HÁLÓZATI RENDSZEREK
ÉS SZOLGÁLTATÁSOK
TANSZÉK

HÁLÓZATOK ALAPJAI ÉS ÜZEMELTETÉSE

Szoftveralapú hálózatok

2023. május 22.

Zsóka Zoltán

BME Hálózati Rendszerek és Szolgáltatások Tanszék

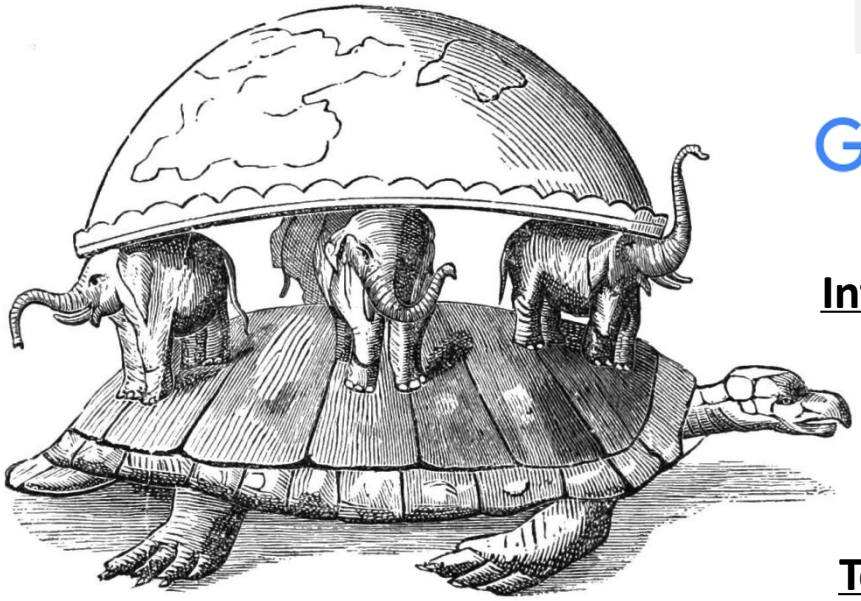
zsoka@hit.bme.hu

MŰEGYETEM 1782

MA MIRŐL LESZ SZÓ?

1. Hálózati funkciók fejlődése
2. Szoftveralapú hálózatok (SDN)
3. A dolgok Internete (IoT)

AZ INTERNET FEJLŐDÉSE



Alkalmazások – folyamatos innováció és megújulás

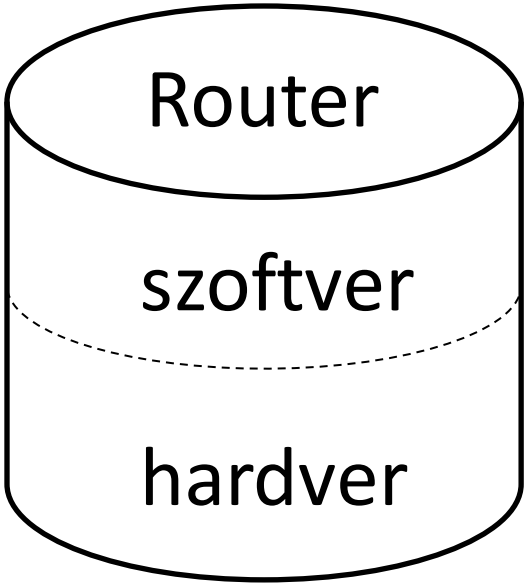
amazon flickr tumblr. twitter
Google skype facebook YouTube tinder

Internet protokollok - változatlanság

TCP/IP, IS-IS, BGP, DNS,
SNMP, …

Technológiák – folyamatos innováció és megújulás

HAGYOMÁNYOS HÁLÓZATOK ARCHITEKTÚRÁJA

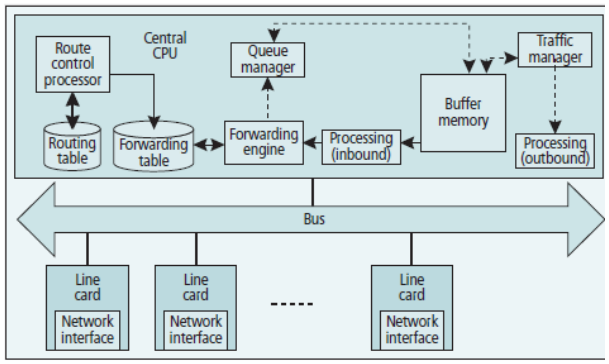


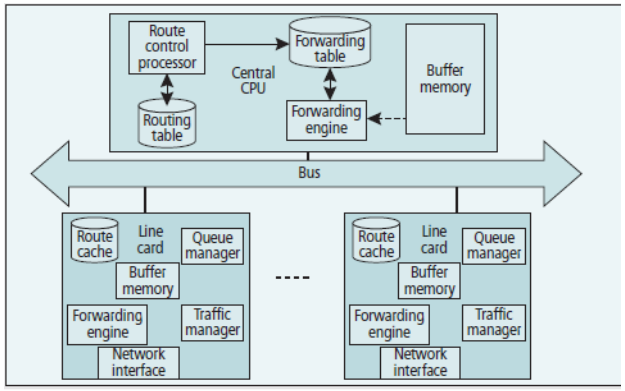
Szoftver kódsorok milliói,
nehezen bővíthető, alakítható

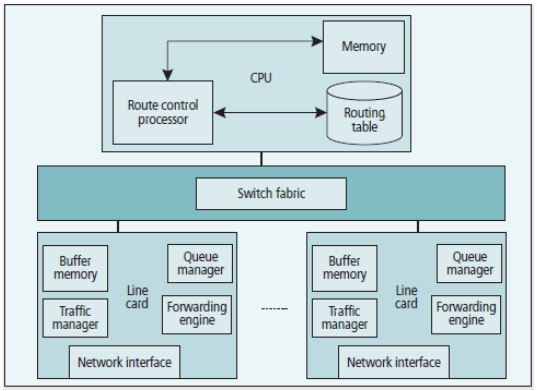
Tranzisztorok tízmilliói, több 10Gb RAM
Energiaigényes

- Vertikális integráció (célhw+célsw): komplex funkciók, protokollok (OSPF, BGP, QoS, forgalomvezérlés / traffic engineering, NAT, tűzfalak, …)

GENERÁCIÓK







Első generációs router

- Egy CPU
- Közös buszra kapcsolódó interfészkártyák
- Kommersz valós idejű op. rsz.
- Szoftverben implementált funkciók: kapcsoló, sorok menedzselése, forgalom menedzselése L2/L3 feldolgozás
- A CPU-t megosztva használta
  - a csomagtovábbítás
  - a routing protokollok
  - a routing tábla frissítések
  - a menedzsment funkciók

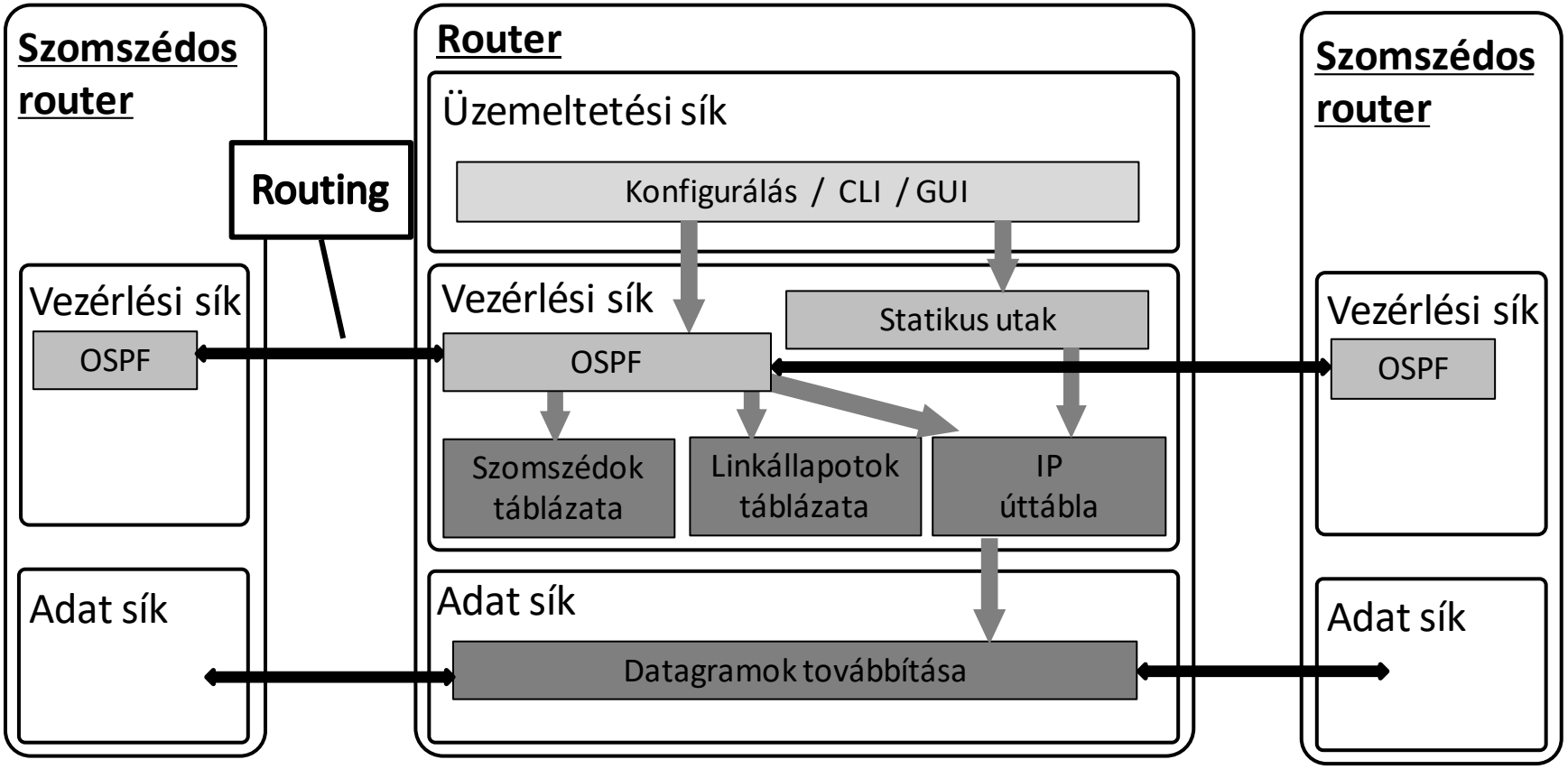
Második generációs router

- Több intelligencia a vonali kártyákon
  - Processzor, memória, továbbításhoz cache
  - Lehetővé téve bizonyos forwarding operációkat
- A vezérlés és a menedzsment továbbra is a CPU-n maradt

Harmadik generációs router

- Funkciók szigorú szétválasztása
  - Forwarding: hardwer alapú
  - Control: software alapú
- A vezérlés és a menedzsment továbbra is a CPU-n maradt
- A közös busz helyett kapcsolómező a továbbítás sebességének növelésére

HÁLÓZATI SÍKOK



- **Üzemeltetési sík**: az eszköz beállításainak kezelése (konfigurálás) és működésének felügyelete
- **Vezérlési sík**: a beérkező információk alapján folyamatosan döntéseket hoz a csomagtovábbítás módjáról – merre menjen a forgalom
- **Adatsík**: a vezérlési síktól kapott utasítások alapján dönt a csomagok sorsáról (továbbítás a megfelelő irányba / eldobás)

Forrás: https://blog.ipspace.net/2013/08/management-control-and-data-planes-in.html

A HAGYOMÁNYOS ARCHITEKTÚRA KORLÁTAI

- A gyártóspecifikus vertikális integráció (monolit hw – op.rsz – hálózati funkciók) miatt
  - a fejlesztések időigényesek, sokszor rugalmatlanok, adott alkalmazáshoz szükségtelen funkciókat is tartalmaznak
  - a különböző gyártók eszközeinek közös rendszerben történő üzemeltetése bonyolult és rosszul automatizálható,

    gyártóspecifikus CLI-k, scriptek (minden eszközt másként) <-> integrált üzemeltető sw rendszer (minden funkciót hasonlóan, egyformán)
  - az üzemeltetés skálázhatósága rossz

TOVÁBBI KORLÁTOK

- Az alapvető hálózati funkció (célcím alapú L3 továbbítás) jól működik, de a gyártóspecifikus vertikális integráció korlátainak következményeként
  - nem elég jól működik
    - a célcím alapú L2 továbbítás (pl. STP korlátai)
    - a L3 forgalomvezérlés (Traffic Engineering), pl. a dinamikus erőforrás lefoglalás problémái miatt (pakolási probléma)
    - a nagyméretű folyamok (elephant flow) továbbítása (statisztika: a folyamok ~5%-a a link sávszélességének ~40%-át is elfoglalhatja)
  - gyakorlatilag nem nagyon működik
    - az elosztottan megvalósuló policyk összehangolása (pl. QoS és hálózatbiztonság)
    - policy alapú (L3 és L4 forrás, nyelő) routing megvalósítása
    - biztonsági funkciók hatékony beillesztése a továbbítási útvonalba

A MEGÚJÍTÁS KORLÁTAI

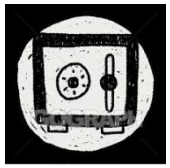
Az innováció korlátai

- Eszközökbe zárt hálózati funkciók (spec hw. + zárt sw.)

- Gyártóspecifikus (zárt) sw interfészek

- Lassú protokollszabványosítási folyamatok

- A gyártók innovációs kapacitásai és üzleti megfontolásai határozzák meg a folyamatok irányát és sebességét



Az üzemletetés korlátai

- Költséges, rosszul skálázódó, lassú folyamatok (hálózati mérnök, CLI, scriptek)

- A konfigurációs hibák okozzák a legtöbb működési zavart

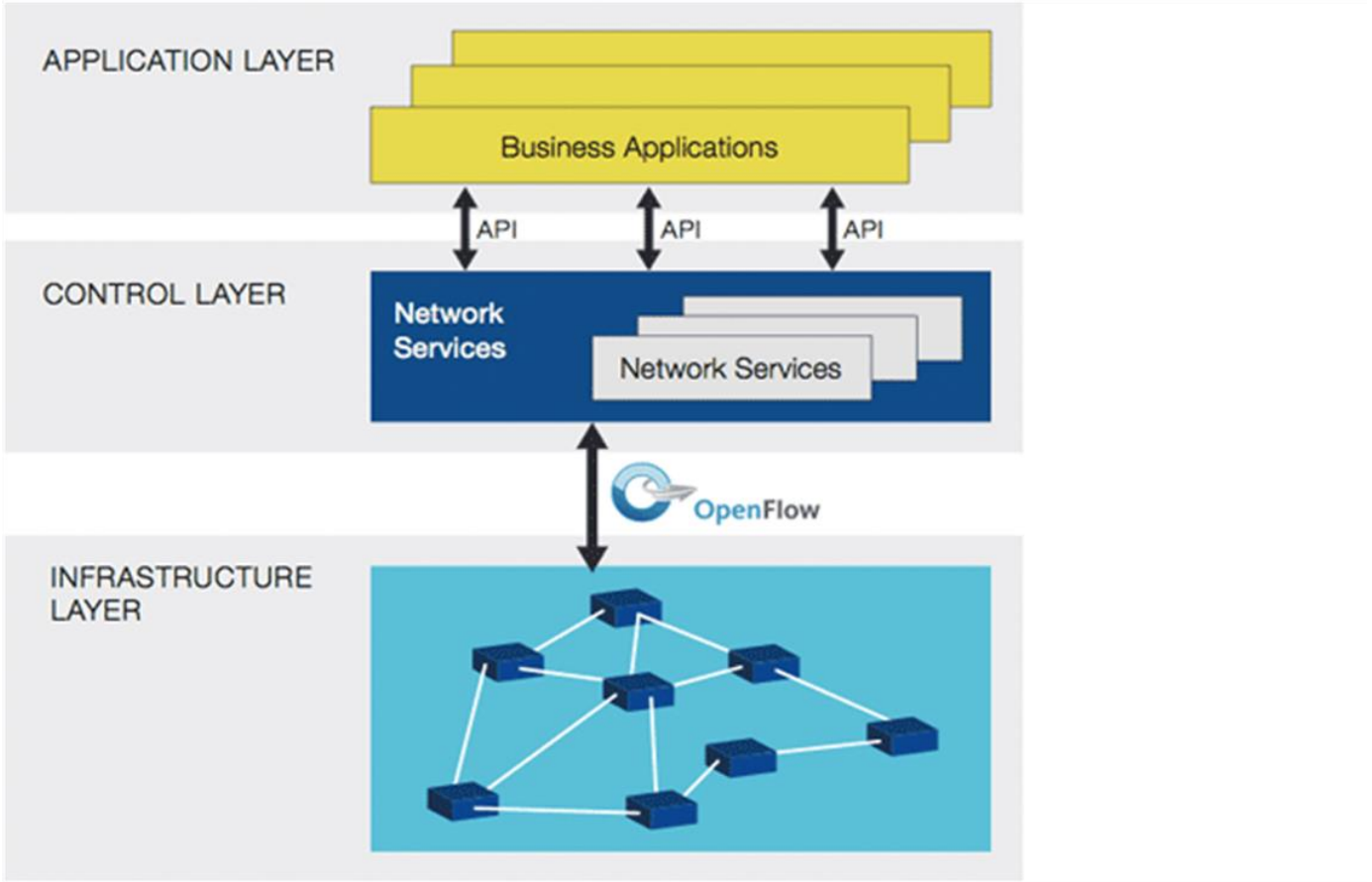
- Szoftverhibák az eszközökben (router sw-e: több millió kódsor, frissítés: ismert hibák ismeretlenre cserélése)

- Kritikus hálózati szegmensek
  - Adatközpontok hálózati megoldásai
  - Otthoni hálózatok

A HÁLÓZATI ARCHITEKTÚRA FEJLŐDÉSI IRÁNYAI

- Szoftver alapú hálózatok
  - erőforrások szoftver alapú vezérlése
  - hatékony kivételkezelés a továbbításban
- Hálózati funkciók virtualizálása
  - az erőforrások egy részének a rugalmasság növelésére fordítása
  - a hálózati funkciók minél nagyobb hányadának tisztán szoftver alapú (nem kell speciális hw) megvalósítása

ÁLTALÁNOS ARCHITEKTÚRA

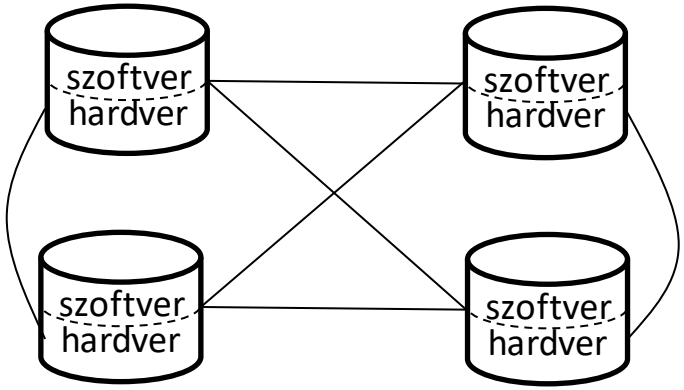


MA MIRŐL LESZ SZÓ?

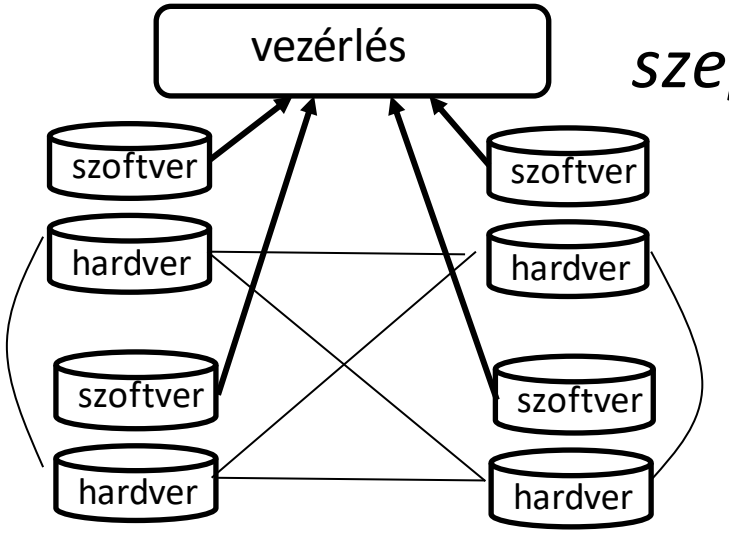
1. Hálózati funkciók fejlődése
2. Szoftveralapú hálózatok (SDN)
3. A dolgok Internete (IoT)

SDN: A VEZÉRLÉSI ÉS ADATSÍK SZÉTVÁLASZTÁSA

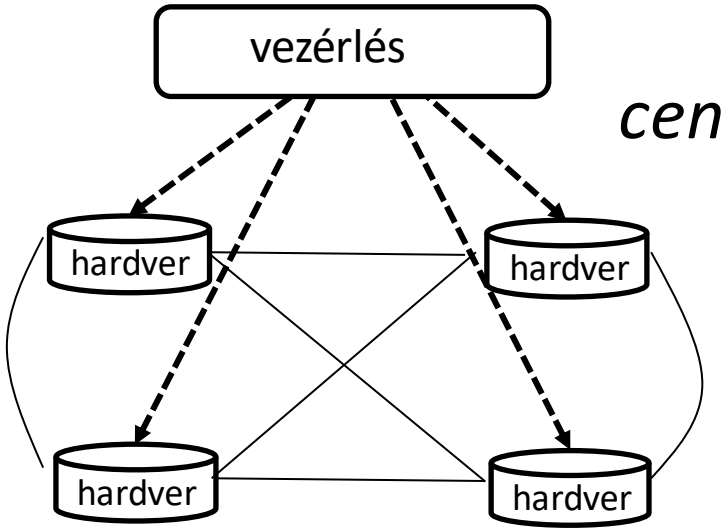
integrált



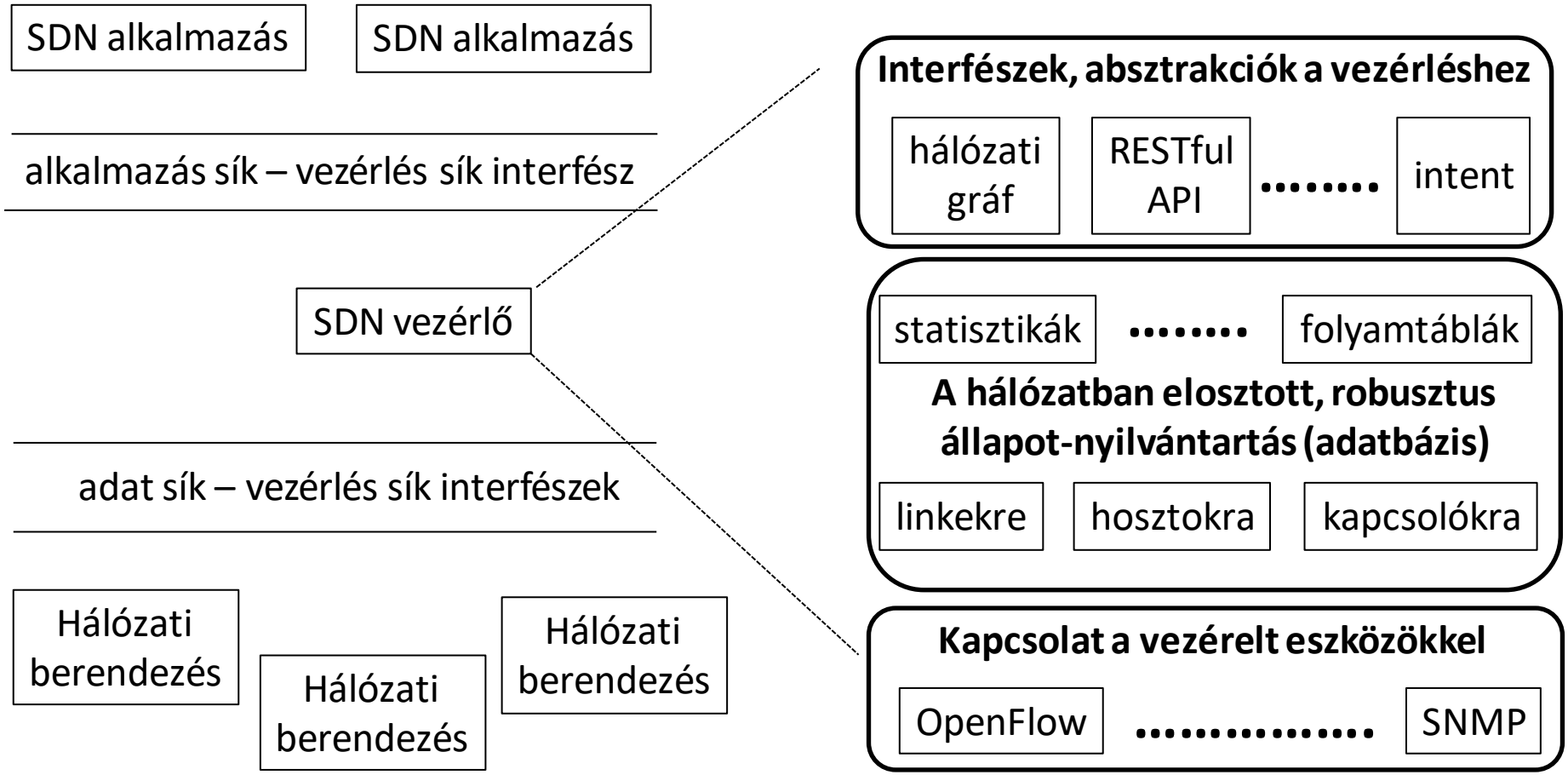
szeparált



centralizált



SDN ARCHITEKTÚRA



SDN – ADATSÍK

- Forgalomtovábbító eszközök
  - Gyors, egyszerű kapcsolóeszközök hardverben megvalósított forgalomtovábbítási képességekkel
  - A kapcsolótábla tartalmát a vezérlő biztosítja
  - API a kapcsolótábla vezérléséhez (pl. OpenFlow)
  - Protokoll a vezérlővel folyó kommunikációhoz (p. OpenFlow)
- Néhány protokollt is képesek futtatni (pl. ARP, LLDP)
- Jól definiált interfészeken kommunikálnak a vezérlési síkkal
  - Funkcióik szoftveresen vezérelhetők
  - Képességeiket hirdetik
  - Eseményekről jelzést adnak
  - A kontroller déli interfészén, Southbound interface, SBI

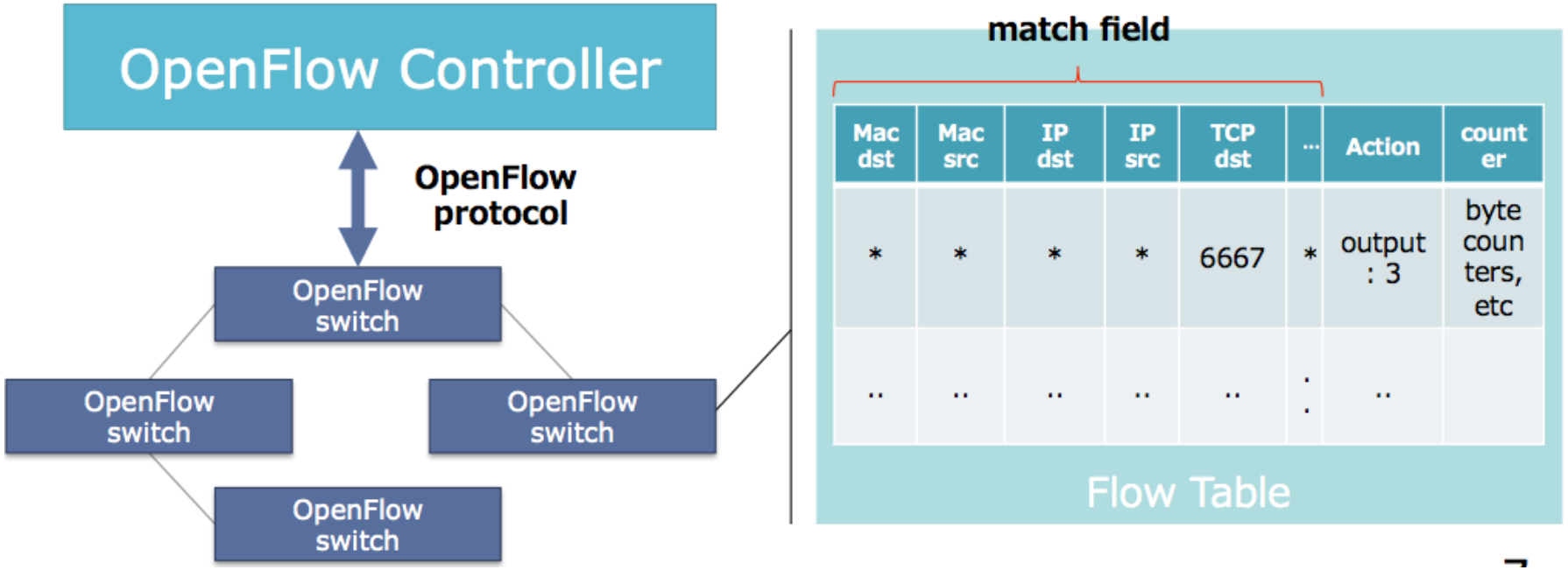
SDN – VEZÉRLÉSI SÍK

- Logikai értelemben központosított
- Alapvető funkciói
  - A topológiára és a hálózati állapotokra vonatkozó információk kezelése
  - Eszközök felderítése
  - Forgalomtovábbítási út meghatározása
  - Biztonsági funkciók
- A vezérlők koordinálása
- Interfész az alkalmazási sík felé
  - Északi interfész, Northbound interface, NBI

SDN – ALKALMAZÁSI SÍK

- Meghatározzák a hálózattól kért erőforrásokat és működést az üzleti és policy szempontoknak megfelelően
- Vezérlési alkalmazások alapvető funkciói
  - Alacsony szintű funkciókra (vezérlő API) alapozott komplex megoldások
  - Routing/forwarding, access control, terhelésmegosztás (load balancing)
- Szükséges lehet az elosztott vezérlők működésének összehangolása (*orchestration*)
- Megfelelő programnyelvek támogatják a fejlesztését (pl. python)

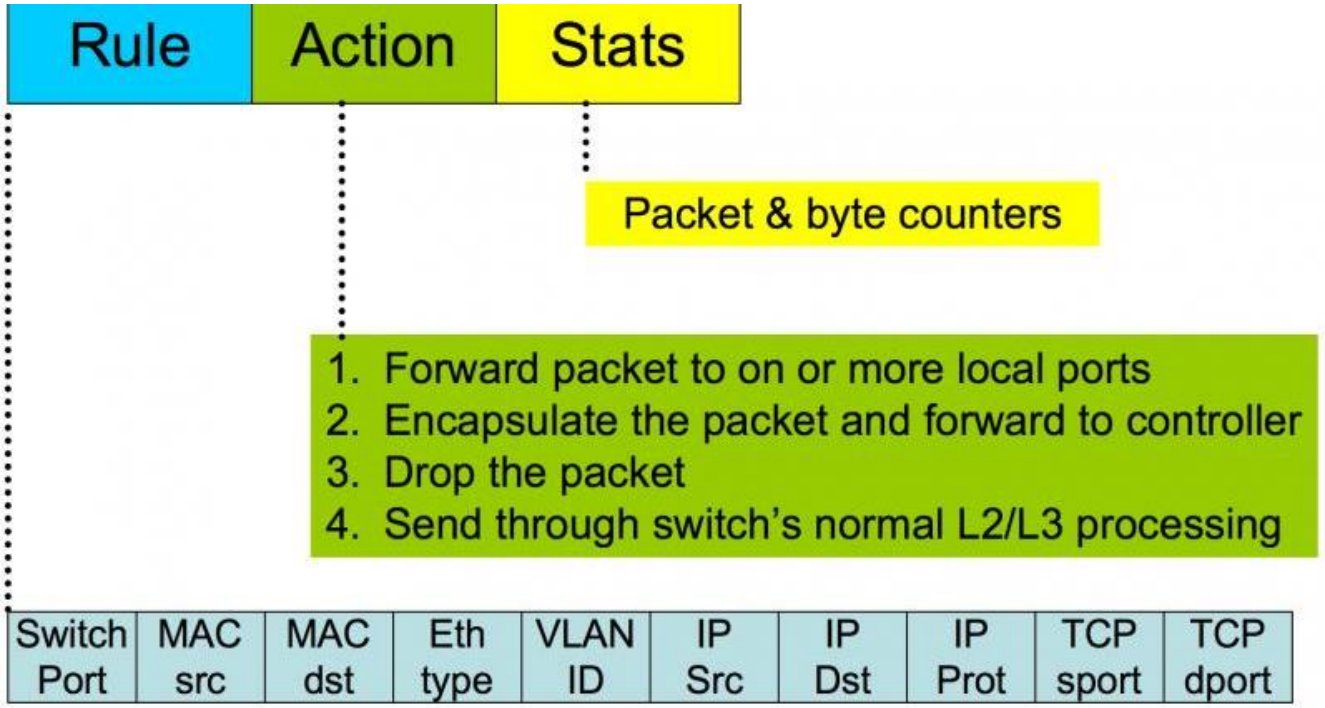
OPENFLOW



- Az SDN megvalósításának egyik meghatározó technológiája
- Nyílt interfész a vezérlési sík és adat sík között
- A kapcsolók TCP felett kommunikálnak a kontrollerrel
  - 6653-as port

OPENFLOW SZABÁLYOK

- L2, L3 és L4 fejléc adatok alapján integrált továbbítási szabályok (flow)
- Flow bejegyzések tábláit tárolják a kapcsolók



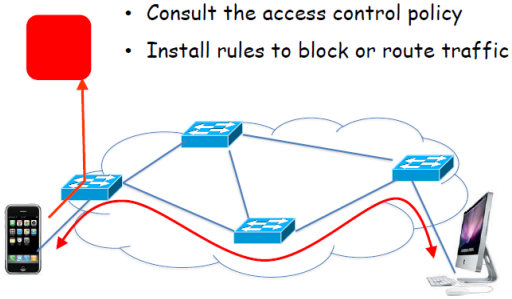
ILLESZTÉSI SZABÁLYOK –PÉLDA

	Input Port	Source MAC	Dest MAC	Ether Type	VLAN ID	Source IP@	Dest IP@	IP Proto	IP SrcPort	IP DstPort
MASKS										
Ethernet Switching	*	*	12:2E	*	*	*	*	*	*	*
IP Routing	*	*	*	*	*	*	1.2.3.4	*	*	*
App Firewall	*	*	*	*	*	*	*	*	*	443
Flow Switching	Port6	12:2E	17:FF	0800	VLAN7	1.2.3.4	4.3.2.1	06	11317	80
VLAN + App	*	*	*	*	VLAN7	*	*	*	*	80
Port + Ethernet + IP	Port6	12:2E	*	0800	*	*	4.3.2.1	06	*	*

OPENFLOW ALKALMAZÁSI PÉLDÁK

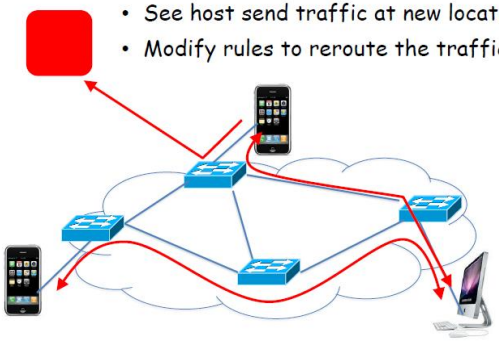
E.g.: Dynamic Access Control

- Inspect first packet of a connection
- Consult the access control policy
- Install rules to block or route traffic



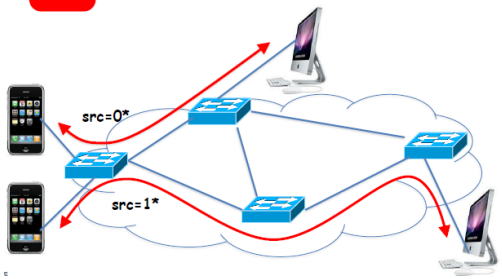
E.g.: Seamless Mobility/Migration

- See host send traffic at new location
- Modify rules to reroute the traffic

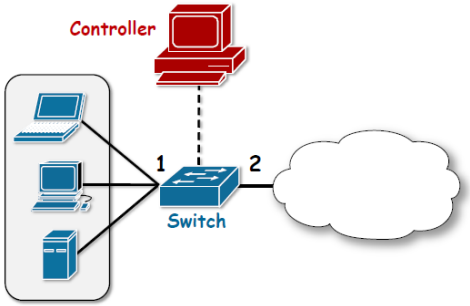


E.g.: Server Load Balancing

- Pre-install load-balancing policy
- Split traffic based on source IP



In-depth Example: Simple Repeater

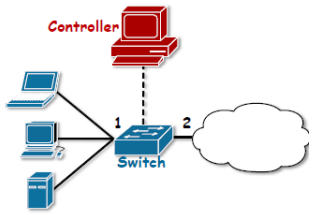


- Simple Network Repeater
  - forward packets received on port 1 out 2 and vice versa

Simple Repeater

Controller (POX) (Pseudo)-Program

```
def handle_packetIn(packet):
    out_port = 2
    if packet.in_port == 2:
        out_port = 1
    flow_mod = ofp_flow_mod()
    flow_mod.match = ofp_match()
    flow_mod.match.in_port = \
        packet.in_port
    action = ofp_action_output()
    action.out_port = out_port
    flow_mod.action = [ action ]
    flow_mod.buffer_id = \
        packet.buffer_id
    send(flow_mod)
```

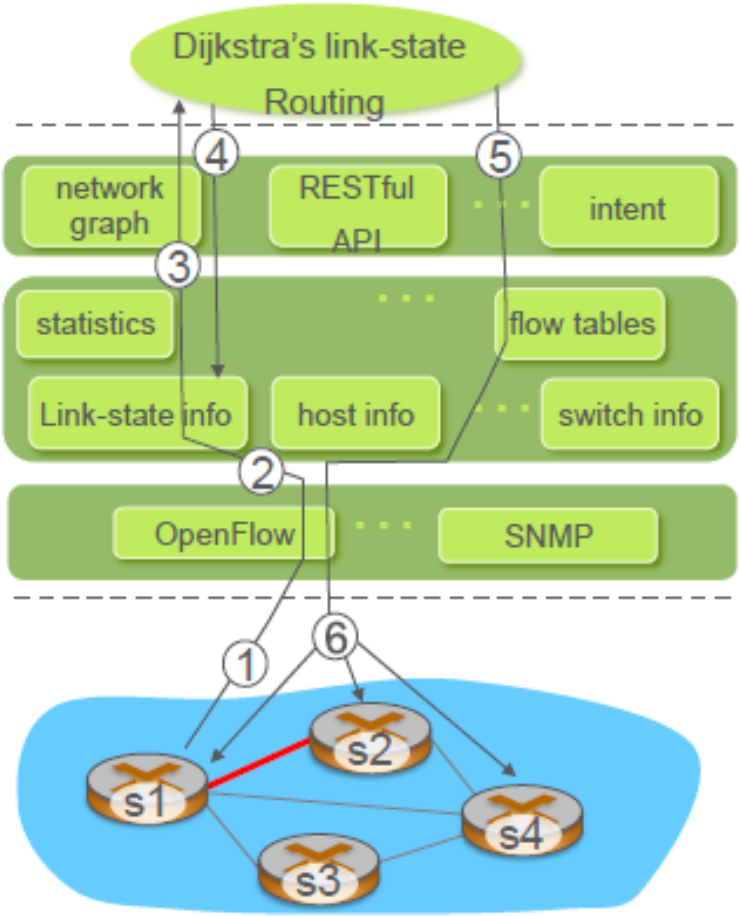


Flow Table

Priority	Pattern	Action	Counters
DEFAULT	IN_PORT:1	OUTPUT:2	(0,0)
DEFAULT	IN_PORT:2	OUTPUT:1	(0,0)

SDN VEZÉRLÉSI PÉLDA

1. S1 érzékeli a link hibáját és OF üzenettel értesíti a kontrollert
2. A kontroller frissíti a linkek adatbázisát
3. A linkállapot változás miatt hívódik az NBI alkalmazás
4. Az aktuális topológián lefut a Dijkstra algortimus
5. A kontroller frissíti a flow táblákat
6. A kontroller szétküldi a flow táblákat



SDN ÖSSZEGZÉS

- Vezérlési sík és adatsík szétválasztása összetett továbbítási szabályok érvényesítésére
- Architektúra
  - Három réteg: infrastruktúra (hálózati eszközök és linkek), vezérlés, alkalmazások a továbbítási szabályok meghatározására
- Centralizált vezérlés (egy/több centralizált vezérlő)
- Jól definiált nyílt API
- Csomagjellemzőkre alapozott továbbítási szabályok
  - L2, L3, L4 header alalpú szabályok:
  - MAC src addr, dst addr, Eth type, VLAN ID,
  - IP src addr, dst addr, prot
  - src port, dst port

SDN PÉLDA

- Számos ingyenes és fizetős kontroller készült már
  - NOX/POX, OpenDaylight, ONOS, Ryu, Cisco DNAC
- A laborban kipróbált megoldás: POX
  - Python alapú
  - OpenFlow 1.0
  - Kész python modulok a protokollfejlécek kezelésére (pl. ARP, Ethernet, IP)
  - Eseményeket kezelő *Listener* funkciókon alapul
  - Bejövő OpenFlow csomag tartalma: amit a kapcsoló nem tudott kezelni
    - Honnan jött?
    - Mi van benne?
    - A problémás PDU felismerése és kezelése
    - Adatok eltárolása, szabály generálása és leküldése, ha kell
    - Üzenet (pl. válasz) összeállítása és szabály nélküli elküldetése a kapcsolóval

MA MIRŐL LESZ SZÓ?

1. Hálózati funkciók fejlődése
2. Szoftveralapú hálózatok (SDN)
3. A dolgok Internete (IoT)

ELŐZMÉNYEK ÉS TRENDEK

- Már a 80-as években működött hálózatra kötött italautomata
- 90-es évek
  - Első okos otthon eszközök
  - Gép-gép (M2M) kommunikáció koncepció
- Dolgok (thing) az Interneten
- Egyre többféle eszközbe kerül bele a lehetőség
  - Talán nem lenne szükséges mindenbe…
  - Ipar 4.0, mezőgazdaság, stb.
  - Otthon, közlekedés, egészségügy, stb.
- Óriási biztonsági kihívás
  - Közvetlen fizikai hatások
  - Csökkenő emberi felügyelet

Kép: CEOWorld Magazine

MI KELL AZ IOT-HEZ

- Hálózat
  - Egyszerű architektúra
    - Thing: szenzor, aktuátor, okos-eszköz
    - Gateway: kapcsolódás a hálózathoz
    - Cloud: adat-feldolgozás, vezérlés
  - Mobilitás támogatás
  - Nagyszámú elem címezhetősége
  - Esetenként önszerveződő képesség
- Felhő, vagy köd – fog computing
  - Bizonyos számításokat helyben célszerű elvégezni
- Biztonság
  - Első sorban hálózati
- Adatkezelés és elemzés
  - Sok adat gyűlhet – big data
- Menedzsment és automatizálás

IOT TECHNOLÓGIÁK

- Meglévő protokollok továbbfejlesztése, vagy csak felhasználása
- Támogatás az OSI modell különböző rétegeiben
  - Az egyszerű architektúra miatt az alsó és a felső rétegek a kiemeltek
  - Biztonság támogatása jó lenne ahol csak lehet (TLS, DTLS)
  - IPv6 (6Lo)
- Példák:
  - Viszony réteg:
    - MQTT – Message Queue Telemetry Transport, adatküldés TCP/IP felett
    - AMQP – Advanced Message Queuing Protocol
    - CoAP – Constrained Application Protocol: RESTful
  - Adatkapcsolati réteg:
    - WiFi, 2G, 3G, 4G, 5G
    - Bluetooth Low Energy
    - ZigBee – kis fogyasztás, kis sebesség
    - LoRaWAN – Long Range Wide Area Network

IOT KIHÍVÁSOK

- Biztonság
- Mobilitás
- Megbízhatóság
- Skálázhatóság
- Menedzsment
- Elérhetőség
- Együttműködés

HÁLÓZATI RENDSZEREK
ÉS SZOLGÁLTATÁSOK
TANSZÉK