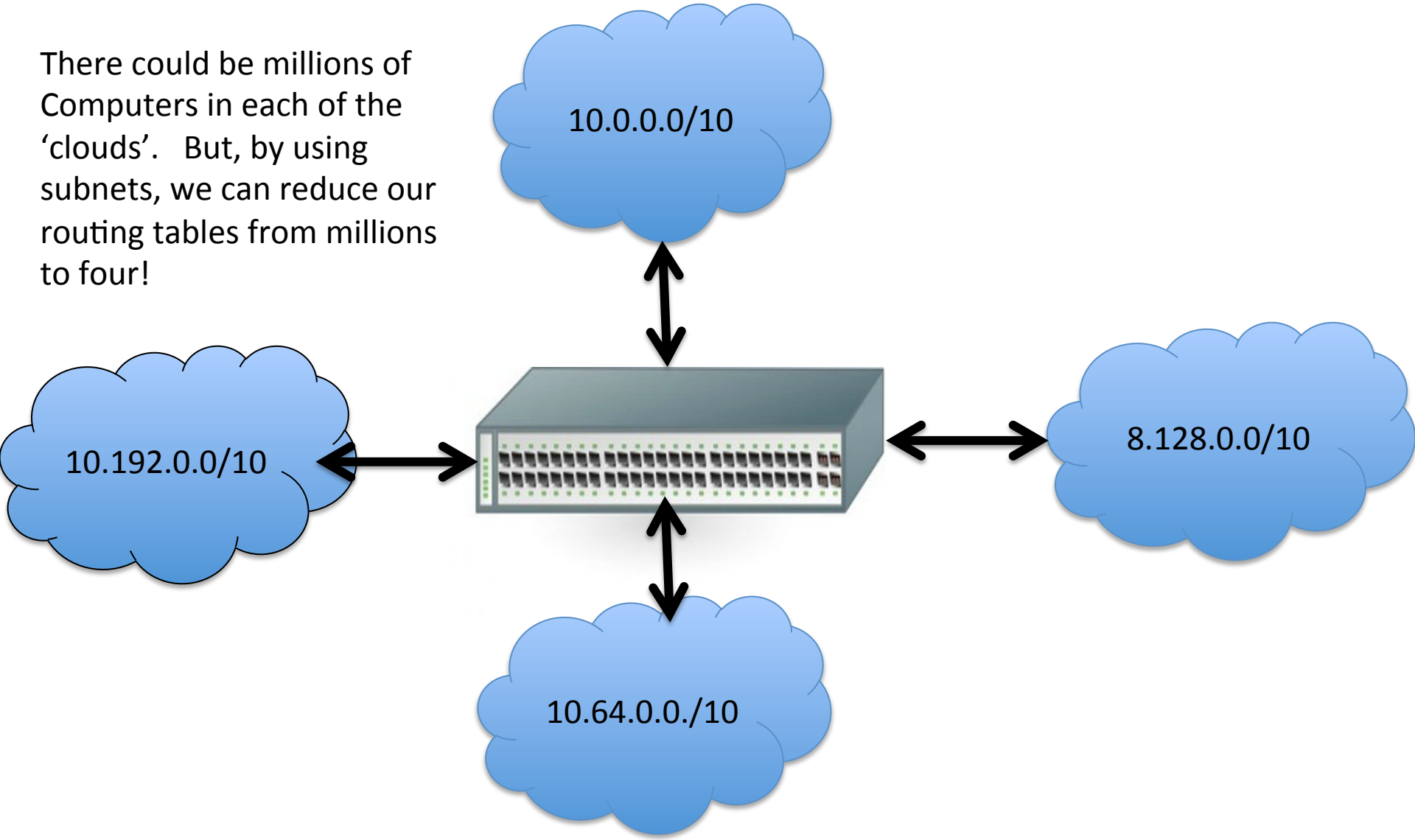
Why Subnets?

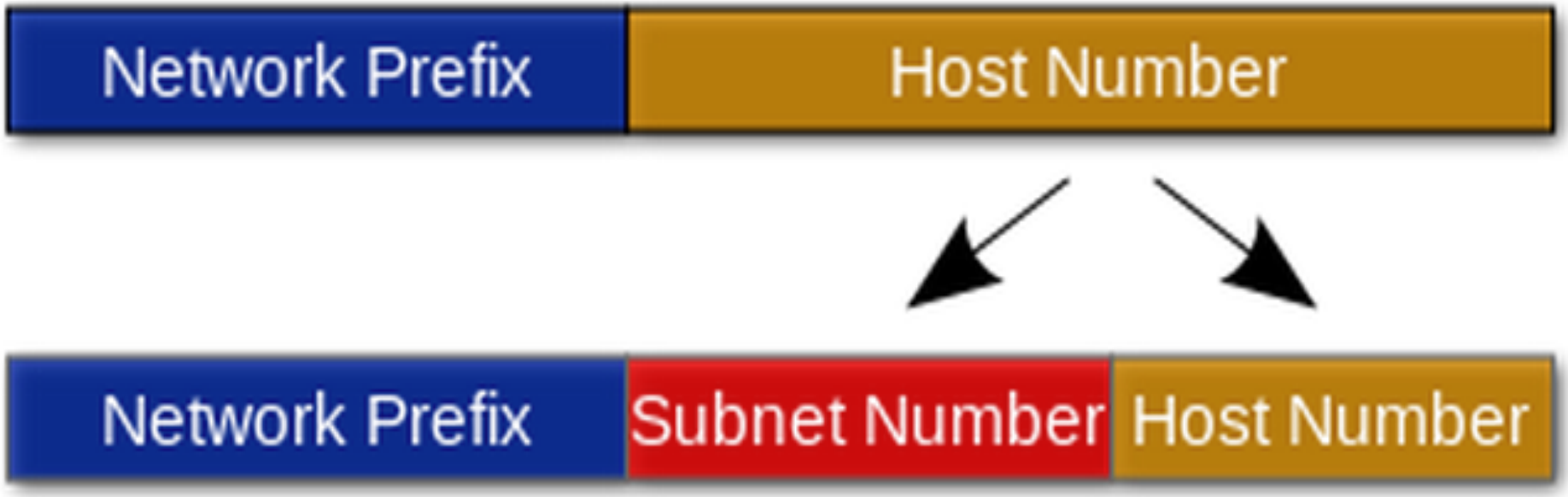
Subnets (Sub Networks) : Logical subdivision of an IP Network

Easier to route packets!

There could be millions of Computers in each of the 'clouds'. But, by using subnets, we can reduce our routing tables from millions to four!



How Do we create a subnet from a given network?



How to figure out a subnet from your network address!

- Given an IP address (10.193.15.26/10) of your network, figure out the various subnets needed.
- In our example, we need 5 subnets of equal size

Calculate how many subnets you will need, in terms of bits! How many bits, minimum, are needed to have the five values required!

How to figure out a subnet from your network address!

- Take the original given IP address (10.193.15.26/10) of your network, in binary form:

$00001010.11000001.00001111.00011010_2$

- Take the original given IP Mask, in binary form:

$11111111.11000000.00000000.00000000_2$

Calculate how many subnets you will need!

3 bits are required to represent 5 subnets (or levels)!

How to figure out a subnet from your network address!

What do we know? We need 5 subnet addresses of equal size, from a given network of IP address (10.193.15.26/10).

- We now have 5 IP Mask (Original bits in BLUE, and the extended bits in RED), in binary form:

$00001010.11000000.00000000.00000000_2$

$00001010.11001000.00000000.00000000_2$

$00001010.11010000.00000000.00000000_2$

$00001010.11011000.00000000.00000000_2$

$00001010.11100000.00000000.00000000_2$

We will have 3 unused subnets (we only require 5), but that is ok!

We can treat all 5 as given subnets.

Notice the original blue bits are identical, the original subnet from the given network of 10.192.0.0/10. The red bits are using the free bits, and allocating 5 new subnets!

How to figure out a subnet from your network address!

What do we know? We need 5 subnet addresses of equal size, from the original dotted netmask of 255.192.0.0 by 3 bits, to 255.248.0.0. We extend the mask by 3 bits:

- We now have 1 IP Mask (extended bits in RED), in binary form:

11111111.11111000.00000000.00000000$_2$

Great! So … Now what?

Now you can resolve for the *different* subnets:

$11111111.11111000.00000000.00000000_2$ - MASK

$00001010.11000000.00000000.00000000_2$ - 1st Subnet

$00001010.11001000.00000000.00000000_2$ - 2nd Subnet

$00001010.11100000.00000000.00000000_2$ - Last Subnet

So, these 13 bits on the most significant bits (remember the slash /13 notation!) should *NOT* be changed.

How to figure out the Subnet! Pt2

Just like a single subnet, but now you have several of them!

Remember, those ‘1’ bits in the mask, you cant modify, for each subnet!

All the ‘0’ bits in the new mask, you can modify.

Convert to Dotted Decimal IPv4 Address as Required

$00001010.11000000.00000000.00000000_2$ - First Subnet
(10.192.0.0)
$00001010.11000000.00000000.00000001_2$ - First Client
(10.192.0.1)
….
$00001010.11000111.11111111.11111110_2$ - Last Client
(10.199.255.254)
$00001010.11000111.11111111.11111111_2$ - Broadcast
(10.199.255.255)

How many clients? You can count the '0' in the mask (19) or 32-(# 1's in mask) = 32 – 13 = 19. The number of clients is 2^{19}-2

$2^{(\#\text{ zeros in mask})}$ -2

Convert to Dotted Decimal IPv4 Address as Required

$00001010.11001000.00000000.00000000_2$ - Second Subnet (10.200.0.0)

$00001010.11001000.00000000.00000001_2$ - First Client (10.200.0.1)

….

$00001010.11001111.11111111.11111110_2$ - Last Client (10.207.255.254)

$00001010.11001111.11111111.11111111_2$ - Broadcast (10.207.255.255)

How many clients? You can count the '0' in the mask (19) or 32-(# 1's in mask) = 32 – 13 = 19. The number of clients is 2^{19}-2

$2^{(\#\text{ zeros in mask})}$ -2

Convert to Dotted Decimal IPv4 Address as Required

$00001010.11100000.00000000.00000000_2$ - Last (5^{th}) Subnet (10.224.0.0)

$00001010.11100000.00000000.00000001_2$ - First Client (10.224.0.1)

….

$00001010.11100111.11111111.11111110_2$ - Last Client (10.231.255.254)

$00001010.11100111.11111111.11111111_2$ - Broadcast (10.231.255.255)

How many clients? You can count the '0' in the mask (19) or 32-(# 1's in mask) = 32 – 13 = 19. The number of clients is 2^{19}-2

$2^{(\#\text{ zeros in mask})}$ -2