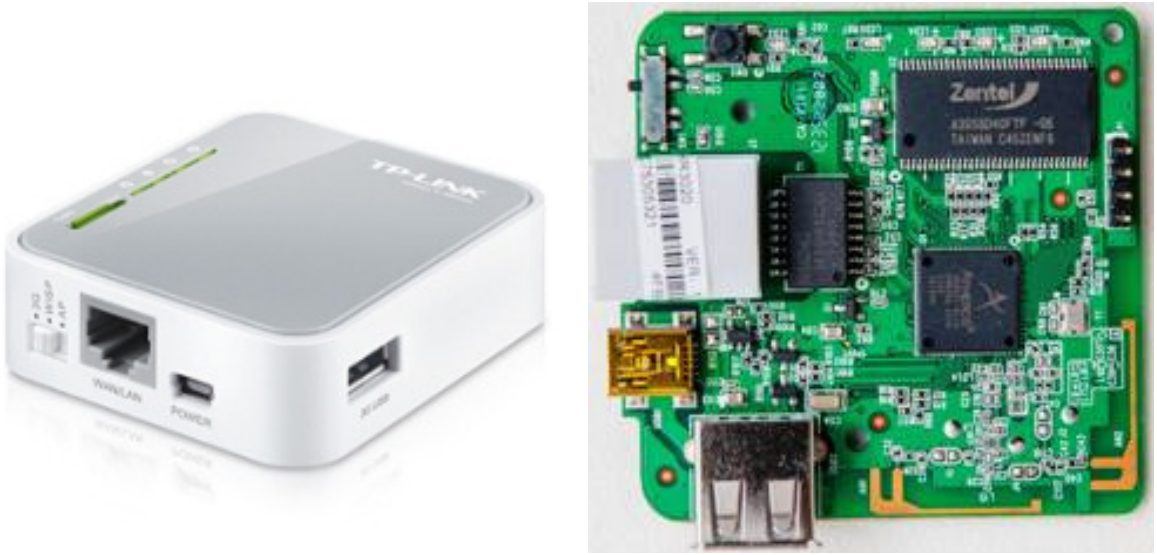

Figura6: Visão Externa e Interna do TL-MR3020 [18].

2.3 *Firmware* OpenWrt

O OpenWrt [19] é um SO (Sistema Operacional), para dispositivos embarcados. É uma distribuição Linux, gratuita e *Open Source* (Código Aberto).

Utiliza-se principalmente em dispositivos embarcados (roteadores sem fio de pequeno porte), foi criado como um *firmware* para que usuários pudessem personalizar ou modificar as funcionalidades dos seus roteadores, adicionando pacotes, e até alterando parâmetros da cama física do protocolo de comunicação.

Como funcionalidades principais, apresenta: serviço de DHCP (*Dynamic Host Configuration Protocol*), encriptação das redes sem fios, WEP (*Wireless Encryption Protocol*), WPA (*Wi-Fi Protected Access*) e WPA 2 (Wi-*Fi Protected Access* 2), reencaminhamento através de porta, UPnP (*Universal Plug and Play*), QoS (*Quality of service*) para vários tipos de tráfego, firewall avançado, modos AP ou *bridge* para a rede sem fio, serviços de DNS (*Domain Name Service*).

O OpenWrt possui um sistema de gerenciamento de pacotes, que podem ser instalados e atualizados de acordo com as necessidades. A lista completa de pacotes para instalação pode ser acessada em [20].

Pode-se funcionar em vários tipos de dispositivos, incluindo roteadores de pequeno porte, *smartphones*, computadores e notebooks. Em [21], você tem a lista completa dos dispositivos de *hardware* suportados pelo OpenWrt.

Gratuito e de código aberto, licenciado sob a licença GPL, possui uma grande comunidade para troca de informações e ajuda mútua.

Para interação com a distribuição, está disponível uma linha de comando (ash shell) através de SSH (*Secure Shell*) ou telnet e uma interface Web(LuCI).

A versão utilizada neste trabalho é a “12.09 - *Attitude Adjustment*”. A Figura 7 mostra a tela inicial do OpenWrt.

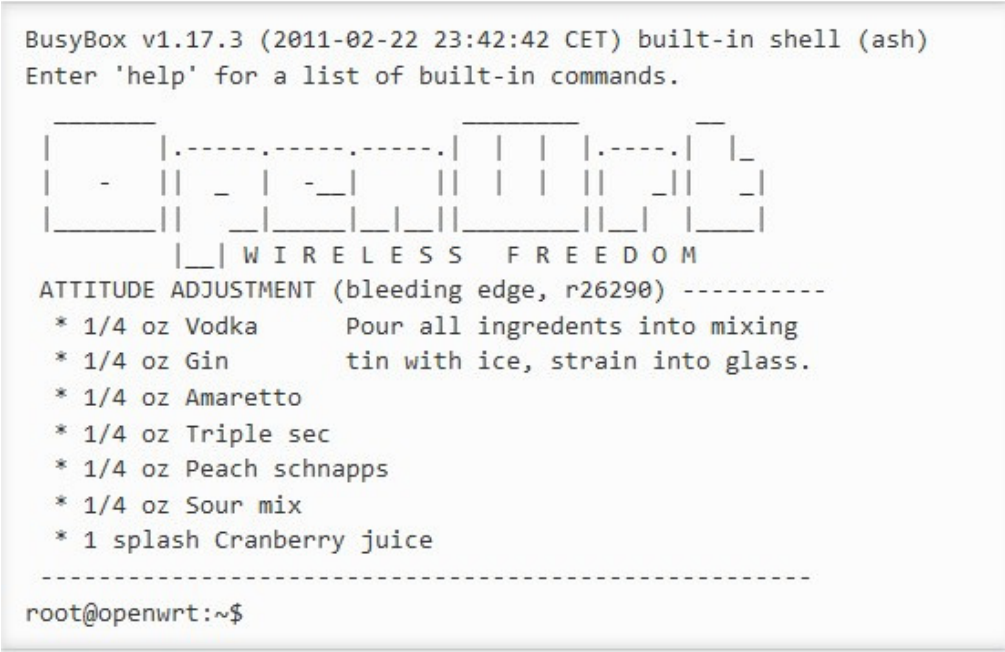

Figura 7: Tela inicial exibida no acesso via SSH e Telnet [22].

LuCI – Interface Web

A interface Web LuCI [23], escrita em linguagem de programação Lua [24], foi desenvolvida para facilitar a manutenção e configuração de dispositivos embarcados, pois a maioria das interfaces de configuração, utilizam Shell-script.

A interface LuCI faz uma divisão em várias partes lógicas:

Tabela 3: Divisão das partes lógicas na interface LuCI

Status	***Routes***
System	***System Log***
Network	***Kernel Log***
Overview	***Processes***
Firewall	***Realtime Graphs***

AFigura 8ilustraa tela inicial "*Overview"* da interface Web LuCI.

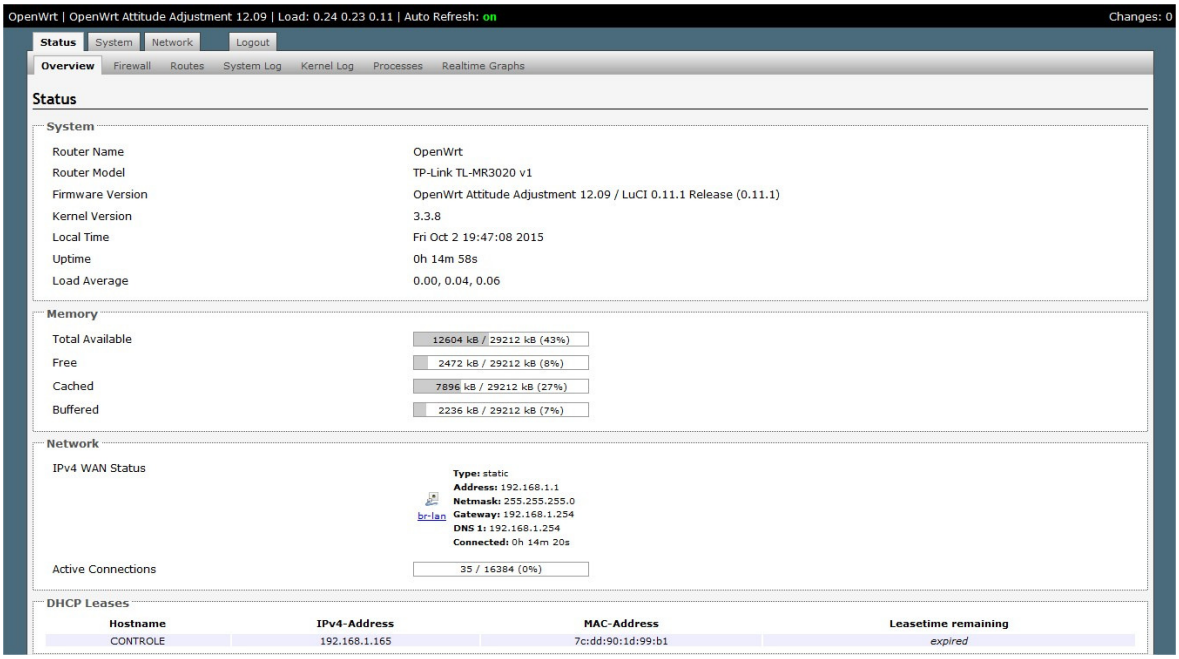

Figura8: Tela inicial exibida no acesso via Web(LuCI).

2.4 *DashBoard* – Painel de Monitoramento

Para uma melhor visualização dos dados coletados, foi desenvolvida uma aplicação em linguagem PHP onde de forma gráfica, utilizando ponteiros e mostradores digitais, as informações são mostradas ao usuário.

Por reunir várias medições/mostradores é também chamada de *DashBoard*, por se tratar de um painel de visualização.

Uma vez na rede IP, este *DashBoard* pode ser acessado via rede Local, ou mesmo pela Internet para ser monitorado.

O código fonte do arquivo da aplicação *DashBoard* é apresentado no Apêndice 8. A Figura 9 ilustra a aplicação *DashBoard* monitorando 6 sensores.

Figura 9: *DashBoard* ProxyIP

3. SNMP – SIMPLENETWORKMANAGEMENTPROTOCOL

O SNMP apresenta-se como um protocolo da camada de aplicação que compõe a pilha de protocolos TCP/IP. É amplamente utilizado como ferramenta na gerência de rede, pois facilita a troca de informações de gerenciamento entre dispositivos [25].

O protocolo utilizado para fazer a comunicação entre a estação de gerência e os dispositivos gerenciados é o UDP (*User Datagram Protocol*), protocolo da camada de transporte da pilha TCP/IP.

Segundo [26], com o crescimento das redes, a gerência se tornou ainda mais complicada, pois tais redes se tornaram cada vez maiores, e complexas. Com esta dificuldade no gerenciamento, formaram-se diversos grupos de trabalho em entidades padronizadoras, buscando criar soluções para diminuir o problema do gerenciamento [27].

O protocolo SNMP foi, inicialmente, uma recomendação do *Internet Architecture Board*(IAB) apresentada no início de 1988, como uma necessidade de uma ferramenta para gerência de redes.Com este objetivo, o SNMP se tornou o protocolo padrão para gerência de redes, tendo sido projetado para ser uma ferramenta básica, e de fácil implementação [26].

O sucesso do SNMP deve-se a solução rápida e fácil para os problemas enfrentados, com alta flexibilidade para inclusão de novos objetos [26].O SNMP baseia-se em dois dispositivos: o agente e o gerente. Cada dispositivo gerenciado possui um conjunto de variáveis que representam informações referentes ao estado atual, sendo que estas informações ficam disponíveis ao gerente.

- **Agente:** processo executado no dispositivo gerenciado, responsável pela manutenção das informações de gerência do dispositivo. Suas principais funções são:

  - Receber as requisições enviadas pelo gerente e enviar os dados relativos a requisição.

- Enviar automaticamente informações de gerenciamento ao gerente, alertas ou condições anormais.

- **Gerente:** Programa executado, geralmente em servidor, que recebe e envia informações de gerenciamento junto aos dispositivos gerenciados. Possui comunicação com um ou mais agentes. Responsável pelo monitoramento, relatórios e decisões na ocorrência de falhas.

  A Figura 10 ilustra o relacionamento de um gerente, agente e objeto gerenciado.

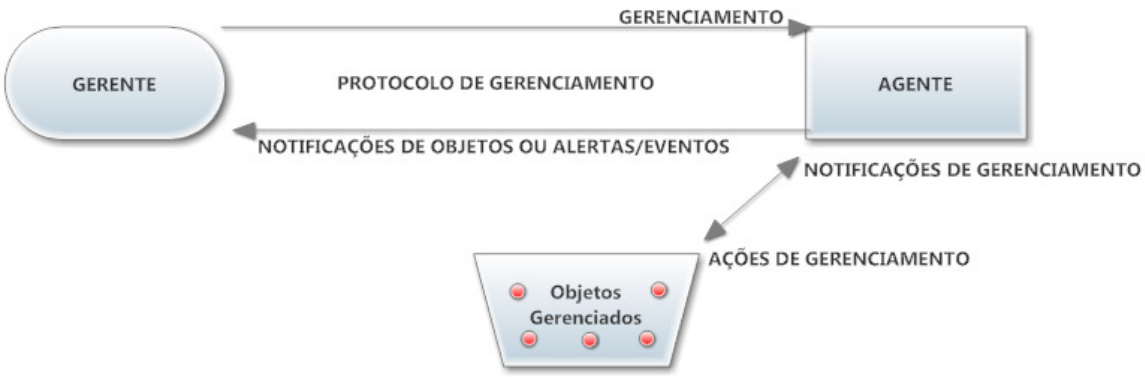

Figura 10: Relacionamento de um gerente com o objeto gerenciado.
Fonte: elaboração própria

3.1 Versões do SNMP

Com uma solução completa para o problema da gerência de redes, o SNMP passou por diversas versões. Foi padronizado através do uso de RFCs(*Request For Comments*) da IETF (*Internet Engineering Task Force*).

Atualmente, existem três versões do SNMP:

- **SNMP Versão 1 (SNMPv1):**Primeira versão do protocolo, padronizada pela RFC 1157 [28] e publicada em 1990. Prevendo a sua operação sobre os protocolos *User Datagram Protocol* (UDP) [25]. A segurança é o principal problema[26]**,** já que trabalha com o

conceito de comunidades que são trafegadas em texto limpo (plaintext) pela rede, ou seja, as senhas ficam visíveis a qualquer equipamento. Esta versão estabeleceu-se como base para o SNMP como se conhece hoje.

- **SNMP Versão 2 (SNMPv2):**A segunda versão foi lançada em 1993 com algumas melhorias, que são descritas pela MIB-2 da RFC 1213 [29]. Uma importante implementação, foi a autenticação, que deixou o protocolo com mais segurança. Esta implementação foi descrita na RFC 1441 [30] e RFC 1442 [31]. Mesmo com a implementação da autenticação, o protocolo ainda não era tão seguro e facilmente manipulado [26]. Houve a necessidade de aprimorar o protocolo para melhorar a segurança, esta melhoria foi incorporada na versão 3.

- **SNMP Versão 3 (SNMPv3):**Na terceira versão, o foco foi a melhoria na segurança e na privacidade dos dados, que são descritas nas RFCs 3410 [32], 3414 [33], 3415 [34], 3418 [35].

3.2 Operações

O protocolo possui três operações básicas, sendo elas: “get”, “set” e “trap”.Como possui estrutura de pacotes, as operações podem afetar mais de um objeto. Por exemplo, uma operação “get”, pode buscar o valor de diversos objetos, utilizando uma única requisição.

A tabela 4 descreve as principais operações do SNMP.

Tabela 4: Principais operações do protocolo SNMP

Operação	Origem – Destino	Mensagem
GET	Gerente -> Agente	get-request
GETNEXT	Gerente -> Agente	get-next-request
GETBULK	Gerente -> Agente	get-bulk-request
SET	Gerente -> Agente	set-request
TRAP	Agente -> Gerente	trap
INFORM	Qualquer -> Gerente	inform-request

- **GET:**utilizado para requisitar um valor de uma informação de gerência. Na operação, o gerente requisita um valor de um determinado objeto a um agente;

- **GET-NEXT:**operação parecida com o get, onde é requisitado um valor, porém, neste caso, o gerente está solicitando o próximo objeto, potencialmente desconhecido, a partir de outro objeto;

- **GET-BULK:**uma melhoria do get-next que possibilita solicitar grandes requisições, principalmente de tabelas;

- **SET:**utilizado para definir o valor de uma determinada informação de um objeto. O gerente informa o valor para o objeto gerenciado no agente;

- **TRAP:**de maneira autônoma, o agente envia um aviso para o gerente. Utilizado para alertas e condições anormais. É uma operação de via única, ou seja, ela não possui confirmação de recebimento pelo gerente.

- **INFORM:**parecida com o trap, é utilizada para alertas e condições anormais. O diferencial em relação ao trap é a obrigatoriedade de confirmação de recebimento. Pode ser utilizada entre agente-gerente ou gerente-gerente.

3.3 OID - *Object Identification*

Cada recursogerenciável (descrição do sistema, valor de sensores, etc.) é considerado um objeto e associado a um identificador de objeto (OID).Uma OID é uma seqüência numérica para identificação de um objeto específico.A tabela 5 demonstra exemplos de identificações de objetos.

Tabela 5: Exemplos de objetos gerenciados

OID	Descrição
1.3.6.1.2.1.1.20.101.1	Carga da CPU
1.3.6.1.2.1.1.21.101.1	Leitura do Sensor 01 – Umidade do Ar
1.3.6.1.2.1.1.22.101.1	Leitura do Sensor 02 – Temperatura do Ar
1.3.6.1.2.1.1.23.101.1	Leitura do Sensor 03 – Luminosidade
1.3.6.1.2.1.1.24.101.1	Leitura do Sensor 04 – Chuva
1.3.6.1.2.1.1.25.101.1	Leitura do Sensor 05 – Umidade do Solo
1.3.6.1.2.1.1.26.101.1	Leitura do Sensor 06 – Voltagem de Entrada

4. IMPLEMENTAÇÃO DO PROXY IP

Esta seção descreve em detalhes toda a implementação do Proxy IP, tanto o *hardware*, quanto o *software*.O Proxy IP provê a conexão entre os sensores e a Internet.Ele possui os modos de AP e CLIENTE para conexões WIFI.

4.1 Modificação do *Hardware*

A interconexão entre o Arduino e o roteador wireless TP-LINK é feita a partir de uma porta serial TX/RX presente em cada um dos equipamentos.

O Arduino já possui uma porta serial com pinos TX e RX disponível para conexão, não precisando de qualquer modificação.

Como modificação de *hardware*, para que o Arduino possa se comunicar aporta serial do roteador wireless TP-LINK, foram necessáriasas seguintesmodificações:

- **TP-LINK TL-WR740N / TL-WR741ND**

A TP-LINK utiliza um padrão de pinagem para a porta serial em todos os modelos, porém nestes dois modelos, o pino TX não está conectado originalmente a CPU.

Para que o TX funcione, é necessário fazer uma conexão entre dois pontos na parte de baixo da PCB, utilizando um pequeno fio.

A conexão feita é entre o pino TX da serial com nome “TP28” e o pino menor. A Figura 11 ilustra os pontos que devem ser conectados.

Figura11: Modificação de *Hardware*: Conexão entre dois pontos para funcionamento do TX [17].

A Figura 12 ilustra a PCB após a conexão dos dois pontos.

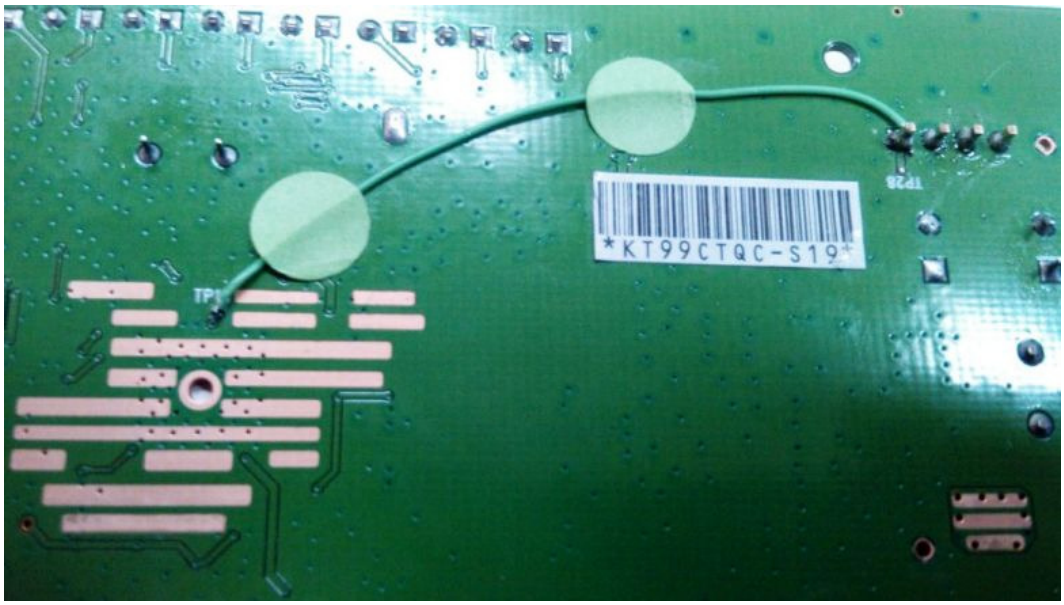

Figura 12: PCB após modificação no pino TX.

Após a modificação no pino TX, é necessário soldar uma barra de pinos no local indicado para a conexão serial.

A porta serial está identificada com o nome “J1” impresso na PCB.

O primeiro pino, ao lado do J1 é o TX, seguido pelo RX, GROUND e por último o VCC (3.3V). A Figura 13 ilustra a PCB após a modificação na serial.

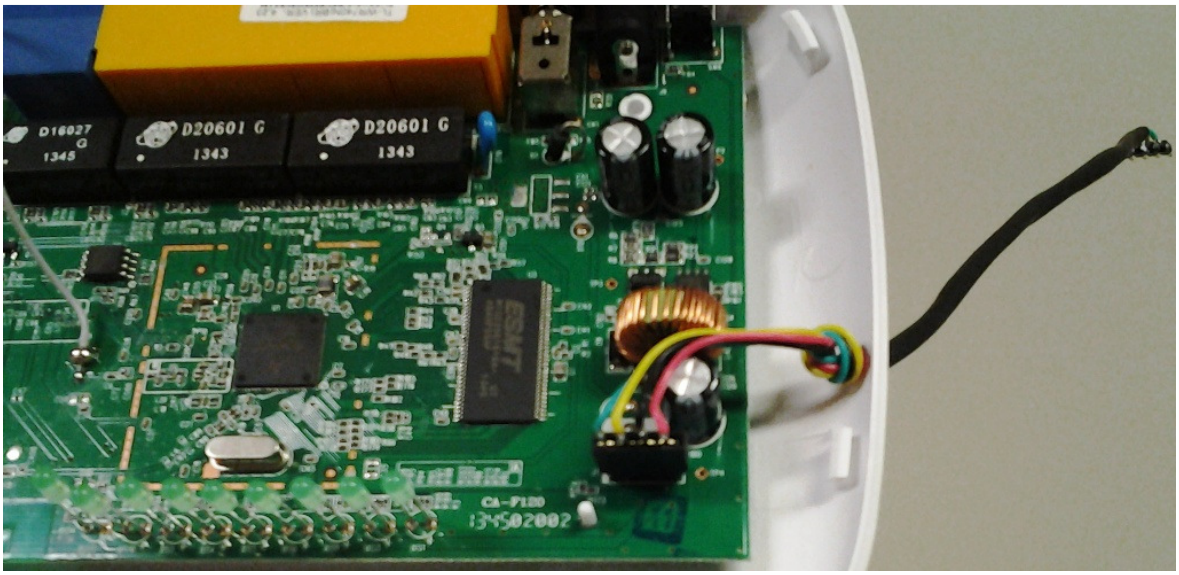

Figura 13: PCB do TL-WR740/741ND após modificação na serial.

- **TP-LINK TL-MR3020**

Diferentemente do modelo acima, a única modificação no TL-MR3020 se resume apenas a soldar uma barra de pinos no local indicado para a conexão serial.

A porta serial está identificada com o nome “P1” impresso na PCB.

O primeiro pino, ao lado do P1 é o TX, seguido pelo RX, GROUND e por último o VCC (3.3V). A Figura 14 ilustra a PCB após a modificação na serial.

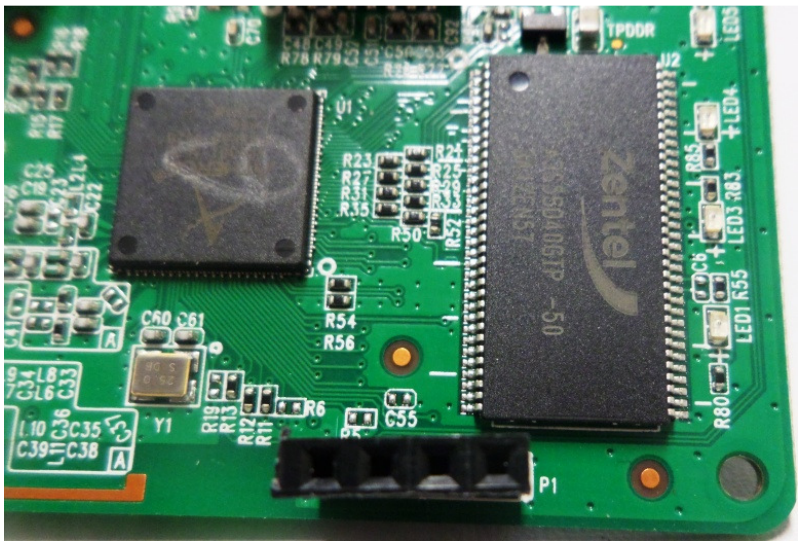

Figura14: PCB do TL-MR3020 após modificação na serial.

4.2 Modificação do *Software*

Para a modificação do *Software*, utilizaremos como exemplo o modelo TP-Link TL-MR3020, pois é o modelo do protótipo final.

Como modificação do *Software*, foram necessárias várias modificações descritas a seguir:

4.2.1 Instalação do *Firmware* OpenWrt

Para atualizar o *firmware* original da TP-Link para o OpenWrt, é necessário primeiramente se conectar ao roteador, via WiFi ou cabo.

Por padrão, o endereço IP do roteador é: 192.168.0.254.

Após a conexão, deve-se acessar as configurações do roteador, que são disponibilizadas via plataforma Web no endereço: http://192.168.0.254 e informar o usuário e senha padrão que é: admin . A Figura 15 ilustra a tela de login.

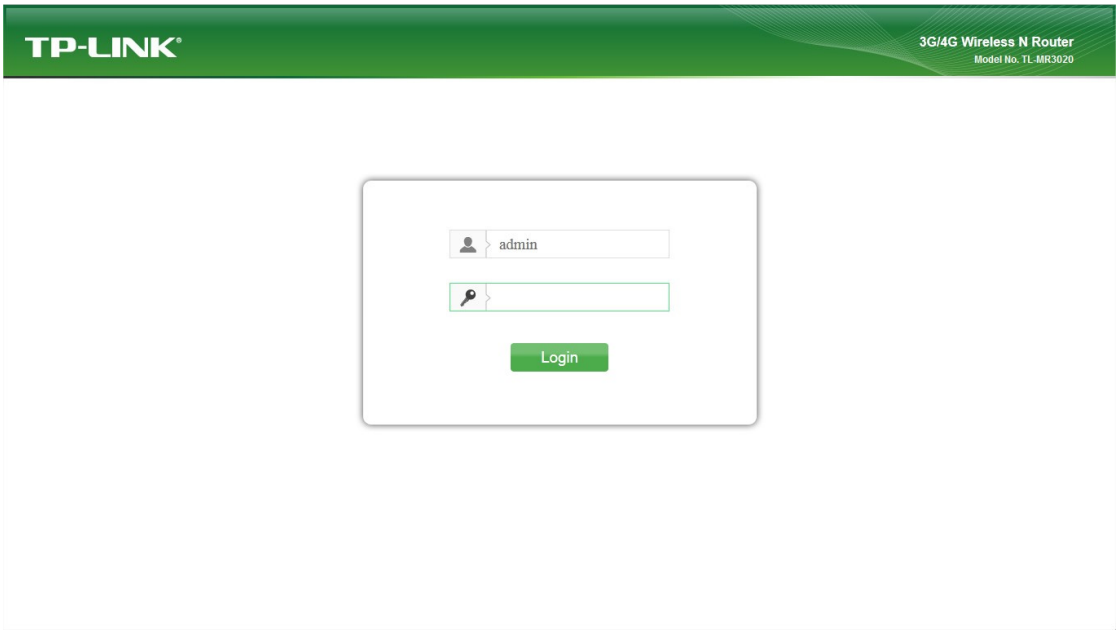

Figura 15: Tela de login *firmware* TP-Link

Após efetuar login, deve-se acessar o menu "*System Tools*", e depois "*Firmware Upgrade*". Na tela "*Firmware Upgrade*", deve-se clicar em "Selecionar arquivo" e informar o caminho do *firmware* OpenWrt previamente já

baixadode[36]. Deve-se clicar em “*Upgrade*”. A Figura 16 ilustra a tela de *upgrade* do *firmware*.

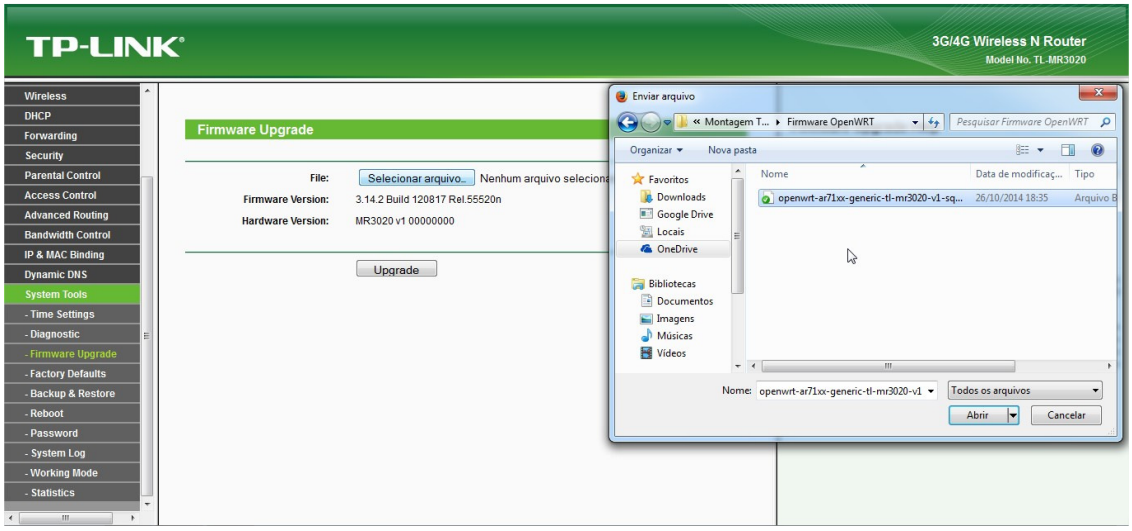

Figura16: Tela *FirmwareUpgrade*

Neste momento, o sistema solicitada uma confirmação para ver se realmente o usuário deseja atualizar o *firmware*. Deve-se confirmar clicando no botão “OK”. A Figura 17 ilustra a tela de confirmação de *upgrade* do *firmware*.

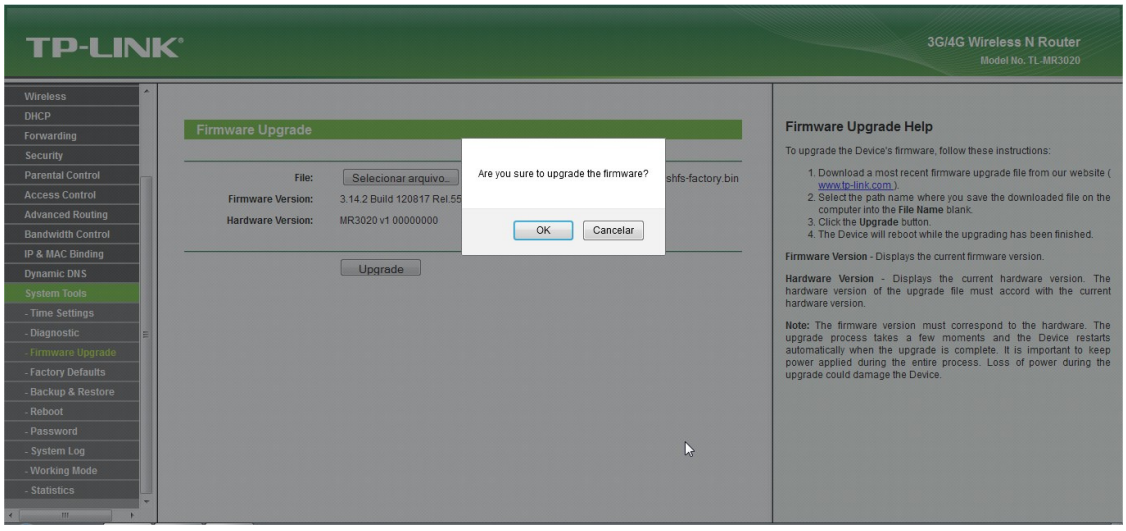

Figura17: Tela Confirmação de *upgrade* do *firmware*

Após a confirmação, o *firmware* novo será instalado e o roteador deve ser reiniciado.

4.2.2 Configurações Iniciais: Senha de root e Rede

Após a atualização de *firmware*, o IP padrão do roteador passa a ser: 192.168.1.1.Por padrão, o OpenWrt vem com o super-usuário: root com a senha em branco, com isso, só será possívelneste primeiro acesso, a conexão via TELNET.Para isso, deve-se utilizar o *software* “PuTTY” que pode ser baixado em [37].Após a conexão ao roteador por WiFi ou cabo, será necessário configurar o PuTTY.Ao abrir o programa, no campo “*Host Name* (or IP *address)*” deve-se colocar o IP do roteador, que é: 192.168.1.1 e no campo “*Connection type*” deve ser selecionado Telnet. A Figura 18ilustra a tela inicial do PuTTY.

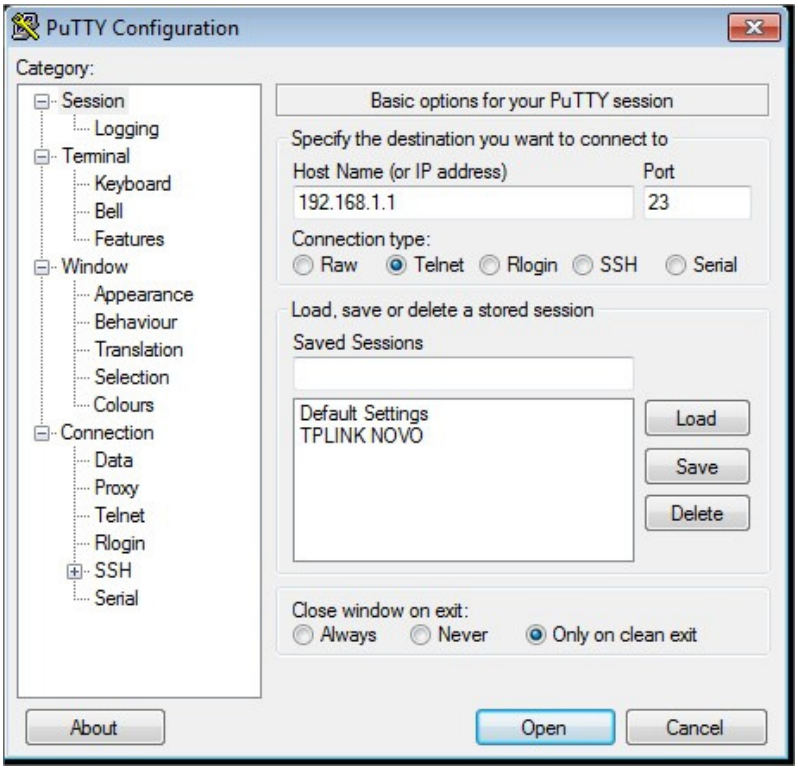

Figura 18: Tela de configuração do PuTTY.

Após a configuração, deve-se clicar em “*Open*”. A Figura 19 ilustra a tela inicial do OpenWrt.

Figura 19: Tela inicial do OpenWrt via Telnet.

Está é a tela do SHELL do OpenWrt.Não será solicitado nenhuma informação de login, como usuário ou senha.

Primeiramente deve-se começar alterando a senha do usuário root: para isso deve-se utilizar o comando: “passwd”. A Figura 20 ilustra o procedimento para troca de senha do usuário root.

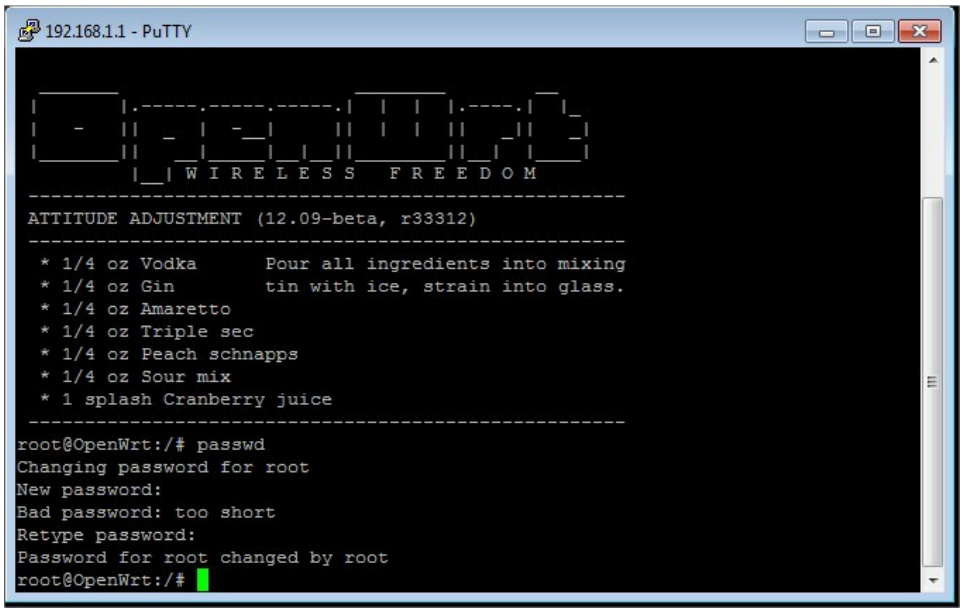

Figura 20: Tela Alterando a senha do usuário root

Pode-se notar que assim que é utilizado o comando “passwd”, o sistema pede uma nova senha, e depois pede a confirmação da nova senha, deve-se digitar a mesma senha escolhida nos dois campos.

Após a troca da senha de root, deve-se desconectar e conectar novamente pelo PuTTY, mas selecionando SSH ao invés de Telnet.

Será solicitado um nome de usuário e senha para se conectar via SSH, deve-se informar o usuário e senha configurados anteriormente. A figura 21 mostra a tela de login via SSH.

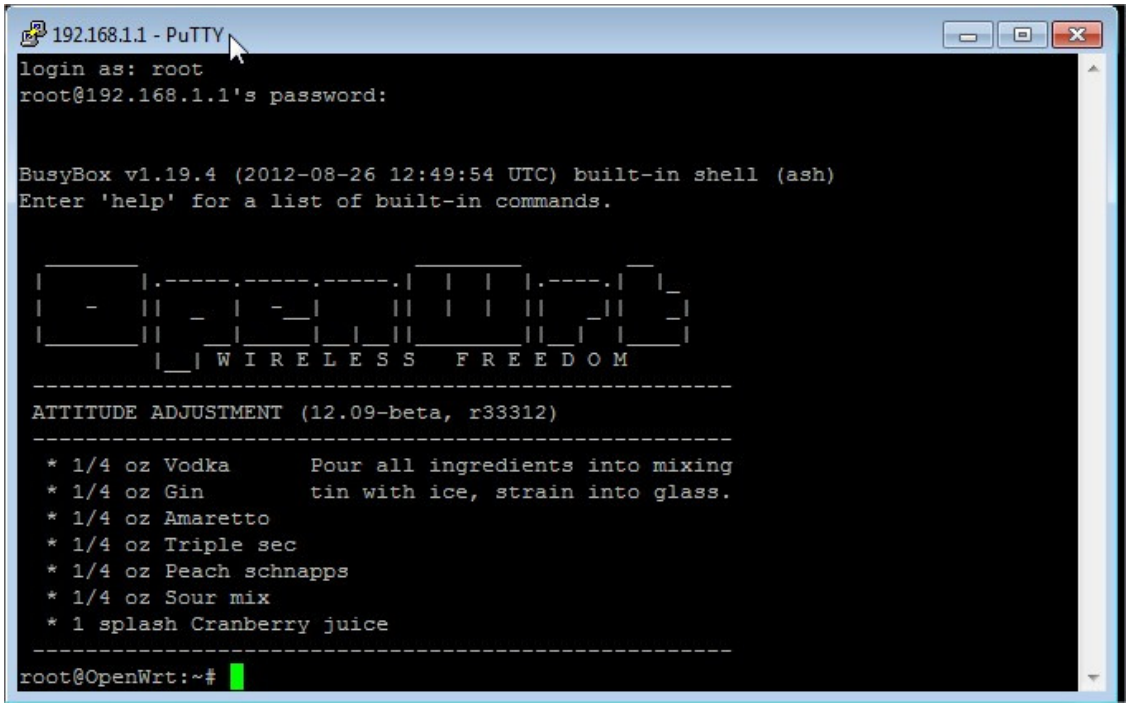

Figura 21: Tela inicial do OpenWrt via SSH.

Deve-se agora prosseguir com a configuração da rede.

Primeiramente deve-se acessar a pasta “/etc/config” onde se localiza o arquivo de configuração da rede “network”. Deve-se editar o arquivo utilizando o editor VIM. A Figura 22 ilustra o comando para edição do arquivo de configuração da rede.

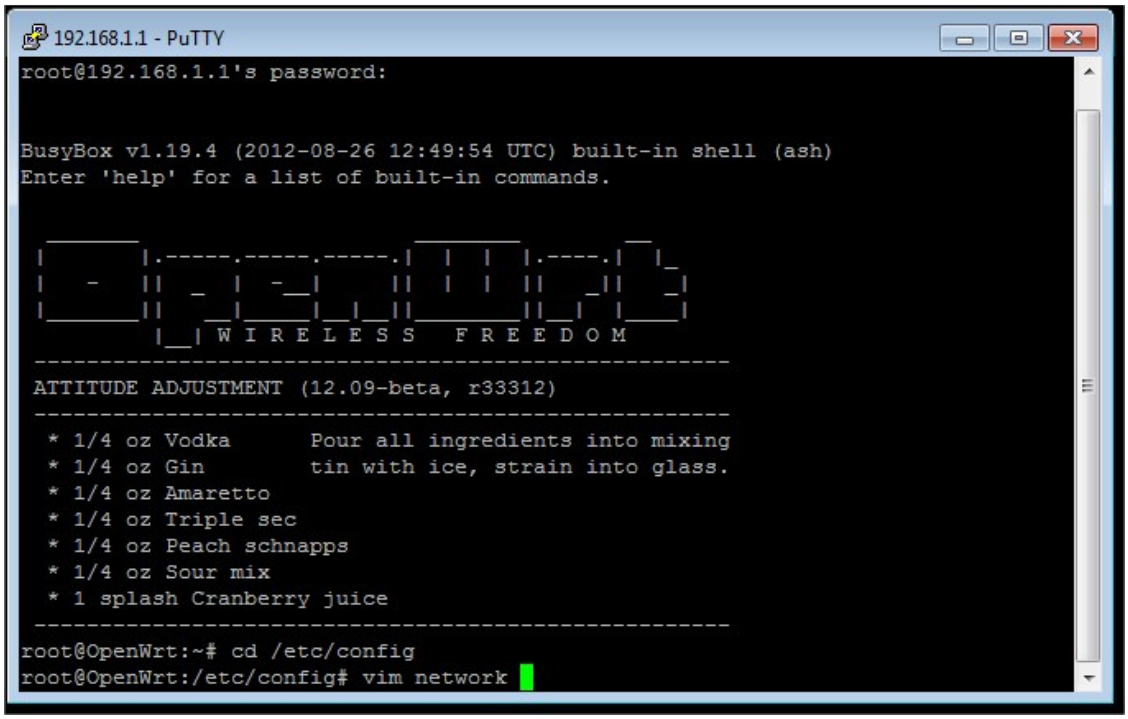

Figura 22: Tela editando arquivo de configuração da rede.

Será necessário a instalação de alguns pacotes novos de *software*, então será necessário a configuraçãodo *gateway* e o dns da interface LAN para ter acesso a Internet.

Deve-se adicionar uma linha abaixo da linha de "*netmask*" da interface LAN e inserir as configurações de *gateway* e na outra linha o dns. A Figura 23 ilustra a configuracao do gateway e do dns.

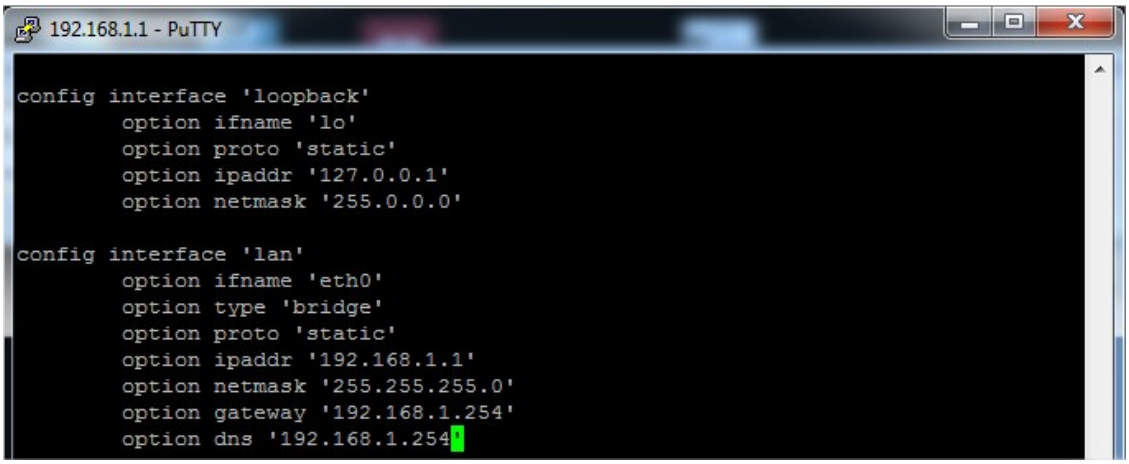

Figura 23: Tela Configurando *gateway* e dns da interface LAN.

Para salvar o arquivo e sair, deve-se digitar “:wq”.

Após voltar para o shell, utilizar o comando: “uci commit network” para validar as configurações, após, deve-se reiniciar o roteador com o comando “reboot”.

4.2.3 Preparação do Pendrive

A memória interna do roteador é muito limitada, vide especificações do *hardware*, então para que seja possível a instalação de novos pacotes de *software*, é necessário a utilização de um Pendrive, que no caso é de 8GB, para que tanto o sistema OpenWrt, quanto os pacotes de *software* e arquivos fiquem salvosno pendrive.

Deve-se preparar o pendrive para utilização, para isso será necessário instalar o *software* de particionamento: “*MiniTool Partition Wizard Home Edition*” [38], em um computador rodando sistema operacional Windows.

Deve-se inserir o pendrive no computador, e abrir o programa“*MiniTool Partition Wizard Home Edition*”. A Figura 24 ilustra a tela inicial do respectivo programa.

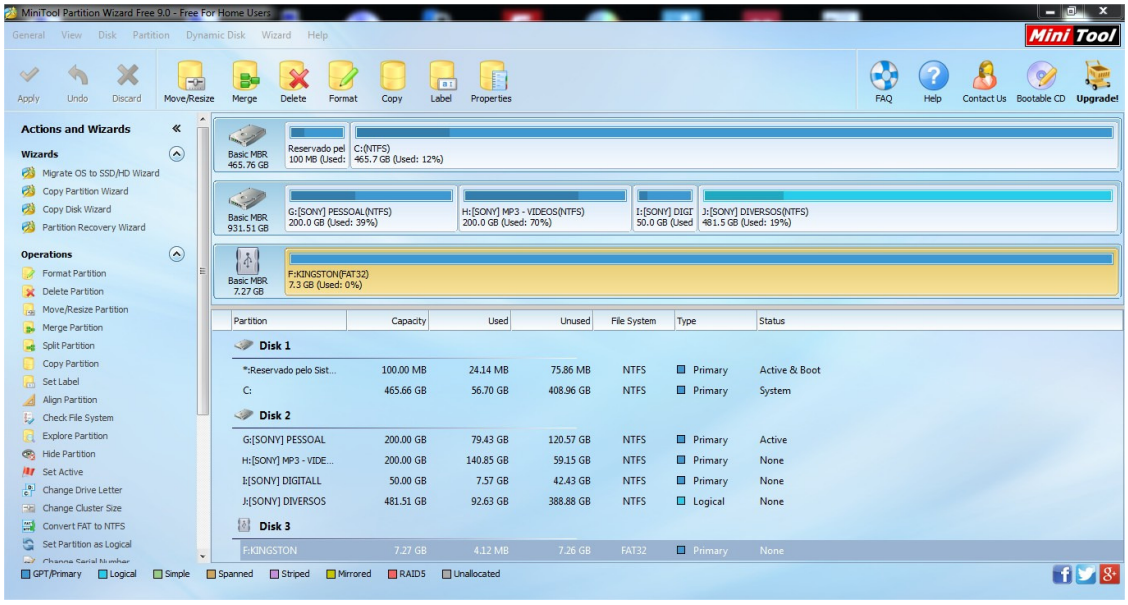

Figura 24: Tela do *MiniTool Partition Wizard*.

Deve-se excluir a partição FAT32 do pendrive, clicar na imagem do pendrive, selecionar, clicar no ícone “*Delete*” do menu. A Figura 25 ilustra o pendrive selecionado e sem nenhuma partição.

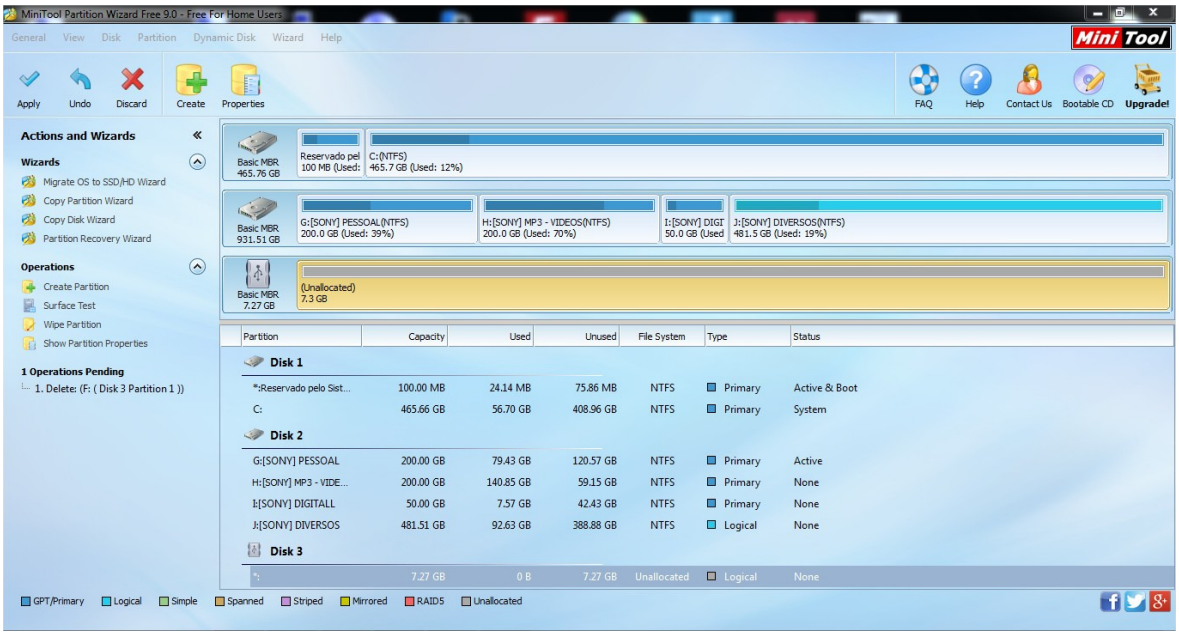

Figura 25: Tela Pendrive sem partições.

Deve-se fazer o particionamento, da seguinte maneira:

Clicar no ícone do pendrive, selecionar, clicar no ícone “*Create*” do menu.

Nas opções, deve-se criar a primeira partição, com tamanho de 1GB, Primária e do tipo Linux Swap. A Figura26 ilustra as configurações da primeira partição.

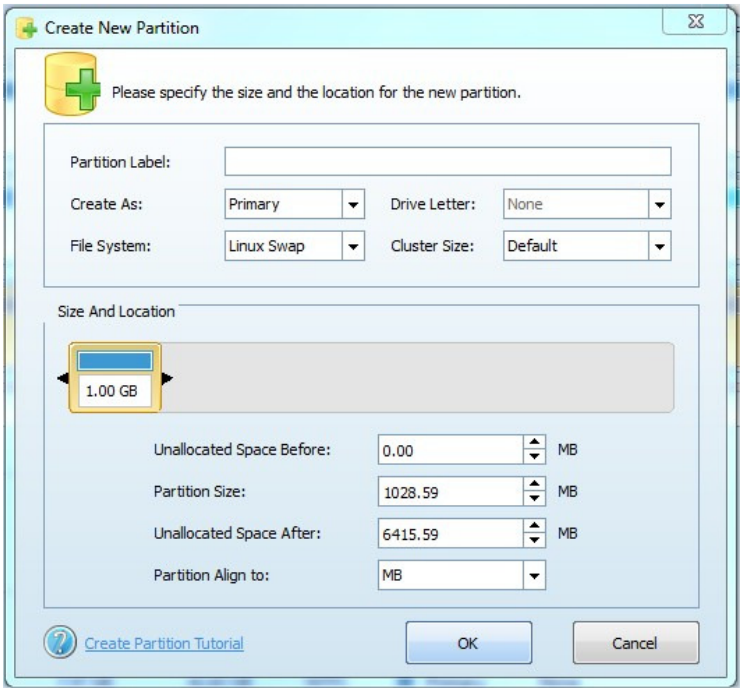

Figura 26: Tela Criando primeira partição

Crie a segunda partição, selecionando o espaço que sobrou no pendrive, clicando no ícone "*Create*" do menu. Selecionar a partição como Primária e tipo Ext4. A Figura 27 ilustra as configurações da segunda partição.

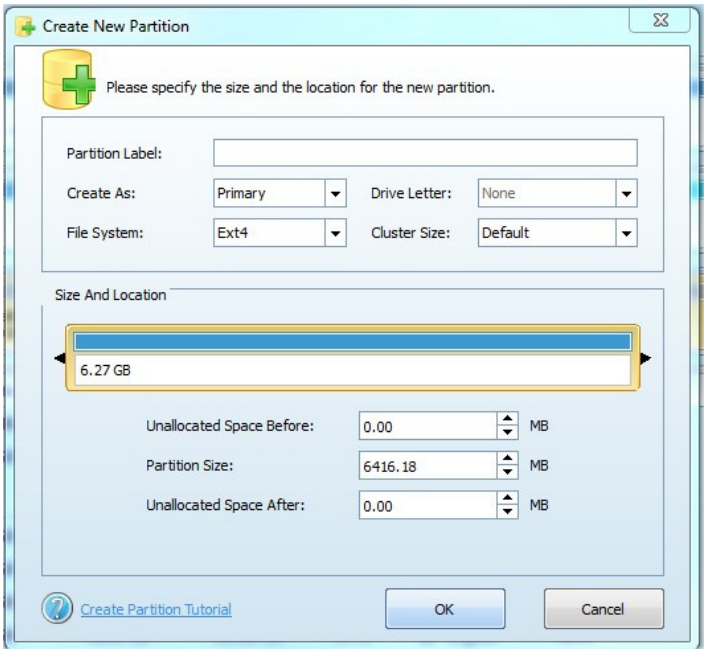

Figura 27: Tela Criando segunda partição.

A Figura28 ilustra como as partições devem ficar após serem criadas.

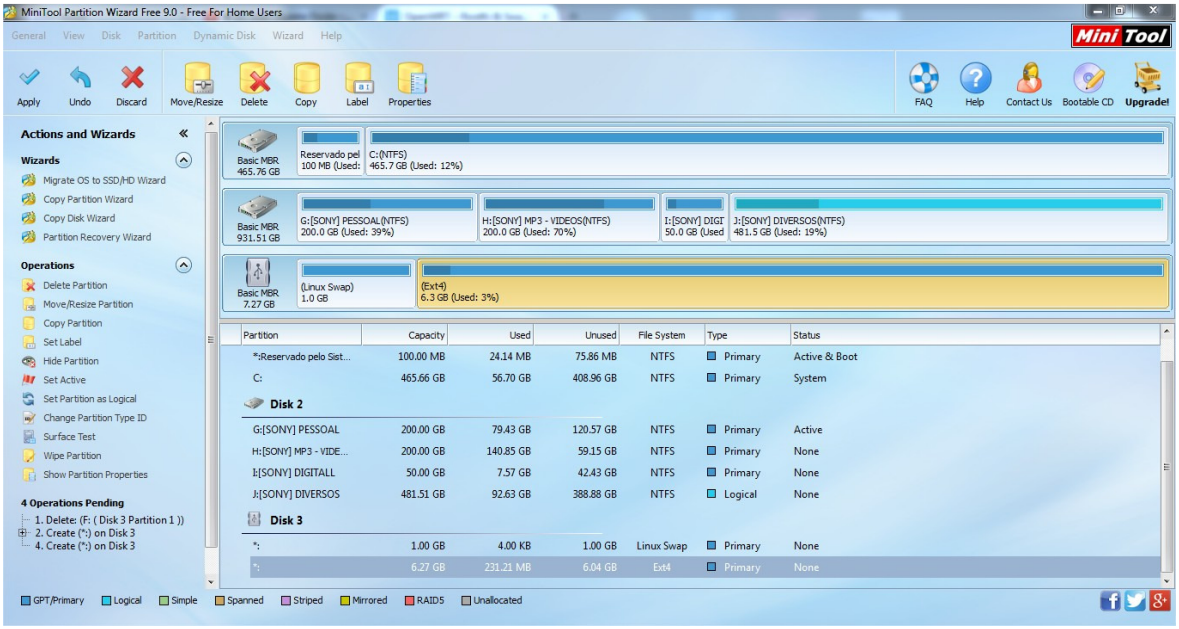

Figura 28: Tela Criando segunda partição.

Aplicar as alterações e salvar, deve-se clicar no botão "*Apply*" do menu. Clicar em "*Yes*" confirmando as modificações. A Figura29 ilustra a tela de confirmação de aplicação das mudanças.

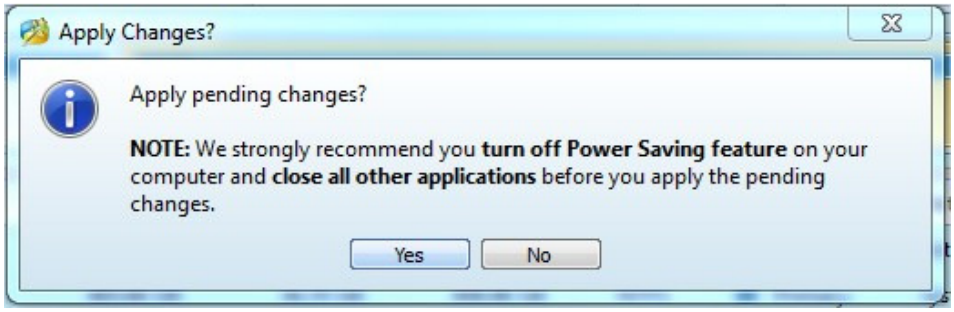

Figura 29: Tela confirmação aplicar mudanças.

Após a finalização, retirar o pendrive do computador e inseri-lo no roteador.

4.2.4 Transferência do sistema para o pendrive

Devem-se executar os comandos abaixo, atualizando a lista de pacotes, e instalando os pacotes necessários:

$ opkg update
$ opkg install kmod-usb-storage-extras
$ opkg install kmod-fs-ext4
$ opkg install block-mount

Após a instalação, deve-se reinicializar o roteador com o comando “reboot”.

Utilizando um computador com sistema operacional Windows, fazer o download do software WinSCP [39].

Após a abertura do programa, deve-se selecionar o protocolo SCP, informar o endereço IP do roteador, que é 192.168.1.1 e os dados de login do usuário root. A Figura30 ilustra a tela inicial do WinSCP.

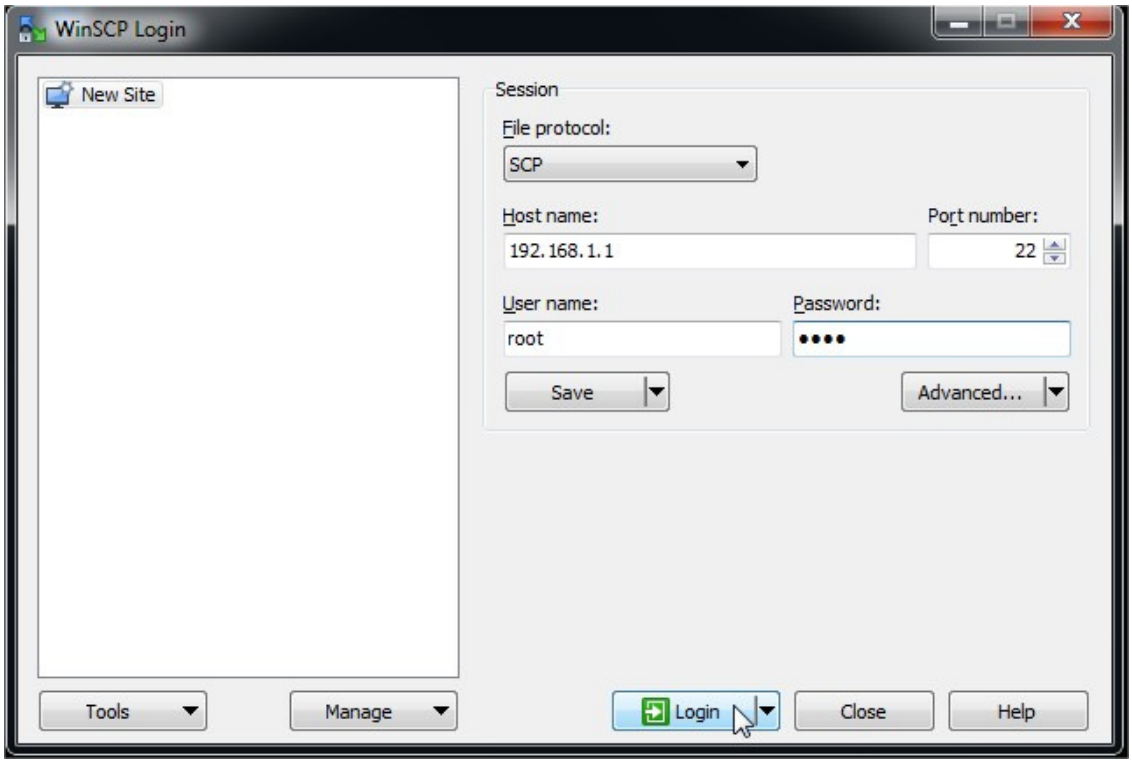

Figura30: Tela Configuração WinSCP.

Deve-se clicar em login, para efetuar a conexão. Após a conexão, a pasta atual, será a pasta raiz “/”.

Acesse a pasta “/etc/config”. Selecione o arquivo “fstab” e com o botão direito do mouse, clique em “Edit”. A Figura 31 ilustra a tela de edição do arquivo “fstab”.

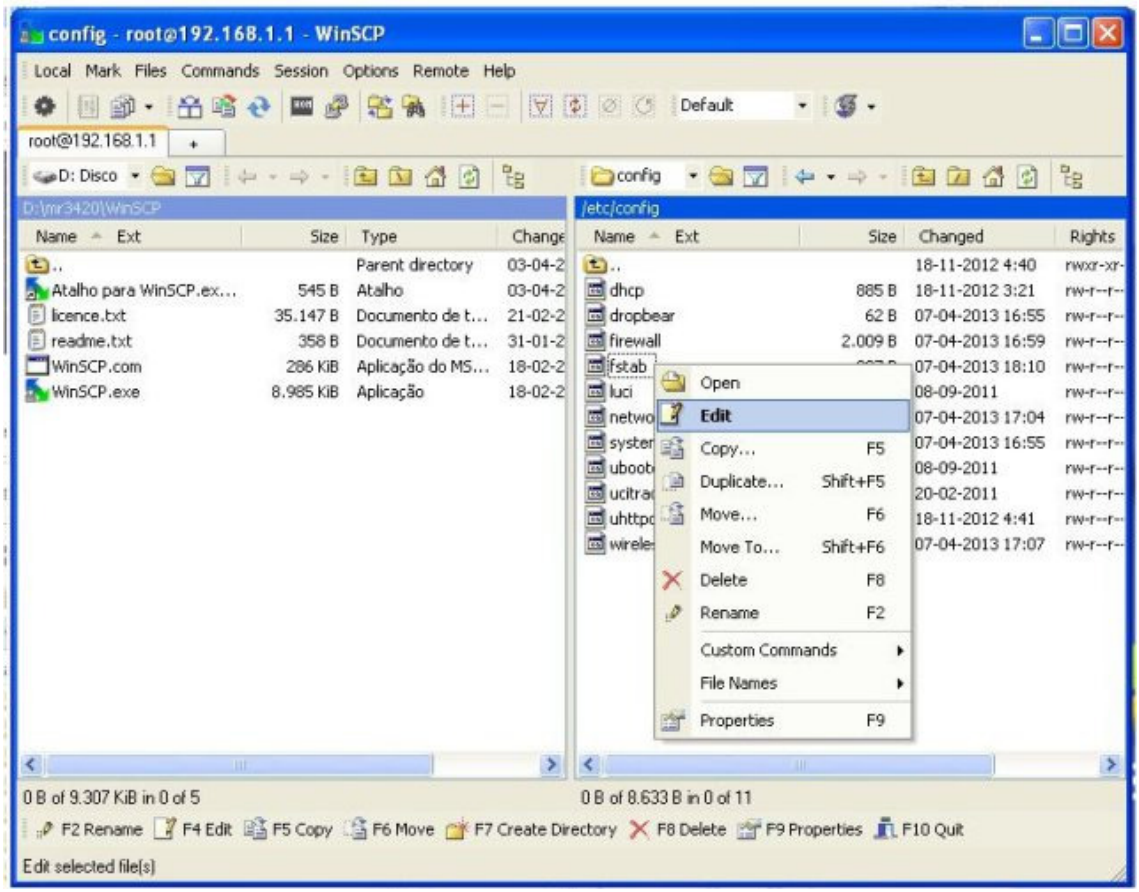

Figura 31: Tela Edit fstab.

A Figura 32 ilustra as configurações necessárias, configurar igualmente.

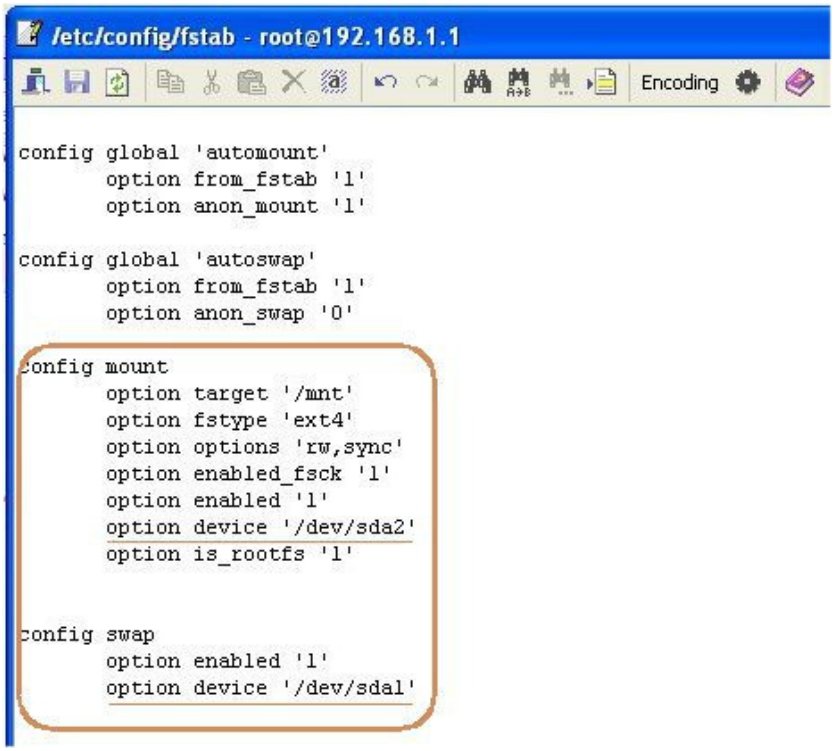

Figura 32: Tela Configuração fstab.

Após o término daedição, é necessário salvar as modificações, clicando no ícone do disquete do menu e reiniciando o sistema com o comando “reboot” via Shell. Deixar o pendrive conectado ao roteador.

O sistema reiniciará já rodando o sistema de arquivos no pendrive, porém toda configuração feita até agora foi perdida, pois o sistema foi zerado. É necessário efetuar as configurações iniciais do item 4.2.2 novamente.

Para verificar a nova disponibilidade de espaço em disco, após efetuar as configurações novamente, acessar a interface Web LuCi no endereço: http://192.168.1.1 , informando os dados de login do usuário root, clicando no menu “*System*” e “*Software*”. A Figura 33 ilustra a tela para verificação do espaço em disco livre.

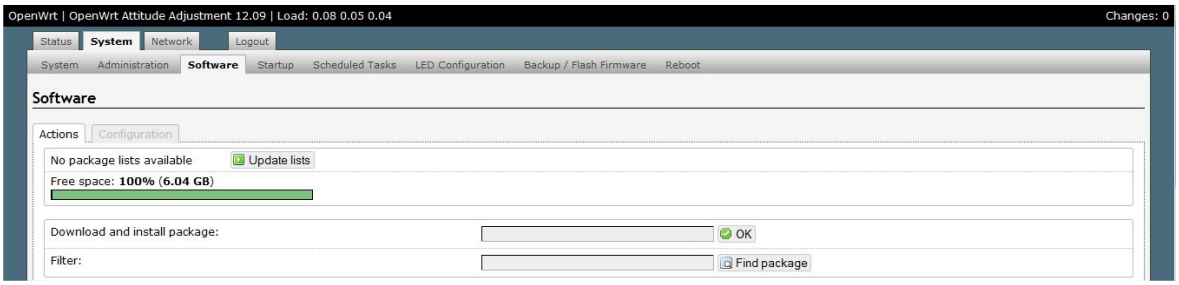

Figura 33: Tela Verificação do espaço em disco livre.

No shell, execute os comandos abaixo, atualizando a lista de pacotes e instalando os pacotes necessários:

$ opkg update

$ opkg install kmod-usb-storage-extras

$ opkg install kmod-fs-ext4

$ opkg install block-mount

Após a instalação, reinicializar o roteador com o comando “reboot”.

Acessar a interface Web LuCi no endereço: http://192.168.1.1, informar os dados de login do usuário root, clicar no menu “*System*” e “*Mount Points*”. A Figura 34 ilustra a tela “*Mount Points*”.

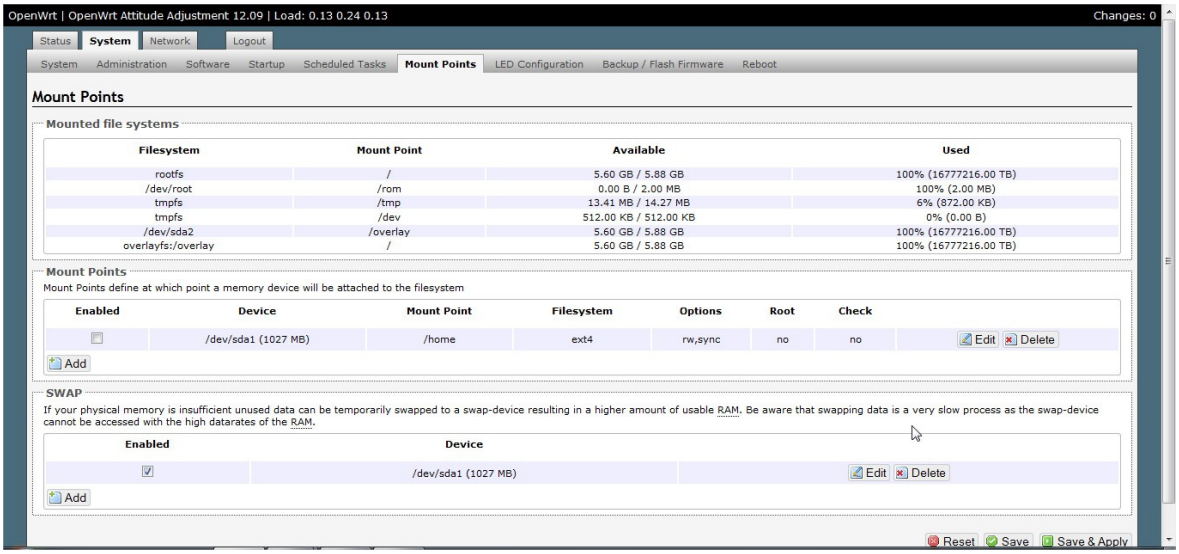

Figura 34: Tela *Mount Points*.

Em SWAP, clicar no botão “*Edit*”, selecionar a partição correta /dev/sda1, habilite a flag “swap”, clique em “*Save & Apply.*”

Em Mount Points, clicar no botão “Edit”, selecionar a partição correta /dev/sda2 e habilitar o ponto de montagem, clicar em “Save & Apply.”

Acessar o menu “*System*” e “*Startup*”, a linha do Initscript “fstab” estará desabilitado, clicar para habilitar. A Figura 35 ilustra a tela “*Startup*” após o “fstab” ser habilitado.

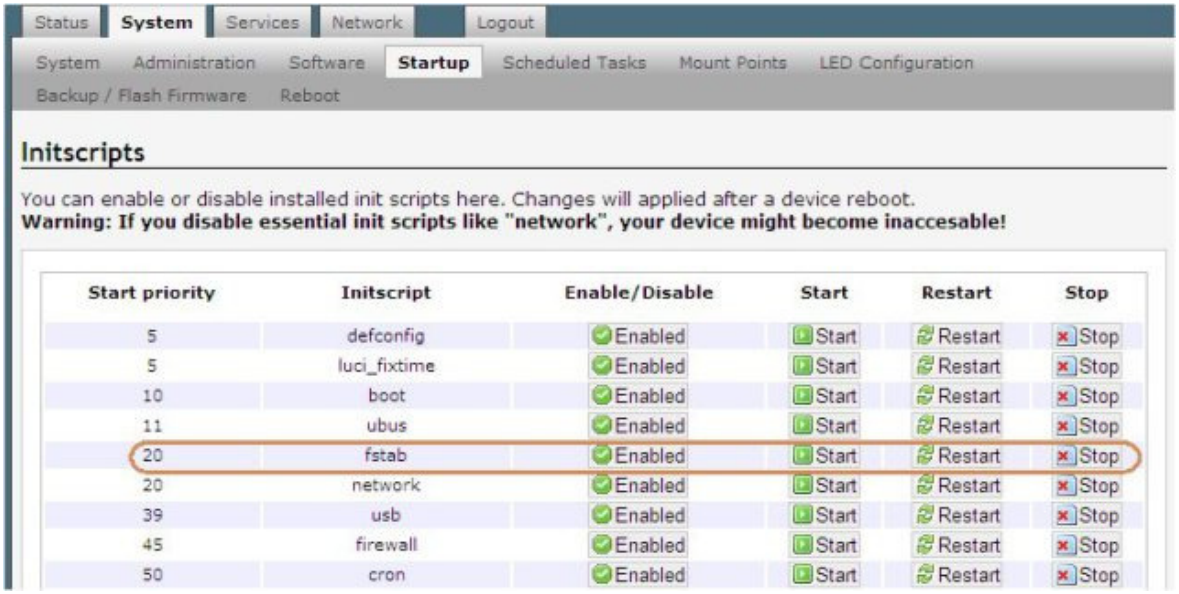

Figura 35: Tela Habilitando o fstab na inicialização.

Deve-se reiniciar o sistema, no menu “*System*” e “*Reboot*”. A Figura 36 ilustra a tela para reiniciar o sistema.

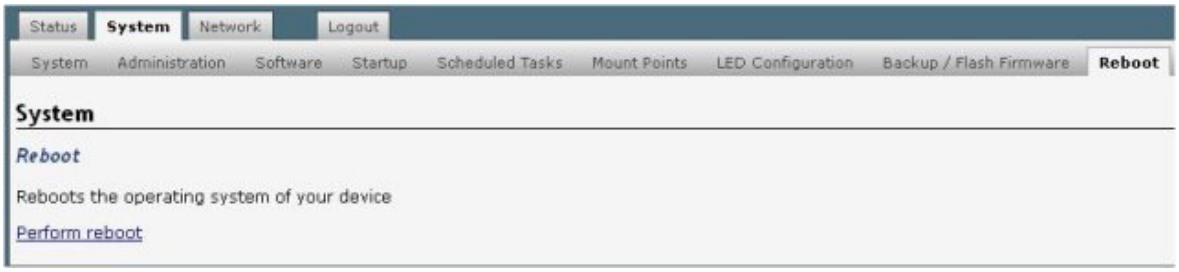

Figura 36: Tela Reiniciando o sistema.

4.2.5 Instalação de pacotes adicionais

No shell, deve-se executar os comandos abaixo para atualizar a lista de pacotes e instalar os pacotes adicionais:

```
$ opkg update
$ opkg remove iptables –force-depends
$ opkg install snmpd
$ opkg install python
$ opkg install pyserial
$ opkg install coreutils-stty
```

Após a instalação, deve-se reiniciar o roteador com o comando “reboot”.

4.2.6 Configurações adicionais

Após a instalação do pacote do snmpd, é necessário habilitar o serviço na inicialização do sistema, para isso, no shell, executar o comando:

```
$ /etc/init.d/snmpd enable
```

4.2.7 Configurando a porta serial

O OpenWrt, por padrão utiliza a porta serial para acesso ao console do sistema.

Para utilizara porta serial para comunicar com o Arduino, deve-se desabilitar o acesso da porta serial para o console. Utilizar os comandos abaixo para editar o arquivo de configuração:

```
$ vim /etc/inittab
```

Com isso o editor vim se abrirá com o arquivo de configuração “inittab”. A Figura 37 ilustra o conteúdo do arquivo “inittab”.

Figura 37: Tela Edição do arquivo inittab.

É necessário remover ou comentar a última linha: ttyATH0::askfirst:/bin/ash –login .

Utilizando o *software*WinSCP, criar o arquivo “setty” no diretório “/etc/init.d” com o conteúdo apresentado no Apêndice 1.

Alterar a permissão do arquivo setty e habilitar para rodar na inicialização com os comandos:

$ chmod 755 /etc/init.d/setty

$ ./etc/init.d/setty enable

Reinicializar o roteador com o comando “reboot”.

4.2.8 Configurando e Habilitando a rede *wireless*

A Figura 38 ilustra os comandos que devem ser executados para configuração da rede wireless.

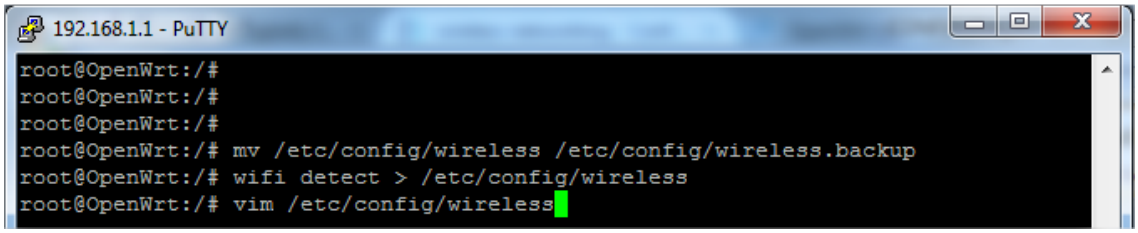

Figura 38: Configuração da rede *wireless*.

Na opção SSID, alterar para o nome da rede de sua preferência e remover a linha comentada. A Figura 39 ilustra o conteúdo do arquivo “wireless” após configuração.

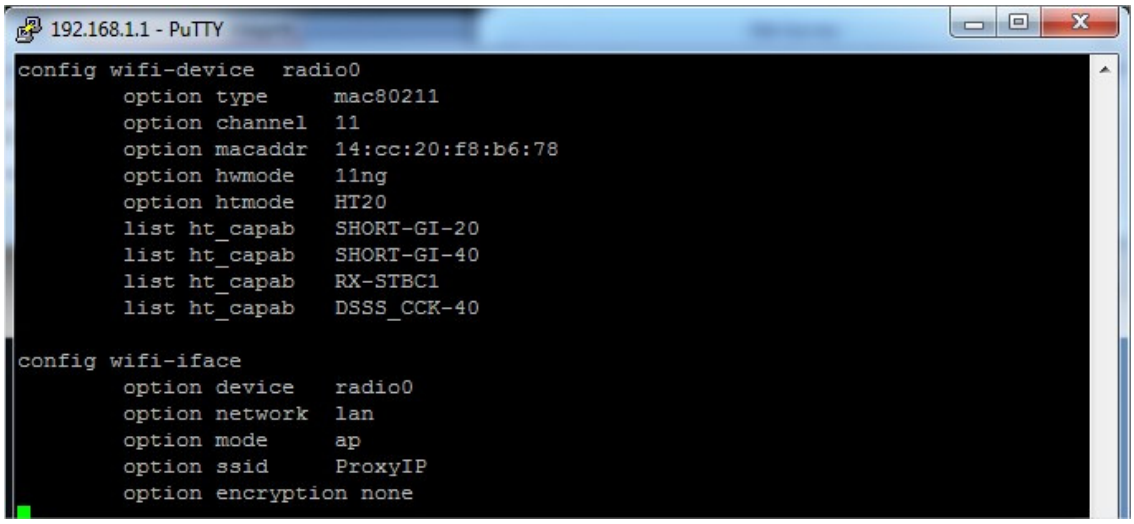

Figura 39: Editando arquivo de configuração da rede *wireless*.

Reiniciar a rede *wireless* com o comando:

$ /etc/init.d/network restart

4.2.9 Troca do servidor web uHTTPd para o Lighttpd.

Executar os comandos abaixo para desabilitar e desinstalar o uHTTPd:

$ /etc/init.d/uhttpd disable

$ /etc/init.d/uhttpd stop

$ opkg remove uhttpd --force-depends

Para instalar o Lighttpd, executar os comandos:

$ opkg update

$ opkg installlighttpd lighttpd-mod-cgi lighttpd-mod-fastcgi lighttpd-mod-access

Após a instalação, enviar o arquivo de configuração do Lighttpd “lighttpd.conf” para o diretório “/etc/lighttpd” utilizando o WinSCP. Sobrescreva o original. O código fonte é apresentado no Apêndice 2.

Reiniciar o servidor Web e habilitar para inicializar com o sistema com os comandos:

$ /etc/init.d/lighttpd restart

$ /etc/init.d/lighttpd enable

4.2.10 Instalando e configurando o PHP 5

Executar os comandos abaixo para instalar o PHP 5:

$ opkg update

$ opkg install php5 php5-cgi php5-fastcgi php5-mod-json php5-mod-session php5-mod-zip libsqlite3 zoneinfo-core php5-mod-pdo php5-mod-pdo-sqlite php5-mod-ctype php5-mod-mbstring php5-mod-gd sqlite3-cli php5-mod-sqlite3 php5-mod-curl curl php5-mod-xml php5-mod-simplexml php5-mod-hash php5-mod-dom php5-mod-iconv

$ opkg install php5-mod-mcrypt php5-mod-openssl php5-mod-fileinfo php5-mod-exif

Utilizar o WinSCP, criar o arquivo “php.ini” no diretório “/etc” com o conteúdo apresentado no Apêndice 3.

Reiniciar e habilitar o PHP para inicializar com o sistema com os comandos:

$ /etc/init.d/php5-fastcgi restart

$ /etc/init.d/php5-fastcgi enable

4.3 Comunicação Serial com o Arduino

A comunicação do ProxyIP com o Arduino se dá via conexão serial.

O pino de transmissão (TX) do Arduino é conectado a porta de Recepção (RX) do ProxyIP, e o pino de recepção (RX) do Arduino é conectado a porta de transmissão do ProxyIP.

Com isso se estabelece uma conexão serial entre os dispositivos, na velocidade já configurada de 9600 bauds.

4.4 Coleta de Dados

Para a coleta de dados das informações vindas do Arduino pela serial, foi necessário o desenvolvimento de um script em linguagem Python, chamado “log.py” que é chamado na inicialização do sistema pelo arquivo “log” e salva todas informações vindas pela serial no arquivo “logs.csv”.

O código fonte do arquivo “log.py” é apresentado no Apêndice 4. O diretório do arquivo é o “/scripts/python”.

O código fonte do arquivo de inicialização “log” é apresentado no Apêndice 5. O diretório do arquivo é o “/etc/init.d”.

O arquivo “logs.csv” será criado automaticamente ao rodar os scripts anteriores. O diretório do arquivo de logs é “/www/proxyip/logs”

4.5 Configuração do SNMP para Múltiplos Sensores

Para a configuração do SNMP para múltiplos sensores, primeiramente foi necessário a configuração do arquivo de configuração do SNMP “snmpd.conf” que fica no diretório “/etc/snmpd”.

Foi adicionado várias entradas de execução, informando o OID escolhido, um nome para o sensor e o caminho do script desenvolvido(sensores.sh). O código fonte do arquivo de configuração “snmpd.conf ” é apresentado no Apêndice 6.

Foi desenvolvido um script chamado “sensores.sh” que possui as rotinas de execução para cada sensor chamado. Podendo rodar comandos Shell ou fazer leituras do arquivo de coleta de dados e mostrar o valor do sensor escolhido. O código fonte do script “sensores.sh” é apresentado no Apêndice 7.

5. APLICAÇÕES E RESULTADOS

Foram implementadas várias aplicações com a utilização da plataforma ProxyIP. A seguir, serão descritas6 aplicações:

5.1 – 1º Aplicação:Monitoramento Remoto de Áreas Urbanas com SNMPvia plataforma aberta Proxy IP

Foi proposto em [40] um experimento que validou a funcionalidade da plataforma Proxy Ip, para realizar o monitoramento remoto de um aspecto do ambiente urbano, utilizando um sensor de luminosidade e realizado monitoramento remoto via SNMP. A coleta dos dados foram realizadas em ambiente urbano, na área externa de uma residência localizada na cidade de Vinhedo-SP e as informações disponibilizadas em tempo real via SNMP.

A Figura 40 ilustra o protótipo da 1º aplicação, sendo destacado o *hardware* utilizado.

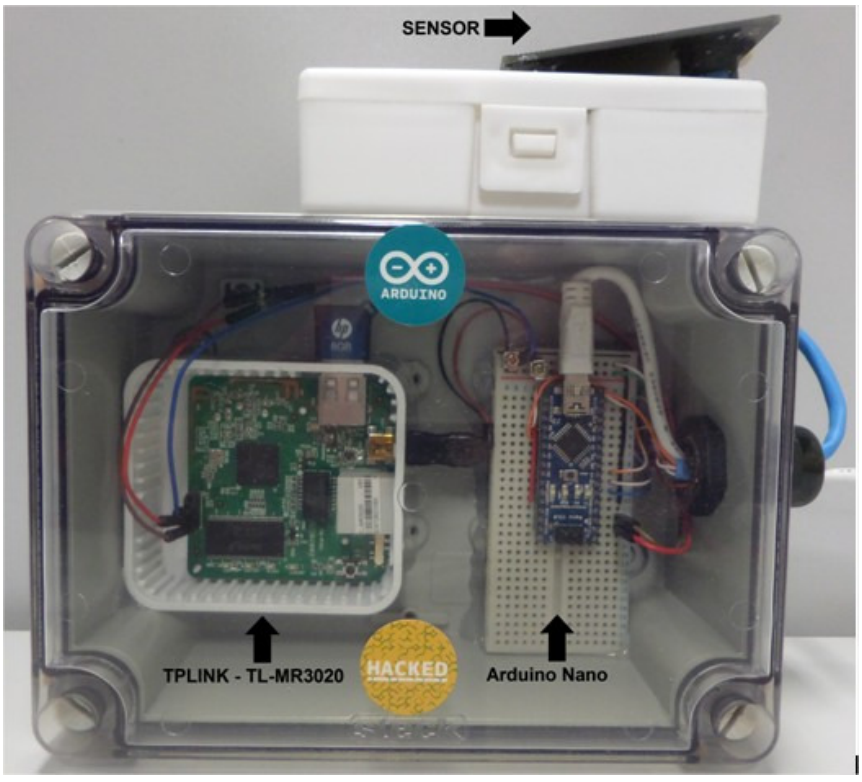

Figura40: Protótipo da 1º Aplicação: ProxyIP V2.0

A Figura 41 ilustra comoresultado da 1º aplicação, o gráfico do monitoramento remoto da luminosidade, no período de 6 horas com intervalos de 15 minutos.

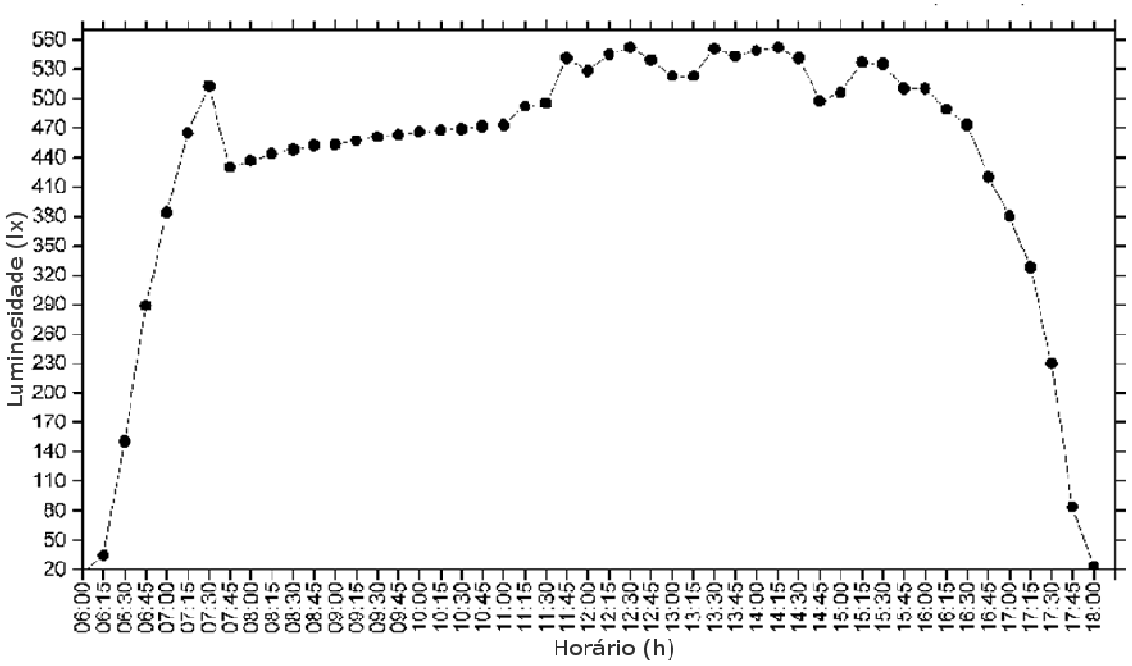

Figura 41: Monitoramento Remoto da Luminosidade

5.2 – 2º Aplicação: Sistema de Irrigação Autônomo e Auto-Suficiente utilizando Energia Solar

Foi detalhado em [41] a construção de um protótipo de irrigação autônomo e auto-suficiente utilizando energia solar como principal fonte de alimentação e a rede elétrica como alimentação secundária. Não sendo necessário a intervenção humana após a instalação, o funcionamento se dá pela utilização de painéis solares, que carregam uma bateria, e esta alimenta uma bomba d´gua. Utilizando um sensor de umidade de terra para acionamento da bomba d´agua quando necessário. A motivação para este trabalho foi buscar um meio de geração de energia limpa e renovável para se implementar um sistema de irrigação que além de economizar água, também economizasse energia elétrica que por sua vez foi gerada sem danos para o meio ambiente, de forma sustentável.

Os dados medidos são disponibilizados em tempo real via Internet utilizando a plataforma ProxyIP e a mesma plataforma se encarrega de fazer a coleta dos dados .

A Figura 42 ilustra o protótipo da 2º aplicação acondicionado em caixa hermética.

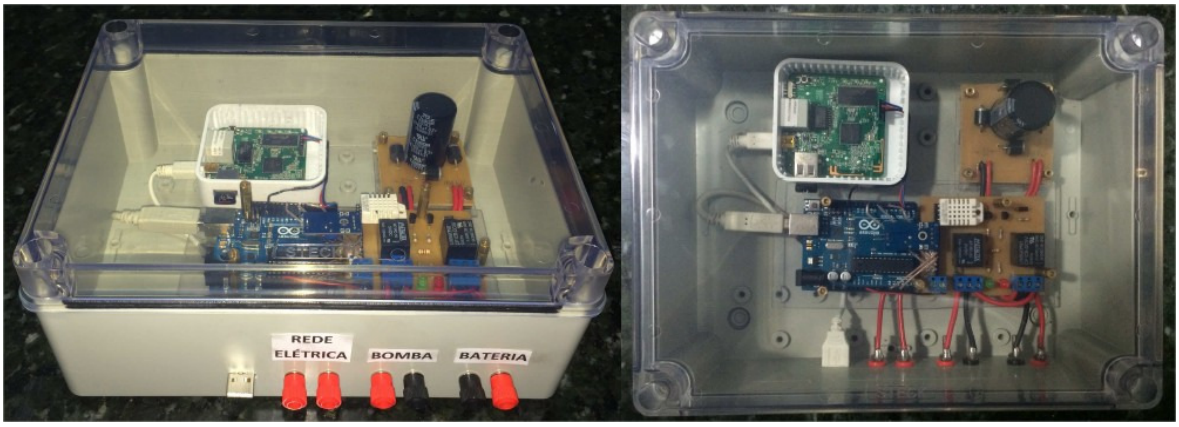

Figura 42: Protótipo da 2º Aplicação

A Figura 43 ilustra como resultadoda 2º aplicação, a *DashBoard* utilizando 4 sensores.

Figura 43: Interface de Monitoramento via Internet

5.3 – 3º Aplicação: Monitoramento do Desperdício de Àgua em Cidades Inteligentes

Foi detalhado em [42] a construção de um sistema para monitorar o desperdício de água, onde utilizando sensores de fluxo de água e de movimento ligados a um microcontrolador e utilizando a plataforma Proxy IP, é possivel alertar o usuário que está ocorrendo desperdício de água. A motivaçao para este trabalho foi a utilização conciente do uso da água, através da utilização de sensores.

A Figura 44 ilustra o protótipo da 3º aplicação e seus sensores.

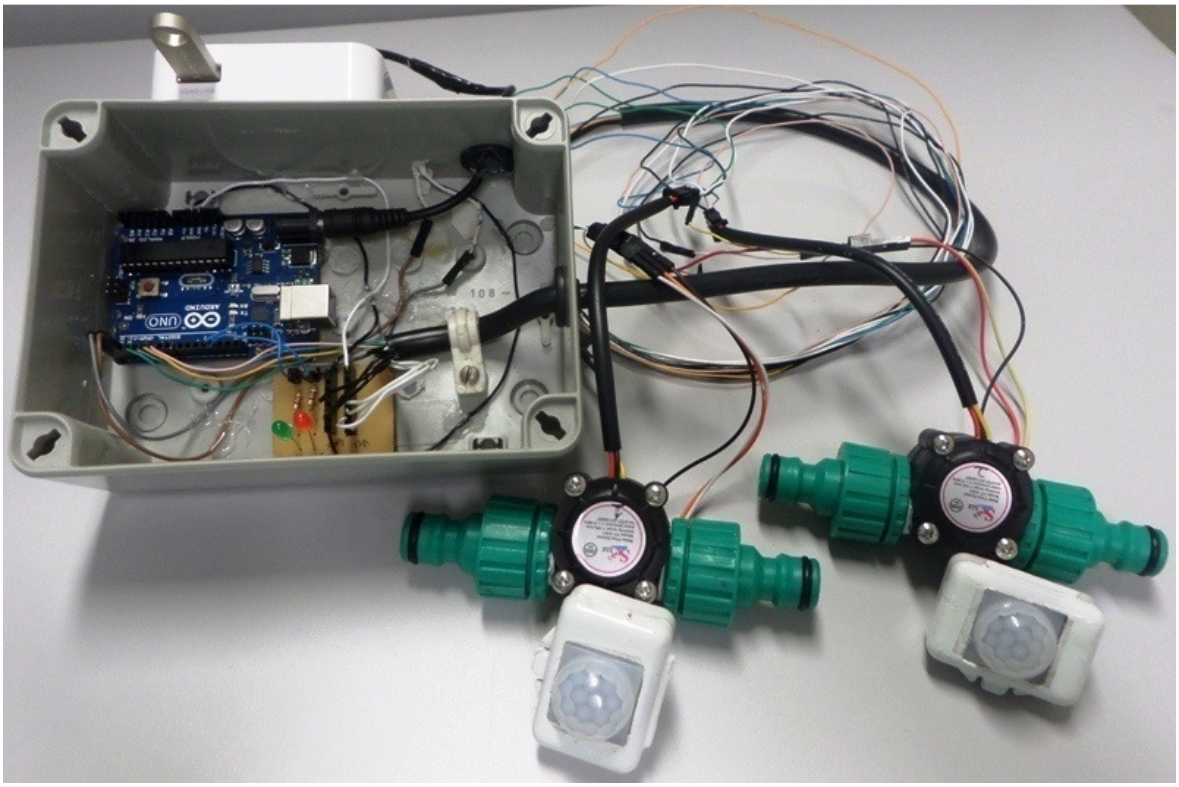

Figura 44: Protótipo da 3º Aplicação

A Figura 45 ilustra como resultadoda 3º aplicação, a *DashBoard* utilizando 2 sensores de vazão, com alerta de desperdício de água.

Figura 45: Monitoramento utilizando o Proxy IP

5.4 – 4º Aplicação: Implementação de Sensores de Temperatura do Ar em Smart Cities com o padrão IEEE802.11

Foi detalhado em [43] a construção de um sistema para aferir a temperatuda do ar dos centros urbanos, utilizando tecnologia sem fio IEEE 802.11 para transmitir grandezas em tempo real. Foi utilizado a plataforma Proxy IP.Ensaio realizado utilizando 6 sensores de temperatura LM35, e efetuando a coleta da temperatura em ambiente urbano. A motivação para este trabalho foi da necessidade de aferição da temperatura do ar dos centros urbanos existentes.

A Figura 46 ilustra o protótipo da 4º aplicação sendo destacado o *hardware* e sensores utilizados.

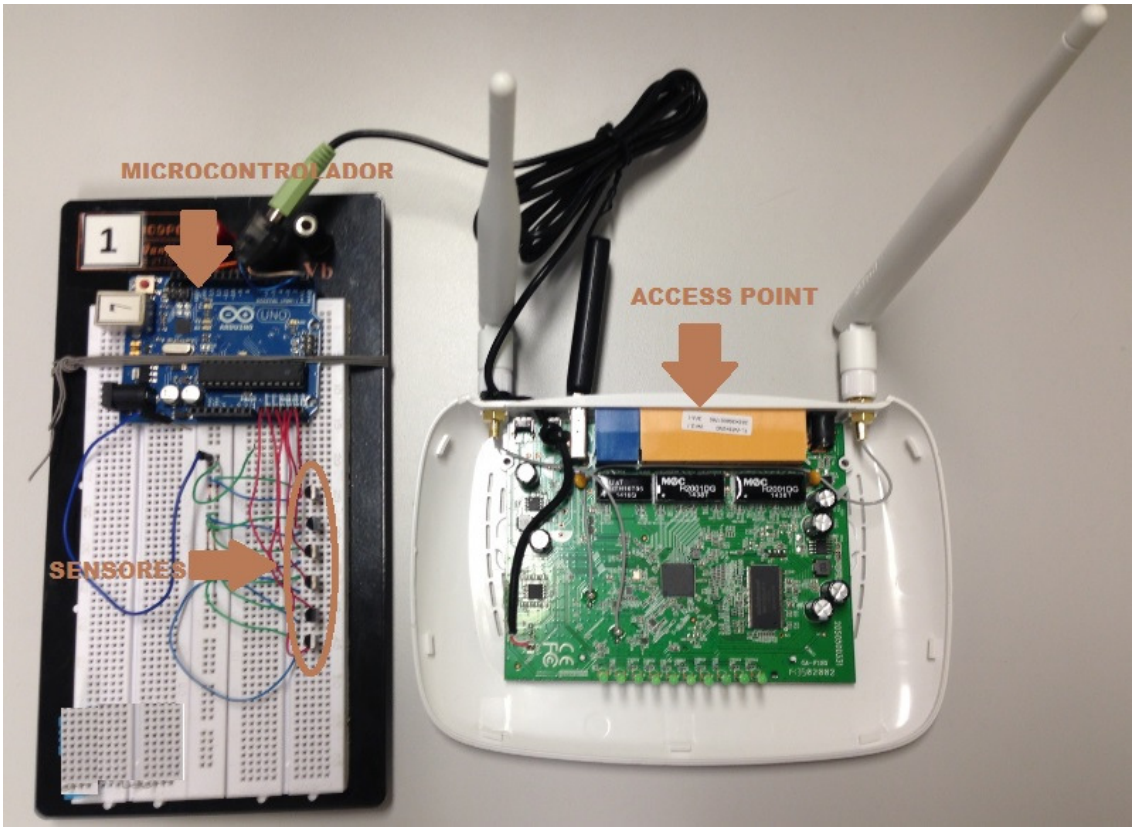

Figura46: Protótipo da 4º Aplicação

A Figura 47 ilustra como resultado da 4º aplicação, o gráfico do monitoramento da temperatura do ar, utilizando seis sensores. Os ensaios foram realizados em um laboratório da PUC-Campinas.

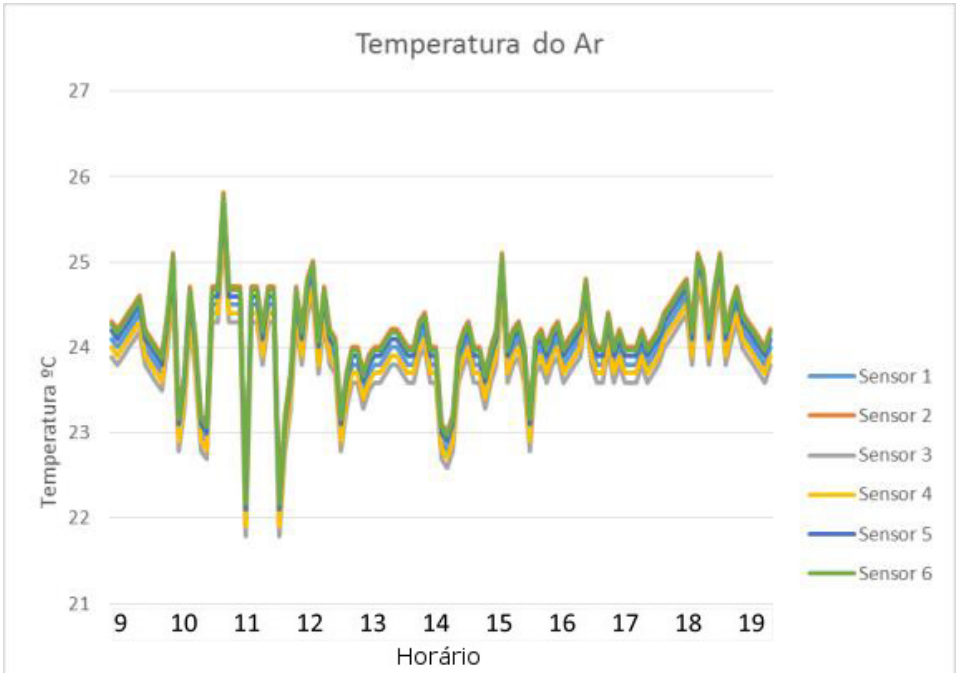

Figura 47: Monitoramento utilizando o Proxy IP

5.5 – 5º Aplicação: Sensoriamento de Nível de Líquidos Utilizando Plataforma Proxy IP para Cidades Inteligentes

Foi detalhado em [44] a construção de um sistema para monitorar o nível de líquidos em cidades inteligentes. Foi desenvolvido um sensor de nível de líquidos utilizando condutividade e foi utilizado a plataforma Proxy IP para coleta e visualização do nível remotamente utilizando o protocolo SNMP. A motivação para este trabalho foi a necessidade de efetuar a medição remota do nível de líquidos.

A Figura 48 ilustra o protótipo da 5º aplicação e o sensor de nível desenvolvido.

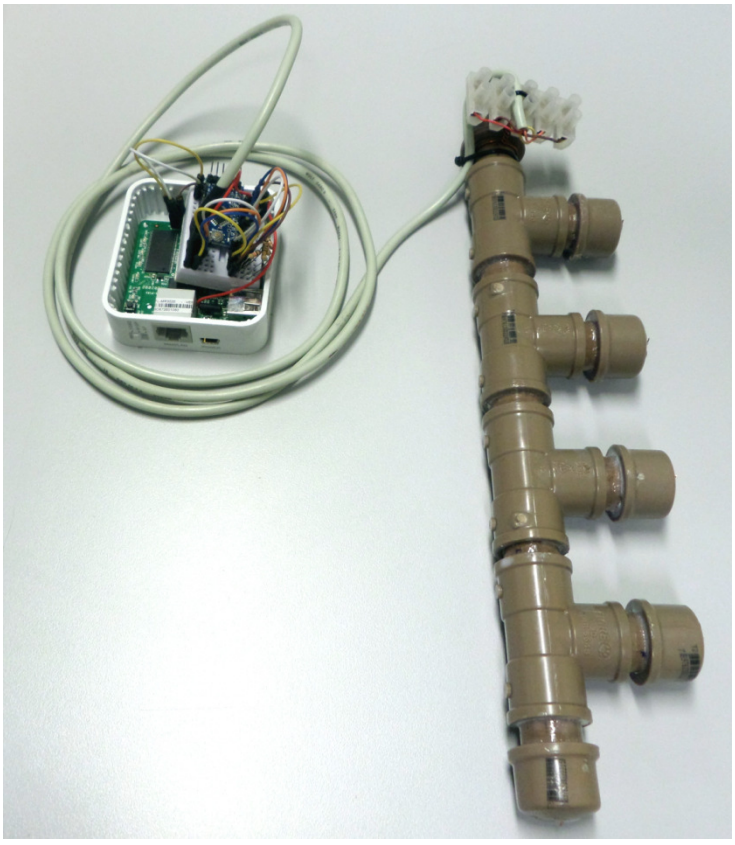

Figura 48: Protótipo da 5º Aplicação

A Figura 49 ilustra como resultado da 5º aplicação, o gráfico do monitoramento do nível de líquido, em todos os seus cinco estágios.

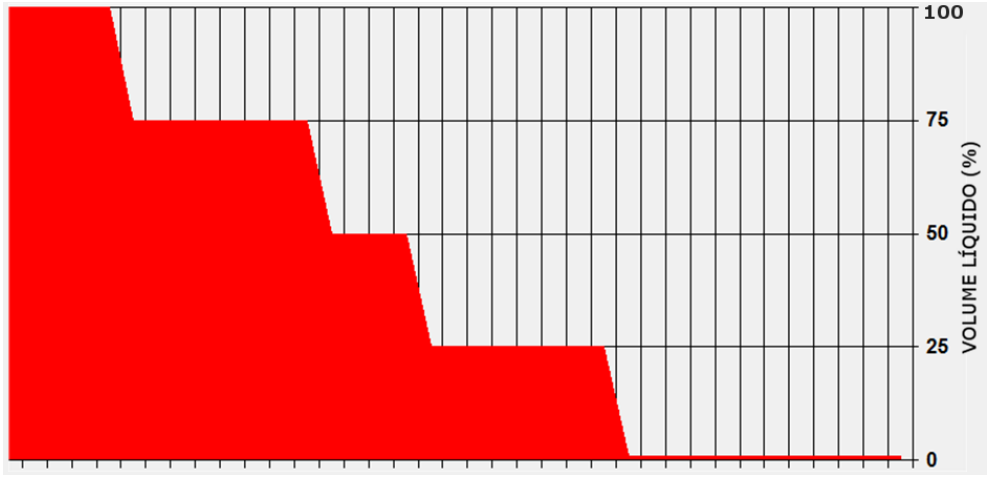

Figura 49: Monitoramento de nível utilizando o Proxy IP

5.6 – 6º Aplicação: Medição de Variação de Tensão em Redes de Baixa Tensão Utilizando Redes sem Fio IEEE 802.11

Foi detalhado em [45] a construção de um sistema para sensoriamento de baixo custo capaz de detectar variações de longa e curta duração nas redes elétricas de baixa tensão. Foi utilizado um sensor de corrente e um de tensão,e foi utilizado a plataforma Proxy IP para coleta e visualização das medições. A motivação para este trabalho foi apresentar uma solução para sensoriamento de baixo custo capaz de detectar variações de tensão de longa e curta duração nas redes elétricas de baixa tensão através de uma rede sem fio utilizando o padrão IEEE 802.11.

A Figura 50 ilustra o protótipo da 6º aplicação sendo destacado o *hardware* e sensores utilizados.

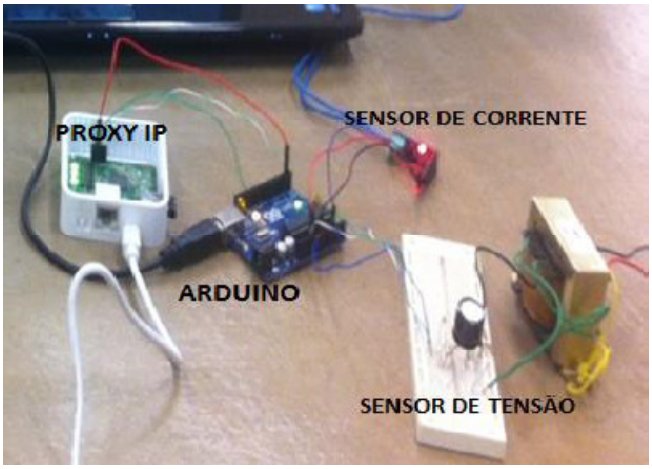

Figura50: Protótipo da 6º Aplicação

A Figura 51 ilustra como resultadoda 6º aplicação, a *DashBoard* utilizando 2 sensores, monitorando tensão e corrente.

Figura 51: Monitoramento de Corrente e Tensão utilizando o Proxy IP

6. CONCLUSÃO

Neste trabalho, foi proposta a implementação da plataforma ProxyIP utilizando múltiplos sensores para Cidades Inteligentes. Foram utilizadas três aplicações para validação de suas funcionalidades e de sua aplicação em Cidades Inteligentes.

Nas aplicações, foifeita a coleta de dados de diversos sensores (luminosidade, umidade do ar, umidade da terra, etc.), para validar os resultados.

Desta forma, há viabilidade de utilização da plataforma ProxyIP para monitoramento de múltiplos sensores para Cidades Inteligentes, uma vez que o monitoramento pode ser via protocolo SNMP ou utilizando a interface Web *DashBoard*.Dando importância também para a coleta de dados que a plataforma realiza, para possível tratamento dos dados posteriormente.

A plataforma possui limitações de hardware, com um limite de sensores que podem ser utilizados. Este limite depende do microcontrolador utilizado, no caso deste trabalho, como foi utilizado o Arduino Nano, o limite é de 14 pinos digitais e 8 pinos analógicos.

Para trabalhos futuros, sugere-se a utilização de um microprocessador mais potente e com mais portas de entrada/saída, para expansão do número de sensores que podem ser utilizados no sistema.

Por fim, sugere-se também uma melhoria no sistema de *DashBoard*, utilizando mostradores digitais.

REFERÊNCIAS

[1] HOLLAND, R. G. Will the real smart city please stand up? City, n. 12, 2008. pp. 303-320.

[2] NAM, T.; PARDO, T. A. Conceptualizing Smart City with Dimensions ofTechnology, People and Institutions. 12th Annual International Conference on DigitalGovernment Research. College Park: [s.n.]. 2011. p. 282-291.

[3] SCHAFFERS, H. et al. Smart cities and the future internet: towardscooperation frameworks for open innovation. Lecture Notes in Computer Science, n. 6656, p. 431-446, 2011.

[4] HERNÁNDEZ-MUÑÕZ , J.M. et al.Smart cities at the forefront of the future internet.Lecture Notes in Computer Science, n. 6656, p. 447–462, 2011.

[5] CHOURABI, H. et al.Understanding smart cities: an integrative framework. In: HAWAII INTERNATIONAL CONFERENCE ON SYSTEM SCIENCES, 45th., 2012, Hawaii. Anais... Albany: Center for Technology in Government, 2012. p. 2289-2297.

[6] CADENA, A.; DOBBS, R.; REMES, J. The growing economic power of cities. Journal of International Affairs, v. 65, n. 2, p. 1-17, 2012.

[7] DIRKS, S.; GURDGIEV, C.; KEELING, M. Smarter cities for smarter growth: how cities can optimize their systems for the talent-based economy.IBM Institute for Business Value: Executive Report, 2010.
Disponível em:
<http://ssrn.com/abstract=2001907>.
[Acesso em 10 12 2015].

[8] ALLWINKLE, S; CRUICKSHANK, P. Creating smart-er cities: an overview. Journal of Urban Technology, v. 18, n. 2, p. 1-16, 2011.

[9] D, Evans. The Internet of Things: How the Next Evolution of the Internet Is Changing Everything. San Jose, CA. Cisco White paper. Apr. 2011, [Online] Disponível em: <http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf>. [Acesso em 01 02 2015].

[10] MercadoLivre Brasil, [Online] Disponível em: <http://www.mercadolivre.com.br>. [Acesso em 2612 2015].

[11] Arduino, [Online] Disponível em: <http://www.arduino.cc>. [Acesso em 12 05 2014.

[12] ATmega328, [Online] Disponível em: <http://www.atmel.com/pt/br/devices/ATMEGA328.aspx?tab=overview>. [Acesso em 10 04 2014].

[13] ATmega168/328 – Arduino Pin Mapping, [Online] Disponível em: <https://www.arduino.cc/en/Hacking/PinMapping168>. [Acesso em 22 04 2014].

[14] Arduino – Boards, [Online] Disponível em: <https://www.arduino.cc/en/Main/Boards>. [Acesso em 12 05 2014].

[15] Arduino Nano, [Online] Disponível em: <https://www.arduino.cc/en/Main/ArduinoBoardNano>. [Acesso em 12 05 2014].

[16] Arduino Getting Started, [Online]
Disponível em:
<https://www.arduino.cc/en/Guide/Windows>.
[Acesso em 15 05 2014].

[17] OpenWrt Table of Hardware – TP-Link TL-WR741ND, [Online]
Disponível em:
<http://wiki.openwrt.org/toh/tp-link/tl-wr741nd >.
[Acesso em 10 09 2014].

[18] OpenWrt Table of Hardware – TP-Link TL-MR3020, [Online]
Disponível em:
<http://wiki.openwrt.org/toh/tp-link/tl-mr3020>.
[Acesso em 18 03 2015].

[19] OpenWrt, [Online]
Disponível em: <http://www.openwrt.org>.
[Acesso em 18 03 2015].

[20] Official OpenWrt P*ackages*, [Online]
Disponível em:
<https://downloads.openwrt.org/attitude_adjustment/12.09/ar71xx/generic/packages/>.
[Acesso em 26 09 2015].

[21] OpenWrt – Table of Hardware, [Online]
Disponível em: <http://wiki.openwrt.org/toh/start>.
[Acesso em 20 09 2015].

[22] OpenWrt First Login, [Online]
Disponível em:
<http://wiki.openwrt.org/doc/howto/firstlogin>.
[Acesso em 20 09 2015].

[23] LuCI – Technical Reference, [Online]
Disponível em:
<http://wiki.openwrt.org/doc/techref/luci>.
[Acesso em 20 09 2015].

[24] A Linguagem de Programação Lua, [Online]
Disponível em:
<http://www.lua.org/portugues.html>.
[Acesso em 27 09 2015].

[25] Simple Network Management Protocol (SNMP) – Cisco Wiki
[Online] Disponível em:
<http://docwiki.cisco.com/wiki/Simple_Network_Management_Protocol>.
[Acesso em 10 07 2014].

[26] STALLINGS, W. SNMP, SNMPv2, SNMPv3, and RMON 1 and 2,
Boston:Addison-Wesley, 1999.

[27] LEINWAND, A.; CONROY, K. F. Network Management: APractical
Perspective. 2a. ed. SanJose: Addison–Wesley, 1996.

[28] SNMP – RFC 1157, [Online]
Disponível em:
<https://www.ietf.org/rfc/rfc1157.txt>
[Acesso em 10 09 2015].

[29] SNMP – RFC1213, [Online]
Disponível em:
<https://www.ietf.org/rfc/rfc1213.txt>
[Acesso em 10 09 2015].

[30] SNMP – RFC1441, [Online]
Disponível em:
<https://www.ietf.org/rfc/rfc1441.txt>
[Acesso em 10 09 2015].

[31] SNMP – RFC1442, [Online]
Disponível em:
<https://www.ietf.org/rfc/rfc1442.txt>
[Acesso em 10 09 2015].

[32] SNMP – RFC 3410, [Online]
Disponível em:
<https://www.ietf.org/rfc/rfc3410.txt>
[Acesso em 10 09 2015].

[33] SNMP – RFC 3414, [Online]
Disponível em:
<https://www.ietf.org/rfc/rfc3414.txt>
[Acesso em 10 09 2015].

[34] SNMP – RFC 3415, [Online]
Disponível em:
<https://www.ietf.org/rfc/rfc3415.txt>
[Acesso em 10 09 2015].

[35] SNMP – RFC 3418, [Online]
Disponível em:
<https://www.ietf.org/rfc/rfc3418.txt>
[Acesso em 10 09 2015].

[36] TP-Link TL-MR3020 OpenWrt Firmware, [Online]
Disponível em:
<http://downloads.openwrt.org/attitude_adjustment/12.09/ar71xx/generic/openwrt-ar71xx-generic-tl-mr3020-v1-squashfs-factory.bin>.
[Acesso em 10 11 2015].

[37] PuTTY Binaries, [Online]
Disponível em:
<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>.
[Acesso em 2210 2015].

[38] MiniTool Partition Wizard, [Online]
Disponível em:
<http://www.partitionwizard.com/free-partition-manager.html>.
[Acesso em 2210 2015].

[39] WinSCP Downloads, [Online]
Disponível em:
<https://winscp.net/eng/download.php>.
[Acesso em 10 09 2015].

[40] Machado, L.; Mota, A.; Mota, L. - SNMP Management of Urban Areas Remote Monitoring via Open Platform Proxy-IP.
1st International Electronic Conference on Remote Sensing
22 Junho–05 Julho 2015; Sciforum Electronic Conference Series, Vol. 1, 2015 , a001; doi:10.3390/ecrs-1-a001.

[41] Gonçalves, C. R.; Machad,o L. F.; Mota, A.; Mota, L. - Sistema de irrigação autônomo e auto-suficiente utilizando energia solar
Cobenge 2015 – XLIII Congresso Brasileiro de Educação em Engenharia
08–11 Setembro 2015; São Bernado do Campo – Brasil.

[42] Riboldi, V. B.; Machado, L. F.; Mota, A.; Mota, L. - Monitoramento do Desperdício de Água em Cidades Inteligentes
Brazilian Technology Symposium (BTSym'15)
2015; Unicamp – Campinas – Brasil.

[43]Oliveira, E. D.; Machado, L. F.; Podeleski, F. S.; Santos, A. A.; Mota, L.; Mota, A. - Implementação de Sensores de Temperatura do Ar em Smart Cities com o padrão IEEE802.11
Brazilian Technology Symposium (BTSym'15)
2015; Unicamp – Campinas – Brasil.

[44]Machado, L.; Mota, A.; Mota, L. - Sensoriamento de nível de líquidos utilizando plataforma Proxy IP para cidades inteligentes.
I Simpósio do Programa de Pós-Graduação Stricto Sensu em Sistemas de Infraestrutura Urbana –PUC- Campinas - SPInfra, 2014, Campinas.

[45]Santos, A. H. R.; Frangeto, C. F.; Machado, L. F.; Lemos, I. P.; Mota, A.; Mota, L. - Medição de variação de tensão em redes de baixa tensão utilizando redes sem fio IEEE802.11.
Revista Sodebras, v. 10, p. 52-57, 2015.

APÊNDICES

Apêndice 1

Arquivo: setty

```
#!/bin/sh /etc/rc.common

# Pos-Graduacao PUC-Campinas - 2015

#

# Script para configurar a porta serial padrao ttyATH0

# na inicializacao do sistema.

#

# Controle de Versao: 1.0


# Inicia o Script apos o script de inicializacao 9 e para apos o 14

START=10

STOP=15


# Comandos ao inicializar sistema.

# Configura a porta serial ttyATH0 na velocidade 9600 e em modo raw

boot() {

        stty -F /dev/ttyATH0 9600 raw min 0 time 4

        }


# Comandos ao iniciar a aplicacao.

# Configura a porta serial ttyATH0 na velocidade 9600 e em modo raw

start() {

        stty -F /dev/ttyATH0 9600 raw min 0 time 4

        }
```

Apêndice 2

Arquivo: lighttpd.conf

```
# lighttpd configuration file

server.modules = (

        "mod_access",

        "mod_fastcgi",

        "mod_cgi",

)

server.network-backend = "write"

server.document-root = "/www/proxyip/"

index-file.names = ( "index.php", "index.html", "default.html", "index.htm",
"default.htm" )

mimetype.assign = (

        ".pdf"    => "application/pdf",

        ".class" => "application/octet-stream",

        ".pac"    => "application/x-ns-proxy-autoconfig",

        ".swf"    => "application/x-shockwave-flash",

        ".wav"    => "audio/x-wav",

        ".gif"    => "image/gif",

        ".jpg"    => "image/jpeg",

        ".jpeg"   => "image/jpeg",

        ".png"    => "image/png",

        ".svg"    => "image/svg+xml",

        ".css"    => "text/css",

        ".html"   => "text/html",

        ".htm"    => "text/html",
```

```
        ".js"    => "text/javascript",
        ".txt"   => "text/plain",
        ".dtd"   => "text/xml",
        ".xml"   => "text/xml"
 )
$HTTP["url"] =~ "\.pdf$" {
        server.range-requests = "disable"
}
static-file.exclude-extensions = ( ".php", ".pl", ".fcgi" )
server.pid-file = "/var/run/lighttpd.pid"
server.upload-dirs = ( "/tmp" )
fastcgi.server = (
        ".php" => ((
                        "socket" => "/tmp/php.socket",
                        "bin-path" => "/usr/bin/php-fcgi",
                        "max-procs" => 1
                )
        )
)
$SERVER["socket"] == ":81" {
        server.document-root = "/www"
       cgi.assign = ("luci" => "/usr/bin/lua")
}
```

Apêndice 3

Arquivo: php.ini

```
[PHP]

zend.ze1_compatibility_mode = Off

; Language Options

engine = On
;short_open_tag = Off
precision    =  12
y2k_compliance = On
output_buffering = Off
;output_handler =
zlib.output_compression = Off
;zlib.output_compression_level = -1
;zlib.output_handler =
implicit_flush = Off
unserialize_callback_func =
serialize_precision = 100

;open_basedir = "/www"
disable_functions =
disable_classes =

; Colors for Syntax Highlighting mode.  Anything that's acceptable in
```

; would work.

;highlight.string = #DD0000

;highlight.comment = #FF9900

;highlight.keyword = #007700

;highlight.bg   = #FFFFFF

;highlight.default = #0000BB

;highlight.html   = #000000

;ignore_user_abort = On

;realpath_cache_size = 16k

;realpath_cache_ttl = 120

; Miscellaneous

expose_php = On

; Resource Limits

max_execution_time = 30 ; Maximum execution time of each script, in seconds.

max_input_time = 60 ; Maximum amount of time each script may spend parsing request data.

;max_input_nesting_level = 64

memory_limit = 32M ; Maximum amount of memory a script may consume.

```
; Error handling and logging


; Error Level Constants:
; E_ALL            - All errors and warnings (includes E_STRICT as of PHP 6.0.0)
; E_ERROR          - fatal run-time errors
; E_RECOVERABLE_ERROR  - almost fatal run-time errors
; E_WARNING        - run-time warnings (non-fatal errors)
; E_PARSE          - compile-time parse errors
; E_NOTICE         - run-time notices (these are warnings which often result
;                  from a bug in your code, but it's possible that it was
;                  intentional (e.g., using an uninitialized variable and
;                  relying on the fact it's automatically initialized to an
;                  empty string)
; E_STRICT                   - run-time notices, enable to have PHP suggest
changes
;                  to your code which will ensure the best interoperability
;                  and forward compatibility of your code
; E_CORE_ERROR     - fatal errors that occur during PHP's initial startup
; E_CORE_WARNING   - warnings (non-fatal errors) that occur during PHP's
;                  initial startup
; E_COMPILE_ERROR  - fatal compile-time errors
; E_COMPILE_WARNING - compile-time warnings (non-fatal errors)
; E_USER_ERROR     - user-generated error message
; E_USER_WARNING   - user-generated warning message
```

```
; E_USER_NOTICE     - user-generated notice message
; E_DEPRECATED      - warn about code that will not work in future versions
;                     of PHP
; E_USER_DEPRECATED - user-generated deprecation warnings
;
; Common Values:
;   E_ALL & ~E_NOTICE  (Show all errors, except for notices and coding standards warnings.)
;   E_ALL & ~E_NOTICE | E_STRICT  (Show all errors, except for notices)
;   E_COMPILE_ERROR|E_RECOVERABLE_ERROR|E_ERROR|E_CORE_ERROR  (Show only errors)
;   E_ALL | E_STRICT  (Show all errors, warnings and notices including coding standards.)
; Default Value: E_ALL & ~E_NOTICE
error_reporting  =  E_ALL & ~E_NOTICE & ~E_STRICT

display_errors = On
display_startup_errors = Off
log_errors = Off
log_errors_max_len = 1024
ignore_repeated_errors = Off
ignore_repeated_source = Off
report_memleaks = On
;report_zend_debug = 0
```

```
track_errors = Off
;html_errors = Off
;docref_root = "/phpmanual/"
;docref_ext = .html
;error_prepend_string = "<font color=#ff0000>"
;error_append_string = "</font>"
; Log errors to specified file.
;error_log = /var/log/php_errors.log
; Log errors to syslog.
error_log = syslog

; Data Handling

;arg_separator.output = "&amp;"
;arg_separator.input = ";&"
variables_order = "EGPCS"
request_order = "GP"
register_globals = Off
register_long_arrays = Off
register_argc_argv = On
auto_globals_jit = On
post_max_size = 8M
;magic_quotes_gpc = Off
magic_quotes_runtime = Off
```

```
magic_quotes_sybase = Off

auto_prepend_file =

auto_append_file =

default_mimetype = "text/html"

;default_charset = "iso-8859-1"

;always_populate_raw_post_data = On


; Paths and Directories


; UNIX: "/path1:/path2"

;include_path = ".:/php/includes"

;doc_root = "/www"

user_dir =

extension_dir = "/usr/lib/php"

enable_dl = On

;cgi.force_redirect = 1

;cgi.nph = 1

;cgi.redirect_status_env = ;

cgi.fix_pathinfo=1

;fastcgi.impersonate = 1;

;fastcgi.logging = 0

;cgi.rfc2616_headers = 0


; File Uploads
```

```
file_uploads = On

upload_tmp_dir = "/tmp"

upload_max_filesize = 2M

max_file_uploads = 20


; Fopen wrappers


allow_url_fopen = On

allow_url_include = Off

;from="john@doe.com"

;user_agent="PHP"

default_socket_timeout = 60

;auto_detect_line_endings = Off


; Dynamic Extensions


;extension=ctype.so

;extension=curl.so

;extension=dom.so

;extension=exif.so

;extension=ftp.so

;extension=gd.so

;extension=gmp.so
```

```
;extension=hash.so

;extension=iconv.so

;extension=json.so

;extension=ldap.so

;extension=mbstring.so

;extension=mcrypt.so

;extension=mysql.so

;extension=openssl.so

;extension=pcre.so

;extension=pdo.so

;extension=pdo-mysql.so

;extension=pdo-pgsql.so

;extension=pdo_sqlite.so

;extension=pgsql.so

;extension=session.so

;extension=soap.so

;extension=sockets.so

;extension=sqlite.so

;extension=sqlite3.so

;extension=tokenizer.so

;extension=xml.so

;extension=xmlreader.so

;extension=xmlwriter.so
```

; Module Settings

[APC]

apc.enabled = 1

apc.shm_segments = 1 ;The number of shared memory segments to allocate for the compiler cache.

apc.shm_size = 4M ;The size of each shared memory segment.

[Date]

;date.timezone =

;date.default_latitude = 31.7667

;date.default_longitude = 35.2333

;date.sunrise_zenith = 90.583333

;date.sunset_zenith = 90.583333

[filter]

;filter.default = unsafe_raw

;filter.default_flags =

[iconv]

;iconv.input_encoding = ISO-8859-1

;iconv.internal_encoding = ISO-8859-1

;iconv.output_encoding = ISO-8859-1

```
[sqlite]
;sqlite.assoc_case = 0

[sqlite3]
;sqlite3.extension_dir =

[Pdo_mysql]
pdo_mysql.cache_size = 2000
pdo_mysql.default_socket=

[MySQL]
mysql.allow_local_infile = On
mysql.allow_persistent = On
mysql.cache_size = 2000
mysql.max_persistent = -1
mysql.max_links = -1
mysql.default_port =
mysql.default_socket =
mysql.default_host =
mysql.default_user =
mysql.default_password =
mysql.connect_timeout = 60
mysql.trace_mode = Off
```

```
[PostgresSQL]
pgsql.allow_persistent = On
pgsql.auto_reset_persistent = Off
pgsql.max_persistent = -1
pgsql.max_links = -1
pgsql.ignore_notice = 0
pgsql.log_notice = 0

[Session]
session.save_handler = files
session.save_path = "/tmp"
session.use_cookies = 1
;session.cookie_secure =
session.use_only_cookies = 1
session.name = PHPSESSID
session.auto_start = 0
session.cookie_lifetime = 0
session.cookie_path = /
session.cookie_domain =
session.cookie_httponly =
session.serialize_handler = php
session.gc_probability = 1
session.gc_divisor     = 100
session.gc_maxlifetime = 1440
```

session.bug_compat_42 = On

session.bug_compat_warn = On

session.referer_check =

session.entropy_length = 0

;session.entropy_file = /dev/urandom

session.entropy_file =

;session.entropy_length = 16

session.cache_limiter = nocache

session.cache_expire = 180

session.use_trans_sid = 0

session.hash_function = 0

session.hash_bits_per_character = 4

url_rewriter.tags = "a=href,area=href,frame=src,input=src,form=,fieldset="

[mbstring]

;mbstring.language = Japanese

;mbstring.internal_encoding = EUC-JP

;mbstring.http_input = auto

;mbstring.http_output = SJIS

;mbstring.encoding_translation = Off

;mbstring.detect_order = auto

;mbstring.substitute_character = none;

;mbstring.func_overload = 0

;mbstring.strict_detection = Off

```
;mbstring.http_output_conv_mimetype=

;mbstring.script_encoding=


[gd]

;gd.jpeg_ignore_warning = 0


[exif]

;exif.encode_unicode = ISO-8859-15

;exif.decode_unicode_motorola = UCS-2BE

;exif.decode_unicode_intel    = UCS-2LE

;exif.encode_jis =

;exif.decode_jis_motorola = JIS

;exif.decode_jis_intel    = JIS


[soap]

soap.wsdl_cache_enabled=1

soap.wsdl_cache_dir="/tmp"

soap.wsdl_cache_ttl=86400

soap.wsdl_cache_limit = 5


[sysvshm]

;sysvshm.init_mem = 10000


[ldap]
```

```
ldap.max_links = -1


[mcrypt]
;mcrypt.algorithms_dir=
;mcrypt.modes_dir=
```

Apêndice 4

Arquivo: log.py

```
#!/usr/bin/env python

import serial

import time

ser = serial.Serial('/dev/ttyATH0', 9600, timeout=1)

#ser.open()

while 1:

    ser.open()

    logfile=open('/www/proxyip/logs/logs.csv', 'a')

    line = ser.readline()

    now = time.strftime("%d/%m/%Y %H:%M:%S", time.localtime())

    a =  "%s %s %s" % (now, line, "\n")

#    print a

    logfile.write(a)

#logfile.flush()

#logfile.close()

#ser.open()

#ser.write("t")

#print ser.readline()

    ser.flush()

    time.sleep(1)

    ser.close()
```

Apêndice 5

Arquivo: log

```
#!/bin/sh /etc/rc.common

START=98
SERVICE_DAEMONIZE=1
PYTHON_BIN="/usr/bin/python"
service_start $PYTHON_BIN /scripts/python/log.py&

start() {
   echo Inicializando
   echo $PATH
}

stop() {
    killall python
}
```

Apêndice 6

Arquivo: snmpd.conf

agentaddress UDP:161

sysLocation office

sysContact bofh@example.com

sysName HeartOfGold

com2sec ro default public

com2sec rw localhost private

group public v1 ro

group public v2c ro

group public usm ro

group private v1 rw

group private v2c rw

group private usm rw

view all included .1

access public "" any noauth exact all none none

access private "" any noauth exact all all all

exec filedescriptors /bin/cat /proc/sys/fs/file-nr

Configuração SNMPD para Multiplos Sensores

exec 1.3.6.1.2.1.1.21 Sensor_Umidade_do_Ar /scripts/sensores/sensores.sh umidade_do_ar

exec 1.3.6.1.2.1.1.22 Sensor_Temperatura_do_Ar /scripts/sensores/sensores.sh temperatura_do_ar

exec 1.3.6.1.2.1.1.23 Sensor_Luminosidade /scripts/sensores/sensores.sh luminosidade

exec 1.3.6.1.2.1.1.24 Sensor_Chuva /scripts/sensores/sensores.sh chuva

exec 1.3.6.1.2.1.1.25 Sensor_Umidade_do_Solo /scripts/sensores/sensores.sh umidade_do_solo

exec 1.3.6.1.2.1.1.26 Sensor_Voltagem_de_Entrada /scripts/sensores/sensores.sh voltagem_de_entrada

exec 1.3.6.1.2.1.1.27 Usuarios_Conectados_no_Wifi /scripts/sensores/sensores.sh usuarios_wifi

Apêndice 7

Arquivo: sensores.sh

```
#!/bin/ash

Arquivo_Coleta=/www/proxyip/logs/logs.csv

U_Ar=`cat $Arquivo_Coleta | tail -n2 | cut -f3 -d' '`

T_Ar=`cat $Arquivo_Coleta | tail -n2 | cut -f4 -d' '`

Lum=`cat $Arquivo_Coleta | tail -n2 | cut -f5 -d' '`

Chuva=`cat $Arquivo_Coleta | tail -n2 | cut -f6 -d' '`

U_Solo=`cat $Arquivo_Coleta | tail -n2 | cut -f7 -d' '`

V_Ent=`cat $Arquivo_Coleta | tail -n2 | cut -f8 -d' '`

U_Wifi=`iw dev wlan0 station dump | awk 'BEGIN {count=0} $1 == "Station"
{count++}; END {print count}'`

case $1 in

   umidade_do_ar )

      echo ${U_Ar}

        ;;

   temperatura_do_ar )

      echo ${T_Ar}

        ;;

   luminosidade )

      echo ${Lum}

        ;;

   chuva )

      echo ${Chuva}

        ;;

   umidade_do_solo )
```

```
        echo ${U_Solo}
          ;;
    voltagem_de_entrada )
        echo ${V_Ent}
          ;;
    usuarios_wifi )
        echo ${U_Wifi}
          ;;
Esac
```

Apêndice 8

Arquivo: *DashBoard*

```
<!doctype html>
<html>
<head>
<title>ProxyIP</title>
<script src="js/gauge.min.js"></script>
</head>
<body>
<center><canvas id="teste"></canvas></center>
<script type="text/javascript">
function Ajax(){
var xmlHttp;
	try{
		xmlHttp=new XMLHttpRequest();// Firefox, Opera 8.0+, Safari
	}
	catch (e){
		try{
			xmlHttp=new ActiveXObject("Msxml2.XMLHTTP"); // Internet
Explorer
		}
		catch (e){
		   try{
				xmlHttp=new ActiveXObject("Microsoft.XMLHTTP");
			}
			catch (e){
```

```
                    alert("No AJAX!?");
                    return false;
                }
            }
        }
xmlHttp.onreadystatechange=function(){
        if(xmlHttp.readyState==4){
            gauge.setValue(xmlHttp.responseText);
                setTimeout('Ajax()',1000);
}
}
xmlHttp.open("GET","S1.php",true); // aqui configuramos o arquivo
xmlHttp.send(null);
}
window.onload=function(){
setTimeout('Ajax()',1000); // aqui o tempo entre uma atualização e outra
}
var gauge = new Gauge({
renderTo : 'teste',
width : 300,
height : 300,
glow : true,
units : '%',
title : 'Umidade do Ar',
```

```
minValue : 0,
maxValue : 100,
majorTicks : ['0','10','20','30','40','50','60','70', '80','90','100'],
minorTicks : 2,
colors : { plate : '#222',
    majorTicks : '#f5f5f5',
    minorTicks : '#ddd',
    title : '#fff',
    units : '#ccc',
    numbers : '#eee',
needle : { start : 'rgba(240, 128, 128, 1)', end : 'rgba(255, 160, 122, .9)' }
} });
gauge.onready = function() {
setInterval( function() {
  }, 3000);
};
gauge.draw();
</script>
</body>
</html>
```