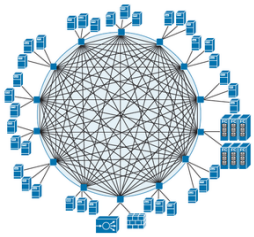
به نام خدا

علی فتحی ۹۴۱۰۹۲۰۵

تمرین سری ۲ شبکه مخابرات داده‌ها – دکتر پاکروان



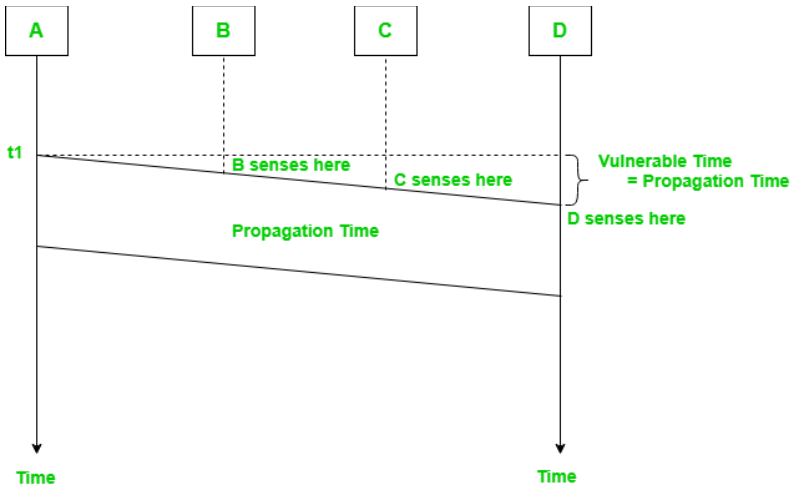
” Data Link Layer ”

Ethernet Network

1. Explain how a minimum packet size can help to detect collisions in Ethernet.

این کار، برای جلوگیری از برخورد (collision) انجام می‌شود، و مشابه دلیل استفاده از PAD در سیستم‌های CSMA دار است.
برای آن که برخورد داده‌ها تشخیص داده شود باید داشته باشیم:

$$T_{frame} > 2t_{propagation} = 2\frac{d}{v}$$



که هرچه سرعت انتقال داده‌ها بیشتر شود (مثلا تبدیل 10Mbps به 100Mbps)، T_{frame} به ازای تعداد bit معین کوتاه‌تر شده پس لازم است packet size بزرگتر شود. اگر این رابطه رعایت نشود، ممکن است برخوردی داشته باشیم و فرستنده آن را متوجه نشود (زیرا داده‌ی مداخله‌گر بعد از اتمام داده‌ی خود فرستنده به دستش می‌رسد).

****

2. Help him out. Compute the threshold of how big packets must be in order for things to work.

با سرعت ،10Mbps اترنت با استاندارد فاصله ۲۵۰۰ متر، طول داده‌ی حداقل ۵۱۲ بیت در هر فریم لازم دارد. با ۱۰ برابر شدن شدن و تبدیل آن به 100Mbps ، زمان T_{frame} یک دهم (۰.۱) شده پس برای برقرار ماندن رابطه ذکر شده، طول داده نیز باید ۱۰ برابر شود، یعنی برابر ۵۱۲۰ بیت در هر فریم یا 5Kb.

****

3. One solution that he came up with is to raise the minimum packet size to the answer from part (2). Suppose that Frank cannot modify the minimum packet size, move the endpoints, lay new cable, or change the software or configuration on the endpoints. You may add new devices to the network. How could Frank change the topology to fix his problems anyway?

اگر نخواهد از PAD کردن (اضافه کردن داده بیخود) استفاده کند، می‌تواند از روشی به نام frame bursting استفاده نماید؛ یعنی وقتی یک فرستنده کانال را در دست گرفت، تا زمان مجاز کانال را برای خود نگه‌دارد و به جای یک فریم، چند فریم بفرستد (اما این کار نیازمند تعریف کردن این قرارداد با همه نقاط شبکه است). با توجه به آن که نقطه مرکزی در دست فرانک است، می‌تواند یک Switched Ethernet به جای سیستم فعلی قراردهد تا کلا از عمل برخورد جلوگیری شود (یا از نظر مفهومی با Buffer ها کار کند).

****

Bit/Byte Stuffing

1. The physical layer transmits a continuous bit stream, while the data link layer segments streams into frames. Why are frames necessary?

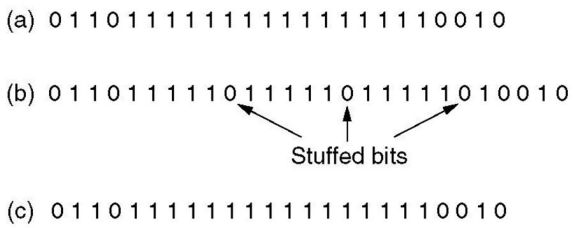
1. فریم بندی باعث می‌شود بتوان عملیات تشخیص خطا (Detection or Correction) را پیاده کرد. زیرا اضافه کردن بیت‌هایی برای انجام این کار بدون مشخص بودن حد و مرز داده‌ها ممکن نیست.
2. در محیط‌های چند دسترسی، MAC ، خطا روی فریم‌ها تعریف می‌شود و مفهوم تداخل و دوباره فرستادن را برای تک بیت مطرح کردن پیچیدگی زمانی را در پی دارد.
3. امکان آدرس دهی مقصد در فریم‌ها وجود دارد (در روش مخابرات packet) اما این کار برای ارسال تک بیتی ممکن نیست.

****

2. Explain how bit stuffing and byte stuffing works, and name a protocol which uses each of these methods.

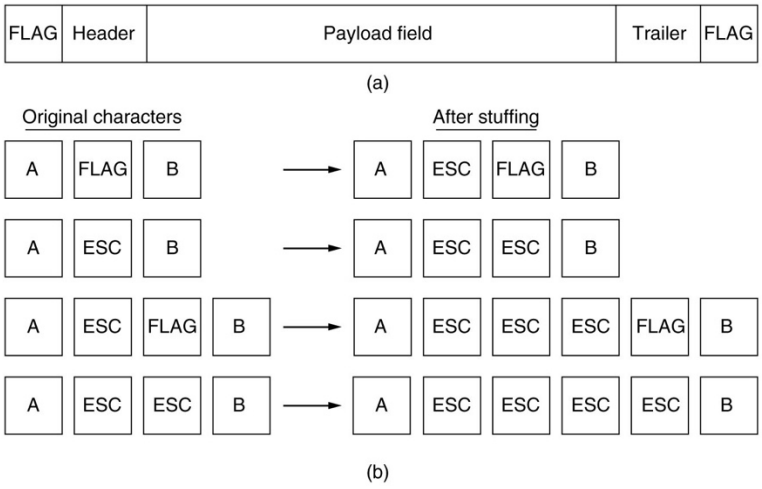
1. **Bit Stuffing**: این روش، در واقع دستوری از سمت Data Link Layer به Physical Layer است، که طی آن یک رشته مشخصی مانند ۰۱۱۱۱۱۱۰ برای مشخص کردن ابتدا و انتهای فریم قرارداد می‌شود؛ سپس برای آن که این رشته در میان داده‌ها در صورت ظهور اشتباه گرفته نشود، بیت‌هایی با قرارداد مشخص در رشته‌های داده جا داده می‌شود (در این مثال، که رشته ۶ عدد ۱ دارد، قرارداد Bit Stuffing به صورت قرار دادن ۰ در میان هر ۵ بیت متوالی ۱ است و سپس حذف آن در هنگام دریافت).

کاربرد: High-Level Data Link Control(HDLC) ، USB، Plesiochronous Digital Hierarch، Synchronous Digital Hierarchy



2. **Byte Stuffing**: در این روش، که در خود Data Link Layer انجام می‌شود، داده‌ها به صورت کاراکتری دیده می‌شوند و ابتدا و انتهای بازه با دو کاراکتر معین به اسم FLAG مشخص می‌گردد. سپس کاراکتر دیگری به نام ESC در صورت تکرار FLAG در میان داده‌های اصلی، به قبل آن Stuff می‌شود و در هنگام دریافت نیز حذف می‌گردد. در صورت وجود ESC نیز دوباره خود ESC اضافه می‌شود و در دریافت نیز به تعداد اضافه شده حذف می‌گردد.

کاربرد: Point-to-Point Protocol (PPP)



****

3. A bit string, 01111011111011111101111110, needs to be transmitted at the data link layer. What is the string actually transmitted after bit stuffing?

با فرض آن که بعد از هر ۵ عدد ۱ متوالی ۱ عدد ۰ اضافه کند:

Data: 01111011111011111101111110
Send: [01111110] 01111011111001111101011111010 [01111110]

****

4. What is the maximum overhead in byte-stuffing algorithm?

این اتفاق در حالتی می‌افتد که تمام Byte ها، FLAG یا ESC باشند. در این وضعیت تعداد Byte ها دو برابر می‌شود و overhead برابر ۱۰۰ درصد داریم.

****

5. The following data fragment occurs in the middle of a data stream for which the byte stuffing algorithm is used: A B ESC C ESC FLAG FLAG D. What is the output after stuffing?

قبل از هر FLAG یا ESC ، یک ESC اضافه می‌کنیم:

Data: A B ESC C ESC FLAG FLAG D
Send: A B ESC ESC C ESC ESC ESC FLAG ESC FLAG D

****

Ethernet Protocol

1. Why is the minimum size of an Ethernet frame 64 bytes?

مشابه آنچه در بخش اول توضیح داده شد، یک رابطه برای رخ ندادن collision میان سرعت انتقال داده و $T_{propagation}$ وجود دارد:

$$T_{frame} > 2t_{propagation} = 2\frac{d}{v_{wave\ in\ copper}}$$

و با توجه به این که Ethernet معمول، با سرعت 10Mbps ، برای جواب دهی تا فاصله ۲۵۰۰ متر طراحی شده خواهیم داشت:

$$T_{frame} = \frac{frame\ size}{bit\ rate}$$

$$\Rightarrow \min(frame\ size) = bit\ rate \times 2\frac{d}{v} = 10Mbps \times 2 \times \frac{2500m}{2\times10^8\ m/s} \approx 250\ bits \approx 32\ Byte$$

که برای آنکه تا حدی قابلیت استفاده در نسل 100Mbps هم داشته باشد، این استاندارد را برای این سرعت هم تا محدوده ۵۰۰ متر Back Compatible کرده‌اند و طول 64 Byte را مشخص نموده‌اند.

****

2. What is the maximum overhead in byte-stuffing algorithm?

همان طور که در بخش قبل توضیح داده شد بیشینه overhead برابر ۱۰۰ درصد است.

****

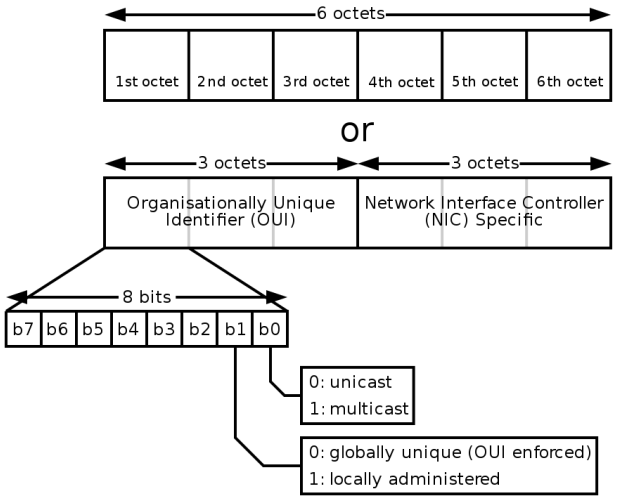
3. What does your Ethernet card do to a frame if it calculates an invalid CRC?

این frame دور انداخته می‌شود. Ethernet Card ، خرابی این frame را اعلام کرده و مجددا ارسال آن را از فرستنده درخواست می‌کند.

****

4. Explain the purpose of MAC address and OUI.

در واقع MAC Address ، مخفف media access control address ، یک شماره مشخص کننده (Identifier) و یکتا در قسمت Data Link Layer برای هر دستگاهی است که از پروتکل Ethernet استفاده می‌کند (البته کاربرد آن به سایر استاندارد های IEEE 802 مانند Wi-Fi نیز گسترده شده است). در پروتکل Ethernet (یا کلی تر در استاندارد IEEE 802)، قسمت آدرس (فرستنده یا گیرنده) با ۶ بایت مشخص می‌شود و می‌توان با این ها، 2^{48} دستگاه متفاوت را آدرس‌دهی کرد. مدل بایت‌ها در شکل زیر آمده است (این مدل به اصطلاح MAC-48 نام دارد):

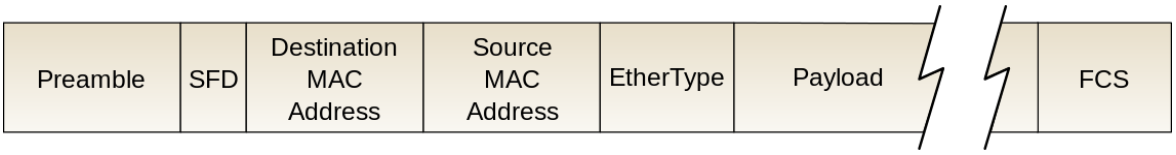


همچنین OUI ، مخفف organizationally unique identifier ، یک شماره ۲۴ بیتی است که به صورت یکتا به یک فروشنده، اداره یا یک شرکت اختصاص داده شده است. OUI ها از IEEE خریده می‌شوند.

****

5. What is start frame delimiter (SFD) in Ethernet frame?

در مخابرات اترنت، ابتدای کار، ۷ بایت (۵۶ بیت ۰ و ۱) تحت عنوان Preamble فرستاده می‌شود تا گیرنده خود را Synchronized کند. سپس زنجیره متوالی ۰ و ۱ ها با ارسال دو بیت پشت سر هم ۱ تحت عنوان SFD خاتمه می‌یابد و بقیه اطلاعات فرستاده می‌شوند.



در واقع. بایت SFD برابر ۱۰۱۰۱۰۱۱ است و به صورت زیر فرستاده می‌شود:

10101010 10101010 10101010 10101010 10101010 10101010 10101010 10101011 .../...

****

Errors

1. Suppose there is a code that adds three bits to every bit pair: a copy of the original two bits and a parity bit (computed off the original pair, not all four bits). For example, 01 becomes 01011, and 11 becomes 11110. How many bit errors can this code detect? How many bit errors can this code correct (i.e., what is the largest k such that the code can correct all possible k-bit flips)? Justify your answer.

قواعد زیر حاکم است:

$B = b_4b_3b_2b_1b_0$
$b_2 = b_4$
$b_1 = b_3$
$b_0 = b_3 \text{ XOR } b_4$

مرحله به مرحله تفسیر می‌کنیم:

- خطا در ۱ بیت: b_4 ، b_3 ، b_2 ، b_1 با بیت متناظرشان چک می‌شوند و در صورت مغایرت می‌فهمیم در میانشان خطایی رخ داده است. سپس به $parity = b_0$ نگاه می‌کنیم که ببینیم با b_4b_4 مطابقت دارد و یا با b_2b_1 . اینگونه بیت خطا را متوجه می‌شویم و آن را تصحیح می‌کنیم. در صورت خطا در $parity = b_0$ نیز از مطابقت ۴ بیت اول و عدم مطابقت این بیت با آنها دقیقا آن را متوجه می‌شویم.
- خطا در ۲ بیت: مشخصا قابل تصحیح نیست. زیرا مثلا هر دو رشته‌ی ۱۰۱۰۱ و ۰۱۰۱۱ می‌توانند با دو تغییر، به ۱۱۱۱۱ تبدیل شوند و تمایزی در انتخابشان هنگام تصحیح نداریم. اما خطا قابل تشخیص است؛ زیرا یا یک تغییر یا دو تغییر در ۴ بیت اول داریم، که اگر ۱ تغییر باشد تفاوت با بیت متناظرش آن را مشخص می‌کند و اگر هم ۲ تغییر باشد، حتما باید در دو بیت متناظر (b_1 و b_3 یا b_2 و b_4) رخ داده باشد؛ که در این صورت $parity = b_0$ با این تغییر نمی‌خواند.
- خطا در۳ بیت: لزوما قابل تشخیص هم نیست؛ برای مثال رشته‌ی ۱۰۱۰۱ میتواند با ۳ تغییر به ۱۱۱۱۰ تبدیل شول که رشته‌ی به ظاهر valid ای است و ما متوجه بی خطا بودن یا ۳ خطا داشتن آن نمی‌شویم.

نتیجه می‌گیریم در ۵ بیت، ۱ خطا قابل اصلاح و ۲ خطا قابل تشخیص است.

****

2. A frame is m bits long and there is a bit error probability p. Suppose we can use an error correcting code that has an overhead of 3 bits and can detect any errors and correct errors of up to 1 bit. What is the probability of a successful transmission?

برای آن که ارسال موفقیت آمیز باشد، باید در فریم دریافت شده خطاها قابل اصلاح باشند؛ و این یعنی در هر فریم حداکثر ۱ خطا ظاهر شده باشد.

$$\Rightarrow P[Successful\ Transmission] = P[error = 0] + P[error = 1]$$

$$= (1-p)^{m+3} + (m+3).p(1-p)^{m+2}$$

****

3. Suppose that a frame is again m bits long with error probability p. Suppose now that an error detection code is used that has an overhead of 1 bit and which can detect any error. Assuming a real time application (no retransmissions) what is the probability of a successful transmission? what values of p will justify the error correction of the previous question and what values of p will make error detection more preferable? (compare their two probability of a successful transmission)

تشخیص خطا در هر صورتی امری مهم در ارسال داده reliable است. تصحیح خطا صرفا روشی برای کاهش تعداد درخواست های مجدد برای باز ارسال فریم‌ها است. در این سوال داریم:

$$\Rightarrow P[Successful\ Transmission] = P[error = 0]$$

$$= (1-p)^{m+1}$$

در مقایسه خواهیم داشت:

$$\frac{P_{detection}}{P_{correction}} = \frac{(1-p)^{m+1}}{(1-p)^{m+3} + (m+3).p(1-p)^{m+2}} = \frac{1}{(1-p)^2 + (m+3).p(1-p)}$$

$$= \frac{1}{1 + (m+2)p.(\frac{m+1}{m+2} - p)}$$

پس حد احتمال خطا، $p = 1 - \frac{1}{m+2}$ است؛ که اگر p بیشتر از این مقدار باشد تصحیح به صرفه نیست (این عدد نزدیک ۱ است و خطای کانال اینقدر زیاد نیست، پس بدون در نظر گرفتن زیاد شدن طول فریم، تقریبا همیشه ارسال با ۲ بیت بیشتر برای تصحیح به صرفه تر است. البته با صرف نظر از بده-بستان افزایش طول فریم!).

****

4. Suppose that data are transmitted in blocks of sizes 1000 bits. What is the maximum error rate under which error detection and retransmission mechanism (1 parity bit per block) is better than using Hamming code? Assume that bit errors are independent of one another and no bit error occurs during retransmission.

دو محاسبه انجام می‌دهیم (error rate را برابر p می‌گیریم):

1) Detection & Retransmission:

$$P[error\ occur\ in\ a\ frame] = 1 - (1-p)^{1001}$$

$$\Rightarrow \mathbb{E}[bit\ transfer] = 1001 \times P[No\ Error] + 2 \times 1001 \times P[Error\ Ocurs]$$

$$= 1001 \times (1-p)^{1001} + 2 \times 1001 \times [1 - (1-p)^{1001}]$$

$$\approx 1001(1 - 1001p) + 2002(1001p) = 1001 + 1001^2 p \approx 1001 + 10^6 p$$

2) Correction (Hamming Code):

$$single\ bit\ error\ Hamming = 10\ bits\ added$$

$$\Rightarrow \mathbb{E}[bit\ transfer] = 1010$$

به طور حدودی، حالت اول وقتی بهتر است $10^6 p$ از ۹ بیت کمتر باشد؛ یعنی به ازای error rate کمتر از 9×10^6 ، تشخیص و باز ارسال راندمان بیشتری دارد.

****

Cyclic Redundancy Check (CRC)

1. Explain the purpose of CRC and its basics.

روش CRC ، یک روش تشخیص خطا است که طی آن به انتهای داده، تعدادی بیت اضافه می‌شود که کارایی‌های زیر را فراهم می‌آورد:

1) تشخیص همیشگی ۱ بیت خطا
2) تشخیص ۲ بیت خطا در بسیاری از حالات
3) تشخیص چند بیت خطا در بعضی موارد
4) تشخیص یک burst کوچک از خطا
5) تشخیص یک burst بزرگ از خطا در موارد خاص

روش آن به طور خلاصه شرح داده می‌شود:

در CRC ، به رشته مبنای ۲ داده‌ها یک چند جمله‌ای متناظر می‌شود:

$$M(x) = 1 \cdot x^9 + 1 \cdot x^8 + 0 \cdot x^7 + 1 \cdot x^6 + 0 \cdot x^5 + 1 \cdot x^4 + 1 \cdot x^3 + 0 \cdot x^2 + 1 \cdot x^1 + 1 \cdot x^0$$
$$M(x) = x^9 + x^8 + x^6 + x^4 + x^3 + x^1 + 1$$

در هر استاندارد نیز یک چند جمله‌ای مشخص برای تقسیم چند جمله‌ای مشخص شده است (به طوری که بیشترین کارایی را داشته باشد). برای مثال در استاندارد CRC-16 :

$$G(x) = x^{16} + x^{15} + x^2 + 1$$

یا در استاندارد CRC-32 :

$$G(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$$

سپس، باقیمانده تقسیم رشته ورودی (M) و یک تعدادی Place Holder = 0 به طول درجه رشته استاندارد، به رشته استاندارد (G) به نام R ، به انتهای رشته اضافه شده و فرستاده می‌شود:

$$CRC = \text{remainder of } \left[M(x) \times \frac{x^n}{G(x)} \right]$$

در گیرنده نیز همین کار تکرار می‌شود و در صورتی که باقیمانده صفر باشد، رشته دریافتی خطا ندارد.

****

2. What are the most used types of CRC and in what protocols are they used?

انواع زیر پر استفاده اند:

- CRC-40-GSM در GSM control channel
- CRC-32 در (HDLC), ANSIX3.66, ITU-T V.42, ISO/IEC/IEEE 802-3 (Ethernet), SATA, PKZIP, Gzip, Bzip2, POSIXcksum, PNG, ZMODEM,
- CRC-30 در CDMA
- CRC-16-IBM در USB

****

3. Assume this is an Ethernet frame, calculate the CRC and check it with the CRC field of the frame.(Make sure to omit the fields of the frame that are not considered in CRC calculation beforehand): 00 10 A4 7B EA 80 00 12 34 56 78 90 08 00 45 00 00 2E B3 FE 00 00 80 11 05 40 C0 A8 00 2C C0 A8 00 04 04 00 04 00 00 1A 2D E8 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 10 11 B3 31 88 1B

قسمت CRC مربوط به رشته بالا، با فرض استاندارد CRC-32 ، برابر ۳۲ بیت انتهایی آن، یا B3 31 88 1B می‌باشد. در کد هم همین نتیجه حاصل شده است:

CRC_32_maker.m | next_32_bits.m

```
function crc = CRC_32_maker(data)
%data_bin_size = 4 * size(data, 2);
data_bin = '';
for i = 1:size(data,2)
    dc = dec2bin(hex2dec(data(i)));
    for j = 1:4-size(dc,2)
        dc = strcat('0', dc);
    end
    data_bin = strcat(data_bin, dc );
end
%data_bin = dec2bin(hex2dec(data))
%for i = 1:data_bin_size-size(data_bin, 2)
%    data_bin = strcat('0', data_bin);
%end
data_bin2 = data_bin;
for i = 1:8:size(data_bin,2)
    data_bin2(i:i+7) = fliplr(data_bin(i:i+7));
end
data_bin = strcat(data_bin2, '00000000000000000000000000000000');
for i = 1:32
    if data_bin(i)=='0'
        data_bin(i)='1';
    else
        data_bin(i)='0';
    end
end
% crc = '11111111111111111111111111111111';
```

Command Window

New to MATLAB? See resources for Getting Started.

```
    '1B8831B3'

>> CRC_32_maker('0010A47BEA800012345678900800450002EB3FE000080110540C0A8002CC0A8000404000400001A2DE8

ans =

    '1B8831B3'
```

****

Investigating an Ethernet Interface

1. Determine the current speed being used by your Ethernet adapter. To this end, run one of the following commands in terminal:

```
sudo dmesg | grep eth
sudo ethtool eth0
cat /sys / class /net / eth0 / speed
```

در این دستگاه، اطلاعات اینچنینی قابل دستیابی نیست (و این دستورات در این کامپیوتر (OSX Based) وجود ندارند؛ لذا هر اطلاعات ممکنی که از شبکه قابل دستیابی است در تصاویر زیر و در بخش بعد آمده‌اند.

****

2. Look at the information on your Ethernet device. Use the following command.

```
ifconfig
```

- What is your MAC address?
- Look this up using an OUI lookup table. Does it match what you expect?
- What is the default frame size (MTU)?
- How many bytes have been received (RX)?
- How many bytes have been transmitted (TX)?
- Has your device seen any collisions?
- Has your device dropped any packets?

```
en0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        ether a0:99:9b:0a:09:1f
        inet6 fe80::cbb:22ce:24d6:261a%en0 prefixlen 64 secured scopeid 0x5
        inet 192.168.1.105 netmask 0xffffff00 broadcast 192.168.1.255
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
p2p0: flags=8843<UP,BROADCAST,RUNNING,SIMPLEX,MULTICAST> mtu 2304
        ether 02:99:9b:0a:09:1f
        media: autoselect
        status: inactive
awdl0: flags=8943<UP,BROADCAST,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1484
        ether 76:a7:c8:b3:7f:62
        inet6 fe80::74a7:c8ff:feb3:7f62%awdl0 prefixlen 64 scopeid 0x7
        nd6 options=201<PERFORMNUD,DAD>
        media: autoselect
        status: active
en1: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        options=60<TSO4,TSO6>
        ether 6a:00:00:6f:a5:30
        media: autoselect <full-duplex>
        status: inactive
en2: flags=8963<UP,BROADCAST,SMART,RUNNING,PROMISC,SIMPLEX,MULTICAST> mtu 1500
        options=60<TSO4,TSO6>
        ether 6a:00:00:6f:a5:31
        media: autoselect <full-duplex>
        status: inactive
bridge0: flags=8863<UP,BROADCAST,SMART,RUNNING,SIMPLEX,MULTICAST> mtu 1500
        options=63<RXCSUM,TXCSUM,TSO4,TSO6>
        ether 6a:00:00:6f:a5:30
        Configuration:
                id 0:0:0:0:0:0 priority 0 hellotime 0 fwddelay 0
                maxage 0 holdcnt 0 proto stp maxaddr 100 timeout 1200
                root id 0:0:0:0:0:0 priority 0 ifcost 0 port 0
                ipfilter disabled flags 0x2
        member: en1 flags=3<LEARNING,DISCOVER>
                ifmaxaddr 0 port 8 priority 0 path cost 0
        member: en2 flags=3<LEARNING,DISCOVER>
                ifmaxaddr 0 port 9 priority 0 path cost 0
        nd6 options=201<PERFORMNUD,DAD>
        media: <unknown type>
        status: inactive
```

در کامپیوتر مکینتاش (OSX Based) نتایج بدست آمده به شکل بالا است. پس برای بدست آوردن RX و TX از دستورات دیگری استفاده می‌کنیم:

```
Name  Mtu   Network       Address            Ipkts Ierrs     Ibytes    Opkts Oerrs     Obytes  Coll
lo0   16384 <Link#1>                        289445     0  473858852   289445     0  473858852     0
lo0   16384 127           localhost         289445     -  473858852   289445     -  473858852     -
lo0   16384 localhost   ::1                 289445     -  473858852   289445     -  473858852     -
lo0   16384 fe80::1%lo0 fe80:1::1           289445     -  473858852   289445     -  473858852     -
gif0* 1280  <Link#2>                            0     0          0        0     0          0     0
stf0* 1280  <Link#3>                            0     0          0        0     0          0     0
XHC20 0     <Link#4>                            0     0          0        0     0          0     0
en0   1500  <Link#5>    a0:99:9b:0a:09:1f 76094937     0 64829797029 60485976     0 28326018141     0
en0   1500  macintoshs- fe80:5::cbb:22ce: 76094937     - 64829797029 60485976     - 28326018141     -
en0   1500  192.168.1     192.168.1.105   76094937     - 64829797029 60485976     - 28326018141     -
p2p0  2304  <Link#6>    02:99:9b:0a:09:1f        0     0          0        0     0          0     0
awdl0 1484  <Link#7>    76:a7:c8:b3:7f:62        0     0          0    14921     0    3213275     0
awdl0 1484  macintoshs- fe80:7::74a7:c8ff        0     -          0    14921     -    3213275     -
en1   1500  <Link#8>    6a:00:00:6f:a5:30        0     0          0        0     0          0     0
en2   1500  <Link#9>    6a:00:00:6f:a5:31        0     0          0        0     0          0     0
bridg 1500  <Link#10>   6a:00:00:6f:a5:30        0     0          0        1     0        342     0
utun0 2000  <Link#11>                            0     0          0        2     0        200     0
utun0 2000  fe80::3260: fe80:b::3260:1b6:        0     -          0        2     -        200     -
```

طبق اطلاعات بالا، دو MAC Address وجود دارد که مطابق زیر است:

6a:00:00:6f:a5:30
6a:00:00:6f:a5:31

در نتیجه برای OUI مقدار زیر را داریم :

6a:00:00

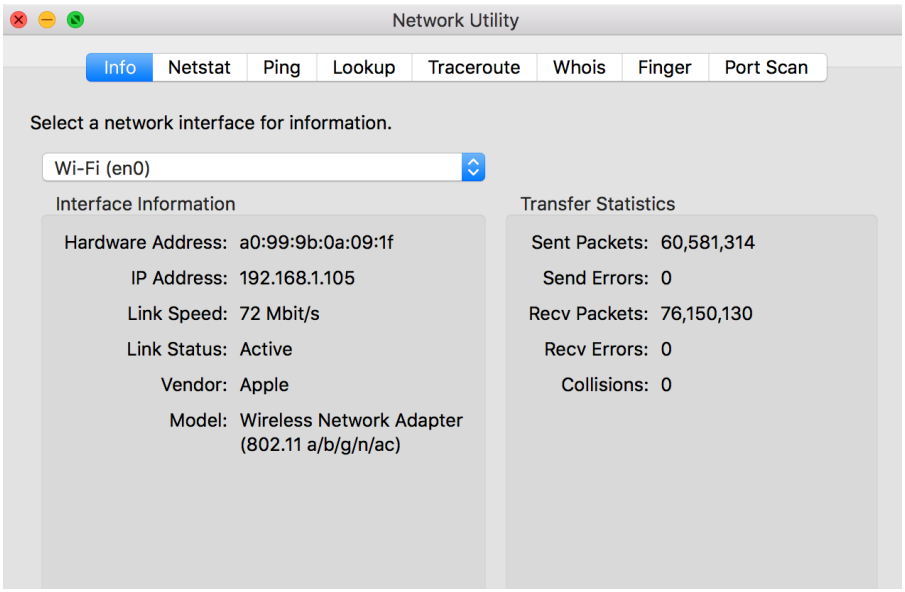
که این مشخص نیست برای چه شرکتی است!

همچنین برای ارتباطات Ethernet در این کامپیوتر (en0 ، en1 ، en1 و bridge) طول فریم (MTU) برابر ۱۵۰۰ است.

برای درگاه en0 روی این کامپیوتر، مقدار RX (Ipkts) برابر ۷۶۰۹۴۹۳۷۵ (معادل ۶۴۸۲۹۷۹۷۰۲۸ بایت) و مقدار TX (Opkts) برابر ۶۰۴۸۵۹۷۶ (معادل ۲۸۳۲۶۰۱۸۱۴۱ بایت) است.

در مورد خطا نیز، مشاهده می‌شود که هم خطای ورودی (Ierrs) و هم خطای خروجی (Oerrs) برابر صفر است.

همچنین برخورد (collision) نیز برابر صفر است و فریمی دچار مشکل نشده است:



❊❊❊❊❊❊❊❊

3. If the collision count is low, can you explain why that is?

احتمالا دلیل آن این است که تعداد دستگاه‌های متصل به شبکه کم است (یک الی سه دستگاه) و در این حالت، احتمال زیادی برای خطا وجود ندارد.

❊❊❊❊❊❊❊❊

Exploring Ethernet Frames

1. Save the selected packet from File¿Export specified Packets and choose Selected packets and upload it with your report.

سه فریم در زیر آورده شده اند؛ اولی با پروتکل TCP ، دومی با پروتکل UDP و سومی با پروتکل ARP.

```
▼ Frame 28618: 1454 bytes on wire (11632 bits), 1454 bytes captured (11632 bits) on interface 0
  ▶ Interface id: 0 (en0)
    Encapsulation type: Ethernet (1)
    Arrival Time: Nov 10, 2018 14:58:39.031999000 +0330
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1541849319.031999000 seconds
    [Time delta from previous captured frame: 0.001626000 seconds]
    [Time delta from previous displayed frame: 0.001626000 seconds]
    [Time since reference or first frame: 10484.482998000 seconds]
    Frame Number: 28618
    Frame Length: 1454 bytes (11632 bits)
    Capture Length: 1454 bytes (11632 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:tcp]
    [Coloring Rule Name: Bad TCP]
    [Coloring Rule String: tcp.analysis.flags && !tcp.analysis.window_update]
▼ Ethernet II, Src: Tp-LinkT_47:76:8f (f8:1a:67:47:76:8f), Dst: Apple_0a:09:1f (a0:99:9b:0a:09:1f)
  ▶ Destination: Apple_0a:09:1f (a0:99:9b:0a:09:1f)
  ▶ Source: Tp-LinkT_47:76:8f (f8:1a:67:47:76:8f)
    Type: IPv4 (0x0800)
▼ Internet Protocol Version 4, Src: 104.66.65.160, Dst: 192.168.1.105
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 1440
    Identification: 0xaea1 (44705)
  ▶ Flags: 0x4000, Don't fragment
    Time to live: 52
    Protocol: TCP (6)
    Header checksum: 0x26c3 [validation disabled]
    [Header checksum status: Unverified]
    Source: 104.66.65.160
    Destination: 192.168.1.105
▼ Transmission Control Protocol, Src Port: 443, Dst Port: 52105, Seq: 5126715, Ack: 734, Len: 1388
    Source Port: 443
    Destination Port: 52105
    [Stream index: 207]
    [TCP Segment Len: 1388]
```

برای این فریم با پروتکل TCP داریم:

Source MAC Address (Src):	f8:1a:67:47:76:8f
Source OUI:	f8:1a:67
Source OUI Owner:	Tp_LinkT
Destination MAC Address (Dst):	a0:99:9b:0a:09:1f
Destination OUI:	a0:99:9b
Destination OUI Owner:	Apple Inc.
Ethernet Type:	IPv4 (0x0800)

❊❊❊❊❊❊❊❊

```
▼ Frame 12: 82 bytes on wire (656 bits), 82 bytes captured (656 bits) on interface 0
  ▶ Interface id: 0 (en0)
    Encapsulation type: Ethernet (1)
    Arrival Time: Nov 10, 2018 12:03:54.952617000 +0330
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1541838834.952617000 seconds
    [Time delta from previous captured frame: 0.101854000 seconds]
    [Time delta from previous displayed frame: 0.101854000 seconds]
    [Time since reference or first frame: 0.403616000 seconds]
    Frame Number: 12
    Frame Length: 82 bytes (656 bits)
    Capture Length: 82 bytes (656 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:ip:udp:data]
    [Coloring Rule Name: UDP]
    [Coloring Rule String: udp]
▼ Ethernet II, Src: AsustekC_eb:36:1a (20:cf:30:eb:36:1a), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  ▶ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  ▶ Source: AsustekC_eb:36:1a (20:cf:30:eb:36:1a)
    Type: IPv4 (0x0800)
▼ Internet Protocol Version 4, Src: 172.20.27.38, Dst: 172.20.27.255
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▶ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 68
    Identification: 0x19c1 (6593)
  ▶ Flags: 0x0000
    Time to live: 128
    Protocol: UDP (17)
    Header checksum: 0x919a [validation disabled]
    [Header checksum status: Unverified]
    Source: 172.20.27.38
    Destination: 172.20.27.255
▶ User Datagram Protocol, Src Port: 60956, Dst Port: 1947
▶ Data (40 bytes)
```

برای این فریم با پروتکل UDP داریم:

Source MAC Address (Src):	20:cf:30:eb:36:1a
Source OUI:	20:cf:30
Source OUI Owner:	AsustekC Inc. (!)
Destination MAC Address (Dst):	ff:ff:ff:ff:ff:ff
Destination OUI:	ff:ff:ff
Destination OUI Owner:	Broadcast
Ethernet Type:	IPv4 (0x0800)

❊❊❊❊❊❊❊❊

```
▼ Frame 37: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) on interface 0
  ▶ Interface id: 0 (en0)
    Encapsulation type: Ethernet (1)
    Arrival Time: Nov 10, 2018 12:03:55.262427000 +0330
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1541838835.262427000 seconds
    [Time delta from previous captured frame: 0.000416000 seconds]
    [Time delta from previous displayed frame: 0.000416000 seconds]
    [Time since reference or first frame: 0.713426000 seconds]
    Frame Number: 37
    Frame Length: 60 bytes (480 bits)
    Capture Length: 60 bytes (480 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:arp]
    [Coloring Rule Name: ARP]
    [Coloring Rule String: arp]
▼ Ethernet II, Src: Dell_b0:dc:0d (14:fe:b5:b0:dc:0d), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  ▶ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  ▶ Source: Dell_b0:dc:0d (14:fe:b5:b0:dc:0d)
    Type: ARP (0x0806)
    Padding: 000000000000000000000000000000000000
▼ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: Dell_b0:dc:0d (14:fe:b5:b0:dc:0d)
    Sender IP address: 172.20.25.129
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 172.20.26.122
```

برای این فریم با پروتکل ARP داریم:

Source MAC Address (Src):	14:fe:b5:b0:dc:0d
Source OUI:	14:fe:b5
Source OUI Owner:	Dell Inc. (!)
Destination MAC Address (Dst):	ff:ff:ff:ff:ff:ff
Destination OUI:	ff:ff:ff
Destination OUI Owner:	Broadcast
Ethernet Type:	IPv4 (0x0800)

❋❋❋❋❋❋❋

2. Based on the Ethernet frame structure, what are the source and destination MAC addresses and Who owns the OUI of them.

این سوال در زیر هر اسنپ‌شات از فریم‌های بالا جواب داده شده است.

****

3. What is the value of the Ethernet type field ?

این سوال نیز در زیر هر اسنپ‌شات از فریم‌های بالا جواب داده شده است.

****

4. What is the purpose of an ARP packet? Select one the ARP packets, what is its destination address and why?

پروتکل ARP ، یا Address Resolution Protocol ، پروتکلی برای مشخص کردن MAC Address از طریق IP است. یک ساختار داده به نام ARP Cache وجود دارد که متناظر با IP هر دستگاه (که برای ارتباط دو دستگاه استفاده می‌شود) یک MAC Address (که برای ارتباطات در لایه فیزیکی استفاده می‌شود) ذخیره کرده است. در این حالت IP با MAC Address به لایه فیزیکی اطلاع داده می‌شود. اما در صورت آن که این IP در حافظه ثبت نشده باشد، پای ARP به میان می‌آید که یک پیام به صورت Broadcast (با آدرس مقصد ff:ff:ff) برای همه دستگاه‌ها ارسال می‌کند و MAC Address متناظر با IP را می‌پرسد و دستگاهی که صدا شده جواب می‌دهد و مخابرات ادامه می‌یابد.

****

5. There is no CRC field in the frames you captured, search and explain the reason.

زیرا نرم‌افزار Wireshark، مستقیما فریم ارسالی و دریافتی را نشان نمی‌دهد؛ بلکه فریم دریافتی ابتدا توسط برنامه دیگری CRC اش بررسی می‌شود (و عملیات لازم روی آن انجام می‌شود)، سپس قسمت اصللی اطلاعات داده فریم وارد نرم‌افزار Wireshark می‌گردد.

**** پایان گزارش ****