“LANs”

- توی این قسمت می خوایم راجع به شبکه های محلی سوئیچی یا switched local area network یا switched LAN صحبت کنیم. شمایی از این شبکه ها توی شکل زیر نشون داده شده :

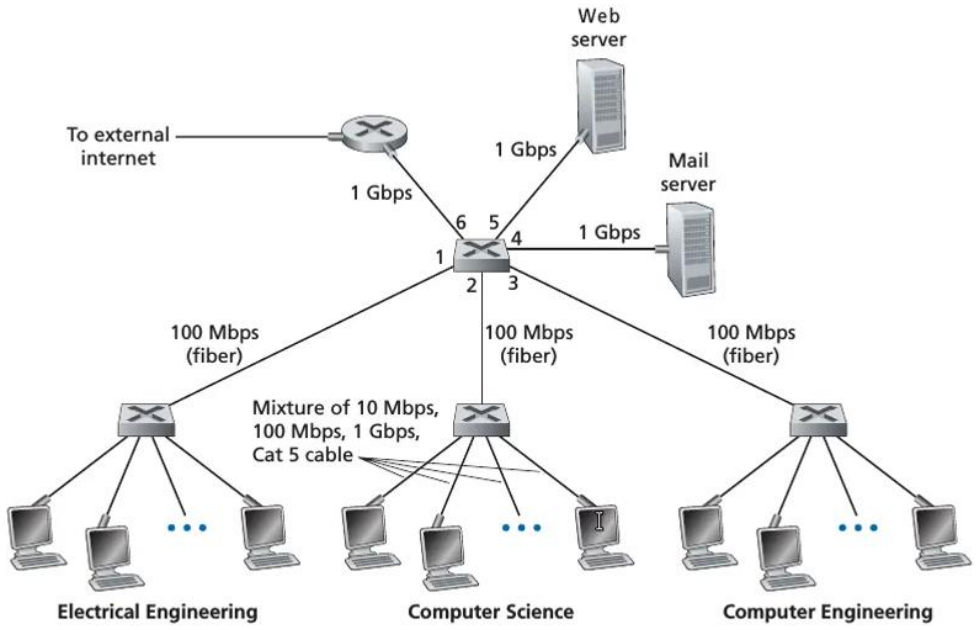


Figure 6.15 ♦ An institutional network connected together by four switches

شبکه هایی هستن که اتصالات بین host ها در داخل شبکه ی محلی مون توسط وسایلی به نام switch شکل می گیره. حالا این سوئیچ ها ممکنه خودشون سلسله مراتبی باشن(مثل شکل بالا) یعنی ابتدا host ها به یه سری سوئیچ متصل میشن و خود این سوئیچ ها از سوئیچ های دیگه ای با پهنای باند بزرگتر استفاده می کنن تا ترافیکشون نهایتا

تجمیع بشه و به gateway سازمان یا موسسه ای که این شبکه مربوط به اون سازمان یا موسسه هست، منتقل بشه.

- سوئیچ ها وسایل لایه ی 2 هستن ، یعنی تا لایه ی 2 رو می فهمن و درکی از لایه ی 3 و لایه های بالاتر ندارن و برای فورووارد کردن frame ها از IP address استفاده نمی کنن. به جای آدرس های IP ، از آدرس های فیزیکی یا MAC address در سوئیچ ها برای فورووارد کردن frame ها استفاده میشه.
- سیر مطالب توی این جلسه :

1 - Addressing و پروتکل ARP که برای تبدیل آدرس IP به MAC address استفاده میشه.

2 - Ethernet

3 - جزئیات عملکرد switch ها

- MAC addresses

- آدرس فیزیکی یا MAC address آدرسیه که مشابه آدرس IP به هر interface داده میشه اما سه تفاوت با آدرس IP داره :

1 - آدرس های IP ورژن ۴ ، ۳۲ بیتی هستن ولی MAC address ها (حداقل اونایی که توی Ethernet و WiFi استفاده میشن) ۴۸ بیتی هستن.

2 - آدرس های IP به صورت dotted-decimal نمایش داده میشن ولی هر بایتِ این ۴۸ بیتِ MAC address رو به صورت اعداد در پایه ی هگزادسیمال نشون می دیم و بین هردوتا بایت مجاور هم یه علامت – قرار می دیم. پس ۶ تا عدد که در پایه ی هگزادسیمال نمایش داده شدن ، داریم .

3 - ساختار MAC address ، flat هست برخلاف آدرس های IP که به صورت سلسله مراتبی بودن.

4 - MAC address در سرتاسر شبکه برای هر interface به صورت یکتا هست و با تغییر محل اتصالش به اینترنت یا با تغییر شبکه MAC address عوض نمیشه.

- دلیل استفاده از MAC address در کنار IP address چیه؟

این دوتا آدرس کاربرد مجزا دارن. مثل خودمون که یه آدرس پستی داریم که برای این استفاده میشه که بسته ها در زمان و محل مشخصی به دست ما برسن ، و همچنین کد ملی. کاربرد کد ملی با آدرس پستی متفاوته و ما به هردوش نیاز داریم.

دلیل استفاده از آدرس IP اینه که محل یه دستگاه به طور یکتا در کل شبکه ی اینترنت مشخص بشه و اگه می خوایم برای یه دستگاهی بسته

ارسال کنیم بسته ها به دست اون دستگاه برسه. بنابراین طبیعیه که اگه ما شبکه یا subnet مون رو عوض کنیم باید یه آدرس IP جدید بگیریم که نشون بده ما الان داخل اون شبکه هستیم.

دلیل استفاده از MAC address اینه که با استفاده از آدرس فیزیکی کارایی دستگاه ها در شبکه های محلی بالاتر میره. برای این که این قضیه رو توضیح بدیم باید به این نکته توجه کنیم که در شبکه های محلی frame هایی که توسط یک نود برای یه مقصد به خصوص که در همون شبکه محلی قرار داره، ارسال میشن ، این امکان وجود داره که اون frame تنها توسط نودِ مقصد ، دریافت نشه ؛ توسط سایر نود هایی که در شبکه ی محلی هستن هم اون frame دریافت بشه.

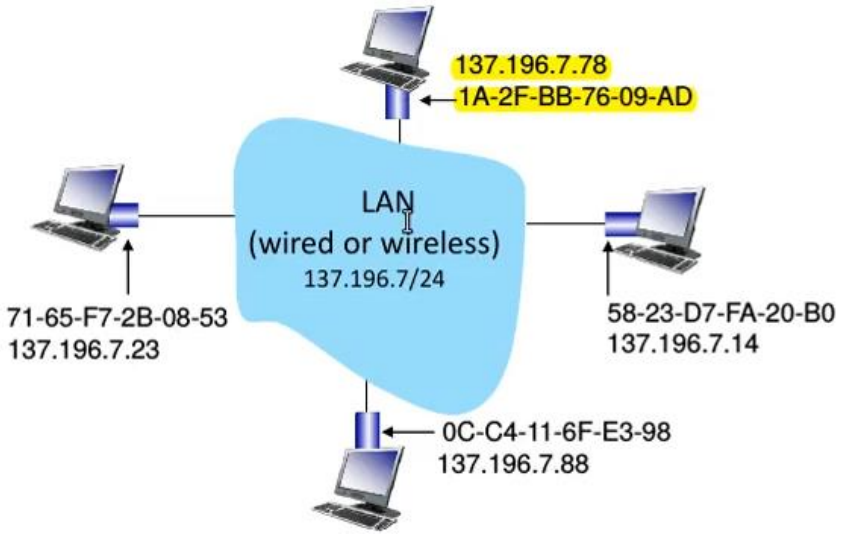
حالا به دو شکل می تونیم این frame دریافتی رو پردازش کنیم:

1 - اگه از MAC address استفاده نکنیم مجبوریم هر frame ای که دریافت میشه یه interrupt به CPU بدیم ، دیتاگرامی که داخل frame هست رو ببریم به لایه ی IP ، و در لایه ی IP ، اون host بررسی کنه که آیا IP address توی هدر IP دیتاگرام هست، مال اون نود هست یا نه ، اگه مال اون نود باشه ، میره پردازشش می کنه و در غیر این صورت دیتاگرام رو دور می ریزه.

2 - اگه از MAC address استفاده کنیم، پردازش اولیه می تونه در سخت افزار انجام بشه. چون قسمتی از کارهای لایه ی لینک در سخت افزار انجام میشه، می تونیم بررسی آدرس MAC رو داخل

سخت افزار انجام بدیم. ببینیم آیا این frame ای که دریافت کردیم، آدرس MAC اش، آدرس MAC ای هست که متناظر با این interface عه، اگه این طور باشه پس frame برای همین interface ارسال شده و به CPU ، interrupt می دیم و frame رو در اختیار لایه های بالاتر protocol stack قرار می دیم و سایر کار ها به صورت نرم افزاری روی اون بسته انجام میشه؛ در غیر این صورت اگه آدرس MAC ، متناظر با این interface نباشه، frame رو دور می ریزیم. این کاربرد MAC address باعث افزایش کارایی دستگاه ها در شبکه های محلی میشه.

- شکل زیر، نشون میده که هر interface ای که داخل شبکه داریم ، (اینجا یه subnet با subnet mask = 24 نشون داده شده) علاوه بر آدرس های IP ۳۲ بیتی ، هر کدوم شامل یه MAC address هستن که این MAC address ها کاملا از هم متفاوت هستن.



آدرس های IP ، قسمت subnet شون مشترکه و قسمت host number شون متفاوته.

- MAC addresses

- برای این که یکتا بودن MAC address ها در کل دنیا تضمین بشه، احتیاج به یه نهادی داره که متولی این یکتا بودن آدرس های MAC interface های مختلف باشه.

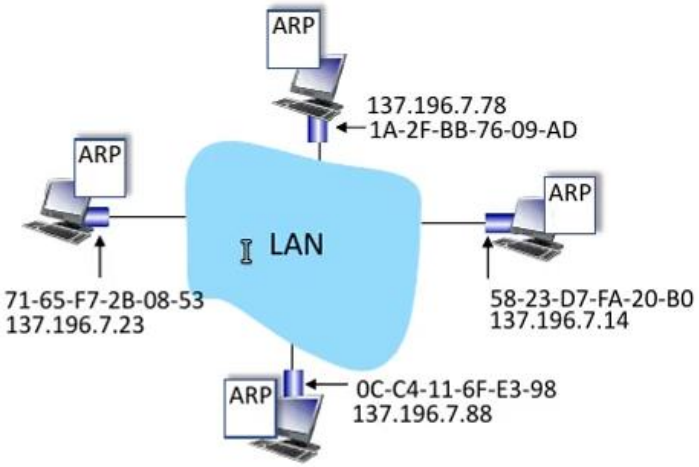
مثلا WiFi chipset ای که روی یه لپ تاپ هست با WiFi chipset ای که روی یه cellphone هست متفاوته . همینطور راجع به کارت شبکه و Ethernet که باید MAC address های متفاوت و یکتایی داشته باشن.

این کار توسط موسسه ی IEEE انجام میشه . هر کمپانی ای که chipset های مربوط به کارت شبکه درست می کنه توسط IEEE یه قسمت از فضای ۴۸ بیتی بهش تخصیص داده میشه و یه سری از بیت های پر ارزش مخصوص اون کمپانی fix میشن و اون کمپانی می تونه بیت های کم ارزش رو تغییر بده و به هر دستگاهی، MAC address مخصوص به خودش رو بده.

- MAC address خاصیت portability داره ؛ یعنی وقتی شبکه ای که بهش متصل هستیم رو تغییر میدیم MAC address مون تغییر نمی کنه و همیشه یه آدرس یکسانه.

- ARP : Address Resolution Protocol

- ما نیازمند روشی هستیم که بتونیم IP address رو به MAC address تبدیل کنیم و برعکس. این کار توسط پروتکل ARP انجام میشه. برای این که لزوم استفاده از چنین پروتکلی رو متوجه بشیم ، شکل روبرو رو در نظر می گیریم:



Interface مربوط به host هایی که به این LAN متصل شدن، دوتا آدرس دارن؛ یه MAC address و یه IP address .

فرض می کنیم نود سمت چپ می خواد برای نود سمت راست frame ارسال کنه. بنابراین نیازه که MAC address گیرنده رو بلد باشه تا فیلد MAC address گیرنده در frame هایی که میخواد ارسال کنه رو درست پر کنه.

فرض می کنیم که IP address گیرنده رو بلده ولی MAC address گیرنده رو نمیدونه. این جاست که پروتکل ARP به کمک ما میاد.
توی این پروتکل ، داخل هر نودی، یه جدولی به اسم ARP table وجود داره و رکورد هایی که داخل این جدولن به فرمت زیر هستن :

<IP address; MAC address ; TTL>

این IP address و MAC address مربوط به یه نود دیگه داخل همین subnet هست. TTL هم زمان انقضای مربوط به هر رکورد رو مشخص می کنه.
جدول ARP داخل هر کدوم از این نود ها، نیازی نیست که IP address و MAC address مربوط به همه ی نود های subnet داخلش موجود باشه ، بلکه فقط اطلاعات نود هایی رو نگه می داریم که قصد ارتباط با اون ها رو داشتیم. مثلا اگه با یه نودی داخل subnet تا حالا هیچ ارتباطی برقرار نکردیم طبیعتا به ازای اون نود هیچ رکوردی هم داخل جدول نداریم.

- عملکرد پروتکل : وقتی نود A می خواد برای نود B ، frame ارسال کنه، اما رکوردی از نود B داخل جدول خودش نداره و به عبارتی MAC address مربوط به نود B رو نداره، یه ARP query ،

broadcast می کنه که این پیام کوئری شامل IP address نود B هست و این کوئری خودش به عنوان payload داخل یک frame قرار می گیره که آدرس MAC گیرنده ی اون frame ، آدرس broadcastِ MAC هست که ۴۸ تا یک در فرمت هگزادسیمالِه :

FF-FF-FF-FF-FF-FF

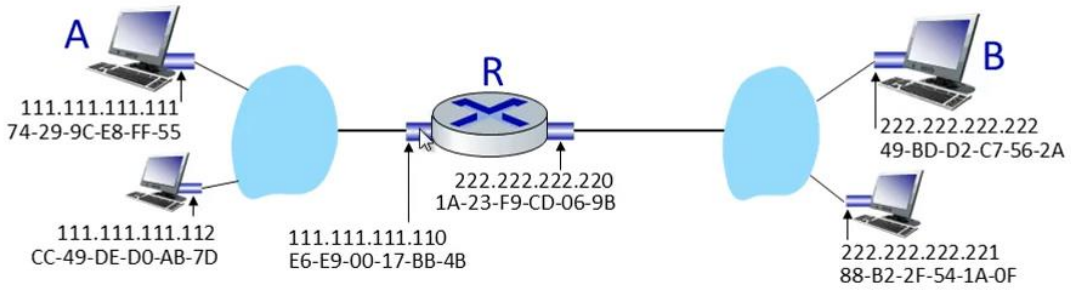
چون MAC address گیرنده این آدرس به خصوص رو داخل frame داره، این frame توسط تمام نود های subnet دریافت میشه و این نود ها این frame رو به ماژول ARP خودشون تحویل میدن .

مثلا این سوئیچ هایی که داخل LAN ما می تونه وجود داشته باشه، وقتی که می بینه آدرس گیرنده، broadcast هست اون frame رو روی سایر پورت ها به جز پورتی که از طریقش frame رو دریافت کرده، ارسال می کنه.

درسته که این frame توسط سایر نود ها هم دریافت میشه اما فقط نود B هست که بعد از بررسی اون متوجه میشه که مخاطبِ این کوئری هست و پاسخی برای نود A ارسال می کنه و ضمن این جواب، MAC address خودش رو به نود A اطلاع میده. این پیام دوم برخلاف پیام اول دیگه broadcast نیست چون نود B بعد از دریافت پیام کوئری از نود A ، MAC address نود A رو می دونه و توی frame ای که براش ارسال می کنه آدرس MAC گیرنده رو برابر با MAC addressِ A قرار میده.

نهایتا نود A بعد از دریافت پاسخ از نود B، می تونه رکورد مربوط به نود B رو در جدول ARP خودش ایجاد کنه.

- مثال : اگه یه نود مثل A برای نود مثل B در یه subnet دیگه بخواد دیتاگرام ارسال کنه گام به گام چه اتفاقاتی میفته؟ (تمرکز روی لایه ی دو و frame های لایه ی لینک هست)



همه ی interface هایی که داخل شبکه وجود دارن یه آدرس IP و یه آدرس MAC دارن و فرض می کنیم نود A ، IP address نود B رو بلده و می خواد براش دیتاگرام ارسال کنه.همچنین IP address مربوط به gateway خودش رو هم بلده ، یعنی نود A ، IP address مربوط به interface سمت چپ نود R رو می دونه که می تونه از طریق پروتکل DHCP به دست آورده باشه.

A ، subnet mask خودش رو هم بلده و این رو هم می تونه از طریق پروتکل DHCP به دست آورده باشه.

A می تونه از طریق پروتکل ARP ، MAC address مربوط به interface سمت چپ نود R رو هم به دست بیاره.

وقتی یه دیتاگرامی می خواد برای B ارسال بشه ، لایه ی IP میاد دیتاگرام خودش که به عنوان source IP address ، IP address نود A داخلش قرار داره و به عنوان destination IP address ، IP address نود B داخلش قرار داره رو به لایه ی لینک تحویل میده ؛ لایه ی لینک میاد می بینه که این دیتاگرام مربوط به نودی هست که داخل subnet خودش نیست، چطوری اینو می فهمه؟ از طریق اون subnet mask ای که در اختیار داره ، destination IP address رو با subnet mask ، mask می کنه و می بینه قسمت subnet part این آدرس IP مقصد ، با قسمت subnet part آدرسی که خودش داره یکسان نیست، بنابراین این نود، نودی نیست که در subnet خودش باشه و بتونه مستقیما بسته رو براش ارسال کنه ؛ بلکه باید باید بسته رو برای gateway router ارسال کنه تا این روتر مسیریابی کنه و بسته رو به مقصد برسونه.

پس لایه لینک میاد داخل frame بندی ای که انجام میده، داخل هدر frame ، MAC address گیرنده رو MAC address مربوط به interface سمت چپ نود R قرار میده تا frame توسط این interface در داخل subnet پردازش بشه و سایر نود های این subnet دیگه این frame رو پردازش نمی کنن. MAC address فرستنده هم که همون MAC address نود A هست.

در گام بعد، روتر بسته رو دریافت می کنه و متوجه میشه که بسته رو باید به interface سمت راست خودش ارسال کنه. مجددا می تونیم فرض کنیم IP address نود B رو که روتر R می دونه ، می تونه از طریق پروتکل ARP ، MAC address نود B رو هم بفهمه. به خاطر همین وقتی مجددا می خواد دیتاگرام رو frame بندی کنه و روی subnet سمت راست بفرسته تا به نود B برسه، این دفعه میاد MAC address گیرنده رو برابر با MAC address نود B قرار میده و MAC address فرستنده رو هم برابر با MAC address مربوط به interface سمت راست روتر R قرار میده و بعد frame رو ارسال می کنه و frame توسط نود B پردازش میشه و سایر نود های این subnet هم frame رو دریافت می کنن اما چون MAC address گیرنده با MAC address خودشون تطابق نداره اونو دور می ریزن.

در نهایت نود B دیتاگرام رو از بسته ای که دریافت کرده استخراج می کنه و این بسته رو تحویل لایه ی IP خودش میده.

- پروتکل ARP شباهتی به پروتکل DNS داره؛ در DNS نام دامنه به IP address ترجمه میشه و در پروتکل ARP ، IP address به MAC address ترجمه میشه .

اما با این حال scope پروتکل DNS با پروتکل ARP تفاوت اساسی داره؛ در DNS ، host ای که نام دامنه اش رو می دونیم و دنبال آدرس

IP اش هستیم ، می تونه در هرجای اینترنت قرار گرفته باشه ؛ اما در ARP ، host ای که آدرس IP اش رو می دونیم و دنبال آدرس MAC اش هستیم ، حتما باید در subnet خودمون قرار گرفته باشه و اگه دوتا نود در دوتا subnet متفاوت باشن ، توسط ARP نمی تونن از MAC address هم مطلع بشن.

- Ethernet

- پروتکل غالب در لایه ی لینک ولایه ی فیزیکی در شبکه های محلی سیمی ، پروتکل IEEE 802.3 یا Ethernet هست.
- دلایل عمده ی موفقیت پروتکل Ethernet نسبت به پروتکل های رقیب:

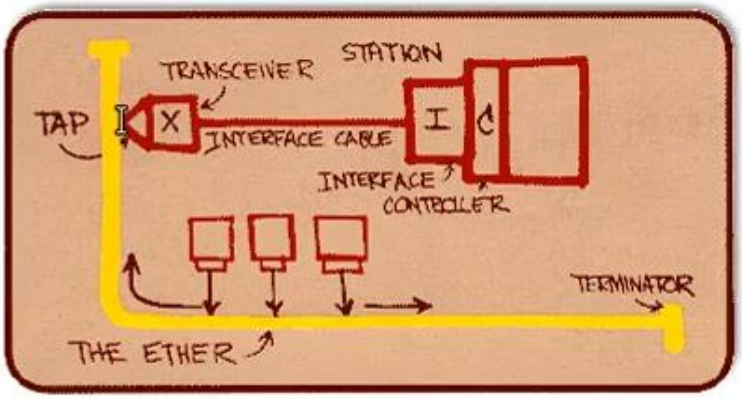
1 - Ethernet زودتر از همه به بازار عرضه شد ، در نتیجه ادمین های شبکه ها با این تکنولوژی آشنا شدن و نسبت به جایگزینی اون با یه پروتکل دیگه اینرسی داشتن .

2 - نسبت به رقبا ساده تر و ارزون تر بود.

3 - Ethernet یه پروتکل پویا و زنده هست و همیشه نسخه های جدید نسبت به نسخه های قبلی سرعت بالاتری دارن. (از 10 Mbps تا 400 Gbps سرعت خودش رو افزایش داده)

4 - به دلیل استفاده ی عمده از Ethernet ، chip های Ethernet به وفور و به صورت اقتصادی در بازار عرضه می شن.

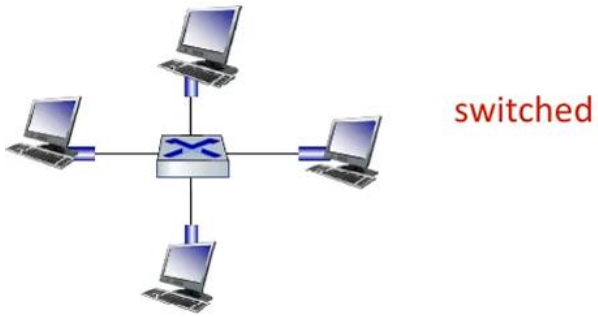
- شکل زیر ، بازسازی شده ی طرحیه که آقای Metcalf مبدع Ethernet در اواسط سال ۱۹۷۵ کشیده و نشون دهنده ی ساختار نسل اول Ethernet هست که به Cable Ethernet هم مشهوره.



توی این ساختار یه کابل کواکس وجود داره که شبیه یه باس عمل می کنه و تمام host ها از طریق این باس به هم متصل میشن و با هم ارتباط دارن. بنابراین Cable Ethernet از یه کانال مشترک استفاده می کنه. پروتکل CSMA/CD به همراه binary exponential backoff به عنوان روش multiple access در این نوع Ethernet استفاده میشه.

- از لحاظ توپولوژی و ساختار ، پروتکل Ethernet در طول زمان دچار تغییرات و تحولات زیادی شده . در ابتدا Ethernet ساختار باسی داشت و به همین دلیل مشکل collision به عنوان یه مشکل مهم در این نوع Ethernet مطرح بود.

ساختار های بعدی ای که ارائه شد، ساختار های ستاره ای بودن که همه ی host ها توسط لینک های point-to-point به یک سوئیچ یا hub متصل هستن و ارتباطشون از این طریق برقرار میشه.



بین ساختار باسی و ساختار ستاره ایِ سوئیچی ، ساختار ستاره ای هابی هم عرضه شد.

- Hub یه دستگاه لایه ی فیزیکی هست و به این ترتیب عمل می کنه که هر بیتی رو از هر پورتش دریافت کنه ، روی سایر پورت ها ارسال می

کنه. به همین دلیل توی ساختار های ستاره ای که مبتنی بر hub هستن همچنان احتمال collision وجود داره و مثلا اگه دوتا frame همزمان از دوتا پورت مختلف به hub برسن نهایتا مخروط شده ی اون دوتا frame توسط نود های اون LAN دریافت میشه.

اما در ساختار ستاره ای مبتنی بر سوئیچ ، collision دیگه وجود نداره.

- سوئیچ یه دستگاه لایه ی دو هست و مبتنی بر store and forward کار می کنه و عملکردش مشابه روتره. با این تفاوت که روتر ها دستگاه های لایه ی 3 هستن و مبتنی بر IP address کار می کنن ولی سوئیچ ها دستگاه های لایه 2 هستن و مبتنی بر MAC address ان اما عمل forwarding در اون ها مشابه هست.
- ساختار Ethernet frame به این نحوه که در شکل نشون داده شده:

ابتدا ۸ بایت preamble داریم ، از این ۸ بایت، ۷ بایتش 10101010 هست، و به منظور ایده آل کردن نودی که قراره این frame رو دریافت کنه ارسال میشن. بایت هشتم ، 10101011 هست یعنی بیت آخر به جای 0 ، 1 عه و به این ترتیب می تونیم مرز بین preamble و سایر قسمت های frame رو متوجه بشیم.

فیلد بعدی ، dest address هست که همون MAC address گیرنده ست و source address هم MAC address فرستنده ست.
یه فیلدی به اسم type داریم که برای عمل های mux و demux استفاده میشه و توسط مقداری که این جا نوشته میشه متوجه میشیم data(payload) رو تحویل چه ماژولی باید بدیم. مثلا اگه data شامل یه دیتاگرام باشه باید اونو تحویل لایه ی شبکه بدیم یا اگه برای پروتکل ARP هست باید تحویل ماژول ARP داده بشه. نقش این فیلد مشابه نقش فیلد protocol number توی لایه ی شبکه و port number توی لایه ی حمل و نقل هست.
فیلد آخر هم CRC هست که قبلا راجع بهش صحبت کردیم و برای آشکار سازی خطا در frame های Ethernet استفاده میشه.

- Ethernet : unreliable , connectionless

1 - پروتکل Ethernet یه پروتکل connectionless هست یعنی سمت فرستنده قبل از اینکه مبادرت به ارسال frame بکنه ، با سمت گیرنده هماهنگی نمی کنه .
2 - Unreliable هست یعنی مثلا اگه مبتنی بر CRC گیرنده متوجه بشه که frame درست هست یه ACK نمی فرسته یا اگه

frame درست نباشه NAK نمی فرسته و عمل retransmission توسط لایه ی لینک انجام نمیشه .

اگه اتفاقی سمت گیرنده افتاد و مثلا مبتنی بر CRC متوجه بشیم خطایی در frame رخ داده اون رو دور می ریزیم و این مسئولیت لایه های بالاتر (مثل TCP) هست که اگه احتیاج دارن که داده ها بدون خطا به صورت end to end برسن، reliable channel رو در لایه های بالاتر ایجاد کنن.

3 - Ethernet همون طور که گفتیم از پروتکل CSMA/CD به همراه binary exponential backoff استفاده می کنه البته استفاده از چنین پروتکلی در برخی از توپولوژی های Ethernet که به صورت باس یا مبتنی بر هاب هستن ، ضرورت داره و توی اون نسخه هایی از Ethernet که مبتنی بر سوئیچ هستن دیگه این پروتکلِ MAC استفاده نمیشه و وجودش ضروری نیست؛ به خاطر این که collision ای رخ نمیده.

- Ethernet standards : link & physical layers

- تا الان استاندارد های خیلی زیادی برای این تکنولوژی عرضه شده که در سرعت، نوع مدیا و ساختار با هم متفاوت هستن. هر استاندارد با یه نام به خصوص شناسایی میشه.
- قراردادی که توی نام گذاری این استاندارد ها رعایت میشه به این شکله که عددی که در ابتدای نام میاد، نشون دهنده ی سرعته ؛ بعد از اون عدد، تقریبا همه ی اسم ها شامل کلمه ی BASE هستن که کابل ها فقط ترافیک Ethernet رو حمل می کنن. چون در تئوری این احتمال وجود داره که ترافیک مربوط به یه تکنولوژی دیگه هم از طریق همون مدیایی که داریم Ethernet رو منتقل می کنیم منتقل بشن. اما تا حالا استاندارد هایی که این اتفاق دراون ها افتاده باشه عرضه نشده.

  نهایتا دوتا کاراکتر انتهایی در نام استاندارد ها وجود داره که نشون دهنده ی نوع مدیا و مشخصه ی مدیا هستن. مثلا کاراکتر T یعنی twisted pair یا زوج سیم و کاراکتر هایی مثل S و F و B یعنی fiber .

این نام ها بیشتر نوع تکنولوژی در لایه ی فیزیکی رو نشون میدن . توی لایه ی لینک استاندارد های مختلف از پروتکل MAC یکسان و ساختار frame یکسان استفاده می کنن.

بنابراین چیزی که توی استاندارد های مختلف Ethernet یکسان و مشابه باقی مونده ، همین ساختار frame یکسان هست.

- Ethernet switch

- سوئیچ ها وسایل لایه ی لینک هستن و عمل store and forward رو روی frame های Ethernet انجام میدن.
- مبتنی بر MAC address گیرنده، میان اون frame ای که دریافت می کنن روی یکی یا بیشتر از یکی از لینک های خروجی ارسال می کنن و روی هر لینک هم پروتکل CSMA/CD رو اجرا می کنن.
- سوئیچ ها وسایل transparent هستن یعنی host ها از وجودشون با خبر نیستن و فرقی نداره که دوتا host از طریق یه لینک مستقیم به هم متصل باشن یا توسط یه سوئیچ میانی این اتصال بین اونا شکل گرفته باشه.
- سوئیچ ها plug-and-play هستن و احتیاج نیست configuration خاصی قبل از استفاده از سوئیچ ها صورت بگیره. مثلا اگه بخوایم چندتا کامپیوتر رو از طریق یه سوئیچ مرتبط کنیم، کافیه که از طریق کابل های شبکه، هر کدوم از اون کامپیوتر ها رو به پورت های اون سوئیچ متصل کنیم.

  از این مکانیزم سوئیچ ها تحت عنوان self-learning یاد میشه که عملیات forwarding frame ها رو به نحو مؤثری انجام میده.

- Switch : multiple simultaneous transmissions

- در switch LAN ها بین هر host و هر پورت سوئیچ یک ارتباط اختصاصی و مستقیم وجود داره و این ارتباط دو سویه هست و به صورت full-duplex صورت می گیره.

مثلا در تکنولوژی ای که از زوج سیم استفاده می کنه، یک زوج سیم برای ترافیک ها از host به سوئیچ وجود داره و یک زوج سیم برای ترافیک در مسیر معکوس. به این ترتیب collision رخ نمیده .

- سوئیچ چطور متوجه میشه که کدوم نود به کدوم پورتش متصله تا بتونه برای ارسال بسته به یک نود تنها بسته رو برای لینک متصل به اون نود ارسال کنه؟

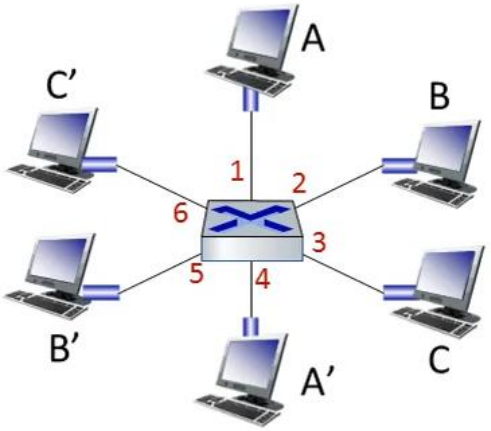
داخل سوئیچ ها جداولی داریم که توی این جداول رکورد هایی به فرم :

(MAC address of host, interface to reach host, time stamp)

هست. فیلد اول همون MAC address گیرنده هست و فیلد دوم مشخص می کنه که گیرنده با اون MAC address به این interface متصله و time stamp هم مشخص می کنه این رکورد ها تا چه مدت اعتبار دارن. این جداول سوئیچینگ، مشابه جداول روتینگ توی روتر ها هستن.

حالا این جداول سوئیچینگ چطور مقدار دهی میشن؟

سوئیچ ها بر خلاف روتر ها دیگه پروتکل های شبیه پروتکل های روتینگ برای پر کردن جداولشون ندارن و به جاش از مکانیزمی به اسم self-learning استفاده می کنن که یه مکانیزم یادگیریه؛



اگه نود A برای نود A' بخواد یه frame ارسال کنه در داخل هدرش به جای source MAC address ،آدرس MAC خودش رو قرار میده. به جای destination MAC address هم آدرس MAC نود A' رو قرار میده. وقتی این بسته توسط سوئیچ دریافت بشه (و اگه جدول سوئیچینگش در ابتدا خالی باشه) ، سوئیچ یاد می گیره که بعد از دریافت این frame که نود A (با توجه به MAC آدرس مبدأ که توی frame هست) به interface شماره ی یک وصله و میاد یه رکورد متناظر با MAC address نود A در جدول ایجاد می کنه :

MAC addr	interface	TTL
A	*1*	*60*

Switch table (initially empty)

برای فوروارد کردن این frame ، رکوردی که به جدول اضافه شده کمکی نمی کنه و ما باید بدونیم نود A’ به چه interface ای متصله.
پس سوئیچ میاد و این frame رو برای روی همه ی لینک ها به غیر از لینکی که frame رو ازش دریافت کرده، ارسال می کنه.
متعاقبا وقتی A’ می خواد به A پاسخ بده، سوئیچ می تونه یه رکورد دیگه بعد از دریافت frame مربوط به A’ (متناظر با نود A’)در جدول خودش ایجاد کنه . در ضمن frame نود A’ رو که می خواد برای A ارسال کنه ، چون رکورد متناظر با A رو در جدول خودش داره ، دیگه نیازی نیست flood کنه (برای همه بفرسته) و فقط از طریق interface شماره ی یک بسته رو می فرسته تا به نود A برسه.

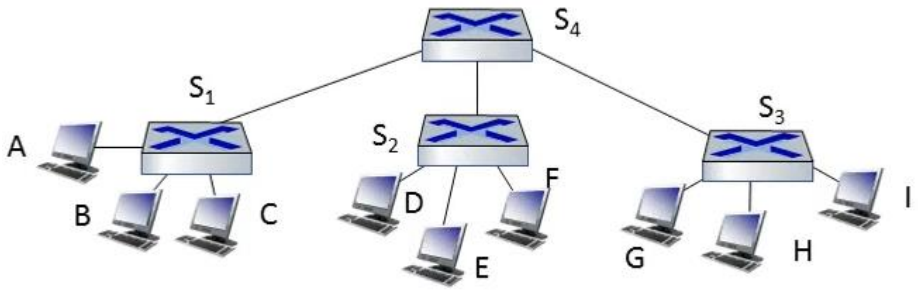
MAC addr	interface	TTL
A	1	60
A’	4	60

الگوریتم frame forwarding/filtering به صورت مرحله به مرحله:

when frame received at switch:

1. record incoming link, MAC address of sending host
2. index switch table using MAC destination address
3. if entry found for destination
   then {
   if destination on segment from which frame arrived
      then drop frame
       else forward frame on interface indicated by entry
   }
   else flood /* forward on all interfaces except arriving interface */

- نکته ای که در مورد سوئیچ ها وجود داره اینه که توی خیلی از شبکه ها این سوئیچ ها تنها یک عدد نیستن بلکه ما مجموعه ای از سوئیچ ها داریم که معمولا به صورت سلسله مراتبی به هم متصل میشن:



توی همچین شبکه هایی که سوئیچ ها به صورت سلسله مراتبی به هم متصلن ، مثلا اگه نود A که به S_1 وصله بخواد برای G که به S_3 وصله ، بسته ارسال کنه، این فورووارد کردن چطور صورت می گیره؟
از طریق همون مکانیزم self-learning ، در مورد این شبکه ها هم در هر کدوم از سوئیچ ها جداول forwarding مناسب شکل می گیره.

- Small institutional network

- خیلی از شبکه های مربوط به موسسات و دانشگاه ها از این ساختار سلسله مراتبی سوئیچ ها استفاده می کنن. در این ساختار ها ، معمولا ترافیک های مربوط به department های مختلف توسط سوئیچ های سطح پایین تجمیع میشن و ارتباط بین خود department های مختلف از طریق یه سری سوئیچ سطح بالاتر تامین میشه و نهایتا هم

لینکی از این سوئیچ های سطح بالا به gateway و اینترنت وجود داره. بعضی از سرور های مهم در داخل اون موسسه هم می تونن به صورت مستقیم و با لینک های با پهنای باند بالا به این سوئیچ های سطح بالا متصل بشن.

- Switches vs routers

- هم سوئیچ ها و هم روتر ها store-and-forward انجام میدن. اما یکی در سطح شبکه هست و مبتنی بر بررسی هدر های IP این کار رو انجام میده و IP address ها مشخص می کنن که بسته باید روی چه پورتی فوروارد بشه، و یکی بر اساس اطلاعات لایه ی لینک هست و مبتنی بر بررسی هدر های لایه ی لینک و آدرس های MAC عمل فورواردینگ رو انجام میدن.
- در هردوی این وسایل ما forwarding table داریم. روتر ها مقادیر این جداول رو توسط routing algorithm ها پیدا می کنن و سوئیچ ها از طریق یادگیری مقادیر این جدول رو پیدا می کنن.
- سوئیچ ها المان های transparent هستن و روتر ها و host ها از وجودشون با خبر نیستن. مثلا توی مثالی که برای توضیح self learning زدیم ، وقتی A می خواست بسته رو ارسال کنه ، MAC address گیرنده رو آدرس MAC نود A’ میذاشت نه آدرس مربوط به

interface شماره ی یک. و اصلا interface سوئیچ ها ، MAC address و IP address نمی گیرن.

اما اگه به جای این سوئیچ ، یه روتر داشتیم ، A برای این که برای A’ یه بسته ارسال کنه، آدرس MAC گیرنده توی هدر بسته ، باید آدرس MAC مربوط به interface روتری باشه که A بهش متصله. بعد هم که روتر می خواد بسته رو برای A’ بفرسته، به عنوان source MAC address ، MAC address مربوط به interface شماره ی ۴ رو قرار میده و برای destination MAC address ، میاد MAC address نود A’ رو قرار میده. بنابراین وجود روتر transparent نیست.

- مثال آخر :)

A Day in the Life of a Web Page Request

این مثال هدفش آشناییِ گام به گام با این رونده که هنگام در خواست یه web page ، از ابتدا تا انتها چه پروتکل هایی و به چه دلیلی اجرا میشن.

این مثال مفصله و ۲۴ تا گام داره. ولی ۴ مرحله ی اصلی اینا هستن:

1 - با پروتکل های DHCP و UDP و IP و Ethernet کار می کنیم.

2 - DNS و ARP

3 - پروتکل های Intra-Domain Routing تا DNS server ها

4 - Web client-server interaction : TCP and HTTP