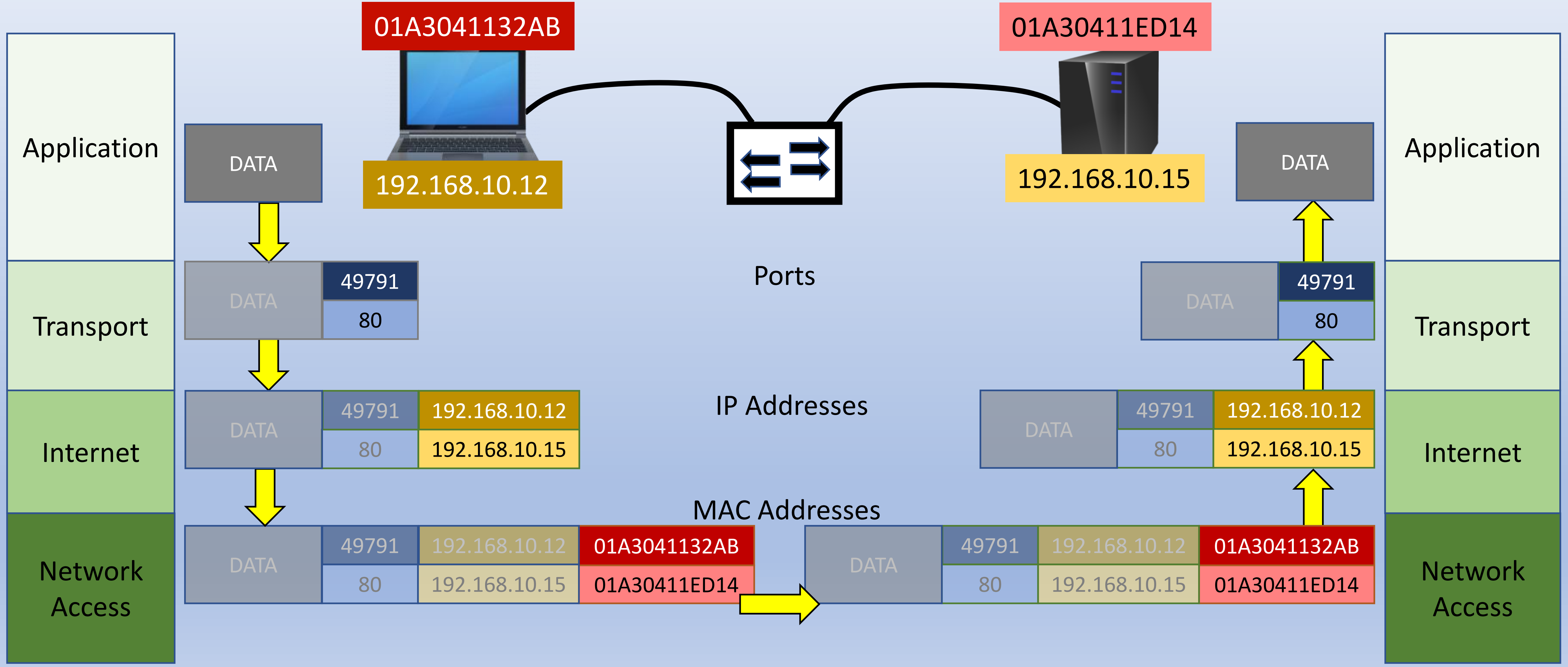
Networking Introduction

OSI vs TCP/IP

<table>
<tr><th>OSI</th><th></th><th>TCP/IP</th></tr>
<tr><td>Application</td><td rowspan="3">Application Protocols</td><td rowspan="3">Application</td></tr>
<tr><td>Presentation</td></tr>
<tr><td>Session</td></tr>
<tr><td>Transport</td><td>UDP TCP</td><td>Transport</td></tr>
<tr><td>Network</td><td>IP</td><td>Internet</td></tr>
<tr><td>Data Link</td><td rowspan="2">MAC</td><td rowspan="2">Network Access</td></tr>
<tr><td>Physical</td></tr>
</table>

Encapsulation Decapsulation



Addressing

- Physical – Media Access Control (MAC Address) of NIC
- Logical – IP Address of NIC
- Network Services – Port numbers

Physical Address (MAC Address)

- MAC Address - basically made up of two parts
  - Vendor Address
  - Random Address
- 013E1FBB23A1
  - 013E1FBB23A1 – First part is the Vendor Address
  - 013E1FBB23A1 – Last part is the Random Address
- How computers communicate with each other

Binary

- Consists of 0 or 1 *(off / on)*
- Binary number can be converted to decimal
- 00000000 bin = 0 dec
- 11111111 bin = 255 dec
- 00000001 bin = 1 dec
- 00000010 bin = 2 dec
- 00000011 bin = 3 dec
- 00000100 bin = 4 dec

For each 1 in binary add the corresponding decimal values:

Binary to Decimal conversion							
1	1	1	1	1	1	1	1
128	64	32	16	8	4	2	1
11111111 bin => 128+64+32+16+8+4+2+1 = 255 dec							

Binary to Decimal conversion							
1	0	1	0	1	0	1	1
128	64	32	16	8	4	2	1
10101011 bin => 128+32+8+2+1 = 171 dec							

Binary to Decimal conversion							
0	0	0	1	0	0	1	0
128	64	32	16	8	4	2	1
10010 bin => 16+2 = 18 dec							

Binary and IP Addresses

- 131.107.1.4
  - Dotted decimal notation
  - Each number is an octet (represents 8 binary digits)
  - 0 = 00000000 (Minimum value)
  - 255 = 11111111 (Maximum value)
- 131 = 10000011
- 107 = 01101011
- 1 = 00000001
- 4 = 00000100
- 131.107.1.4 = 10000011 01101011 00000001 00000100

Logical Address (IP Address)

- IP Addresses describe two things - Network section & Host section
  - Network section: is like the name of a street
  - Host section is like the house number
- Examples:
  - 192.168.0.1
    - 192.168.0.1 – Network section, 192.168.0.1 – Host section
  - 131.107.1.4
    - 131.107.1.4 – Network section, 131.107.1.4 – Host section
  - 10.34.2.1
    - 10.34.2.1 – Network section, 10.34.2.1 – Host section

Original IP Classes (assignable addresses)

- Class A
  - First octet 1-126 (Binary 00000000 – 01111111) *[0 & 127 by convention are not used)*
  - First octet is the network section
  - Last three octets are the Host section
- Class B
  - First octet 128-191 (Binary 10000000 – 10111111)
  - First two octets are the network section
  - Last two octets are the Host section
- Class C
  - First octet 191-223 (Binary 11000000 – 11011111)
  - First three octets are the network section
  - Last octet is the Host section

Subnet Mask

- Shows which part of the IP address is the:
  - Network section
  - Host section
- Determines whether a destination is on:
  - Local subnet
  - Remote subnet
- Traditional Class Subnet masks
  - Class A = 255.0.0.0 = 11111111 00000000 00000000 00000000
  - Class B = 255.255.0.0 = 11111111 11111111 00000000 00000000
  - Class C = 255.255.255.0 = 11111111 11111111 11111111 00000000
- Traditional masks wasted a lot of address
  - Solution = Classless inter-domain routing (CIDR)

ARP – Address Resolution Protocol

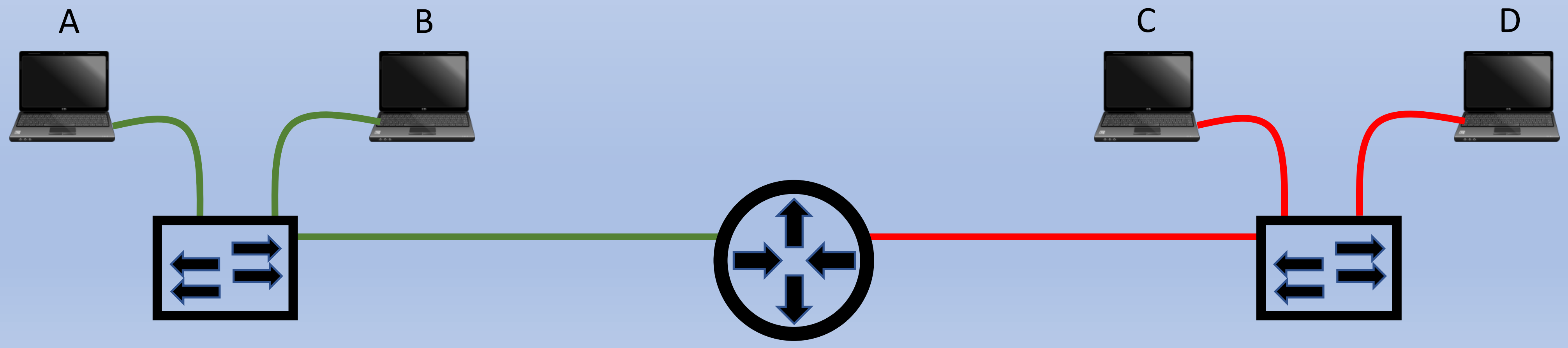
- ARP resolves an IP address into its corresponding MAC address
- ARP request is sent as a broadcast packet
- Asks who owns an IP address
  - IP owner answers with its MAC address with a unicast packet
  - ARP resolution information is cached in memory for about 10 minutes max.
- Routers do not pass broadcasts
  - Cannot ARP for an IP on a remote subnet
  - To communicate with a remote host, route packet through a router
  - Set Default Gateway IP so hosts can be used to find the default router
- Use arp -a to show the contents of the ARP cache
- Use arp -d to delete the contents of the ARP Cache

Determine how to route traffic

- Each time a network transmission happens
  - Sender need to determine if destination is local or remote
- If the destination IP address is on the local network
  - ARP for the destination IP address
- If the destination IP address is on a remote network
  - ARP for the IP address of the Default Gateway

Determine if destination is local or remote

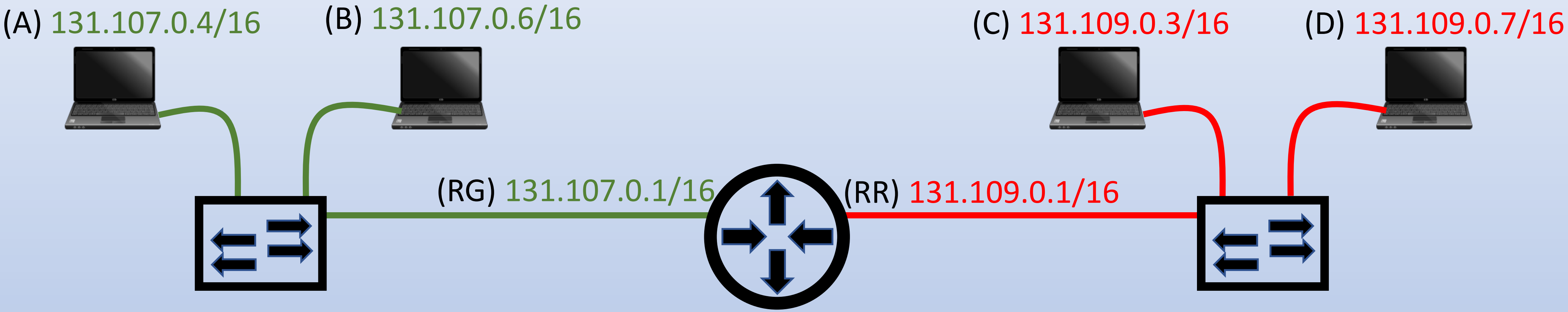
- AND source IP address with subnet mask
- AND destination IP address with subnet mask
- Compare both results
  - if same then local (Computers A & B)
  - if different then remote (Computers A & C)



Boolean AND calculation

BOOLEAN AND				
0	AND	0	=	0
0	AND	1	=	0
1	AND	0	=	0
1	AND	1	=	1

Determining if hosts are local or remote



- A -> B (Local)
  - 131.107.0.4 AND 255.255.0.0 = 131.107.0.0
  - 131.107.0.6 AND 255.255.0.0 = 131.107.0.0
- A -> C (Remote)
  - 131.107.0.4 AND 255.255.0.0 = 131.107.0.0
  - 131.109.0.3 AND 255.255.0.0 = 131.109.0.0

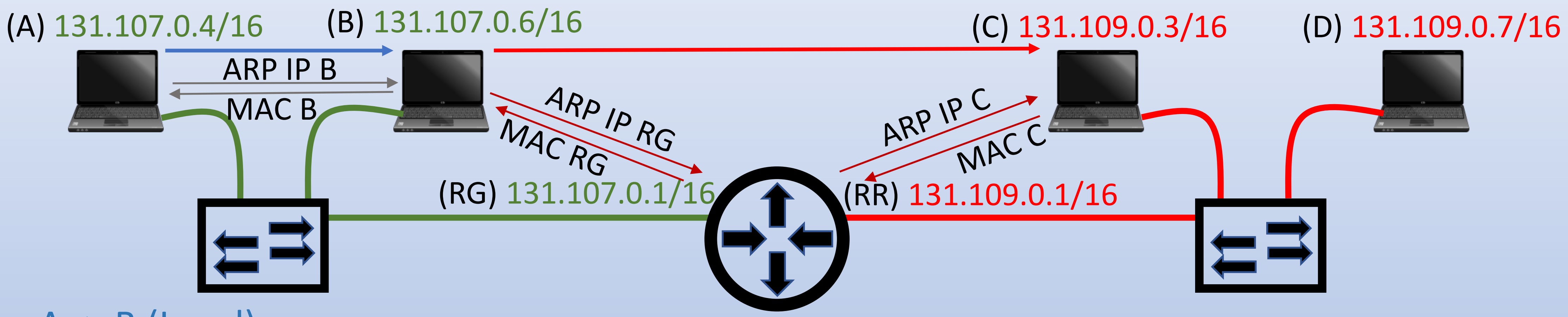
AND calculation (A -> B)

- 131.107.0.4 AND 255.255.0.0
  - 10000011 01101011 00000000 00000100 (131.107.0.4)
  - 11111111 11111111 00000000 00000000 (255.255.0.0)
  - 10000011 01101011 00000000 00000000 (131.107.0.0 - AND Result)
- 131.107.0.6 AND 255.255.0.0
  - 10000011 01101011 00000000 00000110 (131.107.0.6)
  - 11111111 11111111 00000000 00000000 (255.255.0.0)
  - 10000011 01101011 00000000 00000000 (131.107.0.0 - AND Result)

AND calculation (A -> C)

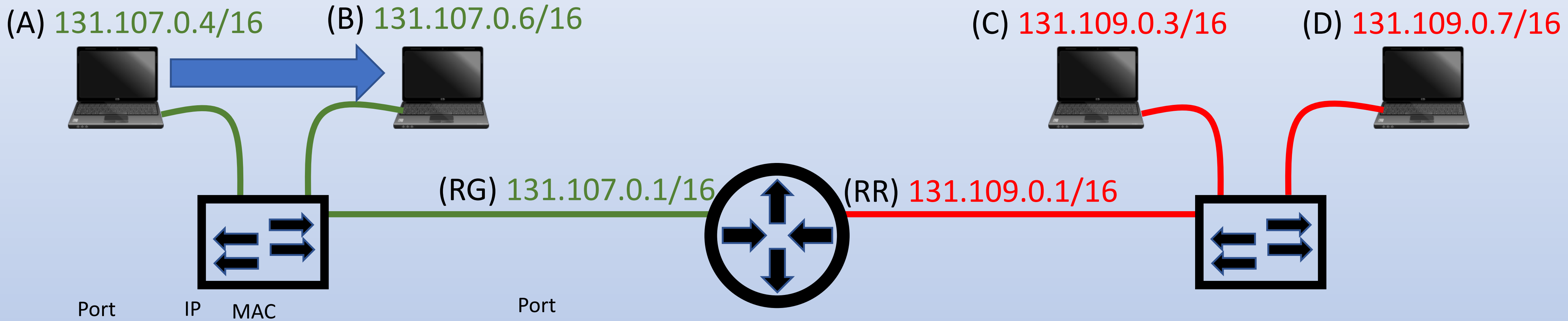
- 131.107.0.4 AND 255.255.0.0
  - 10000011 01101011 00000000 00000100 (131.107.0.4)
  - 11111111 11111111 00000000 00000000 (255.255.0.0)
  - 10000011 01101011 00000000 00000000 (131.107.0.0 - AND Result)
- 131.109.0.3 AND 255.255.0.0
  - 10000011 01101101 00000000 00000011 (131.109.0.3)
  - 11111111 11111111 00000000 00000000 (255.255.0.0)
  - 10000011 01101101 00000000 00000000 (131.109.0.0 - AND Result)

ARP Requests



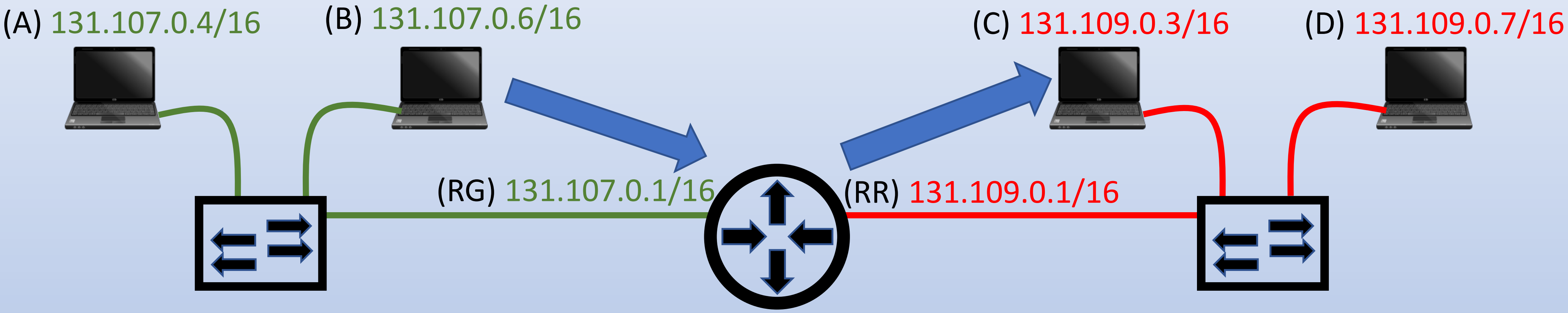
- A -> B (Local)
  - From A, ARP for destination IP B
  - Reply with MAC address from B
- B -> C (Remote)
  - From B, ARP for the router RG IP address
  - Reply with MAC address from RG
  - From Router, ARP for destination C IP address
  - Reply with MAC address from C

Routing local (A -> B [Webserver])



	Port	IP	MAC	
	50331	A	A	SRC
	80	B	B	DST

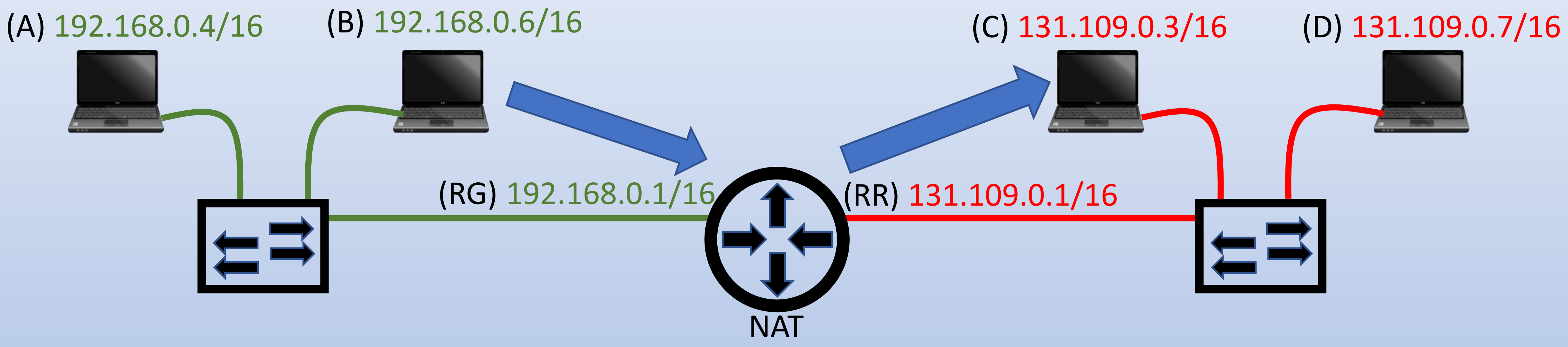
Routing remote (B -> C [Webserver])



	Port	IP	MAC	
	49791	B	B	SRC
	80	C	RG	DST

	Port	IP	MAC	
	49791	B	RR	SRC
	80	C	C	DST

Routing and NAT (B -> C [Webserver])



	Port	IP	MAC	
	49961	B	B	SRC
	80	C	RG	DST

	Port	IP	MAC	
	10001	RR	RR	SRC
	80	C	C	DST

NAT TABLE
192.168.0.6:49961 ---> 131.109.0.1:10001

Subnetting Introduction

Why do we subnet

- To convert a single network into smaller networks
- To reduce wasting of IP addresses
- To control network traffic congestion
- To manage the growth of a network
- To make network security more granular

How do we subnet

- You need to answer several questions before you are ready to subnet
  - What network address are we starting with
  - How many smaller networks do we need (Subnets)
  - How many IP addresses do we need on each subnet (minimum: 1 per NIC)
  - What growth do we expect to happen within the subnets
  - Is it possible to achieve the result with the given conditions

What network are we starting with

- Class A, B, C each have default subnet masks /8, /16, /24
- Some IP addresses on each network have special purposes
- For example:
  - Network ID: 131.12.0.0/16
  - Broadcast ID: 131.12.255.255/16
  - The first address in each subnet is its Network ID
  - The last address in each subnet is its Broadcast ID
  - These two addresses are not used for addressing hosts

Subnetting 131.12.0.0/16

- How many subnets are required
  - 28
- How many IP addresses are required per subnet
  - 1300
- What growth do we expect on these subnets
  - 5% growth in the number of subnets
  - 10% growth in IP addresses
- Max subnets: 28 + 5% = 28 + 1.4 (~ 2) = 30
- Max IP addresses: 1300 + 10% = 1300 + 130 = 1430

Calculate subnets

- Subnets Required: 30
- IP addresses / subnet required : 1430
- Starting Network ID: 131.12.0.0/16
- /16 means
  - That the first 16 bits are network
  - This leaves the last 16 bits for us to slice up into smaller subnets
- How many bits are required to make 32 subnets
- 5 Bits
  - 00000 -> 11111 = is 32 possible numbers (we need 30)
- That leaves 11 bits for hosts
  - 00000000000 -> 11111111111 = 2048 possible numbers – 2 (for Net ID and BC ID)
  - 2046 IP addresses on each of the 32 subnets
- The subnetting is possible!

How is the subnetting performed

- The new subnet mask is the old mask + the network bit required
  - Old mask = 16 bits
  - Subnet bits required for subnetting = 5 bits
  - New subnetting mask = 21 bits
- Old 131.12.0.0/16 in binary looks like this:
  - 10000011 00001100 00000000 00000000
  - 11111111 11111111 00000000 00000000 (16 bits)
- New 131.12.0.0/21 in binary looks like this:
  - 10000011 00001100 00000000 00000000 (Subnet portion Host portion)
  - 11111111 11111111 11111000 00000000 (21 bits)

How to create the first subnet

- 10000011 00001100 00000000 00000000
- The first subnet would look like this in binary:
  - 10000011 00001100 00000000 00000000 - Subnet ID
  - 10000011 00001100 00000000 00000001 - First valid address
  - 10000011 00001100 00000111 11111110 - Last valid address
  - 10000011 00001100 00000111 11111111 - Broadcast ID
- In Decimal
  - 131.12.0.0 - Subnet ID
  - 131.12.0.1 - First valid address
  - 131.12.7.254 - Last valid address
  - 131.12.7.255 - Broadcast ID

How to create the second subnet

- 10000011 00001100 00001000 00000000
- The first subnet would look like this in binary:
  - 10000011 00001100 00001000 00000000 - Subnet ID
  - 10000011 00001100 00001000 00000001 - First valid address
  - 10000011 00001100 00001111 11111110 - Last valid address
  - 10000011 00001100 00001111 11111111 - Broadcast ID
- In Decimal
  - 131.12.8.0 - Subnet ID
  - 131.12.8.1 - First valid address
  - 131.12.15.254 - Last valid address
  - 131.12.15.255 - Broadcast ID

How to create the second subnet

- 10000011 00001100 00010000 00000000
- The first subnet would look like this in binary:
  - 10000011 00001100 00010000 00000000 - Subnet ID
  - 10000011 00001100 00010000 00000001 - First valid address
  - 10000011 00001100 00010111 11111110 - Last valid address
  - 10000011 00001100 00010111 11111111 - Broadcast ID
- In Decimal
  - 131.12.16.0 - Subnet ID
  - 131.12.16.1 - First valid address
  - 131.12.23.254 - Last valid address
  - 131.12.23.255 - Broadcast ID

How to create the fourth subnet

- 10000011 00001100 00011000 00000000
- The first subnet would look like this in binary:
  - 10000011 00001100 00011000 00000000 - Subnet ID
  - 10000011 00001100 00011000 00000001 - First valid address
  - 10000011 00001100 00011111 11111110 - Last valid address
  - 10000011 00001100 00011111 11111111 - Broadcast ID
- In Decimal
  - 131.12.24.0 - Subnet ID
  - 131.12.24.1 - First valid address
  - 131.12.31.254 - Last valid address
  - 131.12.31.255 - Broadcast ID