

Hálózatok építése és üzemeltetése

OpenWRT

Mai téma

- ▶ Hálózati eszközök belső felépítése
 - ▶ egy konkrét példán keresztül
- ▶ OpenWRT
- ▶ Edge, core routerek

OpenWRT történet

OpenWRT koktélok

- ▶ OpenWRT: egy GNU/Linux disztribúció
 - ▶ beágyazott rendszerekhez
 - ▶ “főleg” wireless routerekhez
- ▶ Történet
 - ▶ Linksys WRT54G eszköz: nyílt lett a forráskód (GPL)
 - ▶ 2004
 - ▶ ezekre a GPL-es forrásokra
 - ▶ és az uclibc projektből átvett buildroot környezetre alapozva megjelent az első ún. “stable release”
 - ▶ buildroot környezet: különböző beágyazott rendszerekre (melyek különböző architektúrára és CPU-ra épülnek) tudjunk operációs rendszert / firmware-t “készíteni”
 - funkciók: keresztfordítás (cross-compilation), a root filesystem és a megfelelő kernel generálása, bootloader image létrehozása

OpenWRT koktélok

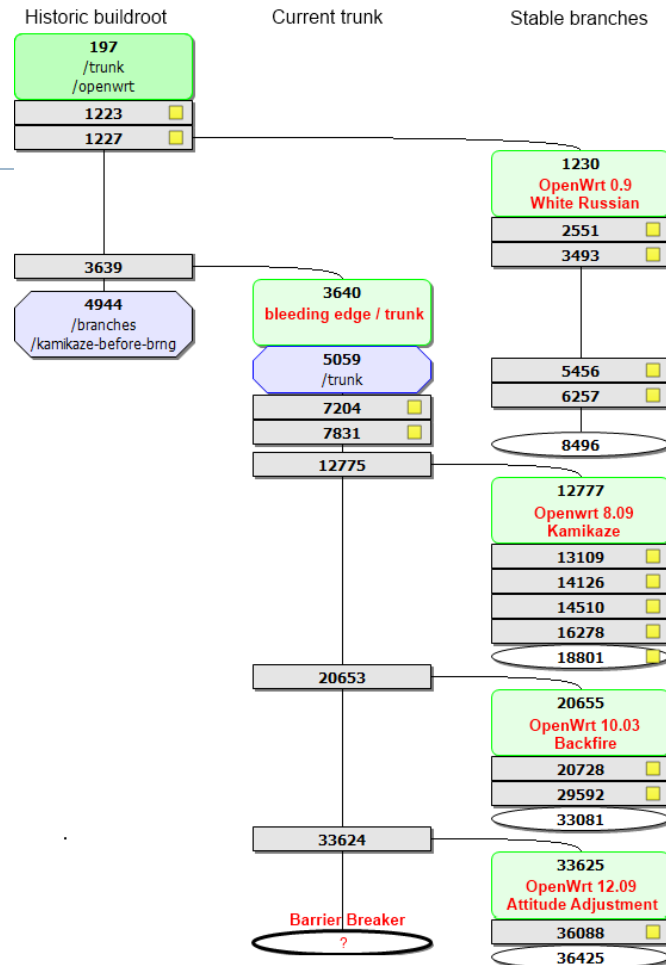
► Történet

► 2005

- sok új fejlesztő csatlakozott a projekthez
- megjelentek az ún. “experimental” verziók
- ennek eredménye lett az első önálló OpenWRT verzió
- ami egy népszerű koktélról a **White Russian** kódnevet kapta
 - a kódnevekkel azóta is követik ezt a jó szokást
 - + indulásnál az éppen aktuális koktél részletes receptjét is megkapjuk (cat /etc/banner)

► 2006 augusztustól

- build környezet alapvető fejlesztése
- eredmény: **Kamikaze** első verziója (majd számos kiadása 2010-ig)



OpenWRT koktélok

► Történet

► Verziók

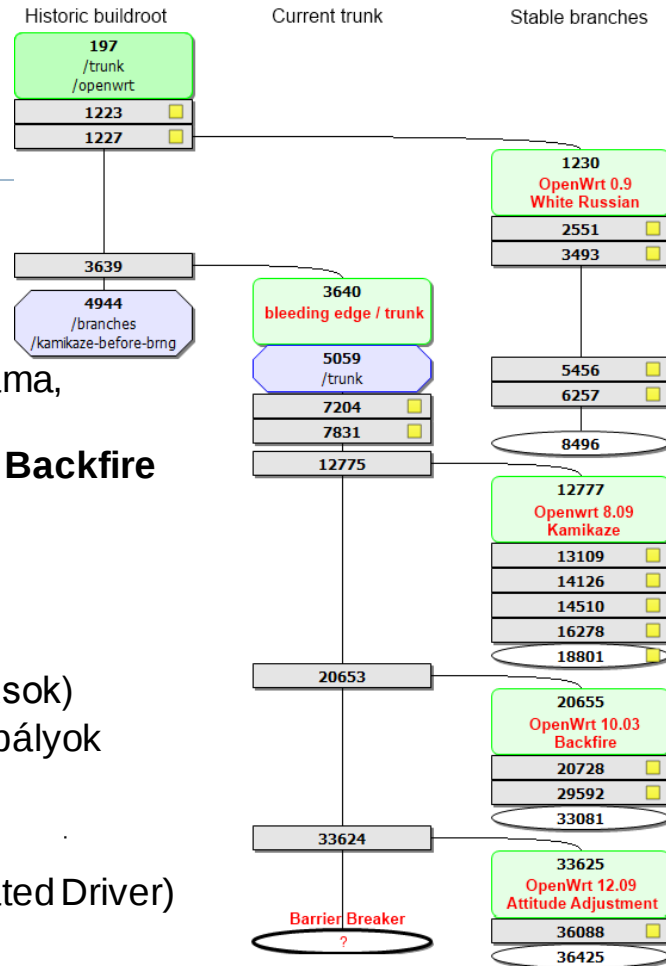
- fő verziószám helyett kódnevek
- ezt egészítik ki különböző számokkal: első kiadás évszáma, hónap [, service release, release candidate sorszám]
- első sokak által használt, sok eszközt támogató verzió a **Backfire** 10.03-as

► 2016-2018: OpenWRT fork: LEDE

- Linux Embedded Development Environment
- Oka: nézeteltérés a fejlesztési folyamatokban (valójában kisebbség kezében voltak a központi erőforrások)
- svn → git, uClibc → musl, demokratikusabb döntési szabályok

► Jelenleg

- OpenWrt név, LEDE fejlesztési modell
- Nincsenek kódnevek a release-ben (elmaradt a Designated Driver)
- stabil verzió: 18.06

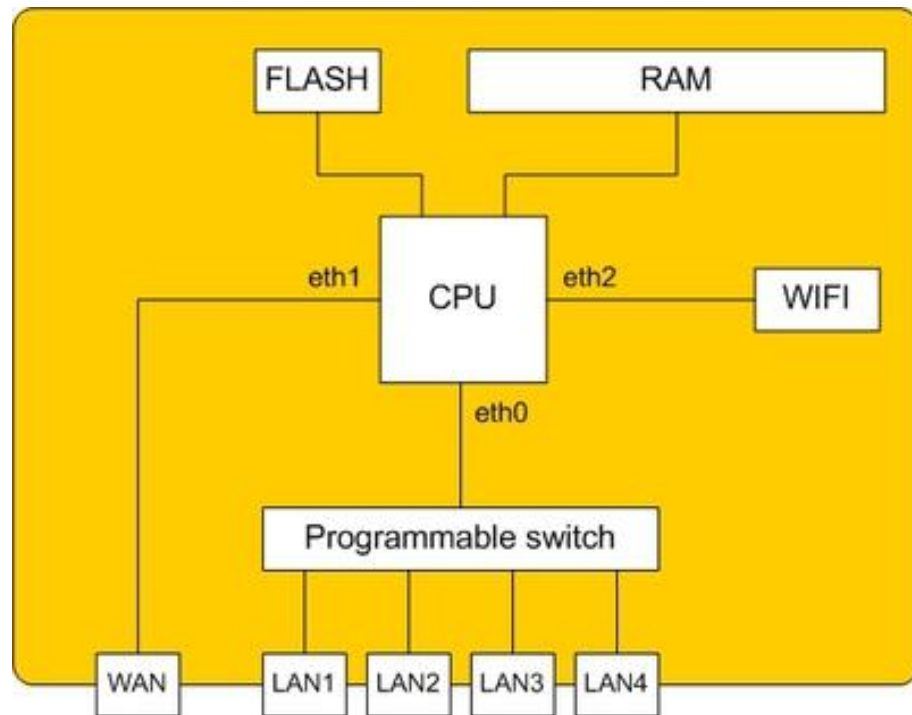


WiFi routerek felépítése



WiFi router (egy lehetséges) felépítése

- ▶ Alacsony teljesítményű CPU
 - ▶ hálózati interfészekkel
- ▶ kisebb flash memória perzisztens adatok tárolására
 - ▶ MTD (Memory Technology Device)
 - ▶ speciális “block device”
 - ▶ (raw flash devices)
- ▶ nagyobb RAM
- ▶ programozható switch (chip)
- ▶ WiFi chip
- ▶ egyéb perifériák
 - ▶ pl. USB eszközök
- ▶ soros interfész, JTAG...
 - ▶ ha baj van



OpenWRT részletek

Partíciók, fájlrendszerek

Partíciók, fájlrendszerek

- ▶ MTD speciális eszköz
 - ▶ speciális partícionálás
 - ▶ speciális fájlrendszerek
 - ▶ nincs MBR, PBR
 - ▶ offsetek megadása (lehet címke is)
 - ▶ kernel, bootloader ezek alapján dolgozik
 - ▶ hierarchikus szervezés
 - ▶ lekérdezés
 - ▶ `cat /proc/mtd`

TP-Link <u>WR1043ND</u> Flash Layout					
Layer0	m25p80 spi0.0: m25p64 8192KiB				
Layer1	mtd0 <i>u-boot</i> 128KiB		mtd5 <i>firmware</i> 8000KiB		mtd4 <i>art</i> 64KiB
Layer2	mtd1 <i>kernel</i> 1280KiB		mtd2 <i>rootfs</i> 6720KiB		
mountpoint	/				
filesystem	mini_fo				
Layer3	mtd3 <i>rootfs_data</i> 5184KiB				
Size in KiB	128KiB	1280KiB	1536KiB	5184KiB	64KiB
Name	<i>u-boot</i>	<i>kernel</i>		<i>rootfs_data</i>	<i>art</i>
mountpoint	<i>none</i>	<i>none</i>	/rom	/overlay	<i>none</i>
filesystem	<i>none</i>	<i>none</i>	SquashFS	JFFS2	<i>none</i>

Partíciók, fájlrendszerek

- ▶ Layer0
 - ▶ közvetlenül látszik a fizikai flash chip
- ▶ Layer1
 - ▶ mtd0: bootloader
 - ▶ mtd5: firmware
 - ▶ mtd4: ART (Atheros Radio Test)
- ▶ Layer2
 - ▶ firmware partíciót tovább osztjuk
 - ▶ mtd1: kernel
 - ▶ mtd2: rootfs
- ▶ Layer3
 - ▶ rootfs partíciót szedjük szét két részre:
 - ▶ mtd3: rootfs_data
 - ▶ név nélküli partíció a fájlrendszer csak olvasható részének (SquashFS)

TP-Link WR1043ND Flash Layout					
Layer0	m25p80 spi0.0: m25p64 8192KiB				
Layer1	mtd0 <i>u-boot</i> 128KiB		mtd5 <i>firmware</i> 8000KiB		mtd4 <i>art</i> 64KiB
Layer2	mtd1 <i>kernel</i> 1280KiB		mtd2 <i>rootfs</i> 6720KiB		
mountpoint	/				
filesystem	mini_fo				
Layer3	mtd3 <i>rootfs_data</i> 5184KiB				
Size in KiB	128KiB	1280KiB	1536KiB	5184KiB	64KiB
Name	<i>u-boot</i>	<i>kernel</i>		<i>rootfs_data</i>	<i>art</i>
mountpoint	<i>none</i>	<i>none</i>	/rom	/overlay	<i>none</i>
filesystem	<i>none</i>	<i>none</i>	SquashFS	JFFS2	<i>none</i>

Kitekintés: SquashFS és union fájlrendszerek

▶ SquashFS

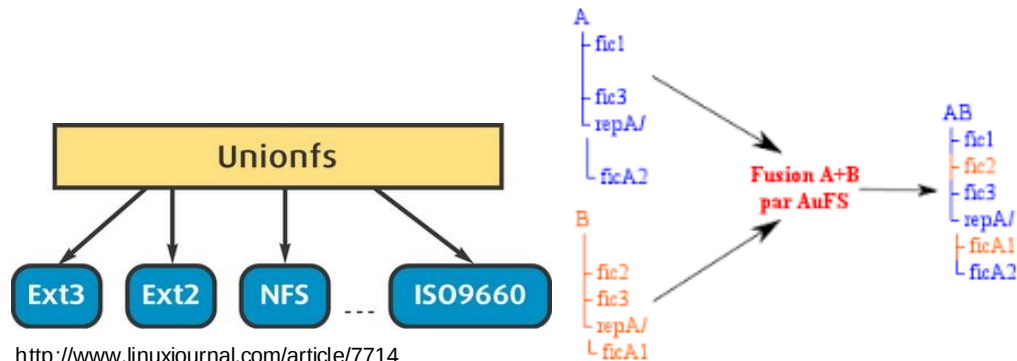
- ▶ tömörített read-only fájlrendszer
- ▶ egy sima fájlban tárolva
 - ▶ egyszerű létrehozás
 - ▶ mksquashfs ...

▶ Union fájlrendszerek

- ▶ több fájlrendszer “összefogása” egybe
- ▶ lehetnek ugyanolyan nevű fájlok
 - ▶ prioritást kell definiálni
- ▶ read-only és read-write fájlrendszerek
- ▶ copy-on-write
- ▶ hány réteget támogat?

▶ Többféle implementáció

- ▶ unionfs (v1, v2)
- ▶ aufs
 - ▶ nem került be a hivatalos kernelbe
 - ▶ de Docker ezt (is) használja
- ▶ overlayfs (by Szeredi Miklós)
 - ▶ mainline Linux kernelben
- ▶ mini_fo
 - ▶ mini fanout overlay file system

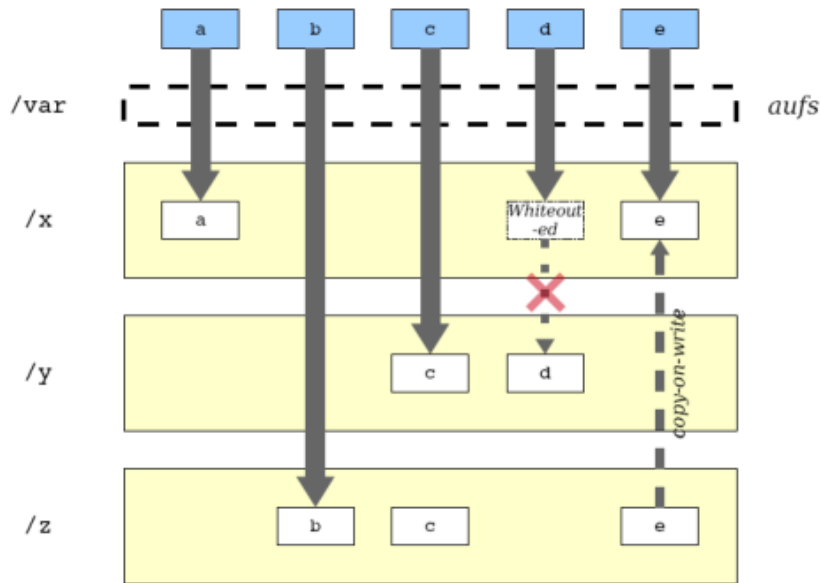


<http://www.linuxjournal.com/article/7714>

<https://www.admin-linux.fr/aufs-unir-les-contenus-de-repertoires/>

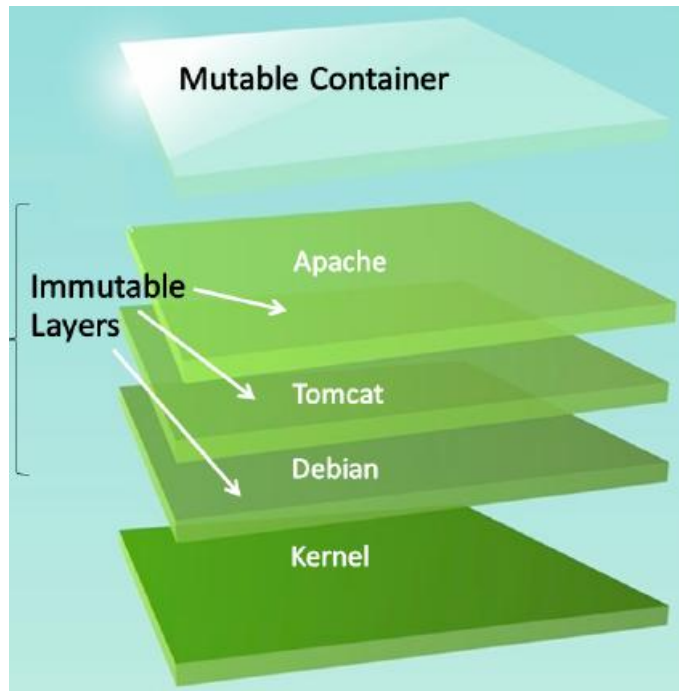
Kitekintés: SquashFS és union fájlrendszerek

Példa: ro, rw fájlrendszerek



<http://d.hatena.ne.jp/dayflower/20080714/1216010519>

Példa: Docker



<https://jaxenter.com/the-resilient-and-highly-scalable-applications-cloud-121731.html>

Mindez az OpenWRT-ben

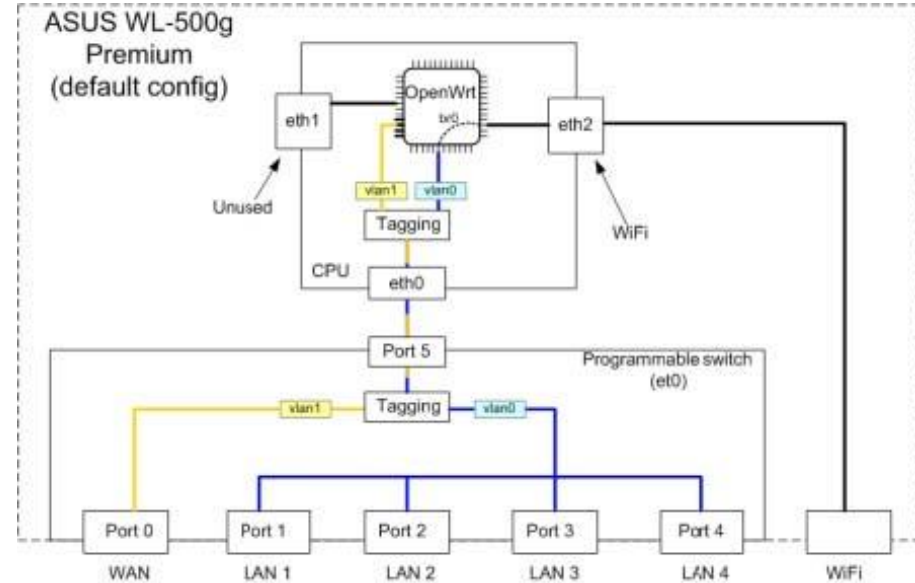
- ▶ fájlrendszer felépítése
 - ▶ /rom
 - ▶ read-only rész (ro)
 - ▶ squashfs
 - ▶ /overlay
 - ▶ írható rész (rw)
 - ▶ JFFS2
 - ▶ (Journalling Flash File System v2)
 - ▶ ide íródnak a különbségek
 - ▶ /
 - ▶ /rom + /overlay
 - ▶ mini_fo fájlrendszer
 - ▶ ami egyesíti az alatta levő fájlrendszereket
 - ▶ tudja, hogy melyik fájlt hol kell elérni

OpenWRT részletek

Hálózatkezelés

VLAN-ok szerepe

- ▶ Programozható switch (chip)
 - ▶ (jobb esetben) tetszőlegesen konfigurálható
 - ▶ VLAN támogatás
 - ▶ CPU-nak meg kell tudni különböztetni, hogy melyik interfészen jött a csomag
 - ▶ (bizonyos funkciók esetében)
- ▶ megoldás: VLAN
 - ▶ bejövő csomagok VLAN taget kapnak (pl. port száma)
 - ▶ CPU-nál VLAN függő virtuális interfészek (eth0.1, eth0.2, ...)

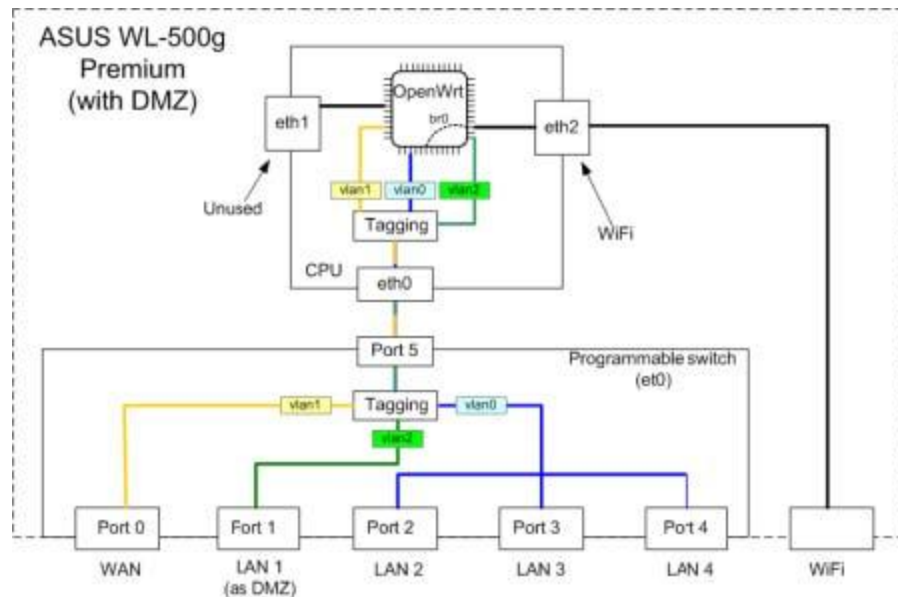


Példa: WAN és LAN portok megkülönböztetése

Forgalom ugyanazon a fizikai interfészen érkezik a CPU-hoz

VLAN-ok szerepe

- ▶ Programozható switch (chip)
 - ▶ (jobb esetben) tetszőlegesen konfigurálható
 - ▶ VLAN támogatás
 - ▶ CPU-nak meg kell tudni különböztetni, hogy melyik interfészen jött a csomag
 - ▶ (bizonyos funkciók esetében)
- ▶ megoldás: VLAN
 - ▶ bejövő csomagok VLAN taget kapnak (pl. port száma)
 - ▶ CPU-nál VLAN függő virtuális interfészek (eth0.1, eth0.2, ...)



Példa: WAN, DMZ és LAN portok megkülönböztetése

Forgalom ugyanazon a fizikai interfészen érkezik a CPU-hoz

OpenWRT részletek

Konfiguráció

Konfigurálás

```
BusyBox v1.22.1 (2014-08-04 22:39:32 CEST) built-in shell (ash)
_Enter 'help' for a list of built-in commands.
```

- ▶ telnet

- ▶ először általában csak ez van
- ▶ password beállítása után: ssh

► ssh

- ▶ általában egyszerűsített Linux shell (busybox)
- ▶ rugalmas konfiguráció
- ▶ UCI
 - ▶ Unified Configuration Interface
 - ▶ paraméterek beállítása egységes interfészen
- ▶ konfig fájlok is ezt használják
 - ▶ pl: /etc/config/network
- ▶ csomagkezelés: opkg
 - ▶ ~egyszerűsített dpkg

[illegible]

```
config interface 'lan'
    option force_link '1'
    option type 'bridge'
    option proto 'static'
    option ipaddr '192.168.1.1'
    option netmask '255.255.255.0'
    option ip6assign '60'
    option _orig_ifname 'eth1 wlan0'
    option _orig_bridge 'true'
    option ifname 'eth1.1'
```

Konfigurálás

- ▶ Korábban tanult eszközök használhatók
- ▶ Egy Linux rendszert kell konfigurálni
- ▶ Használhatjuk routerként, switch-ként is
- ▶ Hálózati funkciók (opkg) csomagokban tehetők fel
 - ▶ dhcp, iptables (nat, firewall), dns
 - ▶ egy alap firmware-ben ezek benne vannak
 - ▶ WiFi kezelés (ez jön majd később!)
- ▶ Számos egyéb program
 - ▶ transmission, asterisk, apache2, ...
 - ▶ webcamera kezeléstől az USB-s külső HDD kezeléséig szinte minden
 - ▶ de ha nincs meg: lefordítható forrásból (cross-compilation)

LuCI: a könnyebb út

Példa:
Network/Interfaces

SobiNet Status ▾ System ▾ Network ▾ Logout AUTO REFRESH ON

Interfaces

Interface Overview

Network	Status	Actions
LAN br-lan	Uptime: 250d 11h 22m 56s MAC-Address: E8:DE:27:F6:F7:AC RX: 144.73 GB (868664824 Pkts.) TX: 1.85 TB (1468893468 Pkts.) IPv4: 192.168.1.1/24 IPv6: FDAD:6E0F:EFD8:0:0:0:0:1/60	Connect Stop Edit Delete
LAN4 eth1.3	Uptime: 250d 11h 22m 56s MAC-Address: E8:DE:27:F6:F7:AC RX: 5.58 GB (9116791 Pkts.) TX: 6.61 GB (8613095 Pkts.) IPv4: 192.168.4.1/24	Connect Stop Edit Delete
WAN pppoe-wan	Uptime: 6d 14h 1m 22s RX: 45.37 GB (35980081 Pkts.) TX: 8.51 GB (22538525 Pkts.) IPv4: 188.143.77.204/32	Connect Stop Edit Delete

Add new interface...

Global network options

IPv6 ULA-Prefix

Save & Apply Save Reset

Powered by LuCI Trunk (svn-r10467) OpenWrt Barrier Breaker 14.07-rc3

LuCI: a könnyebb út

Példa: Network/Switch

SobiNetStatusSystemNetworkLogout

AUTO REFRESH ON

Switch

The network ports on this device can be combined to several VLANs in which computers can communicate directly with each other. VLANs are often used to separate different network segments. Often there is by default one Uplink port for a connection to the next greater network like the internet and other ports for a local network.

Switch "switch0"

Enable VLAN functionality ☒

VLANs on "switch0"

VLAN ID	CPU	Port 1	Port 2	Port 3	Port 4	Port 5	Port 6
Port status:							
	1000baseT full-duplex	1000baseT full-duplex	no link	no link	no link	1000baseT full-duplex	1000baseT full-duplex
1	tagged	untagged	untagged	untagged	off	off	off
2	off	off	off	off	off	untagged	untagged
3	tagged	off	off	off	untagged	off	off

Add

Save & ApplySaveReset

Powered by LuCI Trunk (svn-r10467) OpenWrt Barrier Breaker 14.07-rc3

LuCI: a könnyebb út

Példa:
System/Software

Software

Actions Configuration

No package lists available

 Update lists

Free space: 9% (404.00 KB)



Download and install package:

Filter:

Status

Installed packages Available packages

	Package name	Version
Remove	base-files	155-r42056
Remove	block-mount	2014-06-22-e0430f5c62f367e5a8e02755412977b02c3fc45e
Remove	busybox	1.22.1-2
Remove	dnsmasq	2.71-3
Remove	dropbear	2014.63-1
Remove	firewall	2014-07-19
Remove	fstools	2014-06-22-e0430f5c62f367e5a8e02755412977b02c3fc45e
Remove	hostapd-common	2014-06-03-1
Remove	ip6tables	1.4.21-1
Remove	iperf	2.0.5-1
Remove	iptables	1.4.21-1
Remove	iw	3.15-1
Remove	jshn	2014-07-16-bd388d2b6c2c151bf513c1e449417d18ce02d10b
Remove	jsonfilter	2014-06-19-cdc760c58077f44f40adbef41e1556a67c1b9a9
Remove	kernel	3.10.49-1-94831e5bcf361d1c03e87a15e152b0e8
Remove	kmod-ath	3.10.49+2014-05-22-1
Remove	kmod-ath9k	3.10.49+2014-05-22-1
Remove	kmod-ath9k-common	3.10.49+2014-05-22-1
Remove	kmod-cfg80211	3.10.49+2014-05-22-1
Remove	kmod-crypto-aes	3.10.49-1
Remove	kmod-crypto-arc4	3.10.49-1
Remove	kmod-crypto-core	3.10.49-1
Remove	kmod-crypto-des	3.10.49-1
Remove	kmod-crypto-ecb	3.10.49-1

Összefoglalás

- ▶ Hálózati eszközök belső felépítése...
- ▶ helyett: OpenWRT
 - ▶ egy jó példa
 - ▶ egy beágyazott Linux (firmware)
 - ▶ pl. WiFi routerek vezérlésére
 - ▶ tetszőlegesen testreszabható, konfigurálható
 - ▶ saját programok írhatók hozzá,
 - ▶ futtathatók rajta
- ▶ Érdemes kipróbálni!

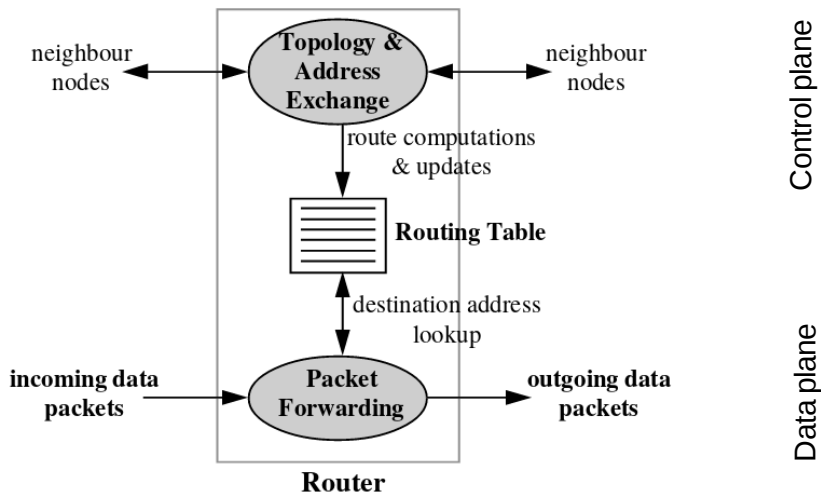


(nem otthoni) routerek

home, access,

edge, core

Routerek általános felépítése



James Aweya: IP Router Architectures: An Overview

A router feladatai

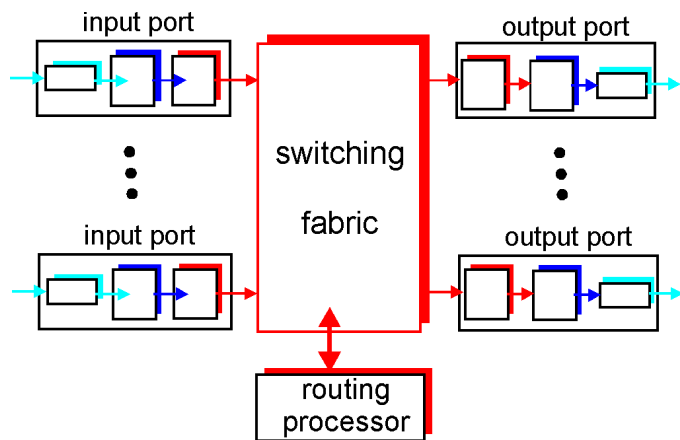
▶ Routing feladatok

- ▶ routing protokoll (OSPF, ...) futtatása, a kapott információk alapján:
- ▶ útvonalak kiszámítása, amit felhasználva:
- ▶ a továbbítási táblázat karbantartása

▶ Csomagtovábbítási feladatok

- ▶ IP csomag validálása
- ▶ Célcím kikeresése a továbbítási táblázatból
 - ▶ helyi, unicast, multicast cím
- ▶ Csomag-élettartam szabályozás
 - ▶ TTL, Time-to-live mező csökkentése
- ▶ Ellenőrzőösszeg újraszámítás
- ▶ Eltérő MTU esetén csomagdarabolás

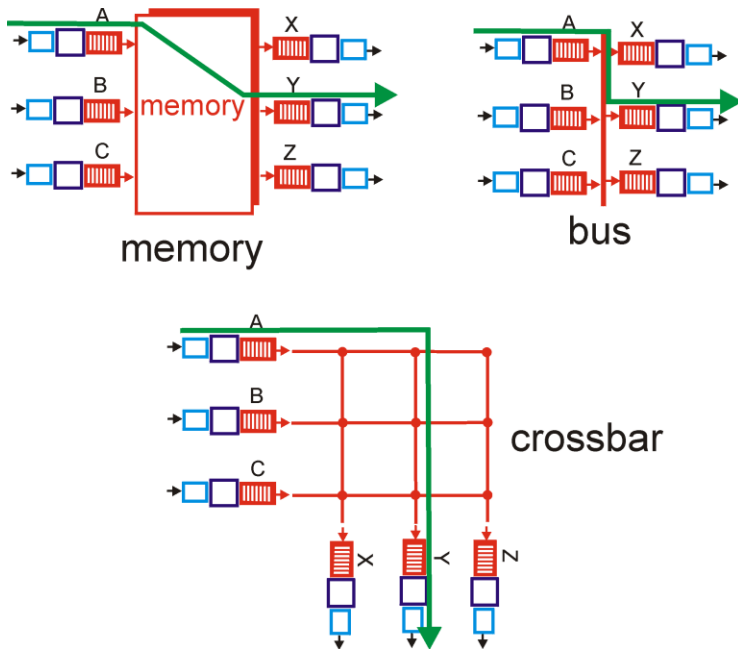
Router architektúrák



http://www2.ic.uff.br/~michael/kr1999/4-network/4_06-inside.htm

- ▶ A portok “line card”-okon csatlakoztathatók
- ▶ A továbbításhoz szükséges legtöbb feladatot a line cardokon el lehet végezni
- ▶ Ideális esetben a routing processor nem végez csomagtovábbítási feladatokat
 - ▶ A line cardok rendelkeznek routing tábla másolataival

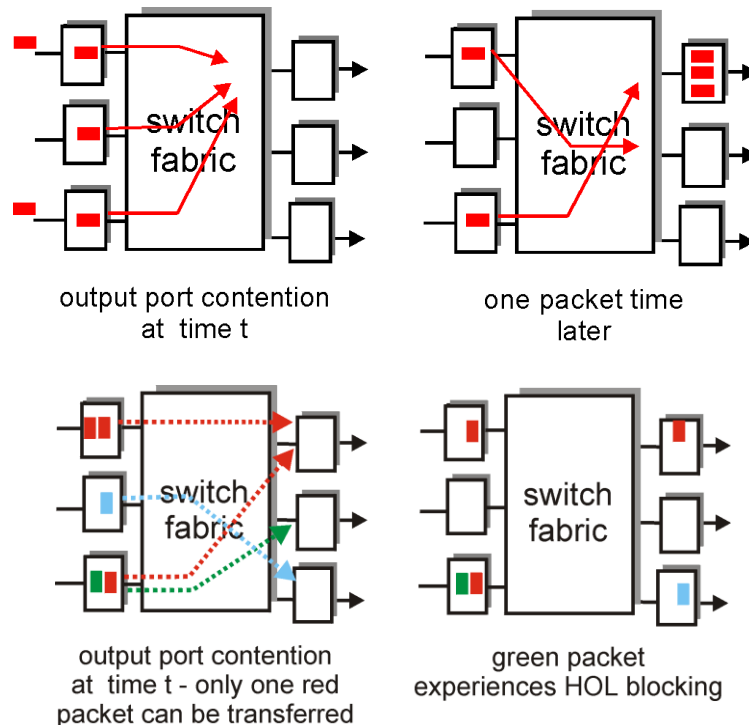
Switching fabric / kapcsolószerkezet



- ▶ Kezdetben hagyományos számítógép architektúrákat használtak (~ OpenWrt)
 - ▶ csomagokat a közös hozzáférésű memóriába másolták
 - ▶ az egyetlen CPU döntött a sorsáról
- ▶ 2nd gen: csomagot a közös buszon továbbították a routing processor beavatkozása nélkül
- ▶ Crossbar: $2N$ busz, N input-output port összekapcsolására
 - ▶ (szokás ilyenkor a csomagot kisebb egységekre darabolni, majd az output porton összerakni)

Sorbanállás, sormenedzsment, QoS

- ▶ Sorok a be- és kimentet is lehetnek
 - ▶ Kimeneti torlódás oka: pl. nagyobb a bejövő összerhelés a kimeneti kapacitásnál
 - ▶ Head-of-the-line blocking:
 - ▶ a switching fabric nem N-szeres sebességgel üzemel,
 - ▶ a pirosat nem lehet még továbbítani, a zöldet lehetne, de a piros feltartja
- ▶ Minőség biztosítása (Quality of Service, QoS)
 - ▶ Nem FIFO sorokkal, pl:
 - ▶ Prioritásos sorokkal,
 - ▶ Weighted Fair Queueing
 - ▶ Random Early Detection (RED)
 - ▶ Torlódás jelzése csomagdobás nélkül (ECN bitben)
 - ▶ TCP estén működik

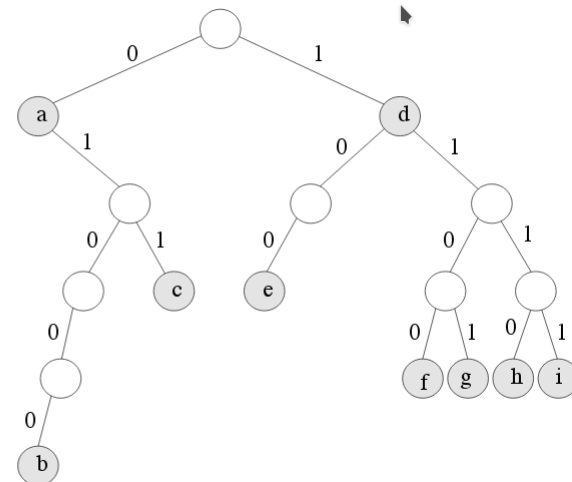


Kimenő port kiválasztása: Longest Prefix Match

- ▶ BGP ~700k bejegyzés
- ▶ Egyszerű bináris keresés
 - ▶ Túl sok memória hozzáférés
- ▶ TCAM (ternary content-addressable memory)
 - ▶ Túl drága a teljes táblázat tárolásához
- ▶ Gyorsítási lehetőségek:
 - ▶ A fa tetejét direktben címezzük
 - ▶ Fából irányított gráfot készítünk
 - ▶ Pl. a alulról összevonjuk az azonos részfákat
 - ▶ Kisebb fa befér a cache-be → gyorsabb a memóriaelérés
 - ▶ ...

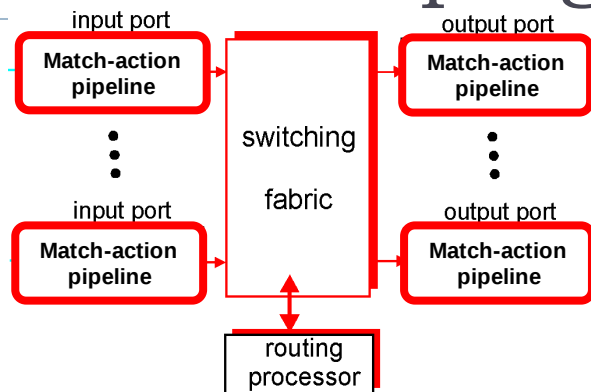
Prefixes

a 0*
b 01000*
c 011*
d 1*
e 100*
f 1100*
g 1101*
h 1110*
i 1111*

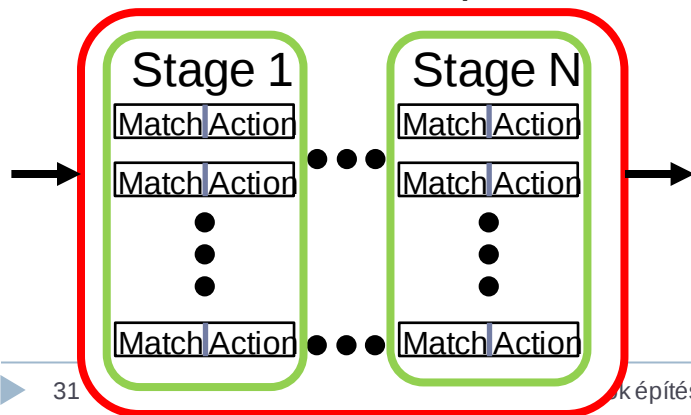


Martin Heusse: Router and switch architecture

Kitekintés: programozható routerek



Match-action Pipeline



- ▶ SDN: Software Defined Networking
- ▶ Lehetséges architektúrák:
 - ▶ A line cardok nem fix műveletsort hajtanak végre: programozhatók match-action pipeline segítségével
 - ▶ A switch teljes működését egyetlen match-action pipeline írja le
 - ▶ ...
- ▶ Match-action pipeline
 - ▶ Openflow, P4, ...