Ch4. P5

(a)

11100000 00 => Link 0

11100000 01000000 => Link 1

1110000 => Link 2

11100001 1 => Link 3

Otherwise => 3

(b)

Prefix match for first address is 5th entry: link interface 3

Prefix match for second address is 3nd entry: link interface 2

Prefix match for third address is 4th entry: link interface 3

P6

00000000 ~ 00111111 => Link 0

01000000 ~ 01011111 => Link 1

01100000 ~ 01111111 => Link 2

10000000 ~ 10111111 => Link 2

11000000 ~ 11111111 => Link 3

Number of addresses for interface 0 = 2^6 = 64

number of addresses for interface 1 = 2^5 = 32

number of addresses for interface 2 = 2^6 + 2^5 = 96

number of addresses for interface 3 = 2^6 = 64

P12

(a)

Subnet A: 214.97.255/24 (256 addresses)

Subnet B: 214.97.254.0/25 - 214.97.254.0/29 (128-8 = 120 addresses)

Subnet C: 214.97.254.128/25 (128 addresses)

Subnet D: 214.97.254.0/31 (2 addresses)

Subnet E: 214.97.254.2/31 (2 addresses)

Subnet F: 214.97.254.4/30 (4 addresses)

(b)

Router 1

11010110 01100001 11111111 => Subnet A

11010110 01100001 11111110 0000000 => Subnet D

11010110 01100001 11111110 000001 => Subnet F

Router 2

11010110 01100001 11111111 0000000 => Subnet D

11010110 01100001 11111110 0 => Subnet B

11010110 01100001 11111110 0000001 => Subnet E

Router 3

11010110 01100001 11111111 000001 => Subnet F

11010110 01100001 11111110 0000001 => Subnet E

11010110 01100001 11111110 1 => Subnet C

P17

a. Yes, based on the observation that the identification numbers of the IP packets generated by hosts behind the NAT are sequentially assigned, we can outline a simple technique to detect the number of unique hosts behind the NAT.

b. If the identification numbers are randomly assigned instead of being sequentially assigned, the outlined technique would not work. In the case of random assignment, the identification numbers do not provide any meaningful information or sequence that can be used to track the number of unique hosts.

Ch5. P3.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| step | N’ | D(t), p(t) | D(u), p(u) | D(v), p(v) | D(w), p(w) | D(y), p(y) | D(z), p(z) |
| 0 | x | [**∞**](https://namu.wiki/w/%E2%88%9E) | [**∞**](https://namu.wiki/w/%E2%88%9E) | 3, x | 6, x | 6, x | 8, x |
| 1 | xv | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 2 | xvu | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 3 | xvuw | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 4 | xvuwy | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 5 | xvuwyt | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |
| 6 | xvuwytz | 7, v | 6, v | 3, x | 6, x | 6, x | 8, x |

P5.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | u | v | x | y | z |
| v | [1](https://namu.wiki/w/%E2%88%9E) | [0](https://namu.wiki/w/%E2%88%9E) | [3](https://namu.wiki/w/%E2%88%9E) | [3](https://namu.wiki/w/%E2%88%9E) | [5](https://namu.wiki/w/%E2%88%9E) |
| x | [4](https://namu.wiki/w/%E2%88%9E) | [3](https://namu.wiki/w/%E2%88%9E) | [0](https://namu.wiki/w/%E2%88%9E) | [3](https://namu.wiki/w/%E2%88%9E) | 2 |
| z | [6](https://namu.wiki/w/%E2%88%9E) | 5 | 2 | [5](https://namu.wiki/w/%E2%88%9E) | 0 |

P7.

(a)

(b)

c(x, w) = 2

c(x, y) = 5

c(x, w) > 6 will change neighbors.

(c)

c(x, w) = 2

c(x, y) = 5

It will not change neighbors.

P11

(a)

(1) Router z

w: , y:

(2) Router w

y: , z:

(3) Router y

w: , z:

(b)

There will be problem because messages can change between w, y, z continuously.

At 31 times, there will no update.

(c) It can cut between y link and z link.

P14

(a) eBGP

(b) iBGP

(c) eBGP

(d) iBGP

P15

(a) I1, because it is the least cost path from 1d to 1c.

(b) I2, because it has the closest next hop.

(c) I1, it has the shortest AS-PATH.

P16

Advertising its route to D through east coast peering point with C

P17

P19

A advertises to B (AS-PATH AW and AV).

A advertises to C (AV).

C uses AS-PATH (CBAW, CBAV, CAV).