ชื่อ-สกุล ดลพัชร เหลืองไพ กระดาษแผ่นที่ 1 รหัสนักศึกษา 59010996

NW Device (Component)

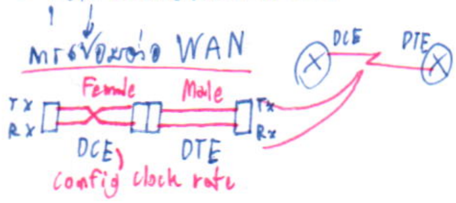
- End-device : PC, NB, printer, scanner
- Intermediary NW device : { NW Access devices / Internetworking devices / Security devices
- NW media : สายต่อ = Copper, Fiber, wireless
- Hub, Repeater : CSMA/CD
- Switches, Bridge : Learning/Flooding/Filtering/Forwarding/Aging
- Routers : Routing (หาเส้นทาง)

การเชื่อมต่อใน LAN

- สาย Crossover A-B
  - Switch ↔ Hub
  - PC ↔ Router
  - อุปกรณ์ชนิดเดียวกัน
- สายตรง (straight) A-A/B-B
  - อุปกรณ์ต่างชนิดกัน ยกเว้น

① ฟ้า ② เหลือง ③ เขียว ④ น้ำตาล
| : สีเต็ม
¦ : สีสลับ

การเชื่อมต่อ WAN



#4 Port @

- 0-1,023 : well know ports (IANA)
- 1,024-49,151 : registed ports
- 49,152-65,535 : dynamic/private ports
- dest. port ∈ well known port
- src. port ∈ สุ่ม

* มีเพื่อให้ข้อมูลที่ส่งกลับมาเข้าถูกเครื่อง

#3 IP @ (IPv4) Logical @

- Class A,B,C /8, /16, /24
- มี domain name แทนการอ่าน host ใน IP unique กันแต่ละ PC
- Private @ : IP ที่สงวนไว้ในแต่ละ class เพื่อใช้ภายในใน private NW (ไม่เชื่อมต่อ internet)
- header service

Ver.	IP Header Length	Differentiated Services: DSCP	ECN	Total Length
Identification			Flag	Fragment Offset
Time To Live	Protocol		Header Checksum	
Src. IP @				
Dest. IP @				
Options (optional)				Padding
8b	8b	8b	8b	

TTL ~ Time To Live

* ของ OS ใช้ TTL 7 bits (ฐาน 8)

NW Diagram : แผนภาพอธิบายภาพรวม NW

- Physical : บอกแผนผังการเชื่อมต่ออุปกรณ์ HW
- Logical : บอก IP/Port ที่แต่ละอุปกรณ์เชื่อมกัน

NW Topology : วิธีการเชื่อมต่อของ NW เช่น Mesh, Bus, Star, Ring, Fully Connected

NW Protocol : กฎในการสื่อสารบนเครือข่าย NW

* ในการทำงานจะใช้หลาย protocol ในการต่อร่วม ทำงานด้วยกันระหว่าง web server และ web client

NW @ :
- Port = Special
- IP @ = Logical
- MAC @ = Physical

NW Types

① แบ่งตามขนาด NW
- Small Home NW : เชื่อมต่อไม่มาก
- Small Office/Home Office : มีการเชื่อมต่อระยะไกล
- Medium to Large NW : เชื่อมต่อหลักพันเครื่อง หลายตึก
- World Wide NW : เชื่อมต่อหลักร้อยล้านเครื่อง เป็นเครือข่ายทั่วโลก

② แบ่งตาม infrastructures
- LAN : มี admin 1 คน/กลุ่ม
- WAN : มีหลาย admin ดูแล policy

Reliable NW

- Fault Tolerance : ต้านทานความผิดพลาด, ใช้งานได้แม้ HW เสีย
- Scalability : รองรับการขยายของระบบ
- Security : บุคคลภายนอก access เข้าไม่ได้
- Quality of Service (QoS) : แบ่ง NW ตามการใช้งาน

PDU	OSI model	TCP/IP model	Name System	Host Config	Email	File Transfer	Web
	Application	Application	DNS	BOOTP, DHCP	SMTP, POP, IMAP	FTP, TFTP, SFTP	HTTP
	Presentation						
Data	Session						
Segment	Transport	Transport	UDP		TCP		
Packet	Network	Internet	IP, NAT	IP Support: ICMP	Routing Protocols: OSPF, EIGRP		
Frame	Data Link	Network Access	ARP	PPP	Ethernet	Interface Drivers	
Bits	Physical						

#2 MAC @ Physical @

- Hex 12 หลัก (4 bits/หลัก) ไม่ซ้ำกันและเปลี่ยนไม่ได้
- IEEE กำหนด 3 bytes แรก เป็น Organizationally Unique Identifier (OUI)
- การส่งข้อมูล } ใน NW
  - Unicast : ส่งไปให้ 1 เครื่อง
  - Multicast : ส่งไปให้หลายเครื่อง
  - Broadcast : ส่งไปให้ทุกเครื่อง

* คำสั่ง

- ipconfig /all : ดู MAC @ ของ Ethernet adapter
- " /release : ปล่อย/ลบ ip ปัจจุบัน
- " /renew : ขอ ip ใหม่จาก DHCP
- nslookup : ดู Domain Name จาก IP (หรือย้อนกลับ)
- arp -a : ดู MAC @ ที่เครื่องเรารู้จัก
- route print : ดู NW ที่เครื่องเรารู้จัก

Cisco IOS

สำหรับการboot Router & Switch
- POST (Power On Serve Test) เช็ค HWครบ?
- Run boot loader SW
- Boot loader low-level CPU initialization (Mini OS)
- " " initializes from flash filesystem
- " " locates & loads default IOS

Cisco IOS device

- Console port (ตอนเริ่มแรก ต้องเข้าทางนี้)
- Telnet
- Secure Shell (SSH)
- Aux port
- * ไม่ใช่ I/O port

Navigating IOS

• โหมด:
- User EXEC >
- Privileged EXEC #
- Global config. (config)#
- Config ย่อย:
  - Interface (config-if)#
  - Routing Engine (config-router)#
  - Line (config-line)#

Basic Config.

① Host names : ตั้ง host name เพื่อให้ PC เข้าถึงได้โดยไม่ต้องเข้าไป console
* เริ่มอักษร, มีอักษร/-/เลข, จบด้วยอักษร/เลข ไม่มี space, ยาว ≤ 64 ตัว

② Limit Access to Device Config.
- ใช้ Banner Msg : บอก/แจ้งผู้ที่ access
- ตั้งรหัสความ Secure ในการ Access device
  - Enable password ccna
  - " secret password
  - Console password
  - VTY password
  - Encryp password ทั้งหมดตอนใน privileged

③ Addressing Devices
: config. interface, ip @
- interface type port
  - " " slot/port
  - " " slot/subslot/port
- ip address ip@ subnet
- no shutdown (ใช้หลัง config แต่ละครั้ง)

④ Verifying connectivity
: ตรวจสอบ/ยืนยันการ config.
- show running-config, interface, ...
- ping ip@ | - traceroute

Routing

Routing : ทำให้ package ส่งไปตามเส้นทางจนถึง dest. โดยอาศัย routing table (บอกเป็นปลายทาง dest.)

Boot-Up process

① Test router HW (PosT, Execute bootstrap loader)
② Locate & load Cisco IOS
③ — " — startup config. (จาก NVRAM)

NW Documentation

- Device names
- Interfaces
- IP & Subnet
- Default gateways (มักใช้ IP ...1/...254)

กำหนด IP ให้ host

Static Assign : config ip เองทีละเครื่อง (คำสั่ง)
Dynamic — " — : ใช้ DHCP กำหนดอัตโนมัติ

Basic Setting - Router

- ตั้งชื่อ device : hostname name
- ตั้ง secret password : enable secret password
- ตั้งค่า banner : banner motd # text #
- config Interface : interface type slot/port
  - ตั้ง IP : ip address x.x.x.x y.y.y.y
  - สั่ง Activated : no shutdown
  - ตั้งค่า serial DCE : clock rate 56000
- config Loopback Interface : (ไม่ใช้ no shutdown)
  - interface loopback number
  - ip address x.x.x.x y.y.y.y

→

Verify Interface Settings

- show ip interface brief
- " " route
- " running-config

(แสดงสรุป)

- " interfaces
- " ip interface

(detail เต็ม)

* Header ของ Layer 2 เปลี่ยน media dependence
  - LAN = MAC @
  - WAN = HDLC encap.

*  decap. ↓ process แปลงตามรอบ routing table ↑ encap

หา Best Path ของ Router

: เลือกทางที่ lowest cost (cost ต่างตามวิธี)
- Routing Information Protocol (RIP)
  > Hop count
- Open Shortest Path First (OSPF)
  > cumulative BW
- Enhanced Interior Gateway Routing Protocol (EIGRP) > BW delay load reliability
- Load Balance : แบ่งส่งไป 2 Path ต่างๆกัน เมื่อ 2 Path มี cost เท่ากัน.
- Administrative Distance (AD) : ระบุความน่าเชื่อถือ (ตั้งใกล้ admin ยิ่งน่าเชื่อ)

Routing Entries

รู้ NW ผ่านใคร
D (NW ปลายทาง [AD / cost] via IP next-hop | time | interface)

รู้ NW จากไหน
(NW ปลายทาง | interface)
L (Link Local)
C (Directly Connected)

* ในการเลือกดู AD > cost

Routing

- Static Routing
- Dynamic Routing
  - Exterior (EGP)
    > BGP
  - Interior (IGP)
    > RIP, OSPF, EIGRP, IS-IS, IGRP

Static Route

- ใช้ resource น้อยในการ maintain
- เปลี่ยนเส้นทางเองไม่ได้
- ผู้ config ต้องเข้าใจ NW ดี
- เหมาะกับ NW เล็ก/มีทางออกเดียว

มี 4 ประเภท
① Standard ② Default
③ Summary ④ Floating

ชื่อ-สกุล ดลพัชร์ เนื่องสุข กระดาษแผ่นที่ 2 รหัสนักศึกษา 59810996

for Staples

Config IPv4 Static Route

⇒ Router(config)# ip route dest-NW@ subnet next-hop@ | itf-name number

* ถ้าหา ip แล้วไม่ match เลย จะส่งออก default route → # ip route 0.0.0.0 0.0.0.0 next-hop@ | itf-name number

Config Interface

⇒ Router(config)# interfac itf-name number

—"— -if)# ip address ip@ subnet

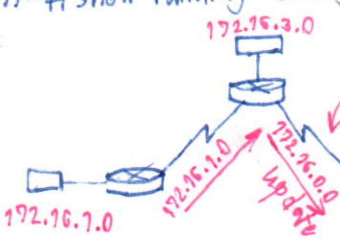
*เฉพาะ DCE serial —"— # clock rate 56000

—"— # no shutdown

Verify

⇒ Router# show ip route (none) | static | network หรือ

—"— # show running-config



เช่น ip = ~.~.0.0

update แบบ classful (ส่ง class) ∴ ทุก ip เป็น class เดียวกัน

Classful Addressing

Class	High Order Bits	Start	End
A	0xxxxxxx	0.0.0.0	127.255.255.255
B	10 ——	128.0.0.0	191. ——
C	110 ——	192.0.0.0	223. ——
Multicast	1110 ——	224.0.0.0	239. ——
Reserved	1111 ——	240.0.0.0	255. ——

CIDR (Classless Inter-Domain Routing)

: update class แบบ classless โดยทำ summarization (ดูจาก บิตที่เหมือนกันจากด้านหน้า ip)

* สั่ง ip route เฉพาะ summarized ip ได้ (Summary Route)

Fixed Length Subnet Masking (FLSM)

xxxxxxxx.xxxxxxxx.xxxxxxxx.sss hhhhh /27

NW Portion — Host Portion

: ความยาว subnet คงที่ตาม /~

Subnet Portion: $2^{8-(32-sm)}$ → 2^3 subnet(ชุด)

New Host Portion: $2^{32-sm} - 2$ → 2^5-2 host/subnet

(.000~ ... .111~) — (.~00000 ... .~11111)

Variable Length Subnet Masking (VLSM)

: การแบ่ง subnet ออกเป็น subnet ย่อย เพื่อให้ยืดหยุ่นในการใช้งาน โดยจะให้แต่ละเครือข่ายใช้ ip เท่าที่จำเป็นได้ (ไม่มี ip เหลือ) โดย ip ที่เหลือก็สามารถนำไปแบ่ง assign ได้ต่อไป

Dynamic Routing Protocols

> Function
- แชร์ข้อมูลระหว่าง router
- อัพเดท routing table เมื่อ router หรือ เส้นทาง (topology) เปลี่ยนแปลง
- หาทางไป dest. ที่ดีที่สุด

> จุดประสงค์
- ครอบคลุม NW ระยะไกล
- รักษาข้อมูลการอัพเดท routing
- หา best path ไปยัง dest.
- หา best path ใหม่ เมื่อ path เดิมไม่สามารถใช้งานได้

> องค์ประกอบ
- algorithm
- routing protocol message (info.)

	Dynamic	Static
Config. Complex.	ไม่ขึ้นกับขนาด NW	เพิ่มตามขนาด NW
ความรู้ admin	advance	ไม่มาก
การเปลี่ยน Topology	เปลี่ยนเส้นทางอัตโนมัติ	admin config ใหม่
Scaling	simple & complex topology	เฉพาะ simple topology
Security	น้อย	มาก
การใช้ทรัพยากร	CPU, mem, link BW	ไม่มีเพิ่มเติม
Predictability (การดูเส้นทาง)	ขึ้นกับ topology ตอนนั้น	เหมือนเดิมตลอด

Dynamic Routing Protocols

- Interior Gateway Protocols (เชื่อมภายในเครือข่าย)
  - Distance Vector Protocols
    - RIP → RIPv2
    - IGRP → EIGRP
  - Link-state Protocols
    - OSPF
    - IS-IS
- Exterior Gateway Protocols (เชื่อมระหว่าง NW)
  - BGP

*Autonomous System : กลุ่ม router ที่อยู่ภายใต้ signal authority

Distance Vector (distance & direction)
- route เป็น vector
- สร้าง NW topology ไม่ได้
- อัพเดทเป็นคาบ/ระยะ

Link State
- สร้าง NW topology ได้
- อัพเดทไม่เป็นคาบ

Load balancing
: การมีเส้นทางไปยัง dest. หนึ่ง 2 ทาง โดยจะส่งข้อมูลไปทั้ง 2 ทางเท่ากัน cost เท่ากัน, table มี 2 บรรทัด

Classful Routing Protocols
: ส่ง class เพื่ออัพเดท

Classless —"— —"—
: ส่ง subnet เพื่ออัพเดท ∴ NW size ต่างกัน

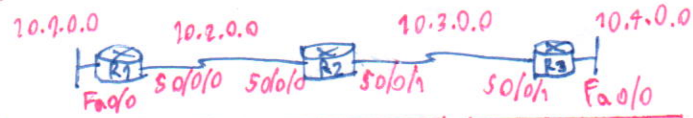
Convergence
: การที่ router ทุกตัวในระบบรู้ว่ามี router down
- Fast : EIGRP & OSPF
- Slower : RIP & IGRP

เลือกเส้นทางจาก
① Administrative Distance (AD) : ยิ่งน้อยยิ่งสำคัญ
② Metric : cost ในเส้นทาง

Distance Vector Routing Protocols

- Periodic update : ส่งข้อมูลให้ neighbor ทุก 30 sec.
- มี Neighbor (next-hop router)
- Broadcast update : 255.255.255.255
- มีการส่ง routing table ไป update

- ข้อจำกัดที่ต้องพิจารณา
  - เวลา convergence
  - scalability
  - การใช้ทรัพยากร
  - implement & maintainance
- state การทำงาน
  - Cold starts (หลัง config)
  - Initial Exchange routing info.
  - Exchange routing table



NW	Itf	Hop
10.1.0.0	Fa 0/0	0
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/0	1

NW	Itf	Hop
10.2.0.0	S0/0/0	0
10.3.0.0	S0/0/1	0

NW	Itf	Hop
10.3.0.0	S0/0/1	0
10.4.0.0	Fa0/0.	0
10.2.0.0	S0/0/1	1

} 2 บรรทัดแรกจาก basic config
} จาก update

* หลังจากตาราง update จะส่ง update ให้ neighbor อีกครั้ง

10.2.0.0	1
10.3.0.0	1

ตารางที่ส่งไป update
* hop count เพิ่ม 1
∴ ขอบระยะไกล

* NW ที่ซ้ำกันใน update จะใช้ข้อมูลที่ AD น้อยกว่า เช่น ที่มาจาก basic config จะ AD น้อยกว่าจากตาราง update

- Periodic Update
  - RIP update timer (default 30 sec)
  - Invalid timer 180 sec
  - Holddown timer 180 sec
  - Flush timer 240 sec
  - * hold state down จะแก้ปัญหา loop ได้
  - * routing info. แสดงเป็น 180 sec จะมีการ flush จริงเมื่อ 240 sec
- Bounded Update : EIGRP
- Triggered Update : อัพเดตเมื่อตารางมีการเปลี่ยนแปลง, ไม่รอ Periodic upd.
- Random Jitter : สุ่มอัพเดตเพื่อลดการชนกันของ update packet

- Routing Loops : เกิดได้จากการที่มี NW หนึ่ง down และ router ที่ใกล้ NW นั้นได้ลบ NW นั้นออกจาก routing table (อัพเดตตามปกติ) แต่ router อื่นๆ ยังไม่รู้ว่ามี NW down จึงส่ง update ซึ่งมี routing ของ NW ที่ down ทำให้เกิด loop ที่ไม่สามารถลบ routing ของ NW ที่ down ได้ และทำให้เกิดปัญหา count to Infinity ของค่า hop ∵ routing จะอัพเดตจาก serial เพราะมี AD เท่ากัน จึงทำให้มีค่า hop เพิ่มเรื่อยๆ
- วิธีแก้ Routing Loops
  1. Setting a Maximum : ให้ hop count สูงสุดได้แค่ 16 ถ้ามากกว่านี้คือ NW down, แก้ count to infinity
  2. Holddown Timers : hold สถานะ down ไว้ ไม่ update route ที่ down (NW)
  3. Split Horizon rule : ไม่ส่งข้อมูล NW เส้นทางเดิมกลับไปทาง itf. ที่ update (ที่รู้จัก NW นั้น)
  4. Routing Poisoning : ตั้ง NW ที่ down เป็น unreachable และส่ง poison ไป router อื่น
  5. ใช้ทั้ง split Horizon & poison reverse
  - * TTL ใช้เพื่อให้ packet ที่เกิด loop ตาย ∵ TTL หมด

RIP v1

- classful, Distance Vector P. ตัวแรก
- metric = hop count (max = 15)
- broadcast update ทุก 30 sec
- * default AD = 120

- Message

L2	L3	L4	L7
Data Link Frame h.	IP Packet h.	UDP segment h.	RIP message

512 Byte, ไม่เกิน 25 route

Bit 0 -- 7 8 15 16 23|24 31

Cmd = 1|2 Ver. = 1	Must be zero
@ Family identifier (2 = IP)	—— " ——
IP@[NW@]	
Must be zero	
Must be zero	
Metric[Hops]	
Multiple Route Entry (ไม่เกิน 25 route)	

RIP h. (มี 3 ช่อง)

Route Entry มี 3 ช่องสำคัญ
- @ Fam.
- IP@
- Metric

: มี 2 ประเภท
- Request : ส่งเพื่อเริ่มทำงานไปยังทุก itf ที่ enable เพื่อขอ routing table จาก neighbor
- Response : รับจาก neighbor พร้อม routing table

* RIP เป็น classful p. (ไม่ส่ง subnet)

- Basic Config. RIP v1

```
R1(config)# router rip
R1(config-router)# network ip@-NW1
——— " ———      # network ip@-NW2
```

/ enable NW

* ให้ R1 รู้จัก 2 NW นี้โดยใช้ NW-ID แล้ว router จะ learn ใน NW ส่วนของ

* อัพเดตแบบ split horizon

* config. ให้รู้จักทุก NW ที่เชื่อม itf ของ router นั้นๆ

- Verify ⇒ #show ip route|protocols, #show running-config
  - #debug ip rip ← แสดงการ update table เป็นระยะ
  - * info. เป็นลักษณะตาราง ip ของ NW พร้อมค่า metric และบอกว่าได้ข้อมูลของนี้มาจากทาง itf ใด

- ไม่รับ อัพเดต แบบเจาะจง itf

```
Router(config-router)# passive-interface name number
```

- Automatic Summarization
  : ยกเลิก rip โดย Router(config)# no router rip แล้ว config. rip ใหม่ตาม Basic Config. RIP, โดย protocol จะทำการ ip summarization ให้อัตโนมัติ สำหรับ ip ที่ subnet เหมือนกัน (summarize แบบ classful)

- Boundary Routers : กลุ่ม NW ของ router ที่ ip class ต่างกัน อยู่ใกล้กัน
  - summarize แบบ classful
  - boundary router (ตัวที่เชื่อมระหว่าง NW) จะ summarize subnet จาก 1 NW หลักส่งไปยังตัวอื่น
- RIP update : มี 2 ข้อ
  1. ส่ง routing update ใน NW เดียวกัน (class เดียวกัน) (boundary) จะส่ง subnet ของ itf ออกไปด้วย
  2. ส่ง routing update ข้าม NW (class ต่าง) จะส่ง classful subnet ของ NW ต้นทางไปด้วย
- ข้อดี : ลดขนาด routing entry & update และเวลา lookup
- ข้อเสีย : ไม่ support discontiguous NW (NW boundary เดียวกัน แต่อยู่คนละที่), เนื่องจาก split horizon จะทำให้ routing update ไม่ไป NW อีก boundary

- Default Route ⇒ (config) ip route 0.0.0.0 0.0.0.0 S0/0/1

ชื่อ-สกุล ดลพัช เหลืองสุวรรณ กระดาษแผ่นที่ 3 รหัสนักศึกษา 58010486

for Staples

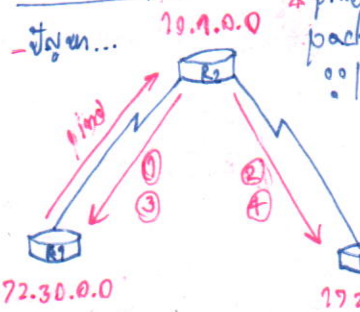
RIPv2

RIPv1	RIPv2
- classful, distance vector p. - ไม่ support discontiguous subnet - —— " —— VLSM - ไม่ส่ง subnet ใน update (NW@ only) - update เป็น broadcast (255.255.255.255) ไม่มี auth.	- classless, distance vector p. - มี next-hop @ & subn ใน update - update เป็น multicast - มี auth.

- ความเหมือนของ V_1 & V_2
  - ใช้ timer ป้องกัน routing loop
  - ใช้ split horizon & split horizon with poison reverse
  - ใช้ triggered update
  - max hop count = 15

- ข้อจำกัด RIPv1
  - Loopback itf.: ping ไปกลับได้, สร้างได้ } virtual itf.
  - Null itf.: ping แล้ว time out, ไม่ตัวตนจริง
    - *config ⇒ R(config)# ip route ip@ subnet Null0
  - Route redistribution: ทำให้ rip รู้จัก static
    - *config ⇒ R(config-router)# redistribute static
  - ถ้า classful เชื่อมแต่ subnet ต่าง จะส่งไม่ได้ ∵ บางครั้ง summarize แล้วดูเป็น over class

- ปัญหา...



*ping ไป R2 ตอบกลับ packet แบ่งไป 2 NW ∵ load balance

discontiguous ↗ *RIP1 ไม่ support ∵ auto summarization

- message RIPv2

0	7 8	15 16	23 24	31
cmd = 1/2	ver. = 2	Must be zero		

Route Entry:

@ Family identifier (2 = IP)	Route Tag
IP@ (NW@)	
Subnet	
Next Hop	
Metric (Hops)	
Multiple Route Entry ไม่เกิน 25	

- Config RIPv2
  - R(config)# router rip
  - ⇒ R(config-router)# version 2
  - *verify ด้วย ⇒ show ip protocols
  - หลัง config ยังมี auto summarization ยกเลิกได้โดยเข้า router rip แล้วใช้
  - ⇒ no auto-summary (ใช้เฉพาะ V2)
  - ใช้ debug ip rip ดูพฤติกรรมได้
  - *หลังยกเลิก auto sum. ใน routing table จะแยกออกมาเป็นหลาย entry ∴ ถ้าไม่เกิด discon. ก็ไม่จำเป็นต้องยกเลิก ∵ เปลือง table lookup

Access Control Lists (ACL)

: เป็น list เพื่อ match เพื่อควบคุมการเข้า/ออกของข้อมูล โดยดักที่ itf ของ router โดยเช็คทีละข้อตามลำดับการตั้ง (คล้าย if-else) ถ้าไม่ match อันไหนเลยจะ match implicit deny (deny any) ไม่ส่ง packet

มี 2 ชนิด (ใช้ควบคุม control & BW IOS compile seq.)

① standard
- เช็ค src-ip@ ของ packet ที่เข้า router
- permit/deny ทุก protocol ที่ match
- ⇒ access-list number permit|deny src-ip@ wildcard
- *config บน router ใกล้ dest. | - number : 1-99, 1300-1999

② Extend
- เช็คทั้ง src/dest.-ip@
- permit/deny โดยระบุ protocol ได้
- ⇒ access-list number permit|deny protocol|port src-ip@ wildcard dest.-ip@ wildcard protocol|port
- *config บน router ใกล้ src. | - number : 100-199, 2000-2699

- Wildcard Masks
  - : เป็น invert ของ subnet masks โดยนำ 255 ลบในแต่ละช่วงด็อต (ทุกช่วงด็อต)
  - - bit ตรง '0' → ตรงกัน
  - - bit ตรง '1' → เป็นอะไรก็ได้ 0/1
  - Ex. 192.168.00000001.00000000
  - 0 . 0 . 11111110 . 11111111
  - 192.168.xxxxxxx1.xxxxxxxx ← x คือ don't care
  - *0.0.0.0 = mark all, 255.255.255.255 = no mark

- กฎการใช้ ACL (F0/0, S0/0/0)
  - 1 ACL/protocol : กำหนดแยกแต่ละ protocol บน interface
  - 1 ACL/direction : inbound และ outbound ตั้งได้อย่างละ 1 ACL
  - 1 ACL/interface *สร้างได้ ACL มีได้แต่ apply ได้แค่ 4

- Config ACL : สร้าง statement list (global config) → apply (itf)
- สร้าง Std ⇒ R(config)# access-list access-list-number deny|permit|remark src src-wildcard
- ลบ ACL ⇒ no access-list
- สร้าง { Ex. R(config)# access-list 2 permit 192.168.1.0 0.0.0.255
  - —— " —— # access-list 2 deny any } *ถ้าเป็น implicit deny จะไม่รู้ว่าจะบล็อกอะไร

- apply แต่ละ itf
  - ⇒ R(config-if)# ip access-group number|name in|out
  - *ยกเลิก ⇒ no ip access-group ↑ ระบุว่าใช้ขาเข้า/ขาออก
  - *แก้ config เดิม ให้ยกเลิก (ทั้งหมด) และ config ใหม่ ∴ ควรเขียนใน T.E. (ลากเส้นใต้)
- verify ⇒ show ip interface name number
- ACL statistics ⇒ R# show access-lists
  - *แสดงข้อมูลการ match แต่ละข้อ

ACL (cont'd)

- config secure VTY port
  R(config-line)# access-class number in|out
  (in = เข้า, out = ออก จาก router)

- Extended ACL : เช็คการเข้าถึงจาก src./dest./protocol/port ได้
- สร้าง Ext. Ex. ⇒ access-list 114 permit tcp 192.168.20.0 0.0.0.255 any eq 23 ← เลข port
  ↳ * ใช้ชื่อได้ → telnet
  ↳ config ⇒ access-list number deny|permit|remark protocol src. [src.-wildcard] [operator operand] [port port-num|port-name] dest. [dest.-wildcard] [operator operand] [port port-num|port-name] [established]
  * ที่อยู่ใน [ ] สามารถละไว้ได้

- Processing Packets with ACLs * ดู table = ผ่าน process
  - Inbound ACL logic : เจอ ACL ก่อนดู routing table
  - Outbound ACL logic : ดู routing table ก่อนเจอ ACL

- DR & BDR : router ตัวแทนสำหรับรับ link-state มาประมวล สร้าง DB แล้วส่งให้ router อื่นๆ เพื่อลดการเกิด collision, ประหยัดการส่ง, ลด process * BDR = Backup DR
- Config. Single-Area OSPFv2
  ⇒ R(config)# router ospf process-id (* มีช่วง (1-65,535) * มีนัยสำคัญ)
  R(config-router)#
  - network ip@-itf-link wildcard area a-id (0)
  - ไม่ update ที่ itf : passive-interface
  - คำนวณ OSPF itf cost : auto-cost
  - router-id

Link-State Routing Protocols

- สร้าง map ของ NW Topology ได้
- สร้าง topology map และเลือก best-path ไปยังทุก dest. (cost = BW)
- ออกแบบเป็น hierachical
- มี Fast convergence
- มี link-state IPv4 IGP 2 ตัว คือ OSPFv2 และ IS-IS
- ใช้ Dijkstra's algo. หา shortest path first (SPF) ไปแต่ละ NW
- การ update
  - learn แต่ละ NW ที่เชื่อมโดยตรง (หา neighbor)
  - แต่ละ router จะส่ง msg ไปหาแต่ละ neighbor เพื่อทักทาย
  - สร้าง Link-state Packet (LSP) และ Flood ไปแต่ละ neighbor
  - สร้าง topology map ให้สมบูรณ์ และคำนวณ best path จาก DB
  - ใส่ OSPF routes ไปยัง routing Table

Dest.	Shortest Path	Cost
10.5.0.0/16	R1 → R2	22

- ข้อดี
  - แต่ละ router สร้าง topological map NW ของตัวเอง เพื่อหา shortest path
  - มี convergence เร็ว (แจ้ง router อื่น และหา SPF ใหม่)
  - LSP ถูกส่งเฉพาะเมื่อ topology เปลี่ยน, โดยส่งเฉพาะข้อมูลที่เปลี่ยน
  - ใช้กับโครงสร้างหลายชั้นได้
- ข้อเสีย
  - การเก็บ DB ใช้ mem. เยอะ
  - ใช้ CPU หนักตอนคำนวณ SPF
  - ใช้ BW เยอะตอน Flood LSP

* เชื่อมต่อแบบมี router ตัวแทน ทำให้ใช้ router น้อย, แก้ปัญหา link เส้นได้, หาแค่ SPF ไปแต่ละ NW น้อย

- OSPF เริ่มเชื่อมต่อ NW
  - สร้าง adjacency กับ neighbor
  - แลกเปลี่ยน routing info.
  - คำนวณ best route
  - เข้าถึง convergence (มีการเปลี่ยน state จนเข้าถึง convergence)

OSPF

* ค่า AD = 110, RIP ค่า AD = 120 OSPF จึงถูกเลือกก่อน RIP

- Command @ DB & table

DB	Table	Command
Adjacency	Neighbor t.	show ip ospf neighbor
Link-state	Topology t.	show ip ospf database
Forwarding	Routing t.	show ip route

- Step การทำงานของ OSPF
  - Find Link & Link State
  - Say Hello
  - Build LSP
  - Flooding LSP & Building DB
  - Building SPF tree & Routing table

* ใช้ได้ทั้ง Single Area และ Multiple Area (ต้องมี Backbone)
* Single Area นั้นจะใช้ Area 0

- Encapsulating OSPF message (header)

Data Link Frame h.	IP Packet h.	OSPF Packet h.	OSPF Packet Type-Specific DB

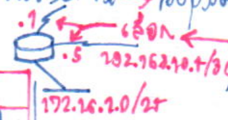
- OSPF Packets

Packet Name	คำอธิบาย
Hello	ทำให้รู้จัก neighbor และสร้าง adjacencies
DB Description (DBD)	เช็ค DB sync. ระหว่าง router
Link-State Request (LSR)	request เฉพาะ link-state record จาก router to router
Link-State Update (LSU)	ส่ง requested link-state record แบบเจาะจง
Link-state ACK (LSAck)	Acknowledge

* Hello packets ถูกส่งทุก (default + ค่าย)
  - 10 sec. บน multiaccess & point-to-point NW
  - 30 sec. บน non-board multiaccess NW

Establish Neighbor Adjacency: Down State → Init State → Two-Way State
Synchronize OSPF DBs: → ExStart State → Exchange State → Loading State
→ Full State ← * ได้ DB สมบูรณ์

- OSPF Router ID ⇒ R(config-router)# router-id 1.1.1.1 (id, เข้าหลัง router ospf p-id)
  * การ set id → ng loopback itf ip → ng Highest active ip (itf)



* ถ้า NW ที่เลือกใช้ ip down ต้อง update DB ∵ ip เปลี่ยน
* ในการแก้ต้องเข้า p-id เดิม
⇒ R(config-router)#network 192.168.10.4 0.0.0.3 area 0

- OSPF cost = ref. BW (10^8 bps) / itf. BW
  config ⇒ auto-cost reference-bandwidth BW_mbps

itf. type	itf. BW
10 Gigabit Eth. 10G	10^{10}
Gigabit Eth. 1G	10^9
Fast Eth. 100M	10^8
Ethernet 10M	10^7
Serial 1.544 Mbps	1.544×10^6
Serial 128 kbps	128×10^3
Serial 64 kbps	64×10^3

* cost ต่ำสุด = 1
- itf BW ปรับได้โดย เข้าไปที่ itf นั้นๆ แล้ว ⇒ R(config-if)# bandwidth BW
* ค่าเปลี่ยนแค่ค่าการคำนวณ แต่ทาง phy. ไม่เปลี่ยน
- OSPF cost : ปรับใน itf ⇒ ip ospf cost cost

- Redistributing OSPF Default Route
  R(config)# ip route 0.0.0.0 0.0.0.0 loopback N
  R(config)# router ospf p-id
  ⇒ R(config-router)# default-information originate
- Redistributing OSPF อื่นๆ
  ⇒ R(config-router)#redistribut ?

DHCP

DHCP : ให้ router แจก IP@ ให้แต่ละ host อัตโนมัติ รวมถึงข้อมูลอื่นๆ ได้แก่
- subnet mask (IPv4) / prefix length (IPv6)
- default gateway @
- DNS server @
- มีวิธี assign ip 3 แบบ (ใน DHCPv4)
  - Manual Allocation : assign โดย admin
  - Automatic Allocation : DHCP assign ip เพื่อถาวร
  - Dynamic Allocation : DHCP assign ตามลำดับการเปิด
- Config DHCPv4 (* ยกเว้นช่วง IP / IP เดี่ยว)
  ⇒ R(config)# ip dhcp excluded-address 192.168.10.1 192.168.10.9
  R(config)# ip dhcp excluded-address 192.168.10.255
  R(config)# ip dhcp pool LAN-POOL-1 ← pool name
  R(dhcp-config)# network NW-IP@ subnet
  R(dhcp-config)# default-router default-gateway-IP@
  R(dhcp-config)# dns-server dns-server-IP@
  R(dhcp-config)# domain-name www.~.com ← (อะไรก็ได้)
  * verify ⇒ show running-config | section dhcp
- PC cmd ⇒ ipconfig /release : ปล่อย IP เก่า
  ⇒ ipconfig /renew : รับ IP ใหม่ / เปลี่ยนจาก ip@ → dhcp
- Assign IP ให้ itf. ⇒ ip address dhcp ← ได้จากตัว router ใกล้สุด

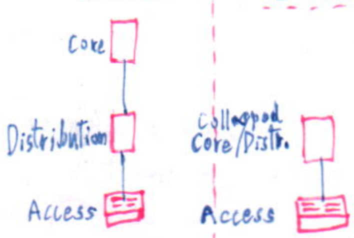
- ใช้ broadcast ข้ามไป ip ที่รับ ⇒ ip helper-address IP@

ชื่อ-สกุล ดวงพัชร เหลืองปู่น กระดาษแผ่นที่ 4 รหัสนักศึกษา 59010496

Basic Switch

- LAN : เครือข่าย NW ที่เชื่อมกัน subnet เดียวกัน admin คนเดียวกัน
- LAN design

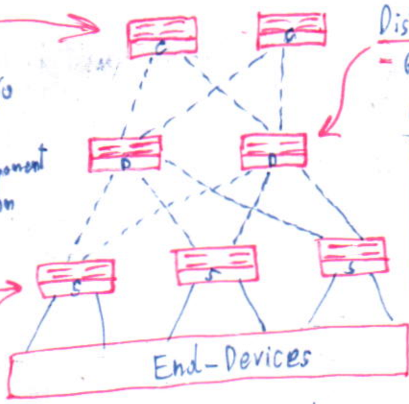
3-Tier | 2-Tier



- ปัจจัยการเลือกอุปกรณ์ใน NW
  - Cost : ขึ้นกับจน. & spd ของ itf.
  - Port Density : sw ต้องรองรับหลายอุปกรณ์ใน NW
  - Power มี PoE (Power over Ethernet)
  - Reliability : sw สามารถ access ได้อย่างต่อเนื่อง
  - Port Speed : spd ในการส่งใน NW
  - Frame Buffer : ความจุ ของการเก็บ frame
  - Scalability : รองรับการขยายตัว ของจำนวน user

Core layer

- Support L3
- Forwarding rate สูง
- มี Gig/10Gig
- มี Redundant component
- มี Link aggregation
- QoS

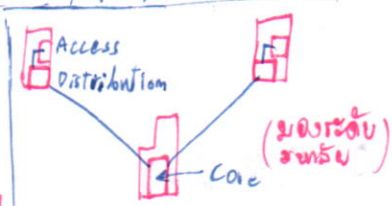


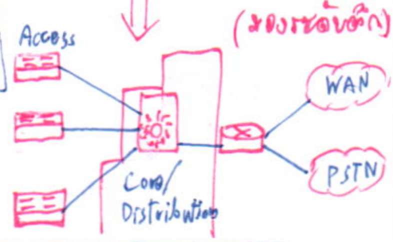
Access layer

- Port security
- VLANs
- มี Gig/Fa
- มี PoE
- มี Link aggregation เพื่อรวม 2 link ให้ BW สูงขึ้น
- QoS

Distribution layer

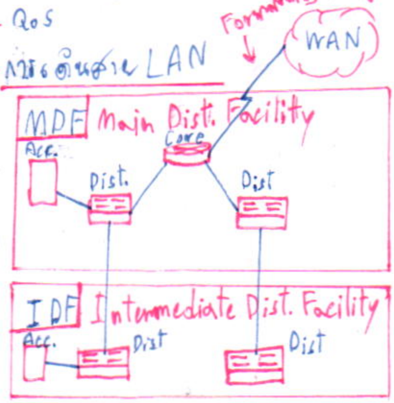
- แบ่งจาก Core layer เพิ่มมาคือ Security policy/ACL





* layer ของ switch ขึ้นกับระดับการมอง

- การเดินสาย LAN



- Server
  - Enterprise S. : ใช้ร่วมกัน, วาง MDF
  - Workgroup S. : ใช้เฉพาะกลุ่ม, วาง IDF

* MDF $\xrightarrow{fiber}$ IDF $\xrightarrow{LAN}$ Acc.

Switch Forwarding Method

① Store-and-Forward : อ่านทั้ง frame เพื่อเช็ค error ก่อนส่ง, เช็ค error ด้วย FCS, auto. buffering, forward ช้า

② Cut-Through Switching : อ่านบางส่วนของ frame เพื่อดู MAC@ แล้วส่ง, forward ทุก 10 ms, ไม่มี FCS และ auto. buf., Fast-Forward ~ 12 Byte, Fragment-Free ~ 64 Byte
* การอ่านแบบ 64 Byte จะลดการเกิด collision

Switching Domains (มีตัวส่ง packet)

① Collision D. : ส่วนที่ device ต่อกัน/เกิด collis.
- ทุก port hub อยู่ collision d. เดียวกัน
- ทุก port sw เป็น collision d. (แยกกัน เป็น collision d. ย่อยๆ)

② Broadcast D. : ส่วนที่อยู่ NW เดียวกัน แบ่งโดย itf router
- sw ส่ง broadcast frame ไปทุก port (ทุก port อยู่ BC เดียวกัน)
- ถ้า sw เชื่อมกัน จะส่ง BC frame ออกทุก port ทุก sw ที่เชื่อม

สรุป CD นับตาม itf router, BD นับตาม port sw (link เดียวนับ 1)

Switch Operation

* sw ทำงานที่ L2 ถ้า CRC เจอ error จะไม่ส่ง

* ส่งข้อมูลเฉพาะ port ที่ระบุปลายทางไว้ใน packet/frame

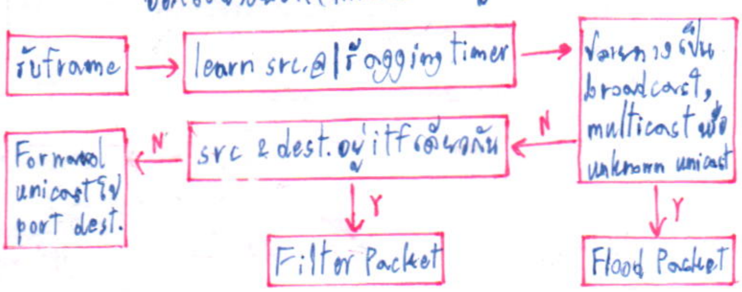
* ดู port ที่ออกจาก Src. @/Mac table

* MAC table เริ่มมาจะว่าง เติมโดยการ learn frame ที่เข้ามา โดย port 1:m MAC@

sw port : dest

port	MAC@

- Learning : learn src. จาก frame ที่เข้ามา
- Agging : เวลาในการเก็บข้อมูลของ src. หนึ่งๆ ถ้าไม่มีข้อมูลจาก src. นั้นนานๆ จะลบข้อมูล src. นั้น (บาง sw ไม่มี)
- Flooding : กรณี sw ไม่รู้จัก MAC@ หนึ่ง จะทำการ Flood ออกทุก enable port, การ Flood ทำเมื่อ 1) broadcast, 2) multicast, 3) unicast (unknown)
- Forwarding : การส่งข้อมูลไป dest. โดยดูจาก MAC@
- Filtering : ป้องกันไม่ให้ข้อมูลที่ส่งให้ dest. ภายใน NW ออกไปข้างนอก (MAC@ นั้นอยู่ใน NW)



Config switch

* ทำเป็น config term. โหมด

itf → interface vlan vlan-num
ip address ip@ subnet
no shutdown
* ทำให้ remote ใน LAN ได้

default gateway
→ ip default-gateway ip@
* ทำให้ remote จากนอกได้

Duplex Comm.
→ interface type num
duplex full | half
speed 100

Auto-MDIX * ทำให้ต่อสายอะไรก็ได้
→ inteface type name
duplex auto
speed auto
mdix auto

Verify → show ip itf-id
→ show mac-address-table

SSH → line vty 0 15
transport input ssh
login local

Verify SSH → show ip ssh
→ show ssh

Port Security * Secure sticky = disable, Max MAC = 1, Violation = shutdown
- Static → switchport mode access
  — " — port-security
  — " — " — mac-address MAC@
- Dynamic → ทำจำแต่เรียน MAC@ เป็น sticky (จาก static)
- Max MAC@ → switchport port-security maximum num
- Violation → switchport port-security violation protect | restrict | shutdown
- Verify Port Secure → show port-security type num

①protect : กัน access port ②restrict : protect แล้วส่ง log ③shutdown : ปิด port เข้าสู่ภาวะผิด secure.

ARP (Address Resolution Protocol)

: แปลง IP → MAC @

- ส่ง IP ; IP & MAC อยู่ NW เดียวกัน
- ส่ง gateway ; IP & MAC อยู่ต่าง NW

* ใช้ arp -a ใน cmd ดู IP ที่ map MAC ใน ARP Cache ซึ่งข้อมูลแต่ละชุดจะหายไปหากไม่ใช้งานระยะหนึ่ง

* ในการส่งข้อมูล ARP request และ ARP reply โดย frame ของทั้ง 2 ต่างกันที่ OPCODE จะเป็น 0x0001 และ 0x0002

* Ethernet frame จะใช้ Broadcast ARP โดยมี dest@ ARP frame ระบุ target IP ไว้ท้าย frame (32 bit)

IPv4 (เรื่อง subneting) มี 2 แบบคือ Classful และ CIDR

- CIDR (Classless Inter-Domain Routing): แบ่ง IP ได้ยืดหยุ่นกว่า
  - Fixed Length Subnet Masking : แบ่ง subnet เท่ากัน (/เท่า)
  - Variable — " — " — " — : แต่ละ subnet ขนาดต่าง (/ต่าง)
- แต่ละ NW ใช้ IP = host IP + 2 → ขนาด 1 block (2^n)

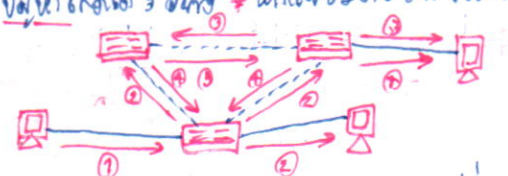
NW	Host	All IP	Subnet Mask
A	126	128	255.255.255.128
B	62	64	— " — .192
C	17	32	— " — .224

ต้องมากกว่าเป็น 2^n ใกล้ๆ จน. ที่ใช้

* วาง block ใหญ่ก่อน

LAN redundancy : มีหลาย link (ทาง) เชื่อมไปยัง sw ตัวหนึ่ง

- ปัญหาเกิดได้ 3 อย่าง * หากเชื่อมต่อ sw เป็น loop



* ถ้า PC ส่ง BC เพื่อ ARP ในตอนแรก
* sw Flood BC ออกทุก port

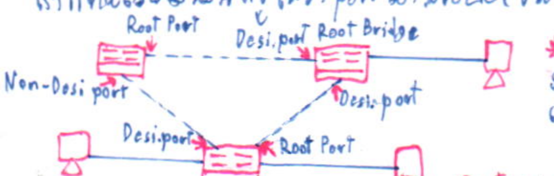
① MAC DB Instability: MAC@ table เปลี่ยนตลอด ไม่เสถียร, sw เกิดการอัพเดท entry เดิม หากมีการรับ frame จาก 2 ที่ (อัพเดทซ้ำซ้อน)

② Broadcast storm: เกิด loop flood ไม่รู้จบ ∵ ไม่มีการลด TTL เหมือนใน router (ใช้งานไม่ได้จนต้องถอด sw ปิด)

③ Multiple Frame Transmissions: PC ได้รับ frame เดิมซ้ำ ∵ มีการ flood ออกทุก port ของ sw

- Spanning Tree Protocol (STP) * ทำแยก VLAN ได้

: block บาง port เพื่อไม่ให้เกิด loop ซึ่งทำให้แก้ 3 ปัญหาได้ ถ้า link เสีย สามารถรู้ได้ว่า port ไหน block (ข้างทาง)/forward



* ก่อนเกิด Root นั้น sw ทุกตัวมองตัวเองเป็น RB

(Root แล้ว)

① Root Bridge : มี 1/NW เลือกจาก Bridge ID ที่น้อยสุด มี 2 ส่วนคือ Priority และ MAC@ (มีเพื่อคำนวณ RB เท่านั้น) โดยดู Priority ก่อน

② Root Port : มี 1 port/Non-RB หา path cost

③ Designated Ports : มี 1 port/segment (collision domain)

④ Non-Designated Ports
⑤ Alternate and backup Ports } block ports

* เริ่มมา ทุก port อยู่ VLAN 1 (default)

- Path cost: เป็น cumulative path cost ระหว่าง RB กับทุก port ของ sw (port/link ที่ติด RB มี cost = 0)

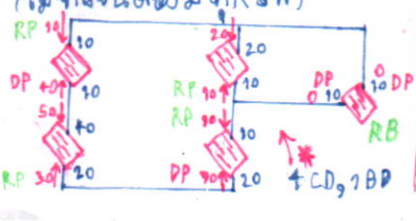
- STP operation

① หา RB จาก min Bridge ID
② หา Path Cost แต่ละแต่ละ sw
③ หา RP แต่ละ sw ดูจาก min cost ของ sw port
④ หา DP ในแต่ละ segment ที่มี RP ดูจาก min cost ใน segment ที่ไม่ใช่ RP

* ถ้า cost เท่ากัน เลือกจาก port ของ RB ที่เลขน้อย เป็นสำคัญ RP

* RP คือ port ที่ใกล้ที่สุดกับรับข้อมูลจาก RB (ในแต่ละตัว)

* DP คือ port ที่ min cost และไม่เป็น RP ใน แต่ละ segment (ไม่จำเป็นต้องมีทุก sw)



- ยกเลิก port
→ interface itf
  no switchport access
  -vlan
- ลบ VLAN
→ no vlan vlan-id
* port ยังคงอยู่ VLAN เดิม ยกเลิกแยกกัน

PVST+ : แบ่งภาระ load ของ RB ไปแต่ละ VLAN (มี RB แยก VLAN)

Bridge Priority	MAC@
2 Byte	6 Byte

⇓

Bridge Priority	Extend System ID	MAC@
4 bit	12 bit	

* ต่อมาพัฒนาเป็น Rapid PVST+ (เร็วขึ้น) เปลี่ยน block port เป็น alternate port เพื่อรับส่งเร็วขึ้น

* มี Edge ports หมายถึง link ที่ไม่ได้เชื่อม sw เพื่อให้ไม่ต้อง spanning tree

- Config STP
→ spanning-tree VLAN num root primary ← ให้เป็น root ในตอนเริ่ม
→ — " — priority priority num ← ตั้ง priority โดยตรง
→ — " — root secondary ← เป็น root ถ้า primary เสีย

- Verify STP → show spanning-tree * show cdp neighbors ใช้สร้าง topology ได้

- Config ส่วนเสริมของ STP
→ spanning-tree portfast
— " — bpduguard enable } Port fast & BPDU guard (เข้า itf)

Config Load balance
→ spanning-tree vlan vlan1 root primary
— " — vlan2 " secondary
— " — vlan1 " primary
— " — vlan2 " secondary
} วิธี① แยก primary & secondary ไว้ vlan กัน
→ — " — vlan1 priority p-num
— " — vlan2 " "
} วิธี② แยก priority ให้เท่ากัน 2 vlan

Config Rapid PVST+ * ทำใน conf. ter.
→ spanning-tree mode rapid-pvst
interface type num
spanning-tree link-type point-to-point

* หากทำ VLAN ใช้ manage sw, sw ธรรมดาทำไม่ได้ ซื้อแพงกว่าที่ใช้เยอะ

VLAN & InterVLAN : Virtual LAN คือการแบ่ง sw ออกเป็นหลาย NW ใน 1 ตัว โดยจะกำหนดว่า port ใดอยู่ NW ใด (NW ⇔ มี BC เดียวกัน), ส่วน Inter VLAN คือการเชื่อม VLAN เดียวกันบนต่าง sw หรือด้วยกันโดย Router

- Trunk: เชื่อม VLAN เดียวกันบนต่าง sw (port ต้องอยู่ VLAN เดียวกัน) หรือเชื่อม Trunk บน sw กับ router เพื่อทำ Inter VLAN (sw มีเพียง port VLAN ได้ทีละ 1)

* sw ที่ทำ VLAN จะ flood BC เฉพาะ VLAN นั้นไม่ใช่ทุก port อย่างเดิม

- IEEE 802.1Q : แทรก tag เพื่อให้รู้ว่า frame มาจาก VLAN ใด, FCS คำนวณ CRC ใหม่ จาก dest. MAC@ – Data (ยกเว้น Native VLAN ที่ไม่ต้องติด tag → ไม่เช็ค VLAN)

Dst MAC	Src MAC	Tag	Type/Length	Data	FCS

Eth. Type (0x8100)	Pri	C	VLAN ID

← * Tag

* sw 2960 & 3560 ทำ VLAN ได้ 4,000 VLAN

① Normal Range VLAN: เลข 1-1005, เก็บใน vlan.dat (ใน flash) มี VTP
② Extend Range VLAN: เลข 1006-4096, เก็บใน running-config (ใน NVRAM)

- สร้าง VLAN
→ vlan vlan-id
  name name

- กำหนด port
→ interface itf | range itf 0/m-n
  switchport mode access
  — " — access vlan vlan-id

- Verify VLAN → show vlan ← verify ด้วย

* เพิ่ม name name ได้

- Config Trunk * อาจเพิ่ม → switchport native vlan id
→ interface
  switchport mode trunk → " — allowed vlan id1, id2, ...

- Verify Trunk → show interface itf switchport

- Config Inter VLAN
① → สร้าง VLAN, กำหนด port, config Trunk ที่ sw
② → no sh (อาจไม่ใช้ IP) itf ที่ router ที่เชื่อม Trunk และสร้าง sub itf โดยแบ่งตามให้ VLAN
→ interface itf num.sub
  encapsulation dot1q vlan-id
  ip address ip@ subnet

ชื่อ-สกุล ดลพัชร เหลืองปั้น กระดาษแผ่นที่ 5 รหัสนักศึกษา 59010496

for Staples

VTP (VLAN Trunk Protocol)

: แก้ปัญหาการสร้าง/แก้ไข VLAN ในsw หลายตัวไม่สะดวก โดยกำหนดให้ sw ตัวเป็น server มีหน้าที่จัดการ/แก้ไข และอัพเดทให้ sw ตัวเป็น client (เชื่อมต่อกันด้วย Trunk) * ไม่รวมการกำหนด port

- สามารถ เพิ่ม/ลบ/เปลี่ยนชื่อ VLAN บน domain เดียวกัน
- VTP msg วิ่งผ่านลิงก์ trunk ISL/IEEE 802.1Q
- Revision num. บอกversionการแก้ config vlan มีค่าจาก 0 – 4,294,927,295 (32 bit)

Feature \ Mode	Server	Client	Transparent
Src VTP msg	/	/	x
Listen to VTP msg	/	/	x
สร้าง VLAN	/	x	/*
ลบ/เปลี่ยน VLAN	/	x	/*

* ไม่มีผลกับ VLAN ใน domain, มีผลเฉพาะ sw

- Config VTP ทำได้ 2 mode
  1. global config. → configure terminal
  2. VLAN config. → vlan database
  - → vtp version 2 ← เลือก ver
  - " domain name } สร้าง domain
  - " password pass }
  - " mode vtp-mode ← เลือก mode
  - * ทำ. global config. mode

- Verify VTP
  - → show vtp status
  - → show vtp counters
  - * เริ่มต้น revision num. = 0
  - * VTP domain เชื่อ msg ที่มี revision num. มากสุด ∴ นำต่อ sw เพิ่มใน Topology ควร clear config.

- VTP pruning : เป็นการเพิ่ม NW BW โดยลด traffic จากการ flood
  - * VLAN 1 เป็น pruning ineligible เสมอ
  - → vtp pruning ← vlan mode
  - → interface itf ← global config. mode
    switchport trunk pruning vlan remove vlan-id

NAT : การ map private ip กับ public ip

(NW @ Translation) * now ใช้ PAT แทนเกือบ NAT

- มี address 4 ประเภท

Local SA In	Local DA Out	→	Global SA In	Global DA Out

- • Inside local @ } เปลี่ยน src จาก private → public
- • – " – global @ }
- • Outside local @ } dst @ เหมือนเดิม
- • – " – global @ }

* แปลงโดย NAT table

① Static NAT : map private กับ public ip แบบ 1 ต่อ 1 โดย fix คู่

- Config Static NAT
  - → ip nat inside source static local@ global@ ← สร้าง map ip@ (1 คู่)
  - interface itf
  - ip nat inside } ระบุ itf ฝั่ง inside
  - interface itf } และ outside
  - ip nat outside }
- Verify static NAT
  - → show ip nat translations
  - → – " – statistics
  - * clear ip nat – " – อาจใช้ก่อนใช้ show stat.

② Dynamic NAT : เหมือนกับ static NAT แต่ไม่ Fix คู่ โดยจับคู่ map จาก public ip ที่ว่าง

- Config Dynamic NAT
  - → ip nat pool name start-ip -end-ip subnet ← สร้าง ip pool
  - access-list acl-num permit src-ip -wildcard ← สร้าง ACL
  - ip nat inside source list acl-num -pool name ← กำหนดใช้ NAT
  - interface itf } ระบุ itf ทั้ง 2 ฝั่ง
  - ip nat inside }
  - interface itf }
  - ip nat outside }
- Verify Dynamic NAT
  - → show ip nat translations
  - * อาจเพิ่ม verbose
  - → show ip nat statistics

③ PAT (Port @ Translation) : map src port กับ src ip ทำให้ map หลาย private ip กับ 1 public ip ได้

- Config PAT มี 2 วิธี
  1. → config เหมือน Dynamic NAT แต่เพิ่ม 'overload' ในขั้น กำหนด NAT ที่ท้ายคำสั่ง
  2. → access-list acl-num permit src-ip -wildcard ← สร้าง ACL
     - ip nat inside source list acl-num -interface itf overload ← กำหนดใช้ PAT
     - interface itf } ระบุ itf ทั้ง 2 ฝั่ง
     - ip nat inside }
     - interface itf }
     - ip nat outside }
- Verify PAT → เหมือน NAT

EIGRP (Enhanced IGRP)

- • เก็บ neighbor & topology table
- • คำนวณ metric ด้วย DUAL
- • interfacing DUAL & routing table
- • รองรับการ filter & ACLs
- • รองรับการ redistributed ต่าง protocol

Feature	
Diffusing Update Algorithm (DUAL)	ใช้ในการหาเส้นทาง (routing algo.)
Establishing Neighbor Adjacencies	สร้างความสัมพันธ์กับ EIGRP router ที่เชื่อมต่อโดยตรง
Reliable Transport Protocol	RTP ทำให้แน่ใจการส่ง packet ระหว่าง neighbor
Patial & Bounded Updates	ส่งข้อมูลเฉพาะเมื่อ path หรือ metric เปลี่ยน (ไม่เป็นรอบ)
Equal & Unequal Cost Load Balancing	ทำ load balance ได้แม้ cost ไม่เท่ากัน

* รองรับ IPv4, IPv6, IPX และ Apple Talk

- • RTP เป็น EIGRP Transport layer protocol (L4), ไม่ใช่ทุก RTP ที่เป็น reliable
- • EIGRP รองรับ authentication

- Packet type
  - • Hello : เพื่อค้นหา EIGRP router ใน NW (ส่ง unreliable, แบบ multicast)
  - • Update : เพื่อส่งข้อมูล routing ให้รู้เส้นทางไปแต่ละ dest.
  - • ACK : ตอบรับ EIGRP packet
  - • Query : request ข้อมูลจาก neighbor router
  - • Reply : ตอบ Query packet

- EIGRP Message : EIGRP packet (protocol 88) ใช้ parameter เป็น 0x0102 ; IP Internal router, เป็น 0x0103 ; IP External router ภายในมี Header (Opcode), Packet type และ AS number

- Config EIGRP * Process ID, ถ้าต้องการให้อยู่ domain เดียวกัน ต้องเลขตรงกัน
  - → router eigrp AS-num
  - network NW-IP@ [wildcard]
  - eigrp router-id 2.2.2.2
- Config Passive Itf → passive-interface itf
- Verify EIGRP → show ip eigrp neighbors
  - → show ip protocols

for Staples

for Staples

- EIGRP initial discovery

1) เมื่อ router เข้าร่วม EIGRP routing domain จะทำการส่ง Hello packet ไปยังทุก EIGRP itf
2) router อื่นๆ เมื่อได้รับ Hello จะเพิ่ม router ตัวใหม่ไปยัง neighbor table
   ส่ง Update packet ที่มีข้อมูล route
   ส่ง Hello packet กลับไปยัง router ใหม่
3) router ใหม่ เพิ่ม router อื่นๆ ที่ส่ง Hello packet มา ไปยัง neighbor table
4) router ใหม่ เพิ่ม update entry ไปยัง topology table (topology table มีเส้นทางทั้งหมดโดย neighbor router และ cost เพื่อไปยังแต่ละ NW)
5) router ใหม่ ส่ง ACK กลับไปบอก router อื่นๆ ว่า ได้รับ update แล้ว (∵ EIGRP update เป็นการส่งแบบ reliable)
6) router ใหม่ ส่ง EIGRP update ไปยังทุก router ที่รู้จัก ยกเว้น router ที่ learn มา (split horizon)
7) router อื่นๆ รับ update จาก router ใหม่ และเพิ่มข้อมูลไปยัง topology table
8) router ที่รับ update ส่ง ACK กลับ router ใหม่

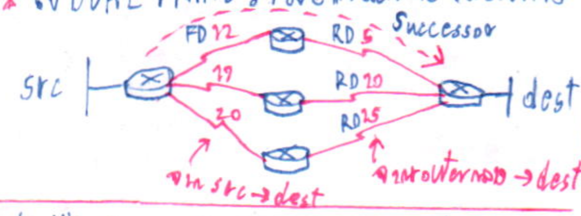
- การคำนวณ EIGRP metric

- คำนวณด้วย DUAL เพื่อหา best path จาก EIGRP metric ได้แก่ BW, Delay, Reliability และ Load (พร้อมค่าคงที่ K_1-K_5) ด้วยสูตร
- $metric = [K_1 \times BW + K_3 \times Delay] \times 256$
- $metric = [K_1 \times BW + (K_2 \times BW)/(256 - load) + K_3 \times Delay] \times [K_5/(reliability + K_4)]$
- คำนวณอย่างง่าย ⇒ $metric = (BW + Delay) \times 256$
  โดย $BW = 10^7/\text{slowest BW}$ (Min BW จากทั้ง EIGRP domain)
  $Delay = \Sigma Delay/10$ (ผลรวม Delay ทั้ง EIGRP domain)
- ดูค่าต่างๆ ในการคำนวณ metric ด้วย → show interface itf
- แก้ BW ใน itf (config-if)# → bandwidth kilobits-value | เรียกได้ ∴ เลือก S&FS router
- Delay วัดจาก เวลาที่ใช้ในการส่ง packet

- DUAL & topology table : มีการเก็บ term (role) ของ router ต่างๆ เพื่อ fast convergence

Term	
Successor S	neighbor router ที่ใช้เส้นทางสั้นสุด (least-cost) ไปยัง dest NW
Fessible Successor FS	ทางสำรองไป dest ซึ่ง RD < FD และไม่ใช่ S (มี >1 ทางได้)
Reported Distance RD	ค่าที่บอกจาก neighbor ว่า cost เท่าไหร่
Fessible Distance FD	ทางที่ cost น้อยสุดไป dest NW (cost ของ S)

* ใช้ DUAL Finite State Machine เลือกทาง



→ show ip eigrp topology
แสดงทุก S และ FS ไปยัง dest
→ show ip route (บางครั้งแค่ S)
แสดง router ของ S และค่า FD
→ show ip eigrp topology all-links
แสดงทั้ง S, FS, not-FS

IPv6 : มี 128 บิต (v4 = 32 บิต) ≈ 340×10^{36} @ (undecillion)

การเปลี่ยนมี 3 แบบ 1) Dual stack : ใช้ทั้ง IPv4 และ IPv6, 2) Tunneling NW ใหม่/นอกเป็น IPv6 ส่วน core หลักเป็น IPv4 โดยสร้างอุโมงค์, 3) Translation ใช้ NAT64 ซึ่งอนุญาตให้ IPv6 คุยกับ IPv4 ได้ (คล้าย global & local concept)

- รูปของ IPv6 @

- จาก 128 บิต แบ่งเป็น 8 Hextet โดย 1 Hextet มี 16 บิต หรือ 4 hex คั่นด้วย ':' ในแต่ละ Hextet

1) Omit leading 0 : ละ 0 ด้านหน้าออกได้
2) Omit all 0 segment : วาง "::" แทนส่วนที่เป็น 0 ทุกตัว โดยจากกี่บิตติดกันก็ได้ แต่ใช้ได้จุดเดียวใน IP

- Prefix length เป็นได้ตั้งแต่ 0 - 128 โดย NW เล็กสุด มีขนาด 64 บิต (ไม่มี subnet)

- IPv6 Unicast @ มี Global Unicast, Link-local, Unique Local

- Static config. ของ Global Unicast @

config ของ itf → ipv6 address ipv6@/prefix-length

- Host config. IPv6

- Manual เหมือน IPv4 โดย default gateway ต้องอ้างอิง link-local หรือ global unicast ของ Gi itf
- Dynamic โดย SLAAC (stateless) หรือ DHCPv6 (stateful)

- IPv6 Static Route config.

→ ipv6 unicast-routing (enable ipv6)
→ ipv6 route ipv6@/prefix next-hop ipv6@ | exit-itf

- Verify IPv6

→ show ipv6 route static
→ show ipv6 route ipv6@

- EIGRP IPv6 config

→ ipv6 unicast-routing
ipv6 router eigrp AS-num
eigrp router-id 2.0.0.0 ↙ v4
no shutdown

- ตั้งค่าที่ itf → ipv6 eigrp AS-num
* ไม่ต้องใช้คำสั่ง network

- Verify EIGRP IPv6

→ show ipv6 eigrp neighbors

- Default IPv6 Route

→ ipv6 route ::/0 next-hop ipv6@ | exit-itf