

Compilando e Instalando OpenWRT com Open vSwitch Instituto Federal do Ceará – IFCE

Christiano Machado da Costa¹

¹Ciência da Computação – Instituto Federal do Ceará – Campus Maracanaú
Av. Contorno Norte, 1753 – 61936-000 – Maracanaú – CE – Brasil

christianomachado10@gmail.com

Resumo. Equipamentos de Redes, como por exemplo roteadores e comutadores, possuem um Firmware. Cada modelo de dispositivo possui características específicas que variam de acordo com o hardware, portanto é necessário uma instalação minuciosa. O Software Livre e de Código Aberto nos trouxe diversas vantagens, sendo uma delas a possibilidade de alterá-los de acordo com nossas necessidades. Com todos estes recursos disponíveis, podemos compilar e instalar o nosso próprio Firmware.

Palavras-chave: Equipamentos de Redes, Software Livre, Código Aberto, Firmware, Roteadores, Comutadores.

Abstract. The Network Devices, for example Routers and Switches, have a Firmware. Each model of devices have specific technical features and they vary according to the hardware, therefore a rigorous installation is required. The Free and Open-source Software brought us several advantages, being one of them the possibility of change them according our needs. With all this available resources, we can compile and install our Firmware.

Keywords: Network Devices, Routers, Switches, Firmware, Free Software, Open-source.

1. Introdução

A informação se propaga de diversas maneiras, desde o rádio até os sistemas mais modernos de telecomunicações. O fato é que uma informação pode ser gerada por qualquer pessoa e passada adiante de diversos modos. A forma mais comum de transmitir tal dado é por meio da Internet, contudo, neste processo há diversas etapas que devem ser seguidas. Comumente em um Sistema de Redes surge a figura dos Roteadores. Eles são equipamentos voltados ao roteamento de pacotes de dados da Rede em questão, ou seja, é traçada uma rota no qual aquele dado irá seguir. Estes dispositivos podem ser encontrados também em redes domésticas ou em alguns ambientes empresariais, sendo o roteador um dos transmissores de sinais *WiFi*.

No âmbito de Redes temos alguns padrões que permitem a comunicação entre diversos dispositivos de diferentes fabricantes. O *Open System Interconnection Model*, ou simplesmente Modelo OSI, é utilizado como referência para o melhor entendimento deste processo de comunicação, ele possui sete camadas sendo estas Camada de Aplicação, Apresentação, Sessão, Transporte, Rede, Enlace e Física. Cada uma possui uma função na interconexão entre os dispositivos. A Figura 1 ilustra a organização deste modelo.

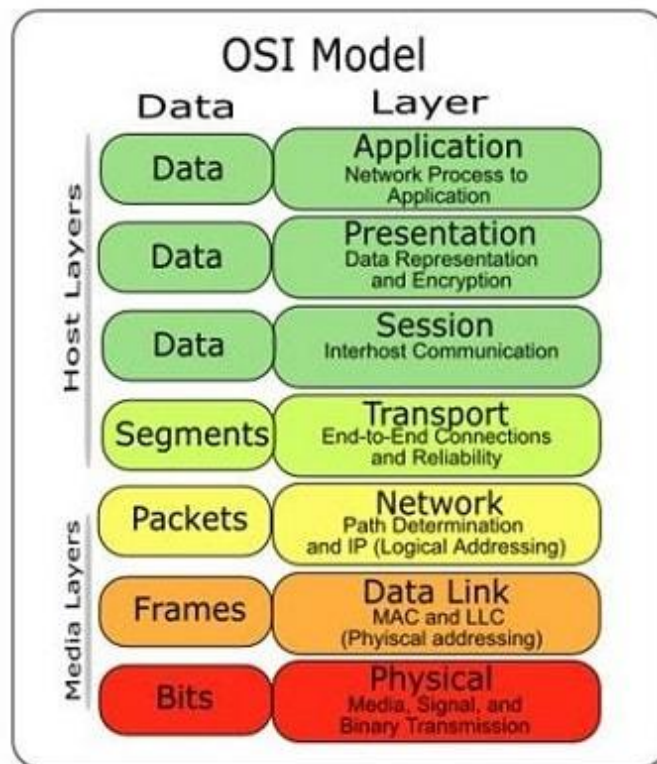


Figura 1. Modelo OSI [Informática 2017]

A Camada de Aplicação (7) permite que o usuário da rede tenha acesso à ela no nível mais alto, ou seja, através do *software*, é nela onde estão localizado a maior parte dos protocolos, como por exemplo o HTTP, SMTP, FTP, DNS, DHCP entre outros. Em seguida temos a Camada de Apresentação (6), que tem como principal função a tradução e a representação dos dados para suas camadas adjacentes. A Camada de Sessão (5) sincroniza e gerencia a comunicação entre os processos de aplicação. A Camada de Transporte (4) é responsável pela comunicação fim a fim entre processos, seus principais protocolos são o TCP e o UDP. A Camada de Rede (3) é a responsável pela determinação do caminho por onde os pacotes irão trafegar, ou seja, ela faz o roteamento dos dados, o protocolo desta camada é o *Internet Protocol* (IP). A Camada de Enlace é responsável principalmente pelo acesso ao meio de comunicação, ela faz com que a camada 3 entenda que não há problemas no meio para a transmissão. A camada 2 trata seus endereços como sendo físicos, diferente da camada 3 que possui endereços lógicos. Por fim, temos a Camada Física é nela que o sinal propriamente dito é transmitido e que o meio é definido, sendo ele cabo, ar, etc.

Roteadores possuem diversas características arquiteturais que variam de acordo com o modelo de uso. Apresentam também uma parte lógica, conhecida como *firmware*, um *software* que possui finalidades específicas para aquele dispositivo. Cada fabricante disponibiliza o *firmware* do seu equipamento para que o mesmo funcione corretamente. Porém existem diversos *firmwares* que fazem as mesmas funções. Um deles é conhecido como OpenWRT, um *software* baseado em Linux, que provê um sistema de arquivos completos com gerenciamento de pacotes de rede. O OpenWRT permite que novas funci-

onalidades, que *firmwares* proprietários não possuem, sejam atribuídas ao equipamento. Sendo muito estável em todas as suas distribuições e não causando nenhum dano ao dispositivo. [OpenWRT 2009].

Com a evolução dos sistemas de comunicação, é normal que novos paradigmas surjam e um deles que está sendo bastante utilizado atualmente é o paradigma de **Redes Definidas por Software** (SDN). Sua principal característica é a separação do plano de dado do plano de controle, respectivamente, os dados da rede e a parte de encaminhamento da informação. Neste caso existem alguns protocolos que trabalham com esta ideia de separação de SDN, sendo o OpenFlow o mais conhecido entre eles [Azodolmolky 2013].

2. Desenvolvimento

Neste capítulo serão tratados a compilação e instalação do *firmware* no dispositivo escolhido. O Sistema Operacional utilizado será Ubuntu 16.04 LTS. Além disso, empregaremos o OpenWRT como *firmware* à ser compilado. No decorrer desta seção entenderemos um pouco das ferramentas utilizadas.

2.1. Preparando Material

Primeiramente devemos escolher qual será o dispositivo para o qual iremos compilar o nosso *firmware* e verificar se o mesmo possui suporte ao OpenWRT, isto pode ser feito na *Table of Hardware* (ToH), em português Tabela de Hardware, disponível em [OpenWRT 2009]. O modelo exemplo será o Roteador TP-LINK TL-WR1043ND v1.11 como mostra a Figura 2.



Figura 2. TP-LINK Modelo TL-WR1043ND v1.11 [LINK 2017]

Feito isto, estudaremos a arquitetura do equipamento afim de preparar melhor e de forma mais eficiente nosso *firmware*, evitando assim falhas graves em nosso dispositivo. No caso do TL-WR1043ND sua *Datasheet* estará disponível no final deste artigo. Caso o OpenWRT dê suporte ao equipamento escolhido, na ToH haverá um *link* que levará à uma página onde estarão algumas informações arquiteturais, como o conjunto de instruções, processador, tamanho da memória flash, etc. É possível também descobrir tais informações através do modelo do equipamento [OpenWRT 2016b]

Devemos agora instalar alguns recursos no nosso Sistema Operacional que serão necessários para fazer a compilação do nosso OpenWRT. Isto pode ser feito executando os seguintes comandos:

Listing 1. Instalando Pacotes para Compilação

```
$ apt-get update
$ apt-get install subversion libssl-dev zlibc zlib1g zlib1g-dev gzip
gawk libncurses5-dev git build-essential
```

Em seguida devemos criar um diretório que receberá o repositório com o material fundamental à compilação, faremos isto utilizando o comando a seguir :

Listing 2. Criando Diretório para os Arquivos

```
$ mkdir openwrt
$ cd openwrt
```

Por fim, faremos o *download* do repositório para darmos início a compilação.

Listing 3. Clonando Repositório para Compilar o OpenWRT

```
/openwrt$ git clone https://git.openwrt.org/15.05/openwrt.git trunk
```

Após realizar os passos acima, seguiremos para o próximo assunto.

2.2. Open vSwitch

Antes de iniciarmos a compilação é importante falar um pouco sobre o Open vSwitch, uma das mais importantes ferramentas de virtualização, contudo é necessário entendermos um pouco sobre virtualização.

A virtualização vem sendo cada vez mais utilizada para várias finalidades, pois ela trás diversas vantagens, elevando ao máximo o uso do *hardware*. Suponhamos um conjunto de servidores Linux que comportam alguns aplicativos necessários para um bom funcionamento de uma determinada aplicação, Figura 3. Para que haja comunicação entre os servidores, cada um possui uma *Network Interface Card* (NIC), ou em português Placa da Interface de Rede, que está conectada a uma rede externa.

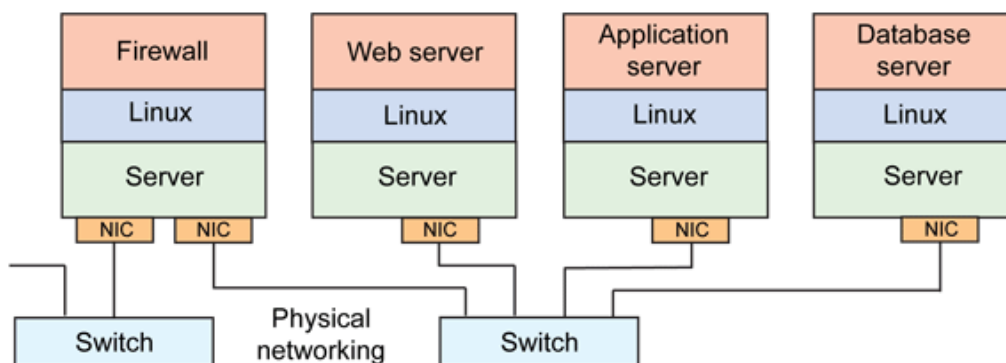


Figura 3. Infraestrutura de Rede Tradicional [Jones 2010]

A inovação por trás da virtualização em servidores é a abstração do *hardware*, permitindo assim que diversos sistemas e/ou aplicativos compartilhem o mesmo *hardware*. O

hypervisor, ou Monitor de Máquina Virtual, realiza esta função. Cada conjunto de sistema ou aplicativo enxerga o *hardware* subjacente como sendo uma máquina completa e não compartilhada, mesmo que tais partes virtualizadas não existam ou sejam compartilhadas por várias *Virtual Machines* (VM), ou Máquinas Virtuais em português. O *hypervisor* pode criar diversas VMs em uma mesmo equipamento físico e cada VM pode ter, por exemplo, um vNIC. Elas podem ser parecidas com NICs físicas reais para as VMs, porém elas representam apenas uma NIC real. A Figura 4 ilustra um pouco esta situação.

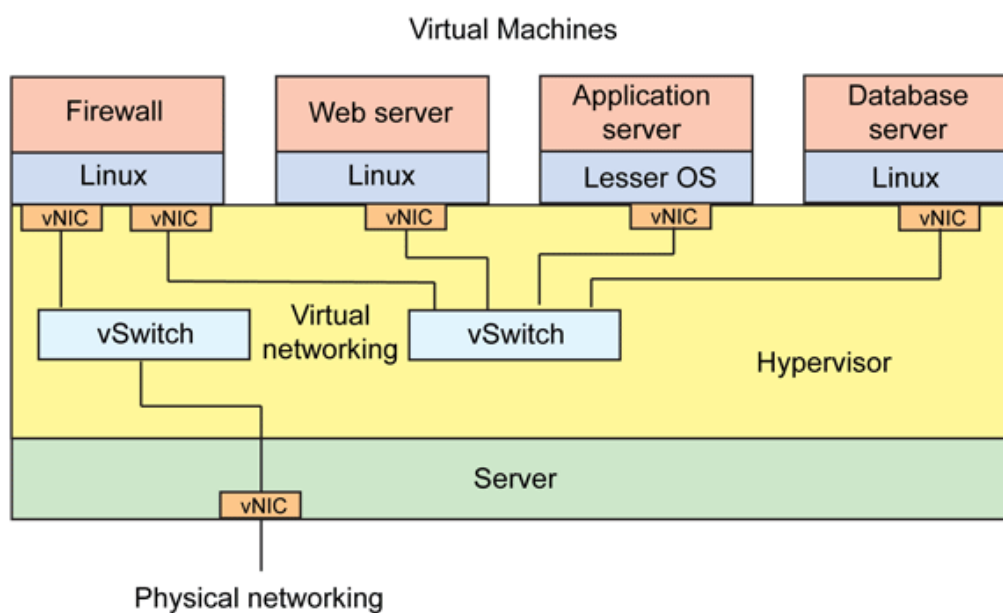


Figura 4. Infraestrutura de Rede Virtualizada [Jones 2010]

O Open vSwitch é um comutador virtual com diversas camadas, está disponível na forma de *software* livre sobre licença do Apache 2.0. Sendo uma das principais ferramentas com recursos e soluções para diversos *hypervisors*, como VirtualBox, Xen, XenServer, etc. O Open vSwitch consiste em um daemon de comutador e um módulo do *Kernel* concomitante que gerencia a comutação baseada no fluxo. Além de tudo isso, é possível executar o Open vSwitch inteiramente em espaço de usuário, contudo poderá haver uma redução no desempenho. [Jones 2010]

Esta ferramenta nos proporciona diversas vantagens, dentre elas a criação de múltiplos *switches* virtuais, permitindo assim que os mesmos possam trazer uma automação na rede através de programação. Além disso, protocolos como o OpenFlow podem ser implementados. O OpenFlow é um protocolo feito para arquiteturas de redes do tipo SDN, onde há separação entre os planos de controle e de dado. Em cada elemento encaminhador há uma tabela de fluxo que é atualizada no plano de controle. Ao chegar um pacote de um fluxo já estabelecido na tabela, ele é comutado pelo *switch* OpenFlow. Já quando um fluxo não está estabelecido na tabela, há um controlador responsável por tratar o mesmo. A Figura 5 mostra, de forma reduzida, a tabela presente em cada *switch* OpenFlow. [Azodolmolky 2013]

Port	Src MAC	Dst MAC	Ether Type	VLAN ID	VLAN ID priority	Src IP	Dst IP	IP Proto	IP TOS bits	Src TCP/UDP port	Dst TCP/UDP port
------	---------	---------	------------	---------	------------------	--------	--------	----------	-------------	------------------	------------------

Figura 5. Tabela de Fluxo Reduzida [Azodolmolky 2013]

2.3. Compilando OpenWRT

Agora podemos dar início ao processo de compilação. Como visto na Subseção 2.1 entraremos na pasta e em seguida movemos o arquivo **feeds.conf.default** para outro arquivo **feeds.conf**, este processo é necessário para ajustar os repositórios de onde serão baixados os arquivos para a compilação. Para fazer isto basta seguir os comandos:

Listing 4. Movendo Arquivo feeds.conf

```
$ cd /openwrt/trunk /
/openwrt/trunk$ mv feeds.conf.default feeds.conf
```

Nós podemos adicionar pacotes específicos, vamos adicionar o pacote LuCI que será a nossa interface web e o Open vSwitch para adicionarmos o módulo do IPSec que trará segurança fim a fim no tráfego da rede a nível da camada IP. Então, execute os comandos abaixo:

Listing 5. Instalando LuCI e Open vSwitch

```
/openwrt/trunk$ ./scripts/feeds update -a
/openwrt/trunk$ ./scripts/feeds install -a -p luci
/openwrt/trunk$ ./scripts/feeds install -a -p openvswitch
/openwrt/trunk$ ./scripts/feeds install -a
```

Após terminar este processo, montaremos o **menuconfig** que nos trará uma interface onde poderemos selecionar todas as bibliotecas necessárias para o funcionamento do *firmware* em nosso equipamento. Execute o comando abaixo e será mostrado uma interface como na Figura 6.

Listing 6. Interface menuconfig

```
/openwrt/trunk$ sudo make menuconfig
```

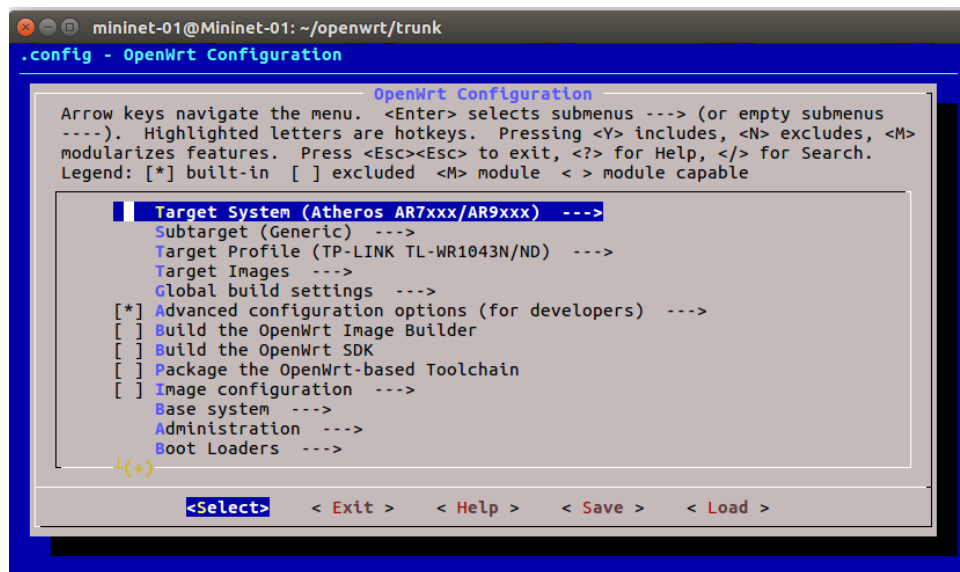


Figura 6. Interface MenuConfig

Ao abrirmos a tela devemos selecionar em **Target System** a plataforma do roteador para o qual o *firmware* será destinado, na ToH do próprio site do OpenWRT possui tal informação, neste caso é **Atheros AR9132**, portanto escolhemos a opção **Atheros AR7xxx/AR9xxx**. Em seguida, na opção **Subtarget** escolhemos a opção **Generic**. O próximo passo é selecionar o **Target Profile** e escolher o modelo do roteador, TP-LINK TL-WR1043ND. Estas instruções são necessárias para configurar o arquivo que será compilado, informando assim a plataforma, esquema arquitetural e o modelo do roteador para o qual vamos compilar o OpenWRT.

Agora, escolheremos o sistema de arquivo do nosso OpenWRT. Em **Target Images** marcamos a opção **squashfs**. Este sistema de arquivos permite que nós possamos "flashar" o *firmware* mantendo os arquivos de configuração. Veja a Figura 7

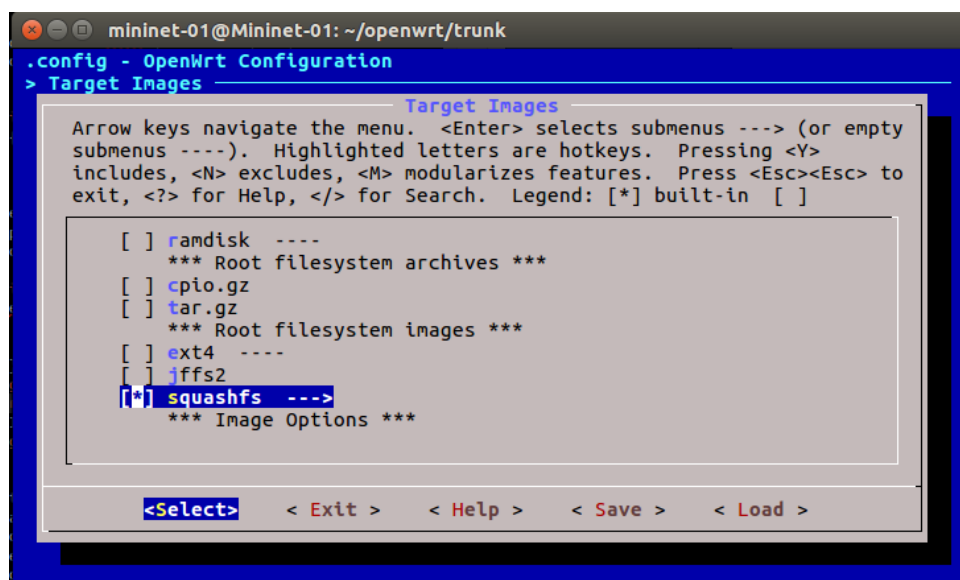


Figura 7. Target Images : squashfs

Salve estas configurações no arquivo **.config**. No terminal digite os comandos abaixo para criar um arquivo padrão de configuração para arquitetura escolhida e em seguida checar os pré-requisitos para compilação.

Listing 7. Configuração Padrão e Pré-requisitos

```
/openwrt/trunk$ sudo make defconfig  
/openwrt/trunk$ sudo make prereq
```

A *cross-compilation* é uma técnica que permite a compilação de um código para um sistema alvo, como por exemplo sistemas embarcados ou microcontroladores. Esta é uma técnica muito eficiente quando tratamos de sistemas muito pequenos, onde um compilador em sua memória interna seria desperdício de espaço. Portanto, estamos aptos a compilar o OpenWRT, a partir do Linux, para que o mesmo funcione no roteador. Porém, é necessário adicionarmos mais algumas bibliotecas para que o nosso sistema atenda de forma eficaz às necessidades do equipamento. Então executemos novamente o comando contido no Listing 6. Vamos adicionar os recursos que serão compilados juntos com o sistema, como por exemplo um editor de texto, para facilitar o acesso uma interface web, dentre outros recursos. A Figura 8 mostra a adição da interface web no sistema. Para fazer isto, basta ir em **LuCI – Collections – luci**.

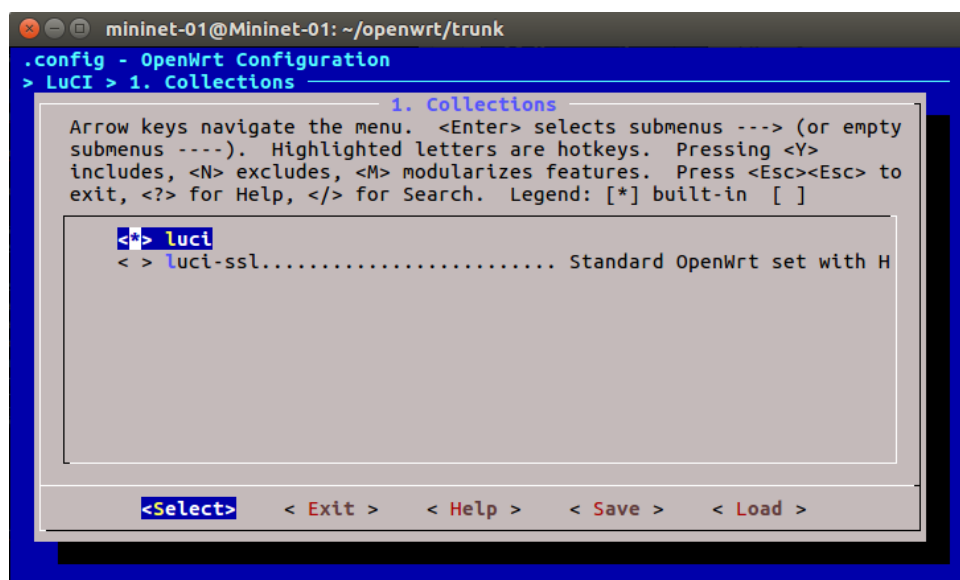


Figura 8. Adicionando Interface Web LuCI

De forma análoga iremos adicionar um editor de texto. Veja a Figura 9, devemos ir em **Utilities – Editors – nano**.

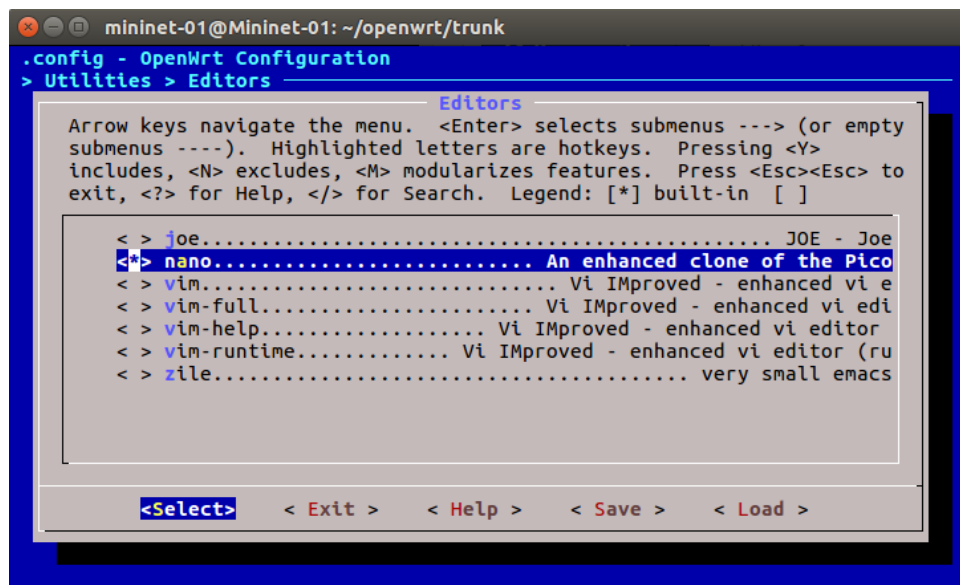


Figura 9. Adicionando Editor de Texto

Devemos fazer com que nosso sistema dê suporte a chamadas de sistema em C/C++, pois algumas bibliotecas que futuramente adicionarmos podem realizar tais ações. Portanto, adicionemos também ao nosso sistema a biblioteca **libstdcpp**. Vá em **Base System** – **libstdcpp** como mostra a Figura 10.

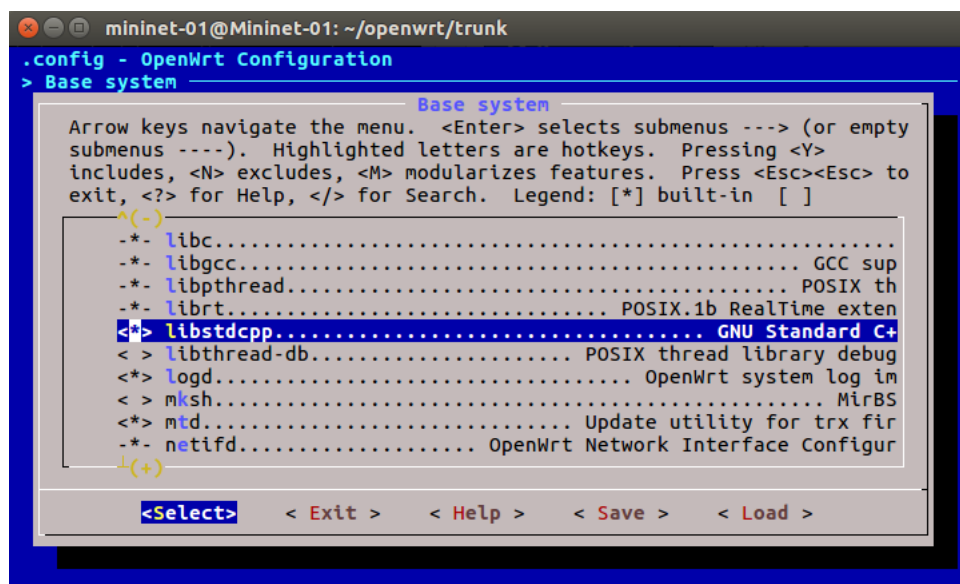


Figura 10. Adicionando Biblioteca de Suporte a Chamadas em C/C++

Este passo é mais específico do equipamento TP-LINK TL-WR1043ND, pois aqui iremos adicionar o suporte a *driver* USB. Para fazer isto, basta ir em **Kernel Modules** – **USB Support** – **kmod-usb-storage** e em seguida **Kernel Modules** – **filesystem** – **kmod-fs-ext4** / **kmod-fs-msdos** / **kmod-fs-vfat**, estes são os módulos núcleos (kmod) que dão suporte aos sistemas de arquivos da partição USB. [OpenWRT 2016c]. O próximo passo

é adicionar o *kmod-nls-cp437* que é um módulo do núcleo para *Native Language Support* (NLS) *Codepage 437*, **Kernel Modules – Native Language Support – kmod-nls-cp437**. [OpenWRT 2016a]. A Figura 11 mostra a adição do **Kernel Modules – USB Support – kmod-usb-storage**.

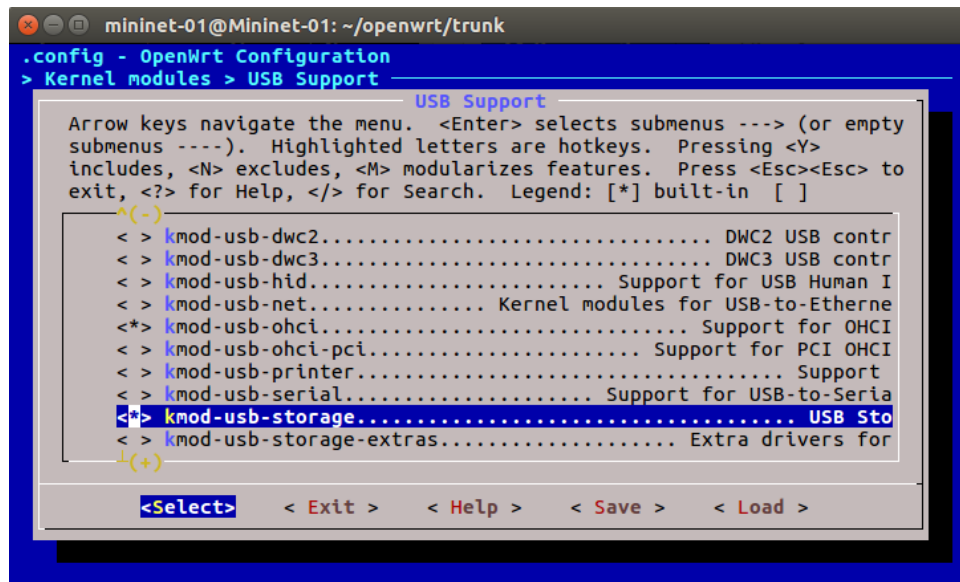


Figura 11. Adicionando Suporte a Dispositivos de Armazenamento USB

O **hostapd** é um AP IEEE802.11 e um autenticador IEEE 802.1XWPAWPA2EAPRADIUS e fará o papel de transmissão do sinal no nosso sistema. [Andrews 2017]. Para adicioná-lo basta ir em **Network – hostapd-utils** como mostra a Figura 12.

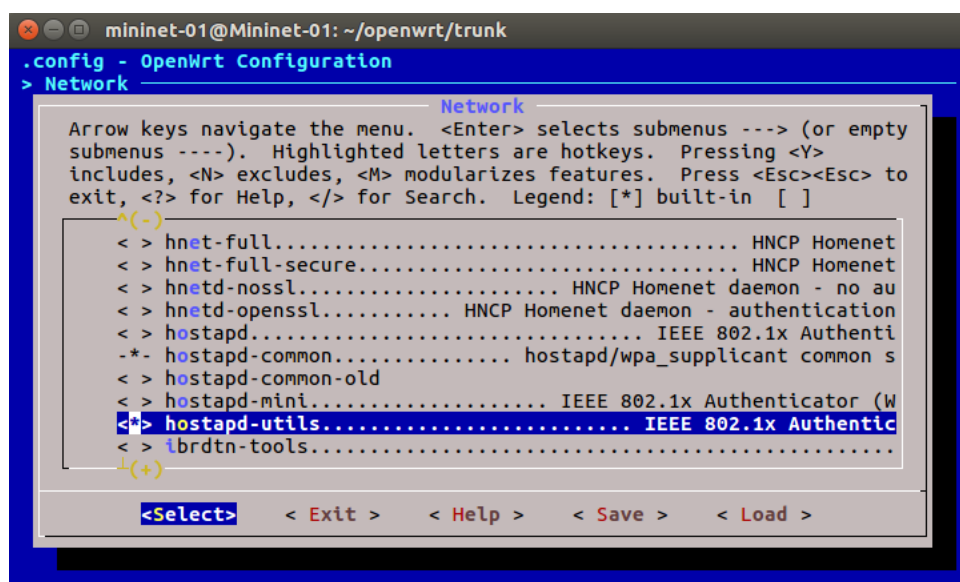


Figura 12. Adicionando hostapd

Como também estamos trabalhando com virtualização alguns recursos como o *TUN/TAP* devem ser adicionados. O *TAP* é um *driver* do kernel de rede virtual que

implementa um dispositivo Ethernet, operando então na Camada de Enlace. O *TUN* é um "túnel" que simula um dispositivo da Camada de Rede e se comunica com o nível mais alto dos pacotes IP. [Jones 2010]. Desta forma iremos adicionar este recurso, **Kernel modules – Network Support – kmod-tun**. Veja a Figura 13.

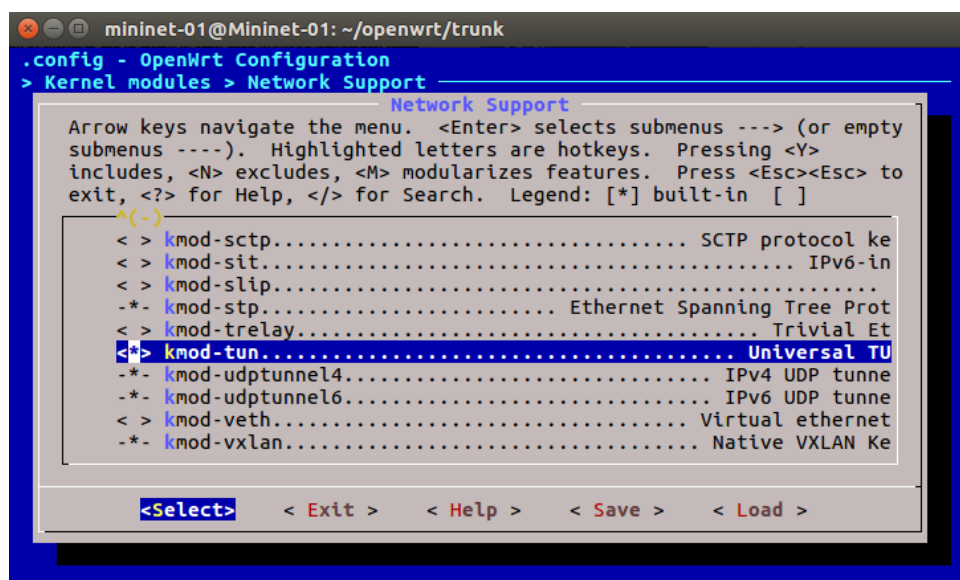


Figura 13. Adicionando TUN/TAP

Esta parte é opcional, mas pode ser necessário caso seu dispositivo não possua uma interface secundária de *Wireless*. Então adicionaremos o *dummy network interface* e *ip link*. Basicamente a *dummy network interface* resolve problemas de *Loopback*. O *ip link* é uma ferramenta similar ao *ifconfig*. A Figura 14 mostra adição do *dummy*, para fazer isto, basta ir em **Kernel modules – Network Devices – kmod-dummy**. Para adicionar o *iplink* basta ir em **Base System – busybox – Network Utilities – ip – ip link** como mostra a Figura 15.

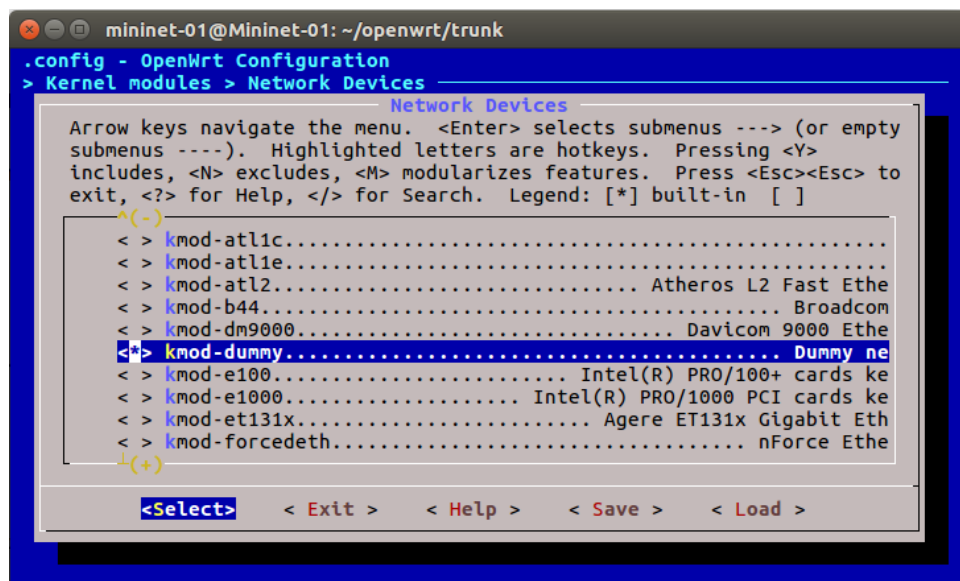


Figura 14. Adicionando Dummy Network Interface

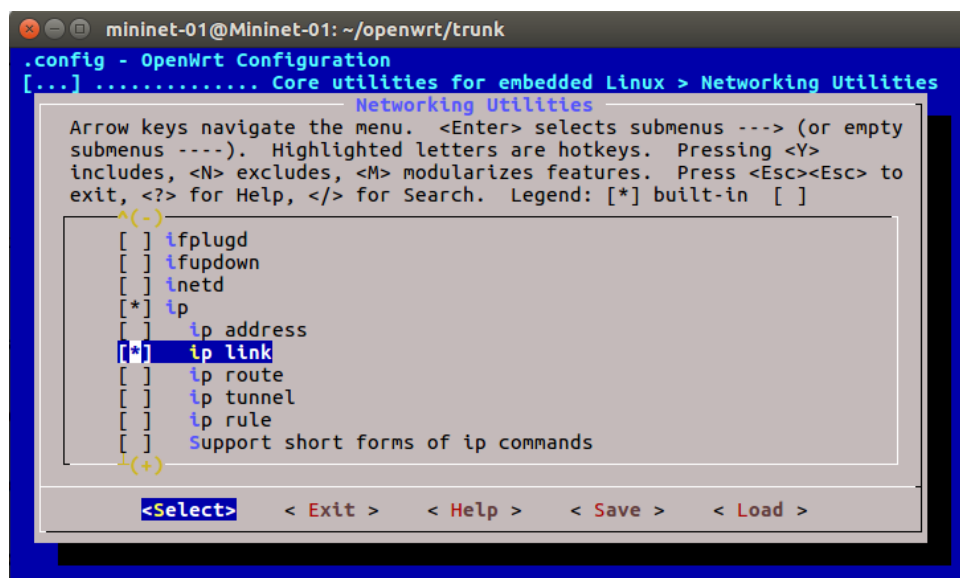


Figura 15. Adicionando iplink

O próximo passo é adicionarmos um *sniffer* ao nosso sistema. Ele nos permite analisar a rede, os pacotes e os protocolos da comunicação. Vá em **Network – tcpdump** como mostra a Figura 16

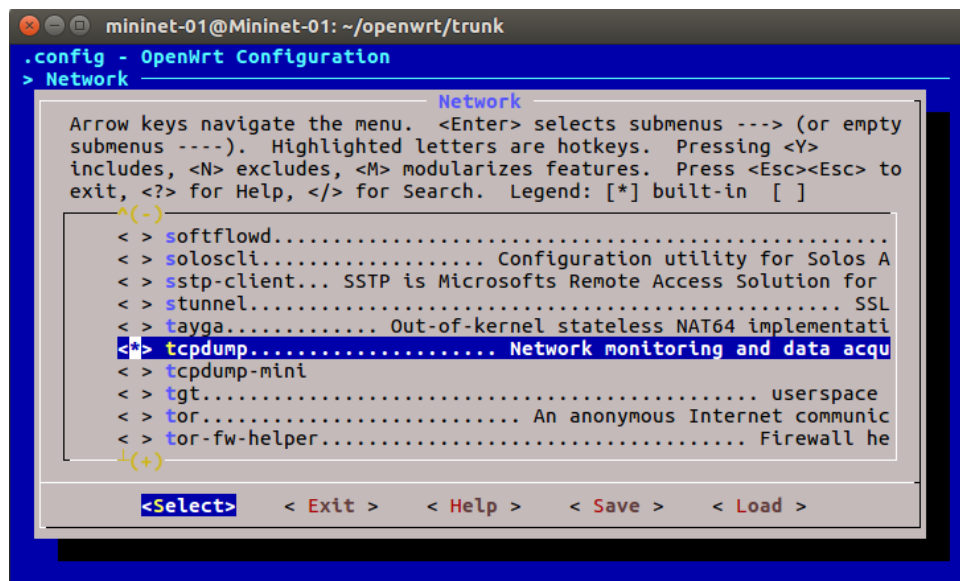


Figura 16. Adicionando tcpdump

Como falado na Seção 2.2 o Open vSwitch é uma ferramenta bastante importante no âmbito de SDN, agora adicionaremos ele ao nosso OpenWRT. Como havíamos adicionado sua biblioteca, podemos usar o **menuconfig** para juntá-lo a compilação. Em **Network – openvswitch – openvswitch-ipsec** como na Figura 17.

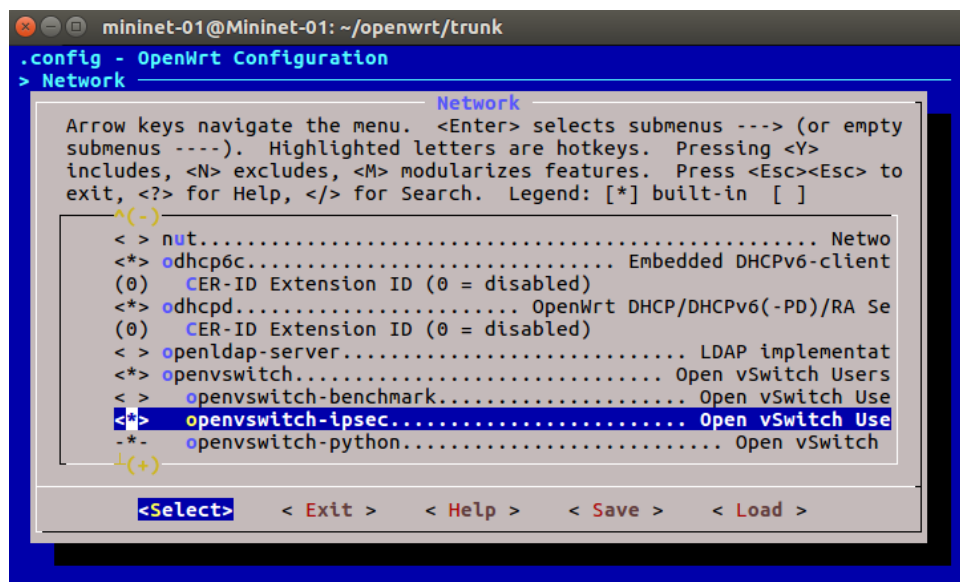


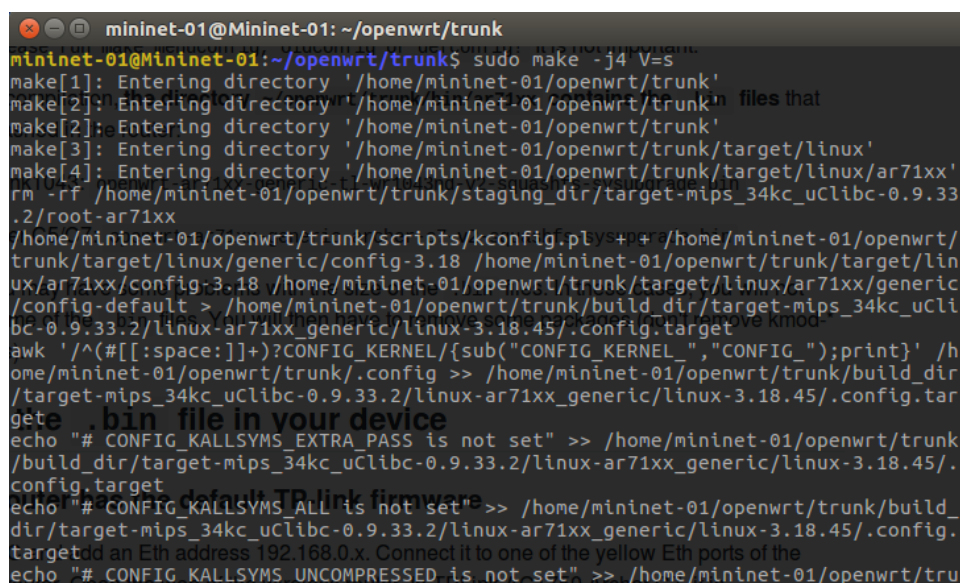
Figura 17. Adicionando Módulo Open vSwitch IPsec

Agora o *firmware* está pronto para ser compilado. Salve e saia do **menuconfig**. Então digite os comandos do Listing 8. Na linha 2, substitua o **X** por um número de *threads* para acelerar o processo, normalmente este número varia de acordo com a capacidade de processamento da CPU do Computador usado para o processo. Ainda na mesma linha o comando **V=s** é utilizado para mostrar informações detalhadas da compilação.

Listing 8. Iniciando Processo de Compilação

```
/openwrt/trunk$ sudo make prereq  
/openwrt/trunk$ sudo make -jX V=s
```

A Figura 18 mostra um trecho do terminal no momento da compilação. Após isto, vá na pasta **openwrt/trunk/bin/ar71xx**, lá estará o arquivo compilado, provavelmente seu nome será algo parecido com **openwrt-ar71xx-generic-tl-wr1043nd-v2-squashfs-sysupgrade.bin**, mas isso pode variar de acordo com suas configurações.



```
mininet-01@Mininet-01: ~/openwrt/trunk  
mininet-01@Mininet-01:~/openwrt/trunk$ sudo make -j4 V=s  
make[1]: Entering directory '/home/mininet-01/openwrt/trunk'  
make[2]: Entering directory '/home/mininet-01/openwrt/trunk'  
make[3]: Entering directory '/home/mininet-01/openwrt/trunk/target/linux'  
make[4]: Entering directory '/home/mininet-01/openwrt/trunk/target/linux/ar71xx'  
rm -rf /home/mininet-01/openwrt/trunk/staging_dir/target-mips_34kc_uClibc-0.9.33  
.2/root-ar71xx  
/home/mininet-01/openwrt/trunk/scripts/kconfig.pl --sync -- /home/mininet-01/openwrt/  
trunk/target/linux/generic/config-3.18 /home/mininet-01/openwrt/trunk/target/lin  
ux/ar71xx/config-3.18 /home/mininet-01/openwrt/trunk/target/linux/ar71xx/generic  
/config-default > /home/mininet-01/openwrt/trunk/build_dir/target-mips_34kc_uCli  
bc-0.9.33.2/linux-ar71xx_generic/linux-3.18.45/.config.target  
awk '/^([[:space:]]+)?CONFIG_KERNEL/{sub("CONFIG_KERNEL_", "CONFIG_");print}' /h  
ome/mininet-01/openwrt/trunk/.config >> /home/mininet-01/openwrt/trunk/build_dir  
/target-mips_34kc_uClibc-0.9.33.2/linux-ar71xx_generic/linux-3.18.45/.config.tar  
get  
echo "# CONFIG_KALLSYMS_EXTRA_PASS is not set" >> /home/mininet-01/openwrt/trunk  
/build_dir/target-mips_34kc_uClibc-0.9.33.2/linux-ar71xx_generic/linux-3.18.45/.  
config.target  
echo "# CONFIG_KALLSYMS_ALL is not set" >> /home/mininet-01/openwrt/trunk/build_  
dir/target-mips_34kc_uClibc-0.9.33.2/linux-ar71xx_generic/linux-3.18.45/.config.  
target  
echo "# CONFIG_KALLSYMS_UNCOMPRESSED is not set" >> /home/mininet-01/openwrt/tru
```

Figura 18. Trecho do Processo de Compilação

2.4. Instalando OpenWRT

Nesta parte final, devemos simplesmente instalar o OpenWRT no roteador escolhido. Provavelmente o roteador estará em alguma subrede padrão, como por exemplo *192.168.1.1* ou *192.168.0.1*. Utilize o comando disposto no Listing 9, o resultado será algo parecido com a Figura 19.

Listing 9. Descobrindo IP do Roteador

```
/openwrt/trunk$ sudo make prereq  
/openwrt/trunk$ sudo make -jX V=s
```



```
mininet-01@Mininet-01: ~  
mininet-01@Mininet-01:~$ ip addr show  
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default  
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00  
    inet 127.0.0.1/8 scope host lo  
        valid_lft forever preferred_lft forever  
    inet6 ::1/128 scope host  
        valid_lft forever preferred_lft forever  
2: enp2s0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP  
    group default qlen 1000  
    link/ether 90:2b:34:f9:af:98 brd ff:ff:ff:ff:ff:ff  
    inet 192.168.1.100/24 brd 192.168.1.255 scope global dynamic enp2s0  
        valid_lft 7179sec preferred_lft 7179sec  
    inet6 fe80::a9ca:d686:c188:82e1/64 scope link  
        valid_lft forever preferred_lft forever  
3: wlp3s0: <BROADCAST,MULTICAST> mtu 1500 qdisc noop state DOWN group default ql  
    en 1000  
    link/ether a0:f3:c1:b8:0a:62 brd ff:ff:ff:ff:ff:ff  
4: enp0s29u1u1c4i2: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fas  
    t state DOWN group default qlen 1000  
    link/ether 6a:fb:7e:39:6f:68 brd ff:ff:ff:ff:ff:ff  
mininet-01@Mininet-01:~$
```

Figura 19. Comando ip addr show no Terminal

Feito isso, acesse o roteador digitando seu endereço IP no navegador. Ao conectar-se com o mesmo, será necessário uma senha para acessá-lo, comumente a maioria dos roteadores possui **Usuário: admin** e **Senha: admin**, caso contrário olhar no manual do fabricante. Ao acessar a interface web, vá em *System Tools – Firmware Upgrade* e basta selecionar o arquivo resultante da compilação. A Figura 20 mostra um exemplo deste passo.

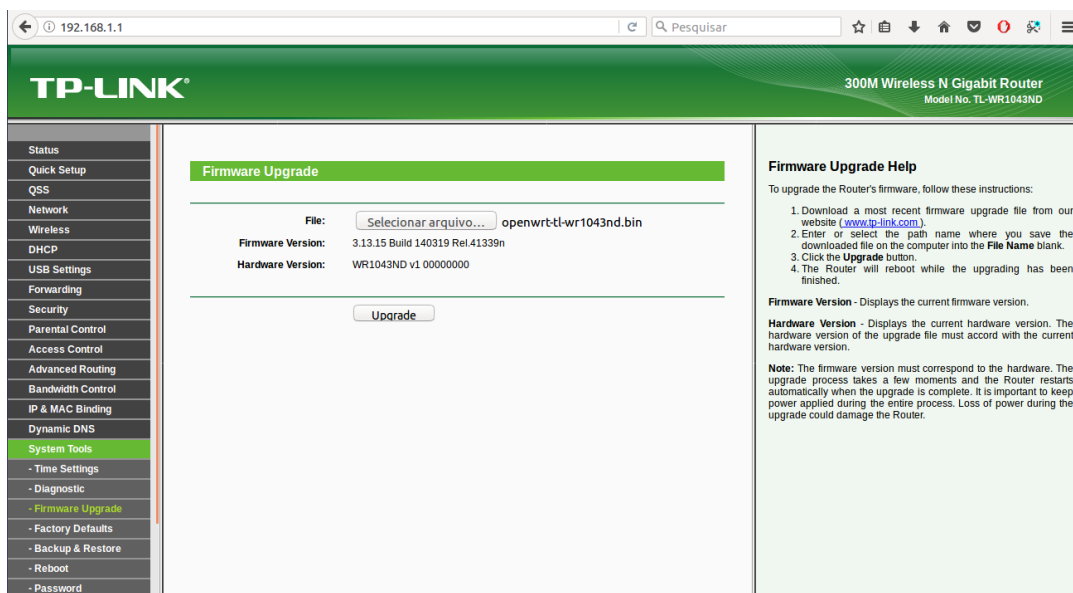


Figura 20. Instalando o novo Firmware

3. Conclusão

Ao utilizar o OpenWRT um leque de opções se abre, pois com ele podemos personalizar o *firmware* do nosso equipamento da maneira que nos for mais adequada. Esta carac-

terística nos traz uma vantagem gigantesca, como por exemplo a adição de um módulo que aumenta a segurança do pacote transmitido. Podemos perceber que o paradigma de SDN está se expandindo cada vez mais e por sua vez este é um conceito muito poderoso.

Diversos outros módulos do Open vSwitch podem ser adicionados ao *firmware*, trazendo assim diversas características positivas extras ao sistema. Porém, isto tem um custo, quanto mais recursos são adicionados, maior o espaço do arquivo final, e como os equipamentos possuem memórias limitadas isso deve ser analisado com cautela na hora da compilação. Além da limitação de armazenamento dos dispositivos há limitações de *hardware* também, como por exemplo, não é necessário adicionar um recurso de armazenamento USB se o próprio equipamento não possui suporte. Recursos adicionados sem utilidade podem causar um grande problema na hora da instalação, podendo resultar até na perda do equipamento. Portanto é necessário tomar cuidado ao compilar e instalar um *firmware*.

Referências

Andrews, T. (2017). hostapd. Documentação do hostapd <https://wireless.wiki.kernel.org/en/users/documentation/hostapd>.

Azodolmolky, S. (2013). *Software Defined Networking with OpenFlow*. Packt Publishing Ltd.

Informática, F. (2017). Modelo osi. Imagem retirada do site <https://faqinformatica.com/modelo-osi-7-camadas-funcoes/>.

Jones, M. (2010). Rede virtual no linux. Computadores virtuais, Open vSwitch e mais algumas tecnologias disponíveis em <https://www.ibm.com/developerworks/br/library/l-virtual-networking/index.html>.

LINK, T. (2017). Tp-link. Informações sobre TL-WR1043ND pode ser encontrada em http://www.tp-link.com.br/download/TL-WR1043ND_V1.html.

OpenWRT (2009). Openwrt. Informações sobre OpenWRT disponíveis em <https://openwrt.org/>.

OpenWRT (2016a). Nls codepage. NLS Codepage 437 e outros tipos de codificação em <https://wiki.openwrt.org/doc/howto/storage>.

OpenWRT (2016b). Table of hardware. Informações arquiteturais através do modelo do equipamento disponíveis em https://wiki.openwrt.org/toh/views/toh_dev_platforms.

OpenWRT (2016c). Usb storage. USB Storage no OpenWRT. <https://wiki.openwrt.org/doc/howto/usb.storage>.