**BÀI TẬP TUẦN 10/12/2024 (edit)**

**Họ và tên: Phạm Mạch Lam Phương**

**MSSV:** [**24210165**](mailto:%324%32%3101%36%35@m%73%2e%75%69t%2e%65%64u.v%6e)

**Bài 1:** Xét 1 router kết nối 3 mạng con [subnet]: Subnet 1, Subnet 2, Subnet 3. Giả sử tất cả các interface trong 3 mạng con này đòi hỏi phải có phần đầu trong địa chỉ là 223.1.17.0/24. Cũng giả sử rằng Subnet 1 được đòi hỏi phải hỗ trợ đến 125 interface, Subnet 2 và 3 mỗi mạng con được yêu cầu phải cung cấp 60 interface. Hãy cung cấp 3 địa chỉ mạng (ở dạng a.b.c.d/x) thỏa mãn 3 ràng buộc này

Bài làm:

**Subnet 1: 223.1.17.0/25**

Giải thích:

Subnet 1 được đòi hỏi phải hỗ trợ đến 125 interface nghĩa là cần ít nhất 127 địa chỉ (bao gồm địa chỉ mạng và broadcast)

Số bit host cần để có 125 host: tìm n thỏa 2^n – 2 >= 125 host => n = 7

Vậy Subnet 1 là 32 – n = 32 – 7 = 25 => 223.1.17.0/25 (128 địa chỉ)

**Subnet 2: 223.1.17.0/26 và Subnet 3: 223.1.17.64/26**

Giải thích:

Subnet 2 và 3: Được đòi hỏi phải hỗ trợ đến 60 interface nghĩa là cần ít nhất 62 địa chỉ (bao gồm địa chỉ mạng và broadcast)

Số bit host cần để có 60 host: tìm n thỏa 2^n – 2 >= 60 host => n = 6

Tổng số địa chỉ của 1 mạng con là 2^6 = 64 (bao gồm địa chỉ mạng và broadcast). Vậy bước nhảy giữa các mạng con là 64 trong octet thứ 4

Vậy Subnet 2 là 32 – n = 32 – 6 = 26 => 223.1.17.0/26 (64 địa chỉ) và Subnet 3 là 223.1.17.64/26 (64 địa chỉ)

**Bài 2:**

* ﻿Xem 1 mạng datagram sử dụng địa chỉ 32-bit. Giả sử 1 router có 4 đường ra/vô (tức 4 interface), đánh số từ 0 đến 3, và các gói tin được chuyển ra các interface theo quy định sau:
* ﻿Mô tả cách thức forwarding table của bạn xác định interface phù hợp cho các datagram hướng tới địa chỉ đích sau:
  + 11100010 01000100 10010001 10010001
  + 11100010 00000000 10010001 10010001
  + A white rectangular box with black lines

    Description automatically generated with medium confidence11100001 10000000 00010001 01110111

Bài làm:

* + 11100010 01000100 10010001 10010001 => Interface 0

**Giải thích**

Phần đầu **11100010** trùng khớp dãy địa chỉ đích **11100010** 00000000 00000000 00000000 → 11100010 01111111 11111111 11111111

Phần sau01000100 10010001 10010001 nằm trong phạm vi 00000000 → 01111111

* + 11100010 00000000 10010001 10010001=> Interface 1

**Giải thích**

Phần đầu **11100010 00000000** trùng khớp dãy địa chỉ đích **11100010 00000000** 00000000 00000000 → 11100010 00000000 11111111 11111111

* + 11100001 10000000 00010001 01110111 => Interface 3

**Giải thích**

Phần đầu **11100001** không trùng khớp với bất kỳ dãy địa chỉ nào => Trường hợp khác (interface 3)

**Bài 3:** Tính Checksum (từng cặp số sau)

Cặp 1:

1000110000000011              
0100111101000101

Cặp 2:

0100000110010000               
0110110011001100

Bài làm:

Cặp 1: 1000110000000011 + 0100111101000101

Thực hiện cộng từ phải sang trái (1 + 1 = 0, nhớ 1).

1 0 0 0 1 1 0 0 0 0 0 0 0 0 1 1

0 1 0 0 1 1 1 1 0 1 0 0 0 1 0 1

------------------------------------

1 1 0 1 1 0 1 1 0 1 0 0 1 0 0 0 => Chỉ có 16 bit không có overflow để xử lý them

Đảo ngược tất cả các bit:

1 1 0 1 1 0 1 1 0 1 0 0 1 0 0 0 => 0 0 1 0 0 1 0 0 1 0 1 1 0 1 1 1

Kết quả check sum 1000110000000011 + 0100111101000101 = 0010010010110111

Cặp 2: 0100000110010000 + 0110110011001100

Thực hiện cộng từ phải sang trái (1 + 1 = 0, nhớ 1).

0 1 0 0 0 0 0 1 1 0 0 1 0 0 0 0

0 1 1 0 1 1 0 0 1 1 0 0 1 1 0 0

------------------------------------

1 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0 => Chỉ có 16 bit không có overflow để xử lý them

Đảo ngược tất cả các bit:

1 0 1 0 1 1 1 0 0 1 0 1 1 1 0 0 => 0 1 0 1 0 0 0 1 1 0 1 0 0 0 1 1

Kết quả check sum 0100000110010000 + 0110110011001100 = 0101000110100011

**Bài 4:** Cho địa chỉ IP 192.168.1.0/24, chia địa chỉ thành các mạng (M) với số lượng host như sau:

* + M1: 50 host
  + M2=M3: 10 host
  + M4: 2 host

Bài làm:

* M1: 50 host

Giải thích:

M1 được đòi hỏi phải hỗ trợ đến 50 host nghĩa là cần ít nhất 52 địa chỉ (bao gồm địa chỉ mạng và broadcast)

Số bit host cần để có 50 host: tìm n thỏa 2^n – 2 >= 50 host => n = 6

Vậy M1 là 32 – n = 32 – 6 = 26 => **192.168.1.0/26** (64 địa chỉ)

* M2 = M3: 10 host

Giải thích:

M2 được đòi hỏi phải hỗ trợ đến 10 host nghĩa là cần ít nhất 12 địa chỉ (bao gồm địa chỉ mạng và broadcast)

Số bit host cần để có 10 host: tìm n thỏa 2^n – 2 >= 10 host => n = 4

Vậy M2 là 32 – n = 32 – 4 = 28 => **192.168.1.0/28** (16 địa chỉ) và M3 là **192.168.1.16/28** (16 bước nhảy)

* M4: 2 host

M4 được đòi hỏi phải hỗ trợ đến 2 host nghĩa là cần ít nhất 4 địa chỉ (bao gồm địa chỉ mạng và broadcast)

Số bit host cần để có 2 host: tìm n thỏa 2^n – 2 >= 2 host => n = 2

Vậy M4 là 32 – n = 32 – 2 = 30 => **192.168.1.0/30** (4 địa chỉ)

**Bài 5:** Cho một datagram có chiều dài là 10000 bytes. Khi datagram này đến router R1 thì bị chia nhỏ vì R1 chỉ cho phép MTU = 2500 byte. Hãy cho biết giá trị của các trường total length, Identification, flag và offset của các fragment sau khi bị chia nhỏ.

Bài làm: IP Dataframe có kích thước <= MTU = 2500

MTU (Maximum Transmission Unit) của R1 là 2500 bytes. Giả sử rằng header của mỗi đoạn là 20 bytes, kích thước dữ liệu tối đa trong mỗi đoạn sẽ là:

2500−20=2480 bytes

2. Tính số lượng đoạn:

Tổng chiều dài của datagram là 10000 bytes. Vì mỗi đoạn có thể chứa tối đa 2480 bytes dữ liệu, số lượng đoạn cần thiết là:

⌈100002480⌉=5 đoạn

3. Tính toán giá trị của các trường trong các đoạn:

* **Fragment 1:**
* Total Length: 2500 bytes
* Identification: Giữ nguyên (giả sử là 12345)
* Flags: More Fragments (MF) = 1
* Fragment Offset: 0
* **Fragment 2:**
* Total Length: 2500 bytes
* Identification: 12345
* Flags: MF = 1
* Fragment Offset: 2480 / 8 = 310
* **Fragment 3:**
* Total Length: 2500 bytes
* Identification: 12345
* Flags: MF = 1
* Fragment Offset: (2480 \* 2) / 8 = 620
* **Fragment 4:**
* Total Length: 2500 bytes
* Identification: 12345
* Flags: MF = 1
* Fragment Offset: (2480 \* 3) / 8 = 930
* **Fragment 5:**
* Total Length: 1080 bytes (10.000 – (2480\*4) + 20)
* Identification: 12345
* Flags: MF = 0 (đây là đoạn cuối cùng)
* Fragment Offset: (2480 \* 4) / 8 =

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Fragment | Total Length | Identification | Flags | Fragment Offset |
| 1 | 2500 | = x | MF = 1 | 0 |
| 2 | 2500 | = x | MF = 1 | 310 |
| 3 | 2500 | = x | MF = 1 | 620 |
| 4 | 2500 | = x | MF = 1 | 930 |
| 5 | 100 | = x | MF = 0 | 1240 |

**Bài 6:** Dùng thuật toán Dijkstra tìm đường đi ngắn nhất từ nút u đến tất cả các nút còn lại. Vẽ bảng chi tiết tìm đường từng bước. Sau đó, vẽ cây đường đi ngắn nhất

A diagram of a network

Description automatically generated

Bài làm:

