Guida Completa Ninux al Routing a Terra

Prefazione

L'obiettivo di questo HowTo è di guidare passo-passo un nuovo ingresso in Ninux nella configurazione del proprio nodo utilizzando il routing a terra, ma è da ritenersi adatta anche per chi voglia aggiornare il proprio nodo al routing a terra.
Questa guida è scritta avendo principalmente in mente i nuovi ingressi in Ninux che non hanno grandi conoscenze in fatto di networking e come tale presuppone il minimo indispensabile: chi sa già come configurare una tipica rete locale (router domestici e interfacce di rete dei propri PC) non dovrebbe avere problemi a seguire questo documento (se ne ha è un bug, segnalatecelo!). I più esperti possono tranquillamente saltare alle parti per loro rilevanti.
Ogni passo necessario è diviso in sottosezioni in base al sistema operativo/ambiente da configurare. Questa guida non tratta di come procedere all'installazione/flash di questi sistemi operativi sui relativi device, ma dove possibile rimanda ad altra documentazione presente nel wiki Ninux o sulla rete.
Nota: Per aprire le immagini di questa guida in alta risoluzione, è sufficiente cliccare col tasto centrale.

Introduzione

Un nodo Ninux, sia esso un nodo foglia o un supernodo, può essere composto da dispositivi hardware (antenne, router, switch, etc) e componenti software (routing, tunnel, firewall, etc) in numero variabile, e da scegliersi tra tante possibilità disponibili. Stando così le cose non è difficile che le funzionalità di rete di un nodo, e specialmente la più importante e cioè il routing del traffico verso gli altri nodi, risultino "sparpagliate" in più di un dispositivo.

Il *routing a terra* è una tipologia di configurazione per un nodo in cui tutta l'attività di inoltro del traffico ad opera del demone di routing (sia esso OLSR, batman o altro), ed in generale il maggior numero di componenti software necessarie, risultano concentrate in un unico dispositivo diverso dall'antenna, detto appunto *router a terra* o *ground router* (GR).

Per capire i vantaggi di questa configurazione facciamo prima una breve panoramica delle altre modalità di configurazione finora più utilizzate nella rete Ninux.

Fino al 2012 la comunità Ninux produceva una versione modificata del sistema operativo AirOS (il firmware di fabbrica delle Ubiquiti, le antenne di gran lunga più utilizzate nella nostra rete) soprannominata *Sburratone*, che integrava il demone di routing OLSR all'interno di AirOS. Al netto di qualche problemino dovuto alla scarsa integrazione tra OLSR e l'interfaccia di AirOS, questo firmware funzionava bene e consentiva di avere un unico pacchetto "pronto". Questa mod era possibile perché la Ubiquiti metteva a disposizione un SDK con cui gli sviluppatori Ninux potevano creare eseguibili di terze parti compatibili con AirOS. Sfortunatamente, a partire dalla versione successiva alla 5.5.2, la Ubiquiti non solo ha smesso di rilasciare l'SDK, ma ha anche revocato le licenze di quelli già rilasciati.

Trovandosi impossibilitati ad aggiornare le vecchie antenne a nuove versioni del sistema base (e quindi patchare eventuali bug o problemi di sicurezza!), e ad installare il vecchio Sburratone sulle revisioni più recenti dell'hardware Ubiquiti, si è deciso di creare un nuovo firmware, questa volta basato sulla

distribuzione embedded open source denominata OpenWRT. Questa nuova mod, basata attualmente sulla versione 12.09 Attitude Adjustment, è stata ribattezzata *Scooreggione*. Essendo completamente FOSS (Free Open Source Software) il nuovo sistema mette al riparo da chiusure improvvise stile-Ubiquiti e consente personalizzazioni molto più estese e più semplici. Tuttavia, questa configurazione comporta delle restrizioni e delle problematiche, sia concrete che potenziali (sulle quali torneremo a breve) che possono essere interamente evitate col routing a terra.

Principi di funzionamento del routing a terra

Come detto, il principio base del routing a terra è il trasferimento di tutta "l'intelligenza di rete" in un unico dispositivo che non sia l'antenna. Un altro modo di ribadire questo concetto è nel senso complementare: il routing a terra rende le antenne "stupide". La loro unica funzione diventa quella di link wireless. Tutto il traffico che l'antenna riceve dagli altri nodi via etere verrà ributtato "senza pensarci" sul cavo verso il router a terra; allo stesso modo, tutto il traffico che l'antenna riceve via cavo dal router a terra verrà spedito nell'etere "senza pensarci". L'antenna, cioè, non prenderà nessuna decisione di nessun tipo sulla destinazione del traffico.
Il router a terra invece verrà configurato con due criteri fondamentali: sarà in grado di distinguere il traffico in arrivo tra un'antenna e l'altra (se è il caso di supernodo con antenne multiple, ovviamente; altrimenti in un nodo foglia distinguerà tra antenna e altri dispositivi eventualmente collegati), e farà girare un'unica istanza del demone di routing che gestirà l'instradamento di tutto questo traffico da/verso gli altri nodi Ninux.

I due "standard di rete" che consentono di attuare questi meccanismi fondamentali per il routing a terra sono i **bridge** e le **VLAN** (Virtual LAN). Per comprendere il routing a terra in ogni sua sfaccettatura è necessaria uno studio approfondito del bridging e delle VLAN, che tuttavia sono concetti che richiedono delle discrete basi di networking e sono comunque troppo vasti da trattare in questa sede. Pertanto per adesso ci limiteremo a darne una descrizione sommaria ma adatta a portare a termine questo HowTo. Per approfondimenti, potete consultare i link consigliati alla fine della guida.

Bridge

Un bridge è un'interfaccia logica (ovvero che non ha una corrispondente "porta" o "interfaccia" fisica) che raggruppa due o più interfacce di rete e che poi si occupa di inoltrare il traffico in ingresso verso quella che rappresenta la corretta destinazione. È del tutto simile a quello che fa uno switch Ethernet, che è un unico blocco che mette in comunicazione più porte ethernet fisiche come se fossero collegate tutte direttamente tra di loro. La differenza è che un bridge è un'interfaccia logica, e pertanto può fare lo stesso lavoro di mettere in comunicazione "tutti con tutti" anche quando ha a che fare con "mezzi fisici" completamente diversi, come porte ethernet, segnali wireless, cavi in fibra, ecc.
Sarà proprio il bridging tra wireless e cavo che imposteremo sull'antenna a replicare "senza pensarci" il traffico da e verso il router a terra. Di fatto, è come se l'antenna *non esistesse neppure* e il nostro router di terra fosse connesso direttamente al nodo al quale ci collegheremo (nodo remoto) con un luuuuuuuuunghissimo cavo virtuale.

[TODO]*inserimento di un box con una spiegazione più tecnica*[/TODO]

VLAN

Se avete mai configurato una semplicissima rete domestica collegando un vostro PC allo switch incluso nel vosto router ADSL, probabilmente lo avete dato per scontato: 1 interfaccia = 1 cavo = 1 porta sullo switch = 1 indirizzo = 1 rete. E nel caso volessimo collegare il nostro PC ad una seconda rete, con altri indirizzi, servirebbero un secondo switch, un secondo cavo e una seconda interfaccia sul PC. I conti tornano anche con quanto detto sopra: uno switch è un unico blocco che agglomera più porte tra loro e le rende parte di un'unica LAN (rete locale).
Tuttavia, alcuni switch mettono a disposizione una funzionalità avanzata chiamata Virtual LAN (VLAN). Le VLAN consentono di partizionare uno switch in tanti "switch virtuali", da un minimo di 16 ad un massimo di 4096, **completamente separati tra loro**. Questi "switch virtuali" (VLAN) saranno composti da un numero di porte variabile da 1 al numero di porte totali dello switch, in (quasi) qualsiasi combinazione ci occorra. Potremmo parlare di VLAN all'infinito, ma le proprietà delle stesse da esporre per l'utilizzo nel routing a terra sono tre:

- Una determinata porta dello switch può far parte, contemporaneamente, di più di una VLAN.
- Questo significa che, sullo stesso cavo collegato a quella porta, potranno viaggiare contemporaneamente flussi di dati appartenenti a reti diverse.
- Possiamo creare VLAN che includono 1 sola porta e dunque creare VLAN dedicate a ognuna delle nostre antenne inserite in una porta del nostro router.

Quando abbiamo detto che le VLAN sono completamente separate tra loro non intendevamo certo che esse non possono comunicare. Tuttavia, ed è fondamentale, ogni comunicazione tra due VLAN non sarà più il frutto della logica dello switch (come detto, sono switch "virtuali" diversi) ma sarà opera di una decisione di routing. Quello che volevamo quando parlavamo di router a terra capace di "distinguere il traffico".

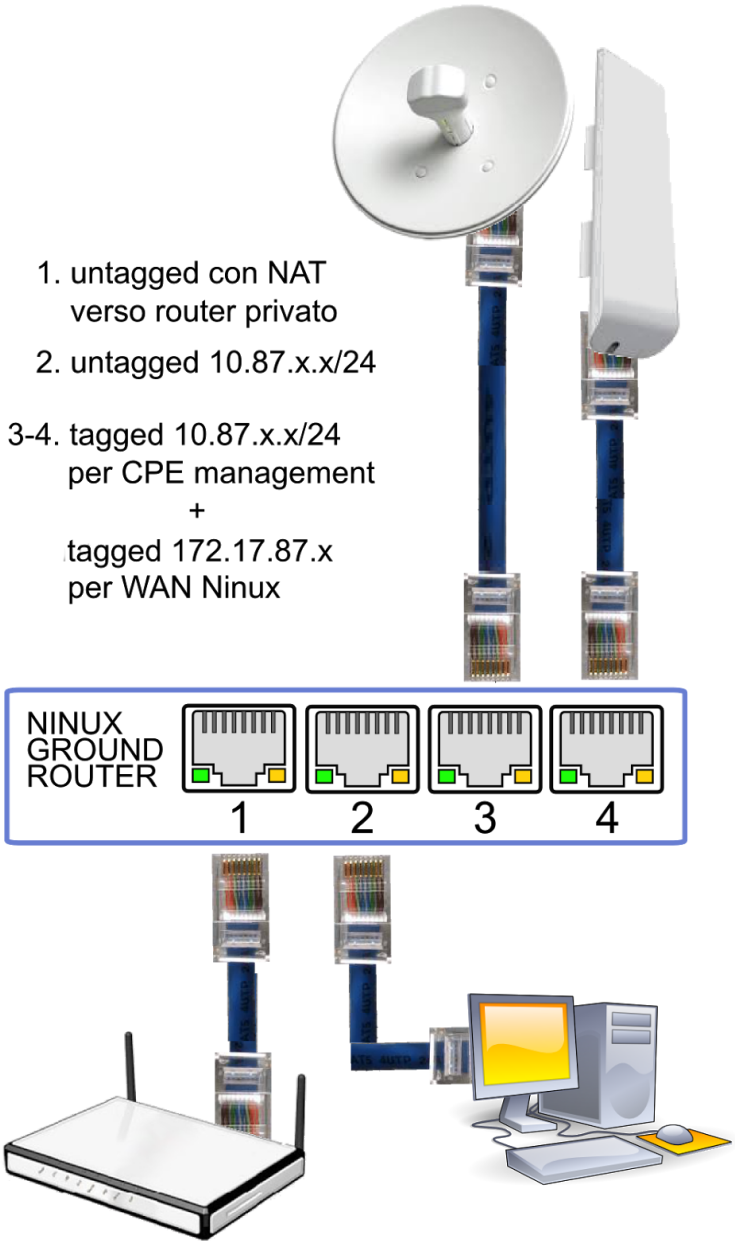
Rimane solo una cosa da dire: se una porta fa parte di più VLAN significa che può ricevere più "flussi di rete". Come farà la porta a distinguere a quale delle sue VLAN appartiene questo flusso di rete in arrivo? La risposta è semplice: il traffico appartenente ad ogni VLAN contiene un'etichetta (**tag**), un numero che varia da 0 a 4095. Il traffico in uscita da questa porta è detto **tagged** (taggato).

Per poter comprendere questo tag e distinguere le VLAN è necessario che l'hardware e il software (router, switch, ma anche i dispositivi finali) supportino lo standard **802.1q** e siano dunque *VLAN-aware*. Questo tuttavia non significa che dispositivi che non supportano e/o non sono in grado di configurare le VLAN non possano far parte di una VLAN e dunque comunicare con le altre porte che la compongono. È infatti possibile impostare, in **una** VLAN, una porta come **untagged** (non taggata). Quello che succederà è che quando sarà necessario inoltrare il traffico taggato in arrivo dalle altre porte della VLAN verso il dispositivo finale collegato ad una porta non taggata, sarà premura dello switch rimuovere prima il tag e trasformare il traffico in normale traffico Ethernet, di modo che possa essere elaborato dal dispositivo finale.
Perchè una porta può essere impostata untagged solo in **una** della VLAN di cui fa parte? Per capire, supponiamo per un attimo che una porta sia untagged in due VLAN diverse, composte da porte completamente diverse, con solo la suddetta porta untagged in comune. Lo switch riceve in ingresso del traffico non taggato su questa porta comune. Come farebbe lo switch, senza nessun tag, a stabilire se quel traffico è destinato alle porte di una o dell'altra VLAN? Non potrebbe. Se invece questo limite è rispettato, lo switch non ha problemi perché ha un'unica scelta possibile e dunque inoltrerà (applicando prima il tag) verso le altre porte della VLAN.

[TODO]*inserimento di un box con una spiegazione più tecnica*[/TODO]

Vantaggi del routing a terra

Possibilità di usare i firmware più adatti: non c'è più alcun bisogno di sostituire il firmware proprietario presente sull'apparato radio con una versione modificata o un firmware del tutto diverso. Anche se quasi sempre un driver open source per la gestione dell'antenna è disponibile, chi li ha usati sa che nel particolare campo del wireless i driver open source sono sempre essere aperti a bug subdoli o a limitazioni dal punto di vista delle prestazioni, del risparmio energetico o dell'affidabilità in generale. Queste situazioni possono essere evitate sfruttando driver e blob binari scritti direttamente dal produttore dalla sua posizione privilegiata di totale conoscenza dell'apparato.
Parco Hardware: col routing a terra l'unico compito dell'antenna diventa quello di stabilire e gestire il link radio e venendo meno, come detto sopra, la necessità di essere compatibili con i firmware Ninux, diventa possibile utilizzare qualsiasi apparato capace di stabilire suddetto link. Allo stesso modo il ground router può essere qualsiasi apparato che abbia 2 o più porte di rete, supporti le VLAN e sia in grado di far girare il protocollo di routing desiderato. Queste caratteristiche sono soddisfatte da una gamma molto ampia di dispositivi.
Prestazioni: oltre al già citato utilizzo dei firmware più ottimali in funzione di ciascun apparato, e alla specializzazione dei compiti di ciascun dispositivo (link radio per le antenne, routing sul Ground Router), la possibilità di far girare un'unica istanza del demone di routing invece di una per ciascuna antenna comporta una consistente riduzione del traffico di segnalazione con conseguente aumento delle prestazioni della rete.
Centralizzazione della configurazione: tutta la configurazione di rete avrà luogo sul ground router che, al contrario delle antenne, può essere posizionato in un luogo decisamente più accessibile. In caso di guasto, di modifica della configurazione, di manutenzione o di sostituzione con un dispositivo più performante, per esempio, non sarà più necessario risalire sul tetto! Inoltre la limitazione dei compiti delle antenne al solo link radio rende estremamente più ridotte le possibilità di blocchi o arresti che costringerebbero a tornare ad operare su di esse.
Flessibilità: il numero superiore di interfacce, il maggior numero di pacchetti software installabili e le prestazioni generalmente superiori di un router di terra rispetto all'apparato radio consentono di realizzare una moltitudine di configurazioni diverse adatte ad ogni esigenza, e di farlo in maniera più diretta e semplice operando su un solo apparato.
Sicurezza: il ground router rappresenta un unico dispositivo "di frontiera" tra la propria rete locale e la rete Ninux, e in quanto tale diventa molto più semplice gestire e tenere aggiornate regole di firewall, tunnel e accessi.



In figura una configurazione classica, una CPE è station verso un AP, un'altra distribuisce come AP oppure è in modalità station, la porta 2 è collegata ad uno switch o ad una workstation in rete 10.87.x.x/24 mentre la porta 1 fà NAT verso una rete privata che non ha routing verso Ninux. Ogni rete è isolata a livello logico grazie alla tecnologia 801.2Q (VLAN) e comunica con le altre mediante forward e routing oppure NAT.

Requisiti

Innanzitutto, **i vostri indirizzi**! Visitate la pagina degli indirizzi sul wiki di Ninux (e le relative sottosezioni dedicate alle isole) e scegliete gli indirizzi coi quali il vostro nodo si collegherà in Ninux. Sarà necessario un indirizzo appartenente alla sottorete 172.16.0.0/12 per ciascuna delle nostre antenne (1 antenna = 1 indirizzo) e poi un'unica sottorete /24 da dedicare alla vostra rete locale.

Come detto, le **antenne** sono a vostra discrezione. L'unica limitazione è che dovranno essere in grado di stabilire e gestire il link radio verso il nodo Ninux al quale ci collegheremo, e pertanto dovranno operare nelle sue stesse frequenze (2,4 o 5 GHz). Per approfondire sulle antenne più utilizzate e consigliate in Ninux, consulta il wiki ma soprattutto il documento *Fondamenti per costuire una rete wireless libera*.

Infine il pezzo forte, il **ground router**. Come detto, il dispositivo che farà da router a terra può essere qualsiasi dispositivo che soddisfi queste caratteristiche:

- Tante porte/interfacce di rete quante sono le antenne + 1.
- Supporto alle VLAN.
- Possibilità di installare il protocollo di routing desiderato, come ad esempio OLSR o Batman. Generalmente questo si traduce direttamente nell'avere un firmware basato su Linux o BSD.

Di hardware che soddisfa queste caratteristiche ce n'è un'infinità, ma al 95% quello di cui avrete bisogno è un **router domestico compatibile con OpenWRT**. Ce ne sono tanti tra cui scegliere, ma consigliamo di optare uno con le seguenti specifiche:

- switch gigabit,
- almeno 64MB di RAM
- almeno 8MB di memoria flash
- privo di bug noti nella gestione delle VLAN.

Torneremo su tutto questo nella sezione dedicata a OpenWRT. Altre possibilità per il router a terra sono:

- un router professionale con un firmware Linux-based, come l'**Ubiquiti EdgeRouter** e il suo sistema EdgeOS (difficilmente battibile come rapporto qualità/prezzo in funzione delle necessità di Ninux).
- Un **computer dedicato con più schede di rete**, su cui poi installare un sistema operativo GNU/Linux o una distribuzione firewall come pfSense.
- Una **board embedded** come le routerboard o le Alix, sulle quali installare OpenWRT, GNU/Linux, pfSense, o simili.

Verrà data per scontata la presenza sul nostro PC di una normale **NIC con porta RJ45 direttamente connessa** alle periferiche da configurare. Se usiamo adattatori USB-to-Ethernet perchè non usufruiamo di una scheda di rete pura oppure se il collegamento passa attraverso uno switch è possibile che si possano verificare anomalie nell'uso di VLAN. Ad esempio, si riesce ad accedere via SSH ma non via HTTPS (interfaccia web).

Infine, questa guida presume che chi configura stia **operando da un sistema operativo Linux**. Se siete utenti di altri sistemi operativi, potete usare una versione avviabile da LiveCD o LiveUSB di una delle tante distribuzioni Linux (le schermate di questa guida si riferiscono ad Ubuntu, ma altre vanno altrettanto bene) e utilizzarla per seguire questo HowTo senza dover procedere all'installazione.
Una volta avviato il sistema, l'unico pacchetto aggiuntivo che sarà necessario installare sarà **vlan**, che contiene l'utility *vconfig* che permetterà di impostare le VLAN sul nostro PC.
È bene, salvo quando diversamente specificato in questo howto, **disabilitare il Network Manager** della distribuzione Linux che stiamo utilizzando. La sua presenza sarà più d'intralcio che d'aiuto durante la configurazione del router a terra. Si può fare comodamente dall'icona nella barra di stato, **togliendo la spunta a *Abilita funzionalità di rete*** o voci simili a questa.
Notazione: Dove compaiono comandi da terminale un prompt "#:" indica che il comando va eseguito

con i permessi di root. Per ottenere i permessi di root definitivamente in un terminale, digitate “sudo su”, inserite la password, e da quel momento in poi ogni comando verrà eseguito come root; in alternativa è possibile anteporre “sudo” di fronte ad ogni comando per eseguire lo stesso come utente root. Un prompt “$:” indica invece che il comando può essere eseguito senza i privilegi di superutente. Un “##” indica invece un commento che illustra la funzione del comando susseguente.

Configurazione Radio

La configurazione della parte **wireless** e di tutto il resto del firmware dell’antenna non cambia col routing a terra rispetto alle altre configurazioni in uso nella rete Ninux, e pertanto non verranno trattate in questa guida.
Ecco cosa andremo a fare: metteremo l’antenna in modalità bridge e sostituiremo il bridge di default tra wireless e LAN0 con uno tra wireless e una VLAN; questa VLAN sarà destinata ad accogliere il traffico in ingresso/uscita da/verso il nodo remoto, proprio come se fosse direttamente connessa ad esso, e sarà diversa (tag VLAN differente) per ognuna delle nostre antenne.
Per consentire la gestione interna dell’antenna (ovvero l’accesso alla sua interfaccia web) estenderemo i normali indirizzi della nostra rete locale alle antenne. Per farlo creeremo una seconda VLAN, **comune** (stesso tag VLAN) a tutte le nostre antenne, e la designeremo come interfaccia per il managing. In questo modo le antenne si comporteranno proprio come se fossero PC sulla nostra rete locale. La creazione della seconda VLAN potrebbe essere evitata, ma solo se il nostro ground router è in grado di gestire traffico taggato e non taggato su una medesima porta. Sfortunatamente molti router OpenWRT non sono in grado di farlo, e pertanto aggireremo il problema taggando sempre anche il traffico “interno”. Approfondiremo la questione nella sezione dedicata a OpenWRT.

AirOS (Ubiquiti)

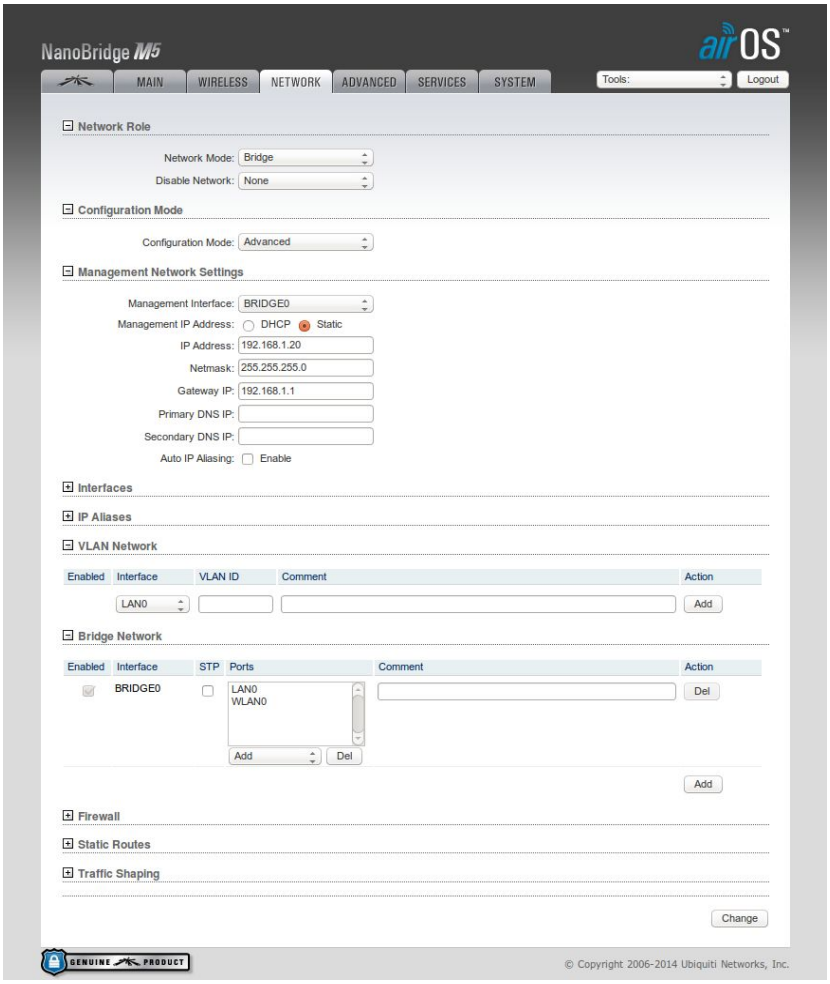
Apriamo un terminale di root e diamo un indirizzo alla nostra scheda di rete:

```
## attivazione interfaccia
#: ifconfig eth0 up
## assegnazione indirizzo
#: ifconfig eth0 192.168.1.2/24
```

Nel browser, logghiamoci nell’interfaccia della nostra antenna (192.168.1.20 se ancora con le impostazioni di fabbrica) e andiamo nel tab **Network**.

Impostiamo l’antenna in modalità bridge (**Network Mode: Bridge**) e attiviamo la configurazione avanzata (**Configuration Mode: Advanced**).

Compariranno molte altre impostazioni di configurazione divise in sezioni; espandiamo già adesso quelle che ci serviranno: **Management Network Settings**, **VLAN Network** e **Bridge Network**.



Per prima cosa **creiamo le VLAN sull'intefaccia LAN0** in VLAN Network. Per farlo ci basterà **compilare il campo VLAN ID** e cliccare sul tasto **Add**. Opzionalmente possiamo lasciare un commento che descrive l'utilizzo della VLAN.

- Per prima creiamo la "VLAN di traffico wireless" col nodo remoto. Essendo il VLAN ID 1 riservato in AirOS, e i VLAN ID 1 e 2 spesso già creati in OpenWRT, è preferibile cominciare dal **VLAN ID 3**. Se abbiamo altre antenne è meglio usare VLAN ID progressivi: 4, 5, 6 e così via.
- Ora creiamo la "VLAN di gestione" interna che renderà le antenne parte della (e accessibili dalla) nostra rete locale. Qualsiasi VLAN ID va bene, ma è preferibile tenersi sul primo lasciato libero dalle "VLAN di traffico wireless" e darne uno minore di 16, ad esempio **VLAN ID 7**. L'importante, se si hanno altre antenne, è configurare la VLAN di gestione sempre con lo stesso ID (7 in questo esempio).

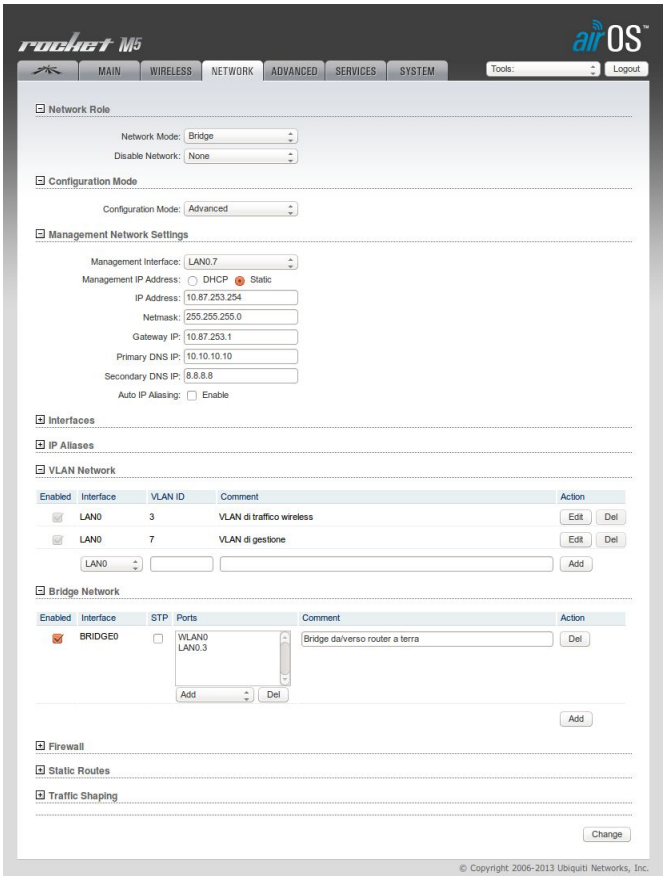
Ora cambiamo l'interfaccia di gestione: in Management Network Settings selezioniamo **Management Interface: LAN0.7** e configuriamola:

- **Management IP Address**: Static
- **IP Address**: un indirizzo della propria subnet Ninux assegnataci, preferibilmente uno "alto" facile da ricordare; ad esempio, se la nostra subnet è 10.87.253.0/24, possiamo assegnare 10.87.253.254 (253, 252, se abbiamo ulteriori antenne).
- **Netmask**: abbiamo una /24, dunque scriveremo 255.255.255.0.

- **Gateway IP**: l'indirizzo del nostro router a terra, che per convenzione sarà il primo disponibile nella nostra sottorete, il ".1". Nel nostro esempio 10.87.253.1.
- **Primary DNS IP e Secondary DNS IP**: Opzionali, da compilare con i proprio server DNS preferiti.

Possiamo infine **sostituire il bridge**: andiamo nella sezione Bridge Network ed eliminiamo il bridge attuale cliccando sul tasto **Del**. Diventa ora possibile creare un nuovo bridge con **Add**. Infine aggiungiamo le interfacce che faranno parte del bridge cliccando sul **menu a tendina Add** e scegliendo prima **WLAN0** e poi **LAN0.X**, dove la X è il VLAN ID della "VLAN di traffico wireless" dell'antenna che stiamo configurando (LAN0.3, LAN0.4, LAN0.5, ecc).

La nostra situazione ora dovrebbe essere esattamente questa:



Siamo pronti per salvare il tutto: clicchiamo su **Change** in basso a destra, e successivamente selezioniamo **Apply** nel messaggio a sfondo azzurro che comparirà in alto. Il dispositivo verrà ora riavviato.

Se tutto è andato bene… dovremmo avere perso l'accesso all'interfaccia web. Per riguadagnare la schermata di AirOS nel nostro browser dobbiamo dapprima configurare una VLAN con ID 7 (la sola che ora consente l'accesso all'antenna) anche sul nostro PC.
Apriamo un terminale di root e digitiamo:

attivazione interfaccia di rete
#: ifconfig eth0 up
creazione vlan di gestione
#: vconfig add eth0 7
attivazione interfaccia logica
#: ifconfig eth0.7 up
assegnamento indirizzo nella stessa subnet
#: ifconfig eth0.7 10.87.253.100/24
test finale
$: ping 10.87.253.254

Dovremmo ora osservare l'antenna **rispondere ai nostri ping** (CTRL + C per terminare). L'interfaccia di AirOS a questo punto è accessibile all'indirizzo impostato sull'antenna in precedenza (10.87.253.254 nel nostro esempio). Adesso possiamo scollegare l'antenna e dare questi comandi

togliamo l'indirizzo dalla VLAN 7 del nostro pc, senza cancellarla
#: ifconfig eth0.7 0.0.0.0

e ricominciare dall'inizio se abbiamo una seconda antenna. Se questa era la nostra unica antenna, possiamo passare al ground router.

RouterOS (Mikrotik)

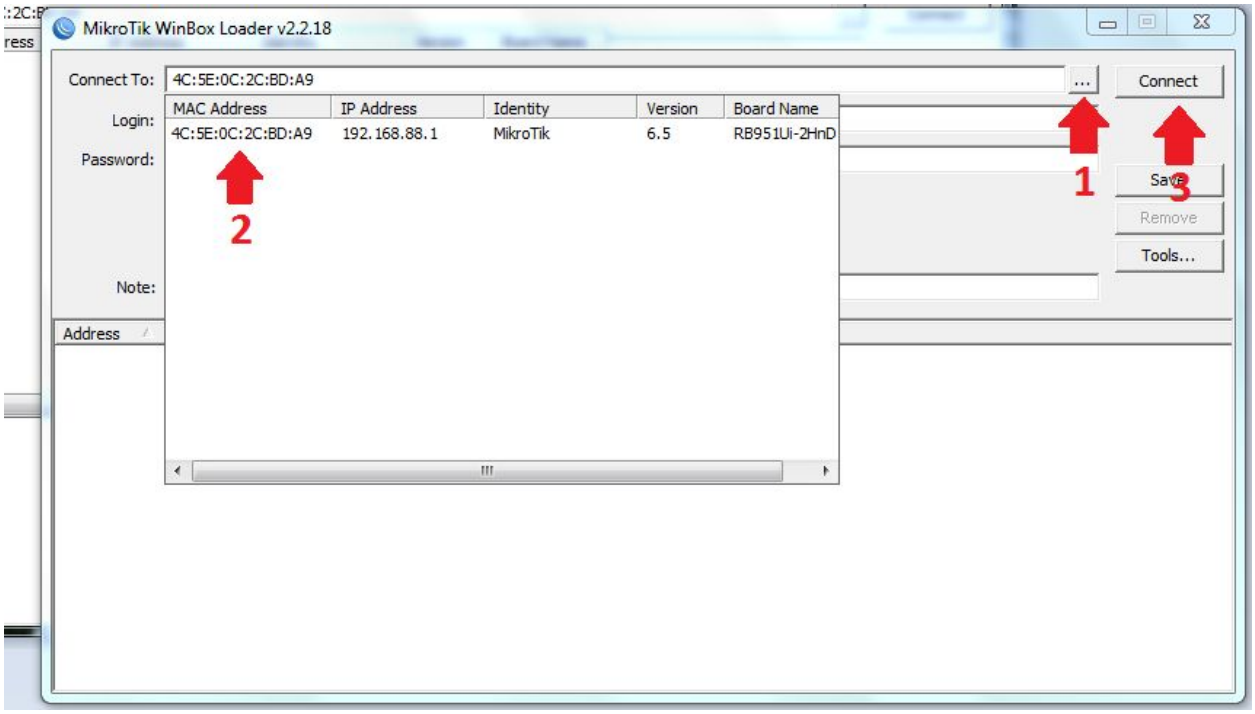
Connessione all'apparato

Per prima cosa, scarichiamo **winbox** dal sito ufficiale mikrotik (da qui) .
Winbox è l'utility per la programmazione delle radio mikrotik. E' possibile anche utilizzare l'interfaccia web ma coprirò esclusivamente la procedura via winbox, in quanto i modelli più vecchi delle routerboard erano accessibili solo tramite questa utility oppure via riga di comando.
Winbox è un eseguibile per windows, perciò se utilizziamo linux, è possibile eseguire l'applicazione tramite wine.

Colleghiamo la radio al pc via cavo e disabilitiamo la scheda wireless (se presente) del pc. Ora lanciamo winbox e premiamo il pulsante **"..."** a fianco di **connect**.

L'utility farà una ricerca delle radio collegate ignorando l'ip su cui esse si trovano.
Una volta comparsa la radio nella lista, clicchiamo sul **mac address**, poi **connect** (eventualmente inseriamo username e password se esse sono state configurate, altrimenti lasciamo "admin" e "vuoto").

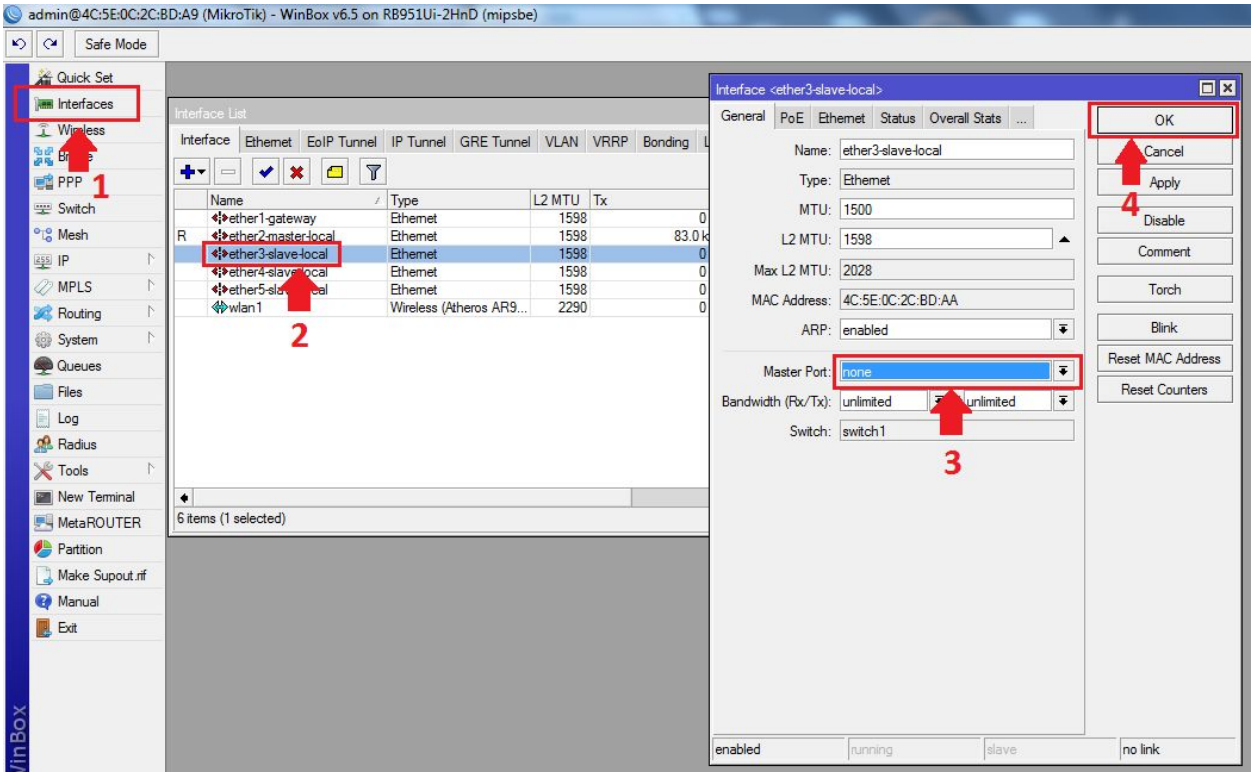
Accedere alla radio tramite mac ci evita di dover riconfigurare l'indirizzo ip del pc.



Configurazione ip, vlan e bridge

Iniziamo.
Dalla barra a sinistra, andiamo in **bridge** ed eliminiamo (con il **"-"** rosso nella barra superiore) il bridge di default, che è quello che collega **wireless** e **lan**.
E' probabile che durante la procedura l' applicazione ci scolleghi dall'antenna ma è sufficiente ricollegarsi con il metodo precedente.
Ora ritorniamo in **bridge > ports** ed eliminiamo (con il tasto **"-"** rosso nella barra superiore)le porte associate al bridge eliminato in precedenza.

Ora andiamo in **interfaces**.
Se la nostra radio ha una sola porta ethernet, passate al prossimo punto, altrimenti eseguiamo "l'isolamento delle porte"
Doppio click sul nome della porta ethernet e in **Master port** scegliamo **none**, poi **ok** per confermare.
Rieseguiamo questa procedura per ogni porta.



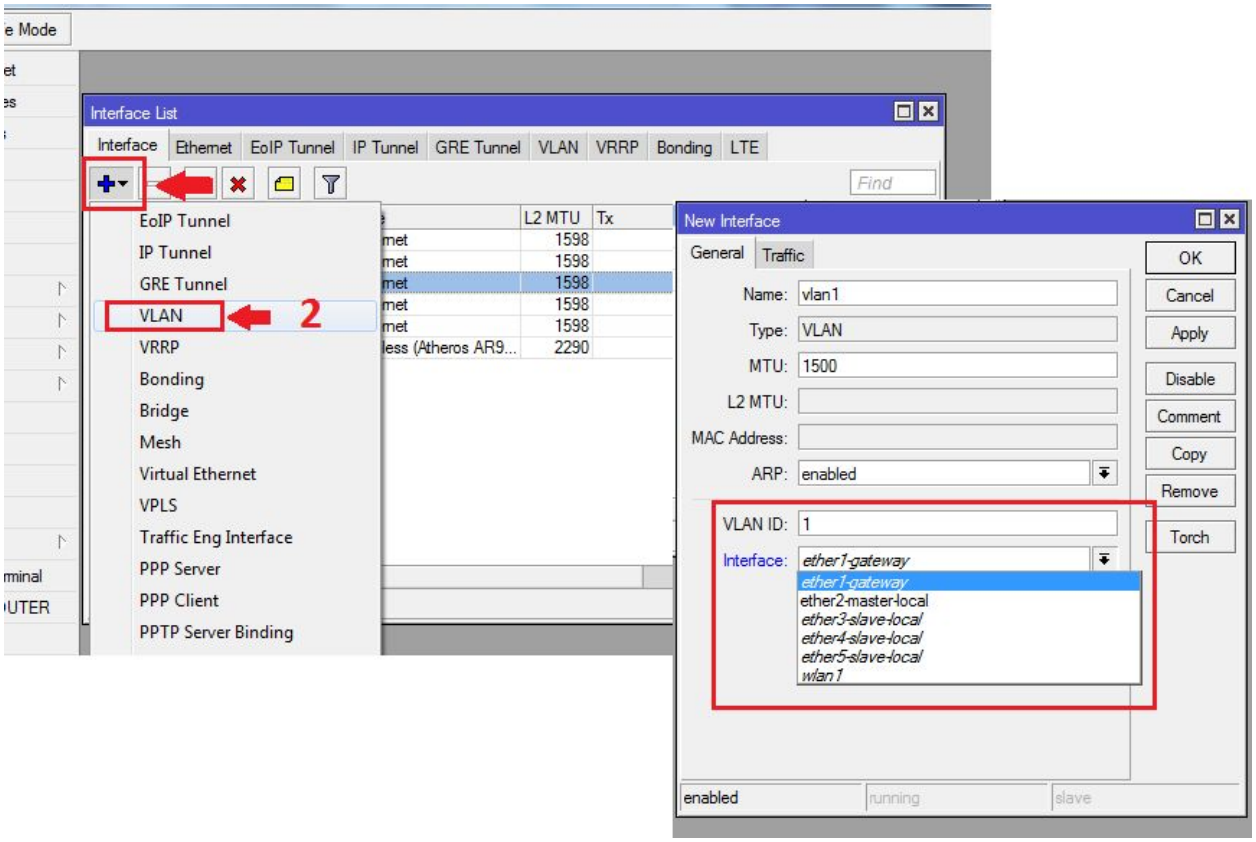
Riprendiamo da qui se abbiamo solo una porta ethernet.
Clicchiamo dalla schermata **interfaces** su **"+"** poi **vlan** e infine nella finestra che comparirà compiliamo i dati della vlan.
Dovremo eseguire questa procedura due volte specificando i seguenti parametri per rispettivamente la prima e la seconda volta:

1. **vlan id** =X, dove X è l'id della vlan di "traffico wireless"
   interface= nome dell'interfaccia su cui collegheremo il cavo
   nome: consiglio un nome esplicativo e che contega il vlan id scelto per avere subito a colpo d'occhio lo scopo di questa interfaccia (ad esempio "vlan3 traffico wireless")
2. **vlan id**=Y, dove Y (diverso da X) è l'id della vlan di "gestione radio"
   interface= nome dell'interfaccia su cui collegheremo il cavo (che deve coincidere con quella selezionata per la vlan precedente)
   nome: come sopra.

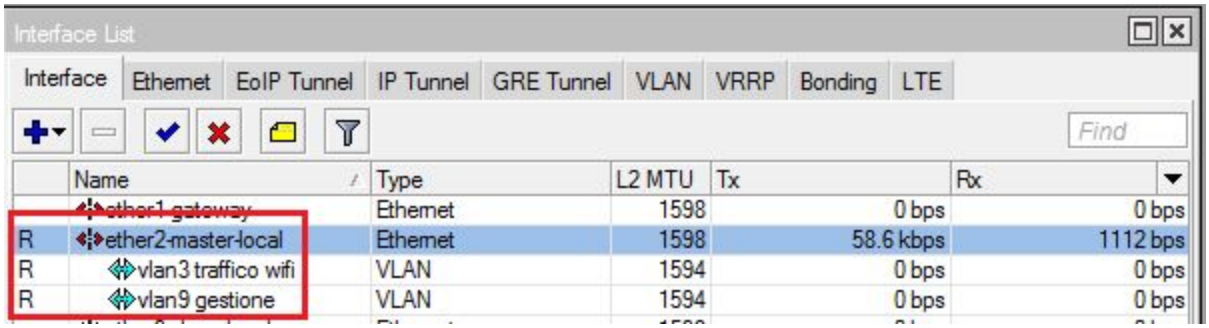
L'accortezza da usare per questi due passaggi è quella di scegliere degli id opportuni: evitiamo 1 e 2 in quanto comuni in openwrt e iniziamo dal 3 in poi, avendo cura di **NON riutilizzare** lo stesso vlan id per il traffico wireless anche su eventuali altre radio (mikrotik e non) e invece di **riutilizzare** lo stesso vlan id di "gestione radio" anche per eventuali altre radio (mikrotik e non).

Ad esempio, utilizziamo 3,4,5,6 per 4 radio diverse nel campo "vlan id di traffico wireless" ma sempre 9 (oppure 10, ma anche 8 o 11) per il campo "vlan id di gestione radio".

Memorizziamoci inoltre questi id perchè poi ci serviranno per configurare il router per il ground routing.

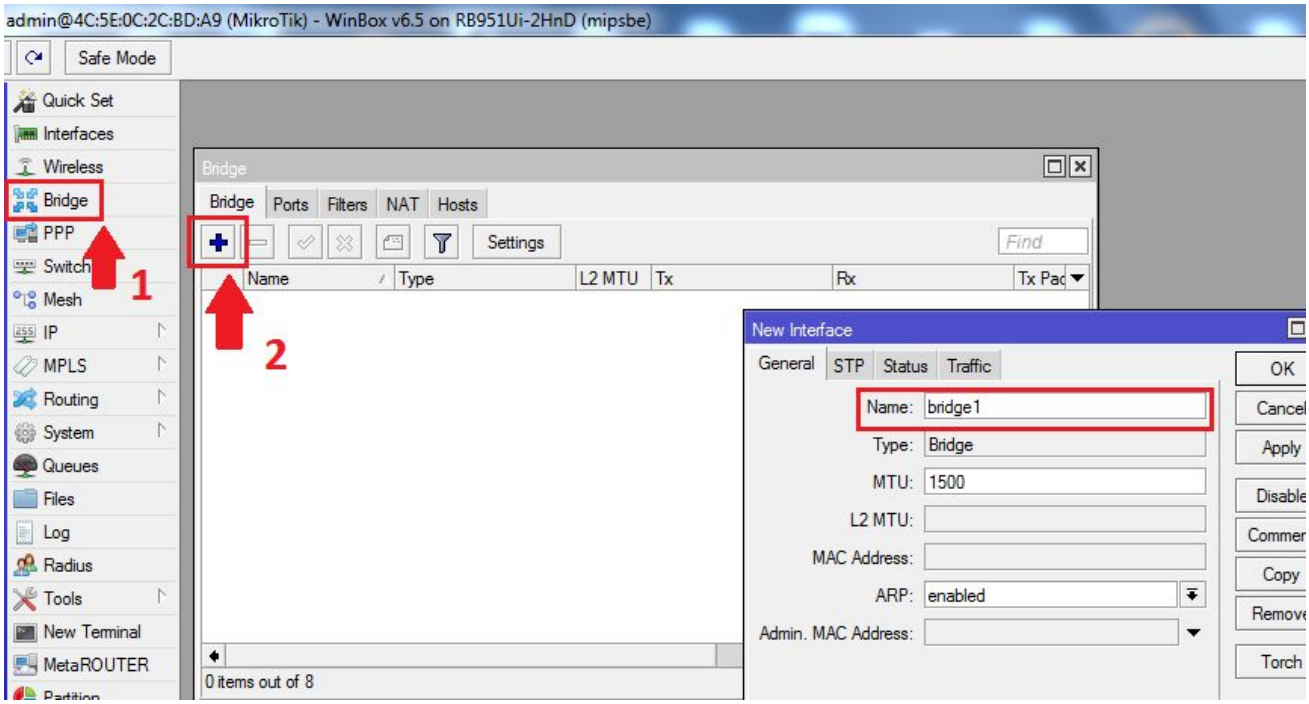


Al termine della procedura dovremmo avere qualcosa del genere:

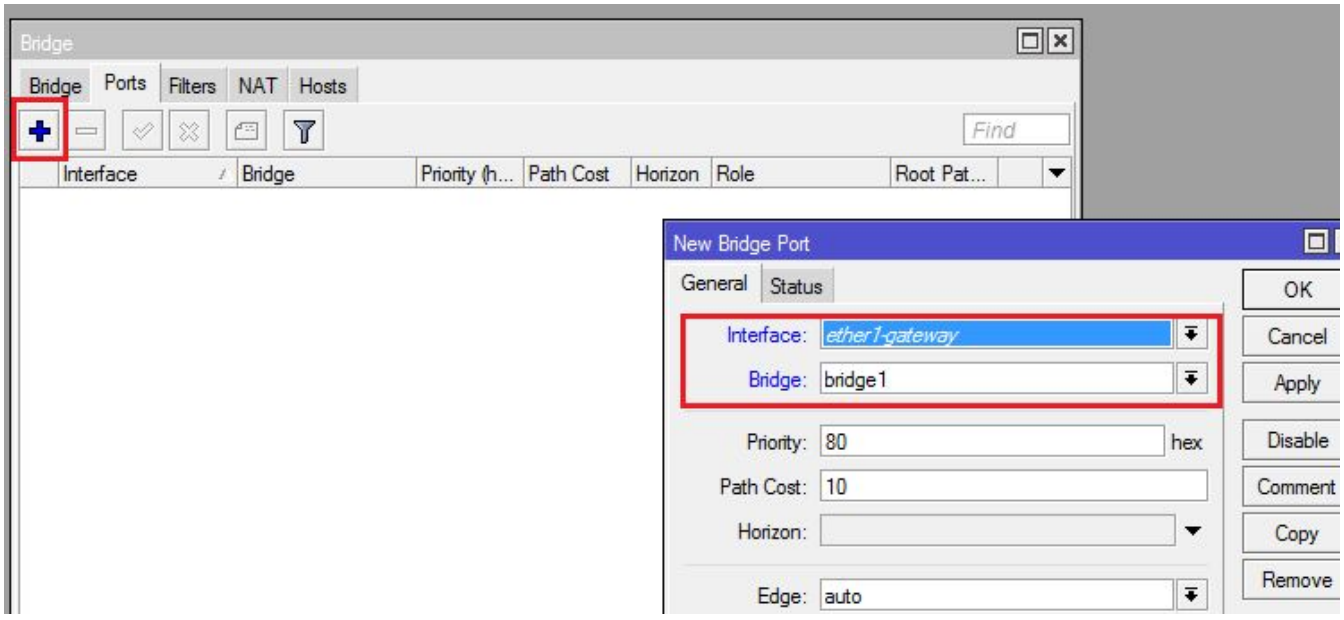


Ora creiamo un bridge tra la "vlan di traffico wireless" e l'interfaccia wireless.

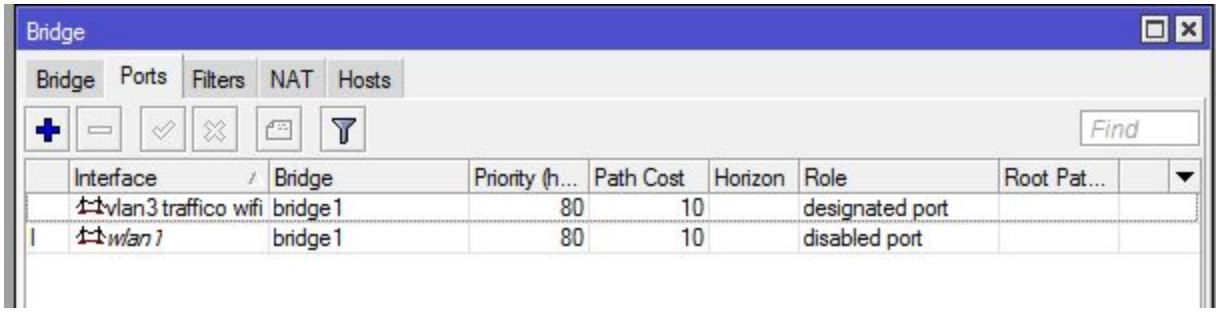
Andiamo in **bridge** , clicchiamo su **"+"** e scegliamo un nome per il bridge. Confermiamo con **ok**.



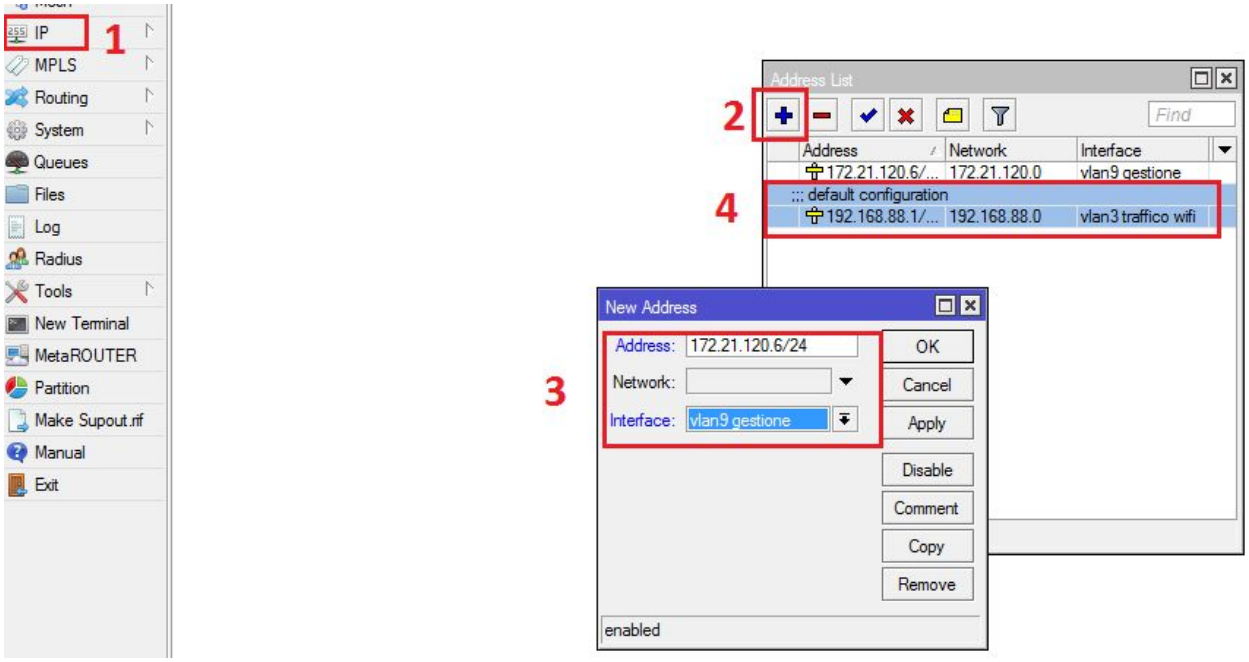
Spostiamoci nel tab **ports** e clicchiamo su **"+"**
Scegliamo dal campo **interface** l'interfaccia di **vlan traffico wireless** e come bridge, il nome del bridge creato al punto precedente.
Ripetiamo la procedura scegliendo questa volta l'interfaccia wifi (probabilmente si chiama wlan1) e come bridge, il nome del bridge come al punto sopra.



Otterremo qualcosa di questo tipo:



Ora non ci resta che gestire gli ip della radio.
Andiamo in **ip > dhcp client** e rimuoviamo tutto quello che troviamo nel tab **dhcp client** (con **"-"** rosso")
Idem andando in **ip > dhcp server**.
Ora in **ip > addresses** clicchiamo su **"+"** e nella schermata che comparirà inseriamo uno degli ip della nostra lan riservata comprensivo dello /CC finale (ad esempio 10.120.0.6/24). Prima di confermare selezioniamo l'interfaccia su cui dovrà essere attivato l'ip, che per noi è l'interfaccia vlan di gestione.
Clicchiamo su **ok** e poi eliminiamo qualsiasi altro ip presente nella schermata.



Fatto!
Se abbiamo altri apparati ricominciamo dall'inizio per la loro configurazione, altrimenti si può proseguire con la configurazione del ground router.

OpenWRT

TODO

Adhoc/Client Mode

Non essendo possibile su openwrt inserire interfacce in client o in ad-hoc all'interno di un bridge dovremo utilizzare il pacchetto trelay e creare questo pseudo-bridge a livello user space.
Per installarlo ci basterà scaricarlo dal packet manager
opkg update
opkg install trelay

dopodiché andremo a configurare /etc/config/trelay inserendo al posto delle 2 interfacce la nostra interfaccia in client/ ad-hoc e la nostra vlan per il traffico verso terra.
Così facendo avremo creato l'equivalente di un bridge e potremo trattare il nostro traffico a terra in maniera trasparente.

Configurazione Ground Router

OpenWRT (LuCI)

Ok, avete flashato OpenWRT fresco fresco sul vostro router a terra e vi siete loggati nell'interfaccia web LuCI (192.168.1.1). Configurarlo attraverso l'interfaccia grafica nel proprio browser è certamente più semplice per chi ha appena iniziato in Ninux, ma richiede delle particolari attenzioni che saranno discusse nel corso di questa sezione.

Cosa c'è da sapere sugli switch in OpenWRT

Nella tipica parte posteriore del tipico router OpenWRT troveremo **4 porte RJ45 numerate (1-4)**, solitamente di colore giallo, e **una singola porta RJ45 di colore blu etichettata come WAN o Internet**. Le prime porte sono quelle che il produttore ha destinato ad essere utilizzate per collegare i singoli PC in un'unica rete locale, mentre la porta blu è utilizzata per collegare la rete locale ad altre reti (Internet, solitamente, attraverso un modem ADSL).
Ora, due casi possono verificarsi all'interno del router:

- Le 4 porte numerate fanno parte di un unico switch, e la porta blu appartiene ad una diversa interfaccia completamente separata dallo switch. Queste due interfacce appena discusse sono note al router come eth0 e eth1.
- Tutte e 5 le porte fanno parte di un'unico switch. In questo caso il router, già con le impostazioni di fabbrica, usa due VLAN preconfigurate (solitamente con VLAN ID 1 e 2) per separare le 5 porte nel solito "4+1". Se questo è il caso, il router chiamerà queste due interfacce eth0.1 e eth0.2.

In generale Linux (sul quale OpenWRT è basato) associa ad ogni VLAN una interfaccia virtuale e la denomina nella forma **ethX.Y**, dove **X** è il numero dell'interfaccia fisica (uno switch, una scheda di rete, ecc.) e **Y** è il VLAN ID (cioè il tag) della VLAN in questione.

Qualsiasi sia il numero di porte RJ45 che faccia parte dello switch, di esso farà sempre parte una porta aggiuntiva speciale che è la **porta CPU**. Questa porta, a differenza delle altre, non presenta un connettore fisico RJ45 accessibile all'esterno ma è invece "saldata" sulla scheda madre del router.

Senza entrare troppo nel dettaglio, questa porta “vede” i pacchetti di tutte le altre e svolge la fondamentale funzione di renderli disponibili al sistema operativo OpenWRT per l’elaborazione (routing, firewall, shaping, ecc).

La necessità di operare con le VLAN rende fastidioso ogni bug che le riguardi. In particolare, tantissimi dei router OpenWRT-compatibili di nostro interesse soffrono di un **bug che rende impossibile fare uscire da una stessa porta sia traffico tagged che untagged** (su due VLAN diverse, ovviamente): se ci si prova si può osservare che quella porta smette totalmente di comunicare oppure continua a far uscire sempre e solo traffico tagged anche nella VLAN dove lo si vuole untagged.
Per quanto fastidioso, questo bug non è particolarmente bloccante: per aggirarlo basta utilizzare la doppia VLAN sull’antenna che abbiamo visto nella sezione “Configurazione Radio”, invece della singola che sarebbe stata necessaria se il nostro router non avesse questo bug; dal punto di vista del ground router invece, questo bug significa semplicemente che ci saranno delle porte (una per ogni antenna) dello switch in meno disponibili per far passare traffico “legacy” (comprensibile dai dispositivi che non supportano le VLAN).

Router OpenWRT-compatibili: Dispositivi consigliati

Marca	Modello	Gigabit	RAM	Flash	Bug VLAN	Easy Install	Uso	Prezzo	Note
TP-Link	TL-WR740N/ND v4.x TL-WR741ND	NO	32 MB	4 MB	SI	SI	Foglia	20 €	
TP-Link	TL-WR841N TL-WR841ND v8.x	NO	32 MB	4 MB	SI	SI	Foglia	20 €	
TP-Link	TL-WR841N v9.x	NO	32 MB ?	4 MB	?	SI	Foglia	18 €	BB
TP-Link	TL-WR1043ND v1	SI	32 MB	8 MB	NO	SI	Foglia	40 €	Fuori Produzione
TP-Link	TL-WR1043ND v2	SI	64 MB	8 MB	?	?	SuperNodo	40 €	BB
TP-Link	TL-WR2543ND	SI	64 MB	8 MB	?	SI	SuperNodo	60 €	
TP-Link	TL-WDR3600	SI	128 MB	8 MB	SI	?	SuperNodo	45 €	
TP-Link	TL-WDR4300	SI	128 MB	8 MB	NO	?	SuperNodo	50 €	
TP-Link	TL-WDR4900	SI	128 MB	16 MB	?	SI	SuperNodo	80 €	BB
Asus	RT-N16	SI	128 MB	32 MB	?	?	SuperNodo	75 €	BB
D-Link	DIR-825 (B1/B2)	SI	64 MB	8 MB	?	?	SuperNodo	? €	AA-T Out of Stock
D-Link	DIR-825 (C1)	SI	128 MB	16 MB	?	?	SuperNodo	? €	BB Out of Stock
LinkSys	WRT160NL	SI	32 MB	8 MB	?	SI	Foglia	65 €	

Allo stato attuale, tra i dispositivi OpenWRT compatibili, le scelte consigliabili sono:

- TP-Link TL-WR1043ND v2.x, per i nodi foglia.
- TP-Link TL-WDR4300, per i SuperNodi.

Preparazione

Se non è già collegato, colleghiamo il router a terra alla porta ethernet del nostro PC. In un terminale di root:

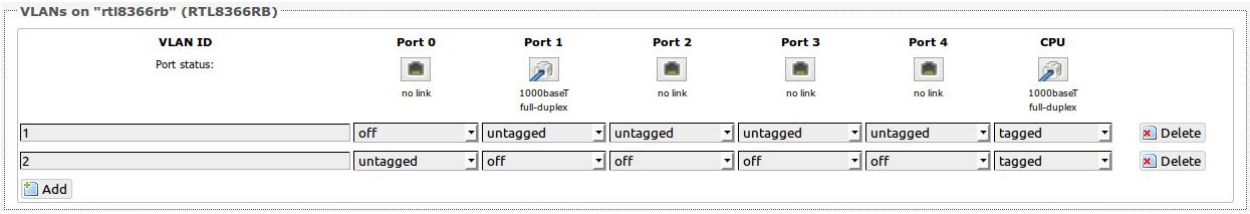
attiviamo l'interfaccia
#: ifconfig eth0 up
diamoci un indirizzo per accedere all'interfaccia del router
#: ifconfig eth0 up 192.168.1.2/24

Ora il router sarà accessibile via browser all'indirizzo 192.168.1.1. Prendetevi il tempo per impostare una password, cosa che farà scomparire il continuo promemoria di OpenWRT.

Analisi dello switch: presenza delle VLAN di default

Altro passaggio facile: basta recarsi in **Network > Interfaces**.
Se l'interfaccia associata alla Network OpenWRT "WAN" è nella forma ethX, significa che la porta blu del router è un'interfaccia a parte rispetto allo switch, sul quale sarà presente un'unica VLAN.

Se invece alla Network OpenWRT "WAN" è associata un'interfaccia ethX.Y, significa che lo switch nel router include tutte le sue porte e le separa per mezzo di 2 VLAN pre-configurate.
Network > Switch dovrebbe confermare la situazione sopra descritta.



Analisi dello switch: numerazione fisica vs numerazione logica

Assicuratevi che il cavo ethernet nel router a terra sia collegato alla porta dello switch etichettata col numero **1 (uno)**. Non preoccupatevi di perdere il collegamento col router: se avete disattivato NetworkManager il router manterrà gli indirizzi anche dopo aver scollegato e ricollegato il cavo. Fatto questo posizionatevi nella schermata **Network > Switch**.
Le icone, e le diciture "no link" e "full duplex", indicano chiaramente se in una porta dello switch è presente o meno un collegamento. Per il momento ignorate la "CPU Port", che ovviamente risulterà sempre collegata.
Sopra le icone è riportato il numero della porta **logica** dello switch. **Quasi sempre questo numero non corrisponde all'etichetta riportata fiscamente sul router**. Appuntatevi su un foglio di carta la corrispondenza tra etichetta fisica e porta logica.

Ora scollegate il cavo dalla porta etichettata 1 (uno) e collegatelo alla porta etichettata 2 (due) sul router. Senza neanche bisogno di aggiornare la pagina, entro qualche secondo noterete che la porta precedente cambierà icona e dicitura in "no link", mente un'altra porta segnalerà "full duplex" e l'icona di collegamento avvenuto. Appuntate questa nuova corrispondenza.
Continuate allo stesso modo per le porte etichettate **3 (tre)** e **4 (quattro)**. Alla fine i vostri appunti riporteranno una situazione del genere.

Porta con etichetta...	Porta logica...
1	2
2	3
3	4
4	1

Nel resto di questo HowTo, quando parleremo di "porta X" intenderemo sempre e solo la **numerazione logica su OpenWRT**. Starà a voi associare a questa la numerazione fisica sul router e collegare i cavi nella porta esatta.

Precauzioni

Operare sugli switch in OpenWRT è un'operazione delicata, ed essere sbattuti fuori dal pannello di configurazione perché la configurazione non è stata applicata o è stata applicata male è questione di un attimo. In questo sfortunato caso saremmo costretti a resettare il router e ricominciare da capo.
Tanto vale farsi furbi e lasciarsi una "porta sul retro" in caso di disastro, che consenta di accedere al router anche in caso di bizze allo switch. Nel nostro caso questa porta sul retro sarà quella blu del router, la WAN.

Andate in **Network > Interfaces** e cliccate su **Edit** nella Network di OpenWRT **WAN**.
Selezionate **Protocol:Static address** e confermate cliccando su **Switch protocol**.
Compilate il campo **IPv4 Address** con 192.168.254.1.
Selezionate **Netmask**:255.255.255.0
Save & Apply e attendete qualche secondo.
Andate in **Network > Firewall > General Settings**. Nella riga "WAN => Reject" variate **Input** da reject a **accept**.
Save & Apply e attendete che i messaggi di applicazione in corso scompaiano.

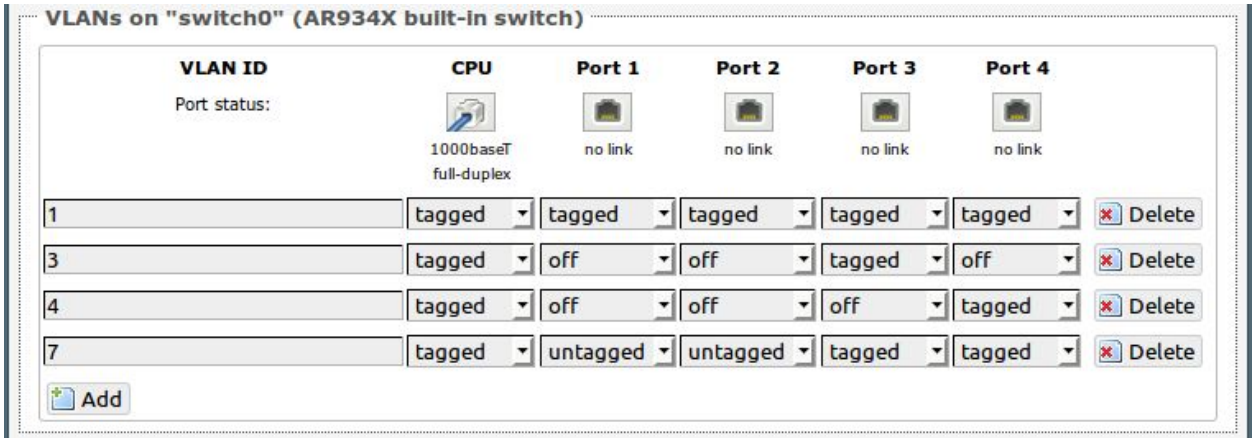
Da questo momento **il router sarà accessibile anche sulla porta WAN**, ovviamente dopo aver impostato sul proprio PC un indirizzo della rete corrispondente, ad esempio 192.168.254.2.

Creazione delle VLAN

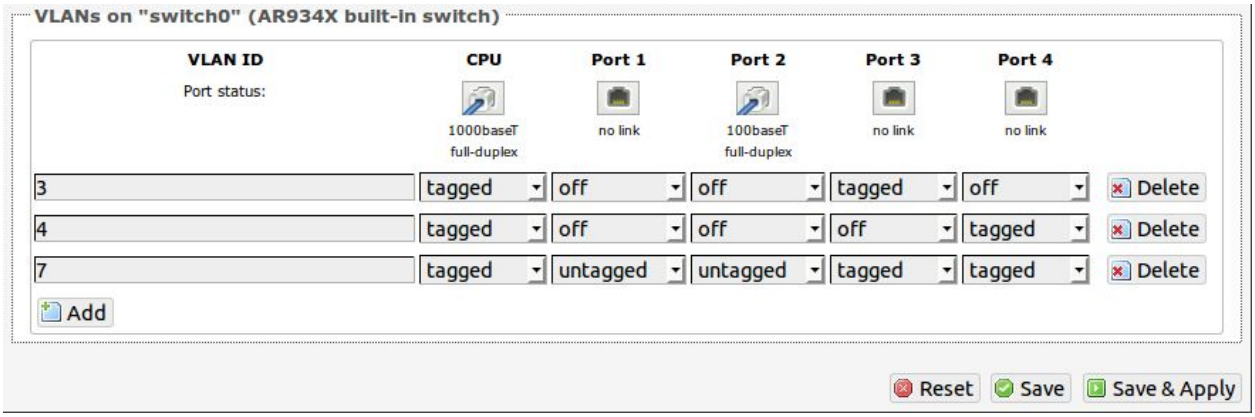
Andate in **Network > Switch**.
Nella VLAN di default che contiene 4 porte numerate untagged (solitamente è quella con VLAN ID 1), **cambiate tutte queste porte in tagged** (compresa la CPU Port).
Create le VLAN a voi necessarie cliccando su **Add** e compilando i campi della nuova riga che comparirà, seguendo attentamente queste regole:

- Qualsiasi VLAN create, assicuratevi che la **CPU Port** sia sempre **tagged**.
- Per **ciascuna** antenna, create una VLAN che abbia
    - VLAN ID pari a quello della “VLAN di traffico wifi” impostato durante la configurazione dell’antenna in questione.
    - la porta alla quale si collegherà l’antenna impostata come **tagged**.
    - tutte le altre porte numerate impostate ad **off**.
- Infine create **una** VLAN che abbia
    - VLAN ID pari a quello della “VLAN di gestione” impostato in comune su tutte le antenne durante la loro configurazione.
    - le porte alle quali si collegheranno le antenne impostate come **tagged**.
    - le porte lasciate libere dalle antenne e disponibili per altri dispositivi, impostate come **untagged**.

Esempio pratico: due antenne configurate come nella sezione dedicata a AirOS, con “VLAN di traffico” rispettivamente aventi ID 3 e 4 e “VLAN di gestione” comune con ID 7. La prima antenna sarà collegata alla porta 3 dello switch, la seconda nella porta 4. Questa la situazione corrispondente, applicando le regole appena enunciate.



Cliccate su **Save**.
Ora rimuovete la VLAN di default, quella che abbiamo variato all’inizio da “tutto untagged” a “tutto tagged”, cliccando sul corrispondente **Delete** e infine facciamo **Save** per ritrovarci con questa situazione finale.



Ora andate su **Network > Interfaces**.
Cliccate **Edit** sulla Network **LAN,** e poi recatevi nel tab **Physical Settings**.
Togliete la spunta da "Ethernet Adapter" e spuntate **VLAN Interface:ethX.7**.
Cliccate su **Save** e poi su **Save & Apply**.

È interamente possibile che il router adesso abbia smesso di rispondervi. Non preoccupatevi (ancora), è normale. **Moltissimi switch richiedono un riavvio** per applicare correttamente le modifiche alle VLAN. **Spegnete e riaccendete il router**.
Assicuratevi che il cavo sia connesso ad una delle porte **non** destinate alle antenne (quelle untagged per intenderci) e dovreste di nuovo poter accedere all'interfaccia di LuCI.
Qualora abbiate 4 antenne e tutto le porte siano quindi state contrassegnate come tagged, per riguadagnare l'accesso al router bisognerà configurare la VLAN ID 7 sul proprio PC come mostrato al termine della sezione dedicata a AirOS; in alternativa potete usare la WAN-backdoor che abbiamo creato in precedenza.

In caso di problemi: se proprio lo switch non vi consente più il collegamento, potete sfruttare la backdoor WAN per accedere a LuCI e resettare il router alle impostazioni di fabbrica e ricominciare da capo.

Creazione e configurazione delle Network OpenWRT

Andate in **Network > Interface**.
Nella Network **LAN** cliccate **Edit**.
Per configurare la vostra rete locale con indirizzamento Ninux, impostate:

- **Protocol:Static address**
- **IPv4 Address: 10.X.Y.1** (primo indirizzo della LAN Ninux che abbiamo riservato per noi nelle tabelle nazionali/regionali/cittadine)
- **Netmask: 255.255.255.0**
- **Gateway:** se Ninux rappresenta il vostro gateway verso Internet impostate **10.X.Y.1**, altrimenti lasciamo vuoto.
- **Use Custom DNS Server: il nostro server DNS preferito**. Se vogliamo poter risolvere eventuali domini interni di Ninux, come *.nnx, dobbiamo inserire un DNS interno a Ninux.
- Potete ignorare tutti gli altri settaggi, a meno che non desideriate qualche personalizzazione particolare (ad esempio un particolare range DHCP).

È il momento di configurare la Network di OpenWRT relativa alla nostra antenna.
Tornate in **Network > Interface**.
Cliccate su **Add new Interface**.
In questa prima schermate, configurate con:

- **Name of the new interface: il nome che preferite**, possibilmente breve e descrittivo del link effettuato, per esempio Antenna1 o M5CasamiaCasazio.
- **Protocol of the new interface: Static address**.
- **Create a bridge over multiple interfaces**: senza spunta
- **Cover the following interface: VLAN Interface ethX.Y**, dove Y è il VLAN ID della "VLAN di traffico wireless" nell'antenna che stiamo configurando.
- Cliccate **Submit**.

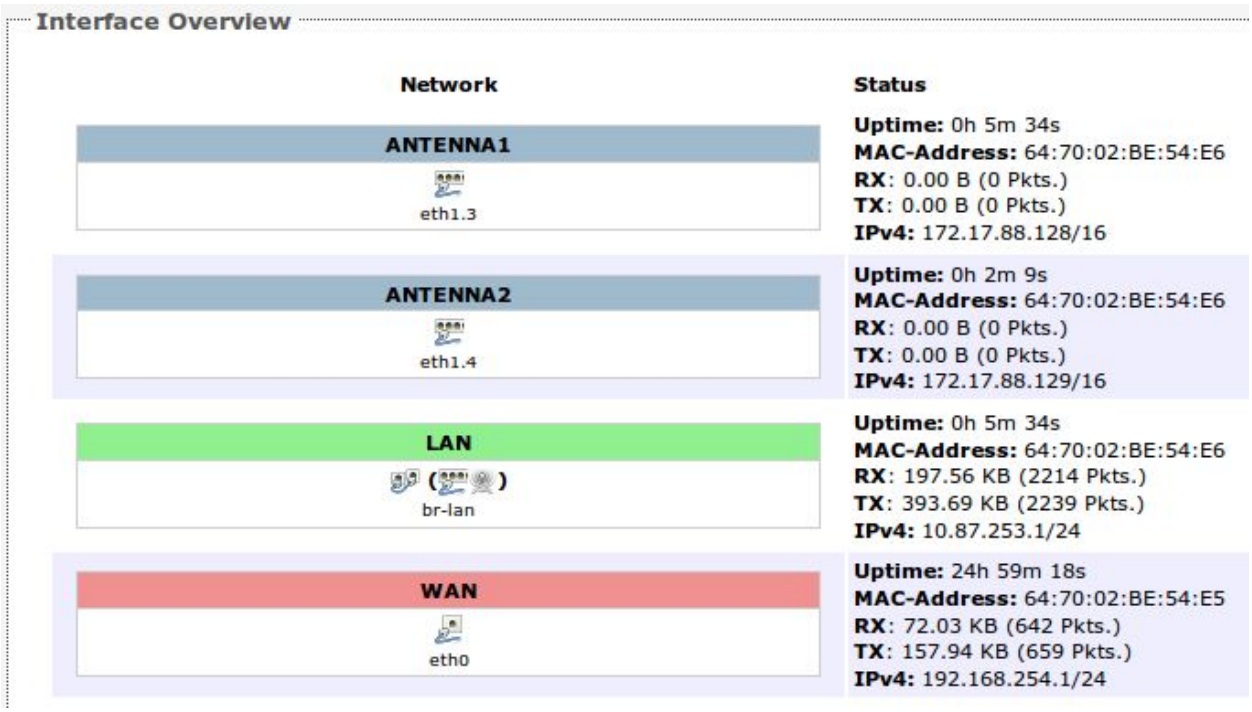
Nella nuova schermata che comparirà, configuriamo gli indirizzi della "VLAN di traffico wireless":

- **Protocol: Static address**.
- **IPv4 Address**: 172.X.Y.Z (l'indirizzo dorsale Ninux che abbiamo riservato per noi nelle tabelle nazionali/regionali/cittadine)
- **Netmask: quella riservata alla nostra isola/città/regione**. La trovate nelle tabelle in GestioneIndirizzi sul wiki di Ninux.

Andate nel **tab Firewall Settings** e nel campo **unspecified-or-create** scrivete **Ninux**.
Cliccate **Save**.

Se abbiamo altri apparati radio, **ripetiamo la procedura anche per le altre antenne**. L'unica differenza è nel passo finale, dove invece di compilare il campo unspecified-or-create selezioneremo direttamente il giù **presente Ninux come firewall zone**.
Quando avete terminato la configurazione, cliccate su **Save & Apply**.

Il router ha ora cambiato indirizzi per assumere quelli della LAN Ninux a noi assegnata. Per riguadagnare accesso all'interfaccia web del router assegniamo al nostro PC un indirizzo della sottorete 10.X.Y.Z/24. a noi assegnata (ovviamente diverso dal .1 finale, che è il nostro router).
La situazione finale assomiglierà a questa:



Routing: OLSR

Questa sezione della guida assume che la nostra build di OpenWRT (ad esempio, Scooreggione) abbia il **demone OLSR già incluso**.
Se non è così e abbiamo disponibilità di un collegamento ad Internet che il nostro router gestirà, ad esempio attraverso attraverso un modem ADSL, adesso è un buon momento per **impostare la nostra connessione ad Internet e attivare il wifi** del router. Non essendo specifiche del routing a terra questa guida non tratterà questi due step, che sono comunque estremamente semplici da realizzare nelle sezioni di LuCI **Network > Interfaces > WAN** e **Network > Wifi > Edit**.
Una volta fatto possiamo recarci in **System > Software** e aggiornare l'elenco dei pacchetti.
Cerchiamo "**olsr**" nell'apposito campo di ricerca **"Find Package"** e passiamo alla **tab "Available Packages"**.
Per installare un pacchetto clicchiamo su **Install e confermiamo**. Ripetiamo la ricerca fino ad installare i seguenti pacchetti:

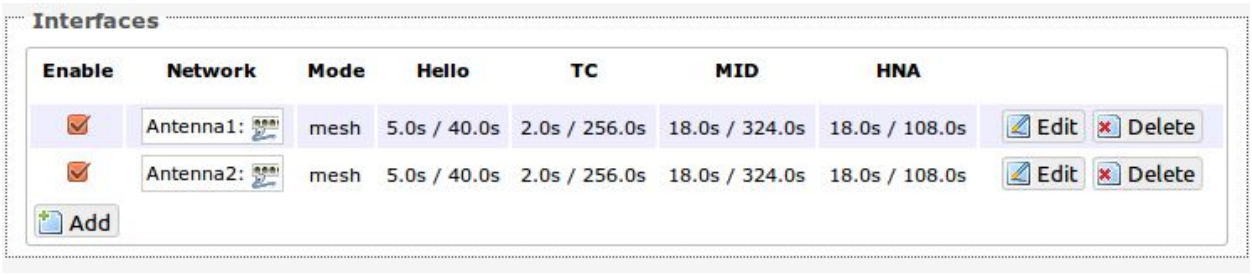
- olsrd
- olsrd-mod-txtinfo
- olsrd-mod-nameservice
- olsrd-mod-arprefresh
- luci-app-olsr

Utilizzando OpenWRT 14.07 Barrier Breaker é necessario anche il pacchetto **luci-lib-json** per il corretto funzionamento dell'interfaccia grafica di OLSR (Status > OLSR).
Una volta installati i pacchetti, spostarsi su **Services > OLSR IPv4**. menu **Plugins** ed assicurarsi che siano tutti **Enabled** (il plugin olsrd_jsoninfo.so.0.0 é di default disabilitato). Se non lo sono, abilitarli e cliccare **Save&Apply**.

Andiamo nella nuova sezione di LuCI che gestirà le opzioni del protocollo OLSR: **Service > OLSR**.

Nella sezione **"Interfaces"** in basso, alla **riga già esistente**, clicchiamo su **Edit**.
Nella sottosezione **Network**, selezioniamo **Antenna1** (o qualsiasi altro nome abbiamo dato alla nostra) e poi **Save**.
Torniamo a **Service > OLSR**. Se abbiamo altre antenne, clicchiamo su **Add** in basso e **spuntiamo la seconda antenna, Save, e così via** fino a creare un'interfaccia OLSR per ogni antenna.
Clicchiamo **Save & Apply**.

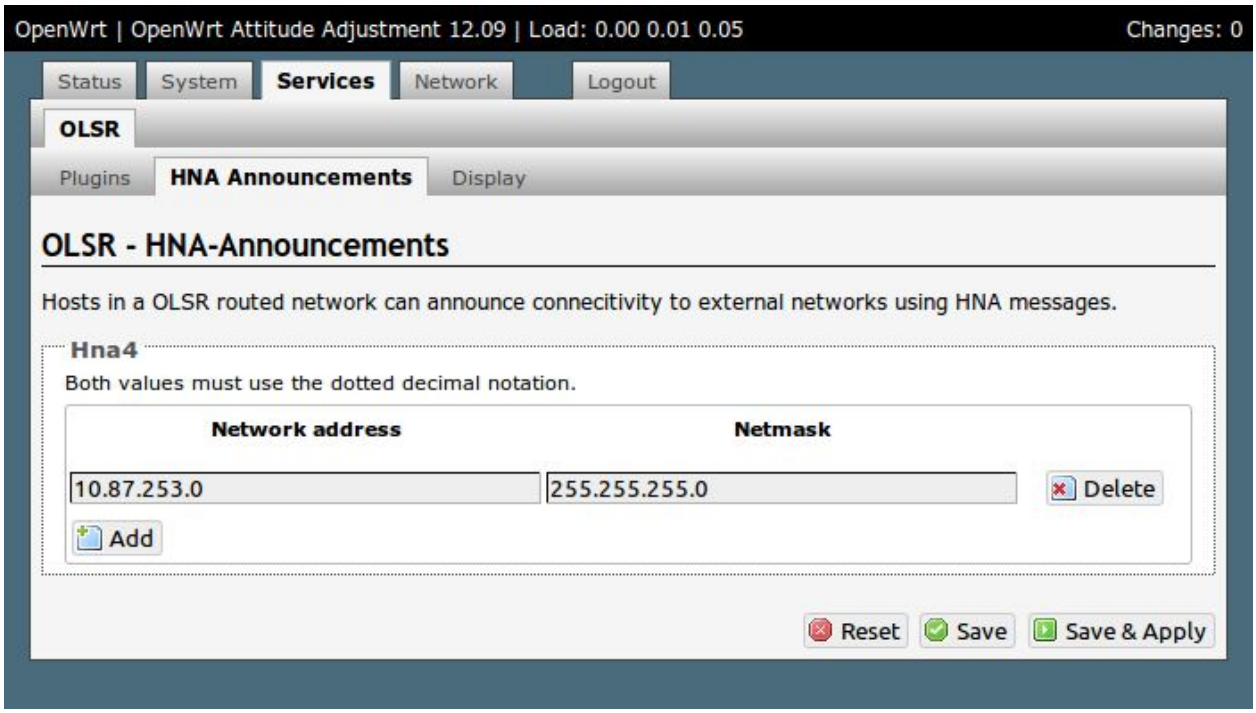
La situazione, nel nostro esempio con 2 antenne, è la seguente.



Andiamo ora in **Service > OLSR > HNA Announcements**.
Clicchiamo su Add e compiliamo i campi

- **Network Address**: quello della **subnet Ninux a noi riservata**. Assicuriamoci che l'indirizzo termini con ".0".
- **Netmask: 255.255.255.0**

Save e **Save & Apply.**



Routing: Batman

TODO

Firewalling

Si tratta di configurare le regole di Firewall per **consentire il transito del traffico Ninux**, sia quello diretto da/verso la nostra rete (Input e Output) sia quello diretto verso gli altri nodi Ninux e che transiterà per il nostro SuperNodo (Forward).

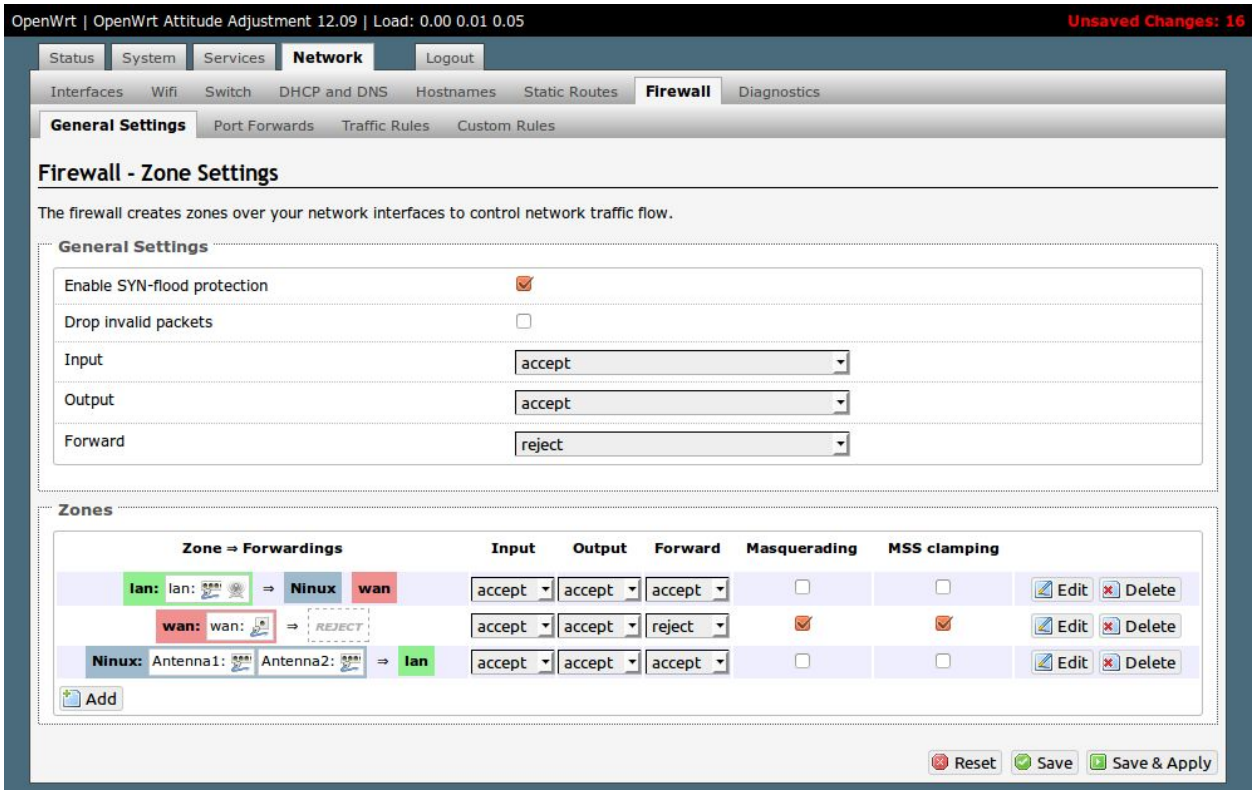
Andiamo in **Network > Firewall**.
Ora consentiremo alla nostra rete locale di comunicare col resto di Ninux.
Clicchiamo **Edit sulla riga che include le nostre antenne Ninux** della sottosezione *Zones* (sarà probabilmente nella forma **Ninux => REJECT**) e compiliamo i seguenti campi:

- **Input, Output, Forward**: tutto ad **Accept**
- **Covered Networks**: la zona che include le nostre antenne, **Ninux** nel nostro esempio.
- **Allow forward to *destination zones***: spunta su **LAN**
- **Allow forward from *sources zones***: spunta su **LAN**

Clicchiamo **Save**.

Ora è il turno di consentire al resto della rete Ninux di comunicare con noi.
In **Network > Firewall**, clicchiamo **Edit** sulla riga nella forma **LAN => Ninux + WAN**.
Variamo solamente **Forward** in **accept**.
Clicchiamo **Save**.

La situazione dovrebbe essere simile a questa:



Possiamo cliccare ora **Save & Apply**.

Passi finali

Abbiamo configurato il routing a terra! Ora non resta che configurare il resto del router normalmente, secondo le opzioni che desideriamo e rimetterlo nella sua location finale.
Assicuriamoci di **collegare ciascuna antenna alla porta dello switch a lei destinata**.
Se non l'avevamo già fatto, colleghiamoci di nuovo alle antenne ed **effettuiamo il collegamento wireless con il nodo remoto**.
Infine colleghiamoci al nostro ground router e andiamo nella sezione **Status > Routes**. Entro breve dovrebbero **apparire le rotte verso gli altri nodi Ninux** apprese attraverso il protocollo di routing che stiamo utilizzando.
Benvenuti in Ninux!

OpenWRT (CLI/UCI)

TODO

Linux Box

Questa guida tratterà il problema usando solamente ciò che si ha a disposizione su ogni macchina linux, quindi si può applicare, con qualche correzione, a tutti i dispositivi su cui possa girare il kernel linux.

Essendo questi comandi temporanei, sarà poi necessario inserirli in uno script di init. Altrimenti fare riferimento alle modalità di configurazione offerte dal sistema operativo.

Configurazione iniziale: 2 Schede di rete:
Eth0 Scheda di rete Gigabit verso Tetto
Eth1 Scheda di rete per la distribuzione in casa

2 Antenne con traffico OLSR sulle Vlan **2** e **3** e managing sulla vlan **7**

Creazione VLAN:
ip link add link eth0 name eth0.**2** type vlan id **2**
ip link add link eth0 name eth0.**3** type vlan id **3**
ip link add link eth0 name eth0.**7** type vlan id **7**

ip addr add 172.X.Y.Z/16 brd 172.X.255.255 dev eth0.**2**
ip link set dev eth0.**2** up

ip addr add 172.X.Y.W/16 brd 172.X.255.255 dev eth0.**3**
ip link set dev eth0.**3** up

Creazione del Bridge tra managing e rete di casa
brctrl addbr br0
brctrl addif br0 eth0**.7**
brctrl addif br0 eth1

ip addr add 10.X.Y.Z/24 dev br0
ip link set dev br0 up

Dopo di chè dovremo installare olsrd, dai repository di debian o usando un binario precompilato da qualcun'altro.

E' possibile installare OLSRD dai repo, ma essendoci solitamente versioni vecchie di OLSRD è consigliato scaricare un binario già compilato o compilarlo in locale.

Una volta installato OLSDR dovemo andare ad editare il file /etc/olsrd/olsrd.conf per configurare il demone di routing:

Questo è uno scheletro del file di configurazione con solamente le istruzioni essenziali

```
DebugLevel 0
IpVersion 4

#RtTable 111
#RtTableDefault 112

#SmartGateway yes
#SmartGatewayUplinkNAT yes
#SmartGatewayUplink "ipv4"

Hna4
{
10.X.Y.0 255.255.255.0
}

#In caso si voglia dare connettività verso internet
#Hna4
#{
#0.0.0.0 0.0.0.0
#}

LoadPlugin "/usr/local/lib/olsrd_httpinfo.so.0.1"
{
PlParam "port" "8080"
PlParam "Net" "0.0.0.0 0.0.0.0"

}

LoadPlugin "/usr/local/lib/olsrd_txtinfo.so.0.1"
{
        PlParam   "port"   "2006"
        PlParam "accept" "0.0.0.0"
}

LoadPlugin "/usr/local/lib/olsrd_mdns.so.1.0.1"
{
   PlParam "NonOlsrIf" "br0"
```

```
}

#Interface "br0"
#{
#Mode "ether"
#}

Interface "eth0.2"
{
Mode “mesh”
}

Interface "eth0.3"
{
Mode “mesh”
}
```

Nel file di configurazione ci aggiungeremo un blocco

```
 Interfaces “Nomeinterfaccia”
{
Mode “mesh”
}
```

Ogni direttiva particolare per questa interfaccia andrà aggiunta all’interno delle parentesi graffe.

A questo punto se abbiamo usato l’insallazione dai repo dobbiamo editare il file /etc/default/oslrd per avviare olsrd insieme al router, se invece abbiamo usato un binario di OLSRD di terze parti dovremo crearci uno script di init per avviarlo all’avvio.

EdgeOS (Ubiquiti EdgeRouter)

Per utilizzare la confgurazione di routing a terra con uno switch edgerouter è fortemente consigliato l’uso di uno switch addizionale, managed o non. Si consigliano i modelli Netgear gs105/8 [Plus] o EdgeSwitch POE.

Collegate le varie antenne allo switch, e una porta dello switch alla porta 1(eth0) del nostro edgerouter. Dovremo configurare le vlan per le varie antenne.

Su EdgeOS è possibile creare interfacce VLAN sia dalla modalità grafica che da linea di comando.
Prima di eseguire una qualsivoglia configurazione con questo prodotto è consigliabile eseguire sia un’immagine completa del supporto di memorizzazione dell’edgerouter, che è una semplicissima

memoria usb, sia eseguire un backup della configurazione di partenza dello stesso.
L'edgerouter appena acquistato parte con una configurazione tale che risponde all'indirizzo 192.168.1.1/24 tramite la porta eth0.
Per eseguire l'immagine della pendrive usb dovete smontare il router stesso svitando le 3 viti nere posteriori e far scorrere la parte superiore ed inferiore in senso opposto.
La scheda madre è avvitata al pannello inferiore del router stesso.
Una volta rimossa la pendrive usb del router, collegarla ad un pc con sistema operativo linux, ad esempio, ed andando a controllare tramite il comando dmesg il nome del device assegnato alla memoria usb, possiamo eseguirne la copia bit a bit andando ad eseguire il seguente comando (assumiamo ad esempio che la nostra distro ha riconosciuto la memoria usb come /dev/sdb):

dd if=/dev/sdb of=/home/<mioUtenteLinux>/EdgeRouterImg.img bs=16M

Visto e considerato che il router di default è configurato con un indirizzo ip che generalmente è assegnato nei router di vari providers internet conviene collegarlo direttamente al nostro pc configurando opportunamente la nostra schede di rete utilizzando la stessa subnet ed un ip a lui adiacente.
Una volta configurato il nostro pc per parlare con l'EdgeRouter Ubiquiti posssiamo iniziare a lavorarci sopra.
Per prima cosa eseguiamo il backup della configurazione del router; una volta acceduto al router tramite l'indirizzo di default 192.168.1.1 con l'utenza standard ubnt/ubnt (nome utente/password), clickare in basso a sinistra su System e poi in basso nella sezione "Configuration Management & Device Maintenance" nella sotto sezione "Back Up Config" clickare su Download ed il router ci farà scaricare un file compresso che contiene sostanzialmente tutta la cartella /config del router.

Per prima cosa ci conviene cambiare utente di default del router in modo tale che soltanto noi possiamo accederci. I comandi da dare sono i seguenti per creare ad esempio un utente con la seguente login (pippo/paperino rispettivamente per username/password):

configure
set system login user pippo authentication plaintext-password paperino
commit
save
exit
exit

Ora possiamo cancellare il vecchio account ubnt/ubnt e per farlo dobbiamo fare la login con il nuovo utente, nel nostro caso pippo/paperino e poi diamo i seguenti comandi :

configure
delete system login user ubnt
commit
save
exit
exit

A questo punto abbiamo sul nostro sistema esclusivamente l'utente che abbiamo precedentemente creato. Nel caso qualche operazione sia andata male e non riuscite ad andare avanti ma riuscite

graficamente ad accedere alla voce System in basso a sinistra, potete ricaricare la configurazione originale che vi siete precedentemente salvati sul pc e ricaricarla sul router per ripristinare il tutto (spero che abbiate saltato il passaggio del backup e men che meno quello dell'immagine del sistema !).

La configurazione finale del router sarà tale che su ogni porta sarà presente una subnet differente ad esempio su eth0 possiamo far arrivare la subnet delle antenne, mentre la porta centrale (eth1) sarà una porta di wan verso il nostro router internet ed infine sulla eth2 possiamo mettere dei servizi annunciati dentro ninux.
Per configurare correttamente la porta eth1 è necessario sapere che subnet il router del nostro gestore internet annuncia; se ad esempio annuncia la 192.168.1.0/24 (generalmente annunciano sempre o la 192.168.1.0/24 oppure la 192.168.0.0/24), cioè assegna indirizzi ip dal 192.168.1.2 al 192.168.1.254 in quanto il 192.168.1.1 è lui stesso, possiamo senza problemi assegnare staticamente alla eth2 un indirizzo ad esempio 192.168.1.50, indirizzo che ovviamente non è utilizzato da nessun altro device della nostra rete.
Per poterlo configurare colleghiamo un cavo di rete sulla eth0, impostiamo la nostra scheda di rete con un indirizzo ip della subnet 192.168.1.0/24 ad esclusione di 192.168.1.1 che è l'indirizzo di default con cui è configurato di fabbrica l'edgerouter, ed colleghiamoci all'indirizzo 192.168.1.1 (come potrete notare già ora avremo una collisione se collegassimo l'edgerouter alla nostra lan del gestore internet) ed utilizzando la console e loggandoci con l'utenza precedentemente impostata scriviamo :

configure
set interfaces ethernet eth1 address 192.168.1.50/24
commit
save

Ora togliamo il cavo dall'eth0 e colleghiamoci all'eth1 per proseguire con la configurazione.

Ora dobbiamo configurare le subnet /24 per la eth0 e la eth2.
Se come subnet libere possiamo utilizzare, ad esempio la 10.150.0.0/24 e la 10.150.1.0/24 possiamo rientrare nel router e dare i seguenti comandi dalla console :

configure
set interfaces ethernet eth0 address 10.150.0.1/24
set interfaces ethernet eth2 address 10.150.1.1/24
commit
save

Volendo per entrambe le subnet possiamo impostare dei DHCP server in modo tale che qualsiasi device connesso che non abbia una configurazione ip, il router gli possa assegnare un indirizzo andiamo graficamente su Services e poi impostiamo i DHCP server che vogliamo impostare andando anche a scegliere il range di ip assegnabili.

Entriamo in modalità configurazione e creiamo le vlan per i flussi di traffico

configure
set interfaces ethernet eth0 vif #nvlan1 address 172.1x.y.z/16
set interfaces ethernet eth0 vif #nvlan2 address 172.1x.y.z/16
commit

save

Al posto di #nvlan1/#nvlan2 dobbiamo mettere la vlan impostata sull'antenna, quindi se abbiamo creato una vlan 4 sull'antenna 4 e la vlan arriva dalla eth0 e come indirizzo sulla 172.16.0.0/16 che ci siamo scelti è 172.16.150.4, l'impostazione da dare correttamente sarà :

set interfaces ethernet eth0 vif 4 address 172.16.150.4/16

Dopo di chè dovremo installare olsrd, dai repository di debian o precompilato a parte.

Su EdgeOS, essendo basato su Debian, è possibile installare i pacchetti con apt-get da qualsiasi repository Debian.
Ogni versione di firmware di EdgeOS è compatibile con una ben precisa versione di Debian.
Ecco un elenco :

- Vers. 1.6.0 : compatibile con Debian Squeeze
- Vers. 1.7.0 : compatibile con Debian Wheezy
- Vers. 1.8.0 : compatibile con Debian Wheezy
- Vers. 1.8.5 : compatibile con Debian Wheezy
- Vers. 1.9.0 : compatibile con Debian Wheezy

E' possibile installare OLSRD dal repo
http://test.ninux.org/~clauz/edgerouter/
oppure lo si può compilare in locale.

Per installare OLSRD da repository Debian, eseguire i seguenti comandi
per diventare root

sudo -s -H

per impostare set apt sources.list (se abbiamo ad esempio un qualunque firmware con pacchetti compatibili con la distribuzione Gnu/Linux Debian Wheezy):

echo 'deb http://ftp.de.debian.org/debian/ wheezy main contrib non-free' > /etc/apt/sources.list

aggiorna apt ed installa il pacchetto olsrd

export DEBIAN_FRONTEND=noninteractive
apt-get update && apt-get -y --force-yes install olsrd

Un consiglio spassionato : **mai mai dare il comando apt-get upgrade** in quanto ci sono un certo numero di pacchetti customizzati per edgerouter che se venissero aggiornati con quelli del repository renderebbero il sistema o non utilizzabile o instabile (Fonte Ubiquiti!).

A questo punto se abbiamo usato l'installazione dai repo dobbiamo editare il file /etc/default/olsrd per avviare olsrd insieme al router. All'interno di tale file si trova la riga :

#START_OLSRD="YES"

eliminare il cancelletto

START_OLSRD="YES"

e salvare.
Se invece abbiamo usato un binario di OLSRD di terze parti dovremo crearci uno script di init per avviarlo all'avvio.
Per la configurazione di OLSRD vedere la sezione "Linux Box"

//TODO inserimento configurazione olsr in caso di installazione pacchetti precompilati non provenienti da repository

pfSense

Configurazione delle VLAN

TODO

Considerazioni premilinari su interfacce e indirizzi

Qundo si configurano le interfacce "wireless" per le antenne bisogna tenere in considerazione quanto segue (si prende come esempio la subnet 172.21.x.x assegnata alle zona Marche, riferire gli esempi alla subnet di zona):

ogni interfaccia ha un indirizzo ip del tipo 172.21.x.x
ogni interfaccia deve essere parte di una sottorete diversa altrimenti pfSense crea un aggregato di rete con un'unica interfaccia come gateway per tutti gli ip della stessa sottorete. A differenza di altri router, pfSense non è in grado di gestire questa situazione (a meno di impostare regole specifiche) neanche quando il routing viene gestito da OLSR.
Per link punto punto si puo' optare per una sottorete /30 ad es: 172.21.1.1/30 che formerà un link con l'antenna remota facente parte della stessa sottorete e avendo ip: 172.21.1.2/30
da notare che questa rete 172.21.1.0/30 contiene I seguenti indirizzi:
172.21.1.0 network
172.21.1.1 host
172.21.1.2 host
172.21.1.3 broadcast
quindi solo .1 e .2 sono host.
allo stesso modo la sottorete successiva, cioè 172.21.1.4/30 sarà:
172.21.1.4 network
172.21.1.5 host
172.21.1.6 host
172.21.1.7 broadcast
con .5 e .6 come host e quindi utilizzabili per le interfacce delle antenne...
...e così via.

Un esempio concreto:

[router A] [router B] [router C]
172.21.1.1/30 <---subnet[01]---> 172.21.1.2/30
172.21.1.5/30 <---subnet[02]---> 172.21.1.6/30

Configurazione delle interfacce

TODO

Routing: OLSR

informazioni riguardo il demone olsrd in pfSense 2.1.3 (FreeBSD 8.3.-p16):
*** olsr.org - 0.6.3-git_-hash_ ***
Build date: 2012-12-20 on packages-8_3-i386.builders.pfsense.org

installare il paccjetto olsrd da:
system → packages → available packages

start:
/usr/local/etc/rc.d/olsrd start

status :
/usr/local/etc/rc.d/olsrd start

Avviare il demone – operazioni preliminari

IMPORTANTE:
per poter avviare il demone e' necessario editare (con ee) /etc/rc.conf.local aggiungendo la riga:
olsrd_enable="YES"

(per poter far ciò il disco deve essere in modalita' rw - To remount file systems as read-write, run: /etc/rc.conf_mount_rw)

ATTENZIONE: il file rc.conf viene cancellato ad ogni riavvio di pfSense.
utilizzare /etc/rc.conf.local

avvio automatico di OLSRD

creare i file rc.local e rc.local.running:
touch /etc/rc.local
touch /etc/rc.local.running

(dopo l'avvio il sistema richiede di poter accedere a questo file)

editare il file rc.local:
ee /etc/rc.local

inserendo la riga:
/usr/local/etc/rc.d/olsrd start

salvare, chiudere, riavviare

file di configurazione

ATTENZIONE VERIFICARE:
se si configura il demone tramite l'interfaccia grafica di pfSense per il servizio OLSRD, il file di configurazione verrà salvato in:
/var/etc/olsr.conf

mentre il demone cerca il file di configurazione in:
/usr/pbi/olsrd-i386/etc/olsrd.conf

pertanto è consigliabile non utilizzare l'interfaccia grafica ma editare il file manualmente:
ee /usr/pbi/olsrd-i386/etc/olsrd.conf

(ee é l'editor di FreeBSD)

*** Configurazione di OLSRD ***

Abilitare interfacce

definire le interfacce usate da olsrd con il parametro Interface:

Interface “[interfaccia 1]” “[interfaccia 2]” “[interfaccia ...]”

le interfacce devono essere definite nella forma richiesta da FreeBSD, ad es:
re0, vr0, ath0, ...

e nel caso delle vlan:
re0_vlan10, re0_vlan11, ...

fare riferimento ai manuali di pfSense e al pannello di controllo per definire correttamente il nome delle interfacce.

Selezionare le reti host da annunciare tramite Hna4
esempio:

Hna4
{
10.21.1.0 255.255.255.0
0.0.0.0 0.0.0.0
}

10.21.1.0 255.255.255.0 identifica la rete (esempio) presente nell'host che viene annunciata da olsrd

Dettagli sulla connettività ad internet

il router connesso ad internet (e quindi con connessione WAN) deve annunciare la propria connettività tramite OLSR. Infatti nel file di configurazione olsr.conf, alla voce Hna4, oltre ad avere 10.21.1.0/24 (la rete di se stesso) dovremmo impostare 0.0.0.0/0.

0.0.0.0 0.0.0.0 indentifica che in questo host e' presente la connessione ad internet e puo' essere usato come gateway. (da impostare solo se necessario)

in questo modo OLSR annuncia ai router collegati che tramite la sua rete wireless (172.21.x.x/30) è possibile raggiungere internet.

Nei router “neighbour” verra' annunciato un default gateway attraverso l'ip dell'antenna che fa parte della rete wireless.

Abilitare plugin server web httpinfo

inserire questa parte nel file di configurazione:

```
# Example plugin entry with parameters:

LoadPlugin "/usr/local/lib/olsrd_httpinfo.so.0.1"
{
        PlParam "port" "88"
        PlParam "Net" "192.168.0.2 255.255.255.0"
}

# PlParam "port" #### e' la porta sulla quale il server risponde
# PlParam "Net" "192.168.0.2 255.255.255.0" #### sono gli host ammessi
```

NOTA: in /usr/local/lib/ si trovano i plugin per olsrd, denominati come olsrd_[nome plugin]

Abilitare plugin txtinfo
(info: http://www.olsr.org/?q=txtinfo_plugin)

il plugin permette di visualizzare un file txt da browser o da terminale .
aggiungere queste righe al file di configurazione:

```
LoadPlugin "olsrd_txtinfo.so.0.1"
{
        PlParam "port" "89"

        PlParam "accept" "127.0.0.1"
        PlParam "accept" "192.168.0.2"
}
```

qui una spiegazione sui parametri:

the default port is 2006 but you can change it like this:
#PlParam "port" "8080"
You can set a single address to allow to connect to txtinfo. If no address
is specified, then localhost (127.0.0.1) is allowed by default.
#PlParam "accept" "127.0.0.1"
#PlParam "accept" "172.29.44.23"
if you set it to 0.0.0.0, it will accept all connections
#PlParam "accept" "0.0.0.0"

- per visualizzare da browser tramite httpinfo digitare ill'ip del router e la porta, es:

192.168.0.1:88

- per visualizzare da terminale (using txtinfo with netcat – nc):

$ echo "/all" | nc localhost [port]

Ricetta PfSense: configurare il NAT in un router con internet (Supernodo con WAN)

prendiamo il [router B] dell'esempio precedente; di default, ed in automatico, pfsense crea una regola di NAT per ogni interfaccia che non sia WAN, verso la WAN, ad es:

WAN – 172.21.1.0/30 – WAN address
(per ogni 172.21.1.0/30 che passa per interfaccia “WAN” assegnare un ip “WAN adrress”)

WAN – 172.21.1.4/30 – WAN address

WAN – 10.21.1.0/24 – WAN address

WAN – 127.0.0.0/8 – WAN address

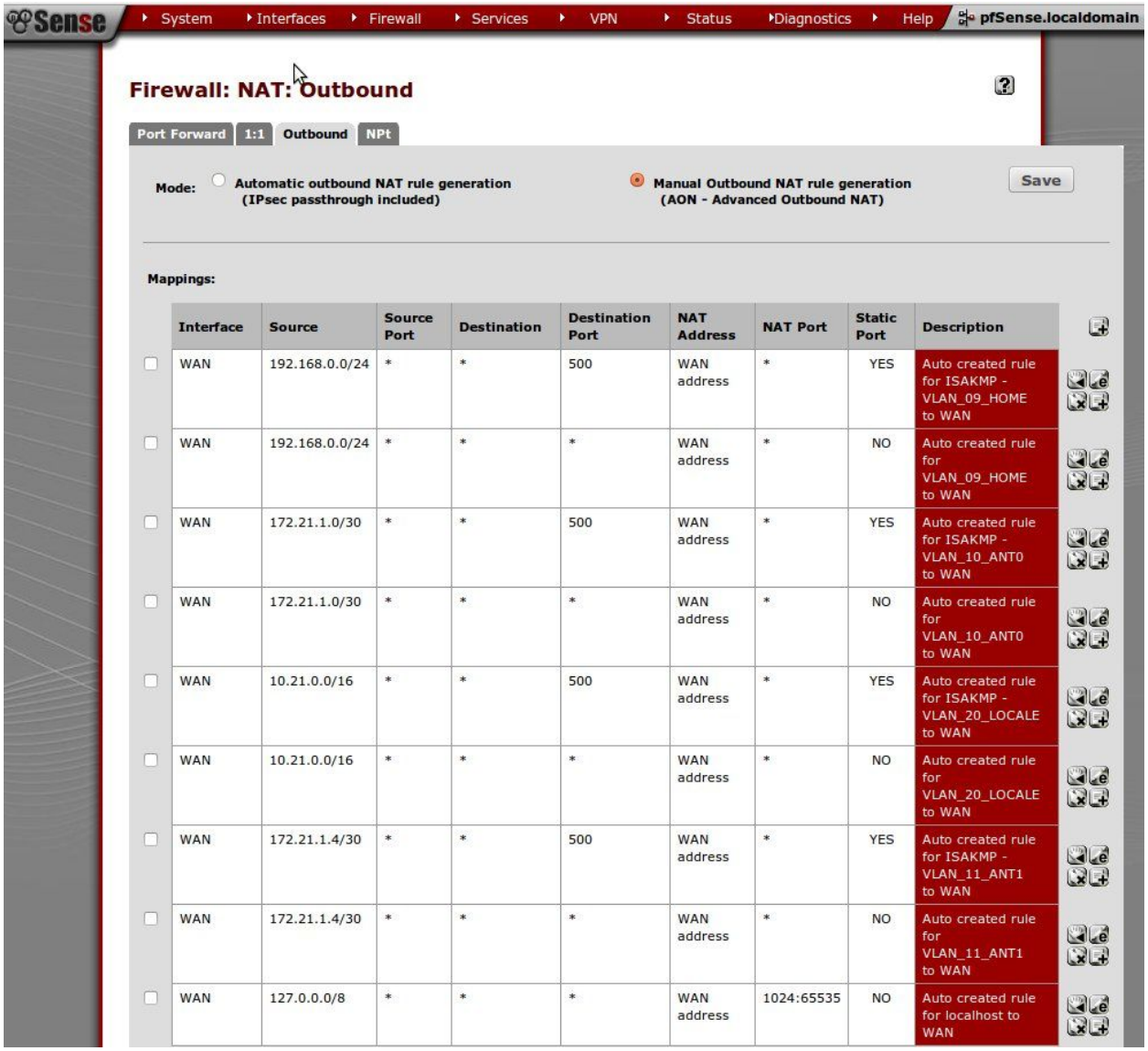
per prima cosa assicuriamoci che ogni subnet del tipo 172.21.1.x/30 abbia la propria regola nel NAT, in modo che ogni richiesta che proviene dalla rete interna viene passata attraverso il NAT mandata in internet tramite WAN.

Poi, avendo configurato la rete 10.21.1.x con subnet /24 sappiamo che non potranno passare sotto NAT indirizzi del tipo 10.21.2.x in quanto facenti parte di una sottorete diversa.

Per includere nella regola del NAT tutti gateway Hna4 di tipo 10.21.x.x che verranno annunciati in rete da OLSR, si imposta /16 come subnet nella regola:

WAN – 10.21.0.0/16 – WAN address

in questo modo tutti I 10.21.1.x...10.21.2.x...10.21.n.x soddisferanno la regola del NAT.



Ricetta PfSense: configurare il NAT in un router senza internet (nodo foglia)

un router in cui non é presente un'interfaccia WAN non presenterà un default gateway; tramite il protocollo OLSR il router vedrà le connessioni ad internet (se presenti) proposte dai router vicini e le imposterà come default gateway

In particolare nella tabella di routing troveremo:

destination gateway
default 172.21.1.x

con 172.21.1.x che è l'indirizzo ip del router che annuncia la connettività ad internet.

PfSense però non crea automaticamente una regola di NAT per le reti che passano attraverso

WIRELESS.

Il traffico di una rete che sta “dietro” a WIRELESS, ad esempio 192.168.1.0/24 che non e' direttamente annunciata da OLSR (come invece avviene per 10.21.x.x/24) deve passare per il NAT interno del router, sia per connettersi ad un qualsiasi host delle reti interne (172.21.x.x che 10.21.x.x) sia per uscire in internet.

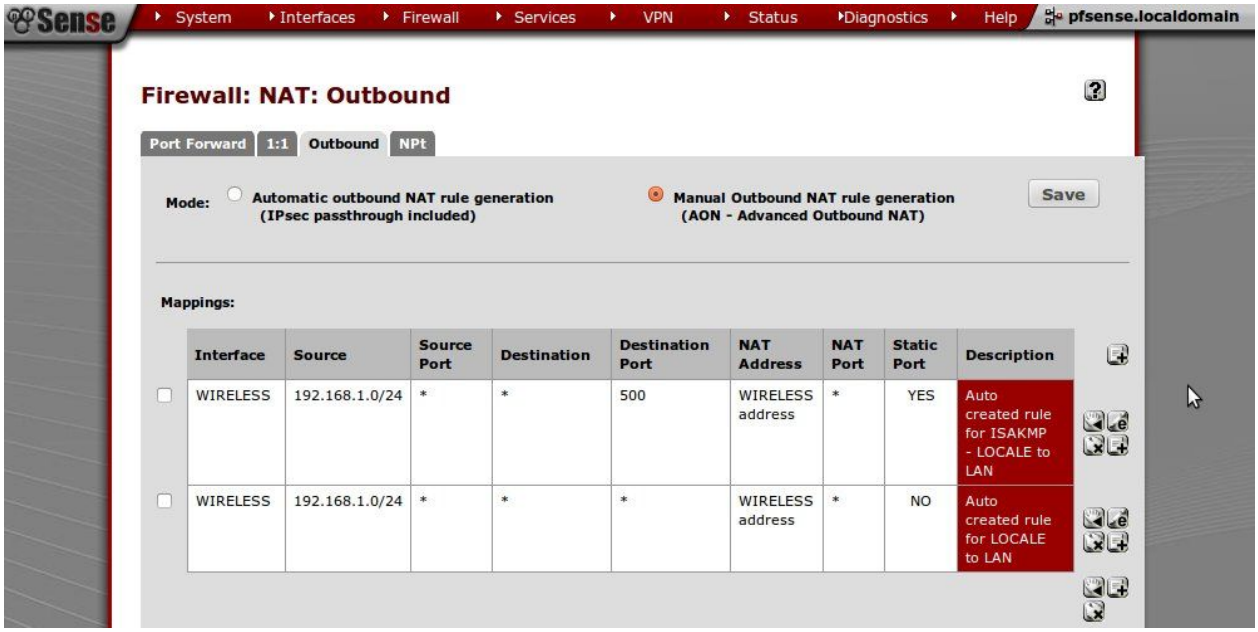
Per far cio' nel router bisogna impostare alcune specifiche regole di NAT:

WIRELESS – 192.168.1.0/24 – WIRELESS address

(per ogni 192.168.1.0/24 che passa per interfaccia “WIRELESS” assegnare un ip “WIRELESS adrress”)

si noti che l'interfaccia WIRELESS ha indirizzo del tipo: 172.21.x.x.
in questo modo ogni richiesta fatta da 192.168.1.x viene mascherata come 172.121.1.x.

(NOTA: verifica NAT su porta 500)



Controllare OLSRD durante il funzionamento

durante la normale attività può accadere che OLSRD si fermi da solo.
Occorre quindi controllare che sia attivo ed eventualmente farlo ripartire:

si utilizza cron...

script posizionato in **/usr/sbin/olsrd-check**
quindi:

```
ee /usr/sbin/olsrd-check
```

si apre l'editor e si incolla questo:

```
#!/bin/sh

if pgrep olsrd
then
echo "olsrd is alive"
else
echo "restarting olsrd"
/usr/local/etc/rc.d/olsrd start
fi
```

lo script deve avere permessi di esecuzione quindi:

```
chmod +x /usr/sbin/olsrd-check
```

a questo punto bisogna eseguire lo script in automatico come operazione pianificata (scheduled)

il modo più semplice é aggiungendo un'operazione alla tabella di **cron.**
(fare riferimento al manuale di cron per ulteriori informazioni:
http://www.freebsd.org/doc/en/books/handbook/configtuning-cron.html)

Cron é un servizio che permette di fare operazioni pianificate scritte in un file con apposita formattazione. Nello specifico aggiungere la seguente riga al file **/etc/crontab:**

```
*       *       *       *       *       root    usr/sbin/olsrd-check
```

in questo modo cron eseguirà lo script una volta al minuto e se necessario riavvierà OLSRD.

P.S. pfSense ha il pacchetto Cron (beta) installabile con interfaccia grafica. Il pacchetto fornisce solo l'interfaccia di gestione di crontab in quanto il servizio cron é già presente in pfSense (e nei sistemi Linux o FreeBSD in generale).

Policy Routing

Editare il file /etc/rc.local ed inserire il seguente codice in fondo:

```
#110 Local routes
#111 RtTable
#112 RtTableDefault
#113 Special Table for /1
#114 blackholes table

#Copy local routes only from table main 254 to table 110
ip route show table 254 | grep -Ev ^default | grep -Ev ^blackhole |
```

```
grep -v ath0 | while read ROUTE ; do
ip route add table 110 $ROUTE
done

#First evaluate local routes
ip rule add from all lookup 110 pref 3

#Private routes to OLSR table
ip rule add to 10.0.0.0/8 table 111 pref 4
ip rule add to 172.16.0.0/12 table 111 pref 4
ip rule add to 192.168.0.0/16 table 111 pref 4

#Evaluate blackholes
ip rule add from all table 114 pref 5

#Lookup default route first from user and then from OLSR
ip rule add from all lookup 254 pref 7
ip rule add from all lookup 112 pref 8

#Blackhole private aggregates
ip route add blackhole 10.0.0.0/8 table 114
ip route add blackhole 172.16.0.0/12 table 114
ip route add blackhole 192.168.0.0/16 table 114

#Ninux IP Addresses to OLSR table
#Decommentare Solo in caso si facciano passare gli ip pubblici ninux all'interno della rete (es bgp vpn)
#ip rule add to 176.62.53.0/24 table 111 pref 4
#Send traffic of public addresses to BGP border routers
#ip rule add from 176.62.53.0/24 table 113 pref 6
#Blackhole Ninux aggregate
#ip route add blackhole 176.62.53.0/24 table 114
```

Troubleshooting

In caso di crash di OLSR

(testato su TP-Link WDR4300 - OpenWrt Barrier Breaker 14.07)

Se il processo OLSR termina inaspettamente, tutta l'attività di routing smette di funzionare.
Occorre quindi controllare che sia attivo ed eventualmente farlo ripartire:

si utilizza cron...
script posizionato in
/usr/sbin/olsrd-check

quindi:

vi /usr/sbin/olsrd-check

si apre l'editor e si incolla questo:

```
#!/bin/sh

if pgrep olsrd
then
echo "olsrd is alive"
else
echo "restarting olsrd"
/etc/init.d/olsrd start
fi
```

lo script deve avere permessi di esecuzione quindi:

chmod +x /usr/sbin/olsrd-check

a questo punto bisogna eseguire lo script in automatico come operazione pianificata (scheduled), quindi accedere all'interfaccia grafica di OpenWrt: System -> Scheduled Tasks

aggiungere la riga:

*** * * * * /usr/sbin/olsrd-check**

poi [submit]

Cron é un servizio che permette di fare operazioni pianificate scritte in un file con apposita formattazione, in questo modo cron eseguirà lo script una volta al minuto e se necessario riavvierà OLSRD.
(https://www.debian-administration.org/article/56/Command_scheduling_with_cron)

FAQ

TODO

Riferimenti

TODO