e-PG Pathshala

Subject : Computer Science

Paper: Computer Networks

Module: IP – Subnets and CIDR
Module No: CS/CN/15

Quadrant 1 – e-text

We looked at the basic features of IP in the previous module. In this module, we look at the problems with the basic class-based addressing, and how subnets, CIDR and other mechanisms have evolved and developed to address them.

Learning Objectives:

To understand how IP addressing has evolved using

- the idea of subnetting and the use of subnet masks
- Class-less addressing mechanism
- Network address aggregation, and
- Network address translation.

15.1 IP addressing – issues

We have seen how the basic IP addressing mechanism divides the available addresses into classes to cater to small (up to 254 nodes), medium (up to 64K nodes), and large (2million nodes) sized networks (classes C, B and A respectively). The number of networks belonging to each class is in reverse proportion – with a possibility of <126 class A networks, 14K class B, and 2million class C networks.

To understand the problem with this addressing mechanism, try guessing this: Which class addresses would be maximum used (more consumption) ?

Not class A – it is not very often that you would need such a large network. Not class C – there are actually plenty of class C network addresses. It is actually class B that was getting consumed at a very fast rate. Note that any network which expected to have more than 250 nodes would ask for a class B address. So it was class B addresses that started getting rapidly used up.

Network addresses were getting used up, but there were many node addresses in each network that were unused. (Remember that if your network had 1000 nodes, you would

have got a class B network assigned, but out of the available 64K addresses assigned to you, you would only be using 1K; the rest of the 63K would be wasted – and cannot be assigned to any other network !). Some internal fragmentation kind of scenario !

Hence, a mechanism to effectively use these unused addresses was first developed – by bringing another level in the hierarchy of IP addressing – "subnets" ! Subnetting allows one network address to be shared by multiple sub-networks. Let us see how this is done.

15.2 Subnetting

Instead of looking at the address as a "network address+ node address", we add one level between the two called the subnet. That is, the node part of address is divided into a subnet part and a node part. So we now look at addresses as network part, sub-network part, and then host part.

Network part	Subnet part	Host part

To understand this, consider a class B address, say, 145.67.0.0. This network can cater to 64K nodes. Assume that we want to share this among 4 sub-networks with 16K nodes each. That means we need 4 sub-network numbers to distinguish between them. So, we use the higher-order two bits of the host part as a subnet address, and the rest of the 14 bits as the node address to identify the 16K nodes.

Network (bits 31-16) : 145.67	Subnet (bits 15:14)	Host (bits 13:0)

Similarly, if we wanted to share the network among 64 different sub-networks with (1K nodes each), the first six bits of the node part of the address will refer to the subnet, and the rest 10 bits will refer to one of the 1K nodes in the corresponding subnet. As you can see, you can extend the idea to any combination of subnets as long as we divide them in powers of 2. When the higher order "n-bits" of the node-part of the address are used to represent the subnet address, we can have 2^n different subnets.

The obvious advantage of this subnet idea is that it allows better use of the address space. If a network needs 1K addresses, it need not take a full Class B address and waste 64K addresses. Instead it can share it with 63 other subnets.

We have logically seen what a subnet is. Physically, what does a subnet mean? It

basically refers to the set of device interfaces with the same subnet part of the IP address, where the devices can physically reach each other *without an intervening router.* In other words, a network connected to an interface of a router could form a subnet. Thus a router could connect to different subnets at its different interfaces. One simple recipe to identify a subnet would be to detach the interface to its router. It would create islands of isolated networks, each of which would be a subnet.

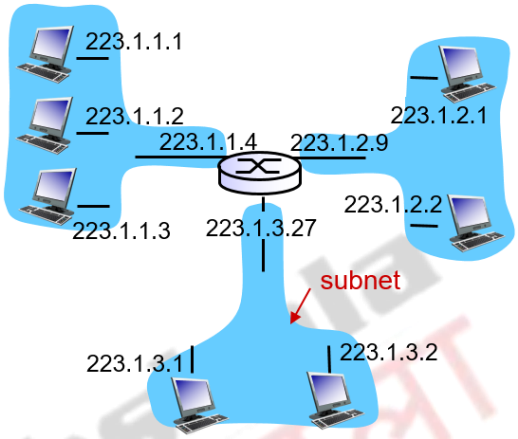


Figure 15.1 A router connecting 3 subnets

Consider Fig. 15.1 which shows a router connected to three different subnets on its 3 interfaces. You can see that one subnet has an address prefix of 223.1.1, another has 223.1.2, and the third has 223.1.3. The three interfaces of the router have these three prefixes respectively. If you disconnect the network with same prefix from the routers interface, you can identify the corresponding subnet.

Another example with many routers and subnets is shown in Fig.15.2.

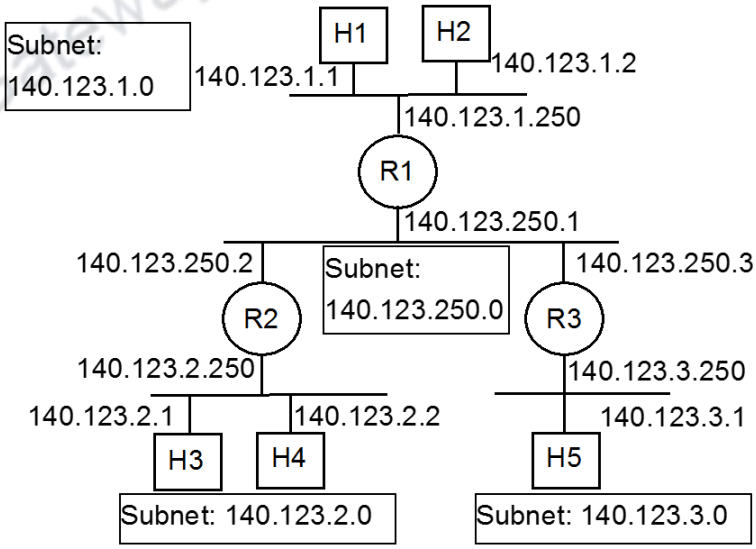


Figure 15.2 Subnets – an example

Router R1 connects two subnets, 140.23.1.0 (with hosts H1 and H2), and 140.123.250.0. R2 is connected to 140.123.250.0 and 140.23.2.0(hosts H3, H4). R3 is connected to 140.123.250.0 and 140.23.3.0 (host H5).

The routers R1, R2 and R3 have to know that the class B address of 140.23.0.0 is being shared by subnets, and they also have to know how the addresses are shared. This information is specified by what are known as ***subnet masks***.

15.2.1 Subnet Masks

Subnet masks define the variable partition of the host part of a class A or class B address. A mask of 32 bits is specified to show the host part and (the network + subnet) part of the address. The bits which are 1 in the mask correspond to the bits in the IP address which correspond to the network+subnet part. The bits which are zero indicate the node part. Let us understand this using an illustration (Fig.15.3).

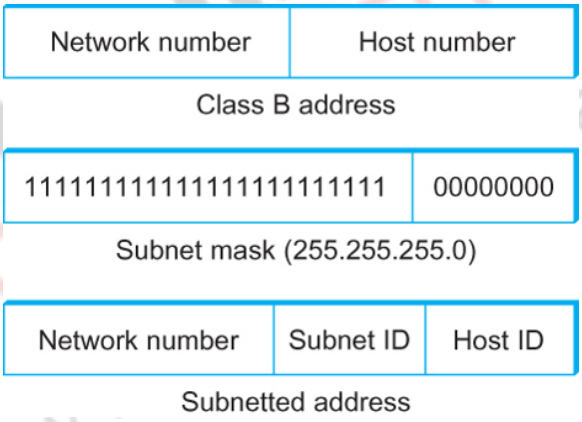


Figure 15.3 Subnet mask

Consider the subnetting of a class B network address, which has 16 bits network part and 16 bits node p

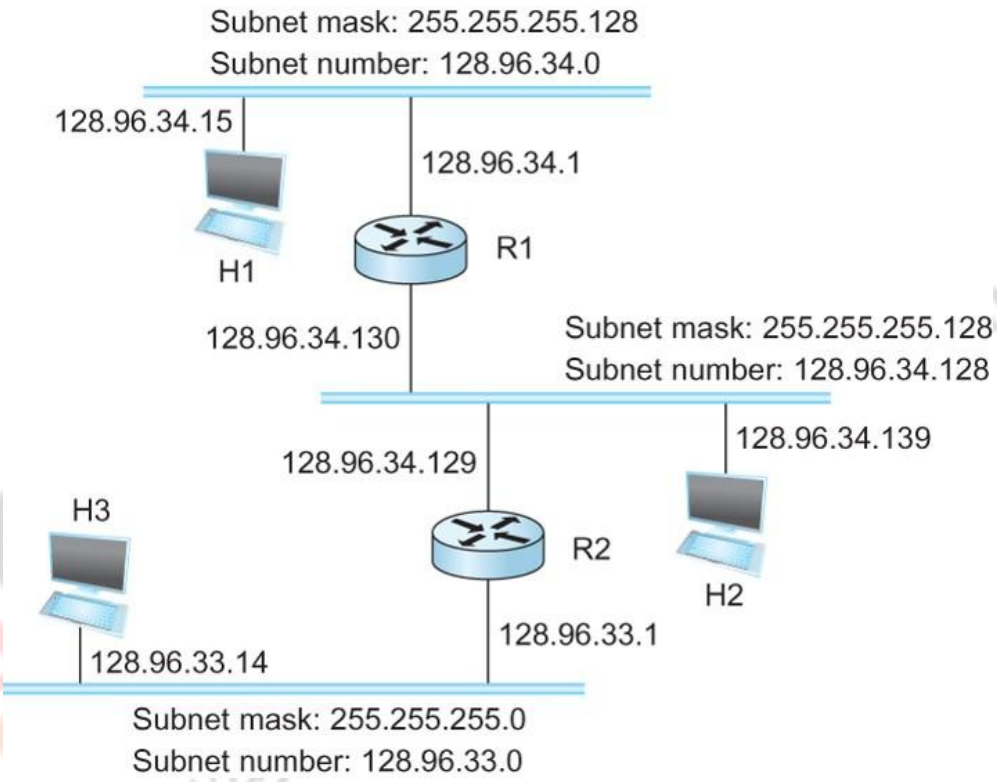
art into subnets having 256 nodes. Each subnet should be able to distinguish between the 256 nodes in it. So the node part of the mask – i.e., last 8 bits (bits 7-0) will be zeroes. With 256 nodes in each subnet, we can have 256 different subnets. So, the next higher order 8 bits (bits 15-8), in the network mask will be ones. The bits 31-16 are the network part – hence the corresponding bits in the mask are set to 1s. Thus we get a mask of 24 ones followed by 8 zeroes, which in the dotted decimal notation is written as 255.255.255.0.

Let us now look at how the forwarding algorithm changes in the presence of subnets.

15.2.2 Forwarding with Subnets

The subnetting idea is normally used only at the edges. That is, the edge routers are the ones that use the subnet mask to identify the network+subnet part of the address, and send the packets to the corresponding interface. The other core routers upstream would treat this entire set of subnets as a single class B (or A) network.

The information in the forwarding tables of the edge router would consist of triples of <subnet number, subnet mask, nextHop>. An example with subnets, and the forwarding table at router R1 are given in Fig. 15.4.



Forwarding table at R1

SubnetNumber	SubnetMask	NextHop
128.96.34.0	255.255.255.128	Interface 0
128.96.34.128	255.255.255.128	Interface 1
128.96.33.0	255.255.255.0	R2

Figure 15.4 Subnet based forwarding table – an example

This table would be used in the following manner: When a packet arrives, its destination address will be masked with each mask in the table, and the masked value will be compared with the corresponding subnet number. If it matches, it will be forwarded to the corresponding nextHop interface/router.

15.2.3 Forwarding Algorithm (with subnets)

The algorithm for forwarding when handling subnets is as follows:

```
D = destination IP address
for each entry < SubnetNum, SubnetMask, NextHop>
D1 = SubnetMask & D
if D1 == SubnetNum
if NextHop is an interface
  deliver datagram directly to destination
else
  deliver datagram to NextHop (a router)
```

15.2.4 Subnets – some interesting notes

In the subnet masks given in the examples, we can notice that the 1's in the subnet masks are contiguous. But it is not necessary for it to be so. However, it is convenient to interpret the addresses when it is contiguous.

Also, as we mentioned earlier, it should be noted that the subnet is not visible from the rest of the Internet. It is an edge-router phenomenon. In that sense, it only allows sharing of IP addresses towards the edge, and hence does not totally solve the address-space usage issue.

However, the technique of using masks gives us an idea of moving towards a class-less addressing scheme, which could solve the problems of the classful addressing scheme.

15.2.5 Classful addressing – problems

As we mentioned earlier, one of the disadvantages of the classful addressing is the inefficient use of address space. Let us look at some numbers to understand this inefficiency. A network with two nodes needs to be given a class C address. It's address usage efficiency is ~ 2/255 (0.7%). A network with 256 nodes needs a class B address – an address efficiency of ~ 256/65535 (0.3%). This is because we need to hand-out addresses in fixed chunks of 3 very different sizes.

Secondly, it also has a scalability issue. When there are many small class-C networks, each of them has to have a routing entry in the routing tables. For instance, if an organization or an autonomous system has 16 small networks (say, 4000 addresses), we could be handing out 16 class-C addresses to it or 1 class B address which can be

subnetted. This would mean that we would have 16 class-C entries in the routing table or 1 class-B entry. The number of routing table entries being large leads to latency in forwarding, while allocating a class-B address leads to address inefficiency (16x255/65535 = 6.2%).

Hence, we need a solution to both these problems. That solution is the class-less addressing (which kind of takes its cue from the masking idea used in subnetting) described below.

15.3 Classless Inter-Domain Routing (CIDR)

Continuing with the same example, for an autonomous system with 16 class C networks, what we do is this. Instead of handing out 16 class-c addresses at random, we hand out a block of contiguous class C addresses. When we do so, we can aggregate these addresses to a single address using the idea of masking. For example, suppose we assign the class C network numbers from 192.4.16 through 192.4.31 to the organization. Look at the network part of these addresses. We can note that the top 20 bits of all the addresses in this range are the same (11000000 00000100 0001). That is, we can now refer to this block using the top 20-bits of the address. In other words, ***we have created a 20-bit network number (which is in between class B network number and class C number) - that is our Classless addressing or CIDR addressing.***

Now to specify that the top 20-bits have to be used to identify the network part, we use our idea of masks. We now have a CIDR-mask of 20-bits, specified along with the address as 192.4.16.0/20.

We use this address format of a.b.c.d/x, where x is the number of bits in the network portion of the address. In this notation, a single class C network number, which is 24 bits long, would be written as 192.4.16/24. Similarly, a class B network would be written as x.y/16.

With this addressing scheme, we solve both the problems mentioned in the classful addressing mechanism. This method balances between minimizing the number of routes that a router needs to have against the need to hand out addresses efficiently.

In addition to that it can be used to ***aggregate*** routes, and have a single entry in the forwarding table to tell the router how to reach a lot of different networks, leading to further efficiency in routing. That is, if there are eight x.y.z.0/24 entries with contiguous numbers pointing to the same nextHop, they can be aggregated to a single x.y.u.0/21 entry (because the first 21 bits would be the same). Such an example is shown in Fig. 15.5.

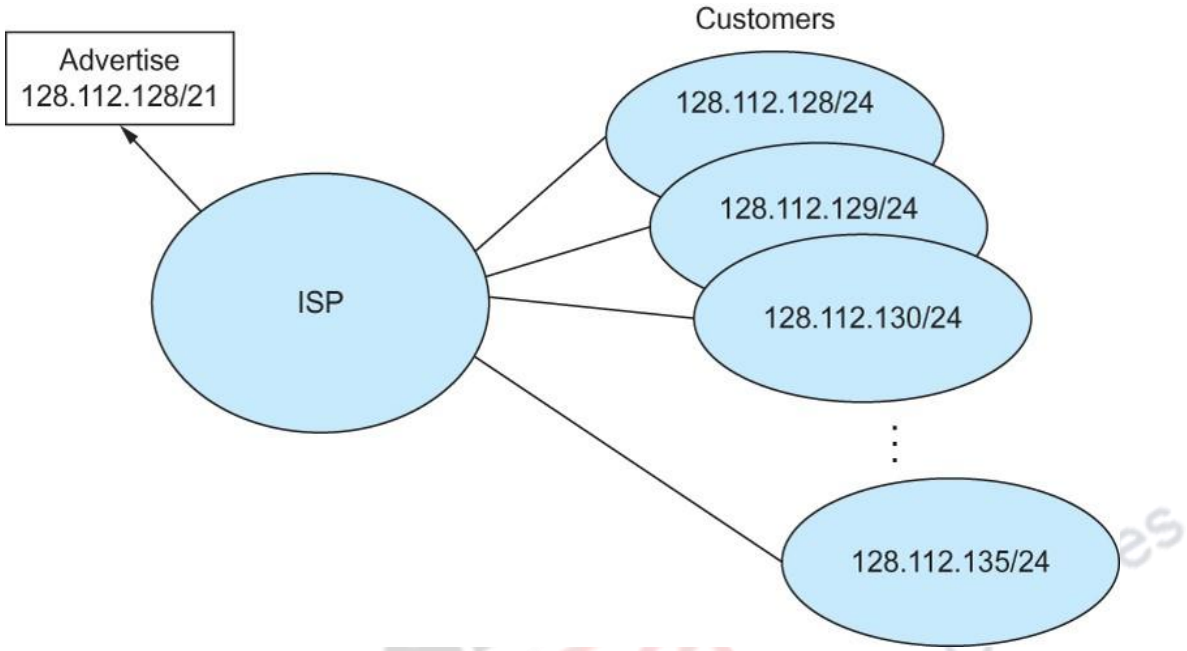


Figure 15.5 CIDR Route Aggregation

An ISP having 8 contiguous class-C addresses assigned to its customers – 128.112.128/24 to 128.112.135/24 can aggregate them to a single block with address 128.112.128/21, and advertise a single entry to its upstream routers. The routers upstream will then have only one entry in their tables for this entire block. Such aggregation can be carried out at each of the routers. Of course, the only requirement for such aggregation is that the blocks of addresses should share a common prefix!

15.3.1 CIDR – Routing

Now, let us look at how the routing protocols handle these classless addresses. They must now understand that the network number may be of any length – 2 to 32 bits. Hence the network numbers are represented as a pair **<value/length>.**

The IP forwarding mechanism in this case, finds a fixed network number in a packet and then looks up that number in the forwarding table. It applies the mask to identify the matching entry. An example of a forwarding table with CIDR addressing is given in Table 15.1.

Consider a packet destined for 128.96.39.10. Where would it go? On applying the masks of length 24, 25 and 26, we can see that it will match the first entry, and will go to interface 0.

128.96.39.10
255.255.255.0

128.96.39.0

Table 15.1 Forwarding table with CIDR addresses

Net/Prefix	Next Hop
128.96.39.0/24	Interface 0
128.96.39.128/25	Interface 1
128.96.40.0/25	Router 2
192.4.153.0/26	Router 3
default (0.0.0.0/0)	Router 4

This is a simple case. Actually, if you notice, it will actually match the last entry (default entry of 0.0.0.0/0 as well. In such cases, where more than one entry is matched we use the principle of “longest prefix match”. That is, we choose that entry which has matched for a longer prefix – a better match because it is more specific !

It is also possible to have non-default prefixes in the forwarding tables that overlap. In such cases also, some addresses may match more than one prefix. For example, both 171.69 (a 16 bit prefix) and 171.69.10 (a 24 bit prefix) could exist in the forwarding table of a single router. A packet destined to 171.69.10.5 clearly matches both prefixes. Again applying out principle of “longest prefix match”, we will choose 171.69.10. On the other hand, a packet destined to 171.69.20.5 would match 171.69 and not 171.69.10.

Thus aggregation and longest-prefix match, combined together give enough flexibility in specifying routes while decreasing the number of entries. This is the major merit of

CIDR.

15.4 Network Address Translation

In the context of IP addressing, another important topic that should be understood is network address translation (NAT). NAT was originally proposed as a partial solution to the address depletion problem. However, it has other advantages in terms of providing security as well. Let us understand NAT from the perspective of address assignment now.

NAT uses the idea of private address spaces and intranets to reduce the IP address requirement. There are a few address blocks that have been identified as private IP address spaces (refer RFC 1597):

10.0.0.0-10.255.255.255

172.16.0.0-172.31.255.255 and

192.168.0.0-192.168.255.255

These addresses are not visible to the internet – they are meant for use in an intranet scenario. What NAT does is that it uses these addresses to identify the nodes inside a network, but exposes this entire network to the outside world as a single IP address. It thus allows hosts with private IP address to have Internet access with a single IP address.

In a sense, it is a short-term solution for the IP address depletion problem – because even a large network is only asking for one IP address – not an entire range of IP addresses!

There are quite a few other benefits that NAT gives. One can change addresses of devices in local network without notifying the outside world. Similarly, one can change the ISP without changing addresses of devices in the local network. Since devices inside the local network are not explicitly addressable or visible to the outside world, it also has a security advantage.

The question in NAT is – how do you map one IP address or a small set of IP addresses to the many devices in the local network and the numerous network applications running on each of them? NAT does this in several ways as explained below.

(i) NAT with pool of IP addresses:

NAT maintains a pool of global IP addresses, and dynamically translates IP addresses

on demand. Or it could statically pre-configure the translations. For example, it could translate addresses as shown below:

10.2.2.2 ==> 140.123.101.30
10.2.2.3 ==> 140.123.101.31

(ii) NAT with Port Address Translation (NAPT) of one global IP address:

This is a commonly used technique, where one global address is used, and the different connections identified by <local IP address, port number> are mapped to <global IP address, different port number> as shown below.

10.2.2.2:1064 ==> 140.123.101.30:5064

10.2.2.3:1175 ==> 140.123.101.30:6175

Since port number field is 16 bits, 64K different connections can be handled. Fig. 15.6 shows an example of this type of NAT.

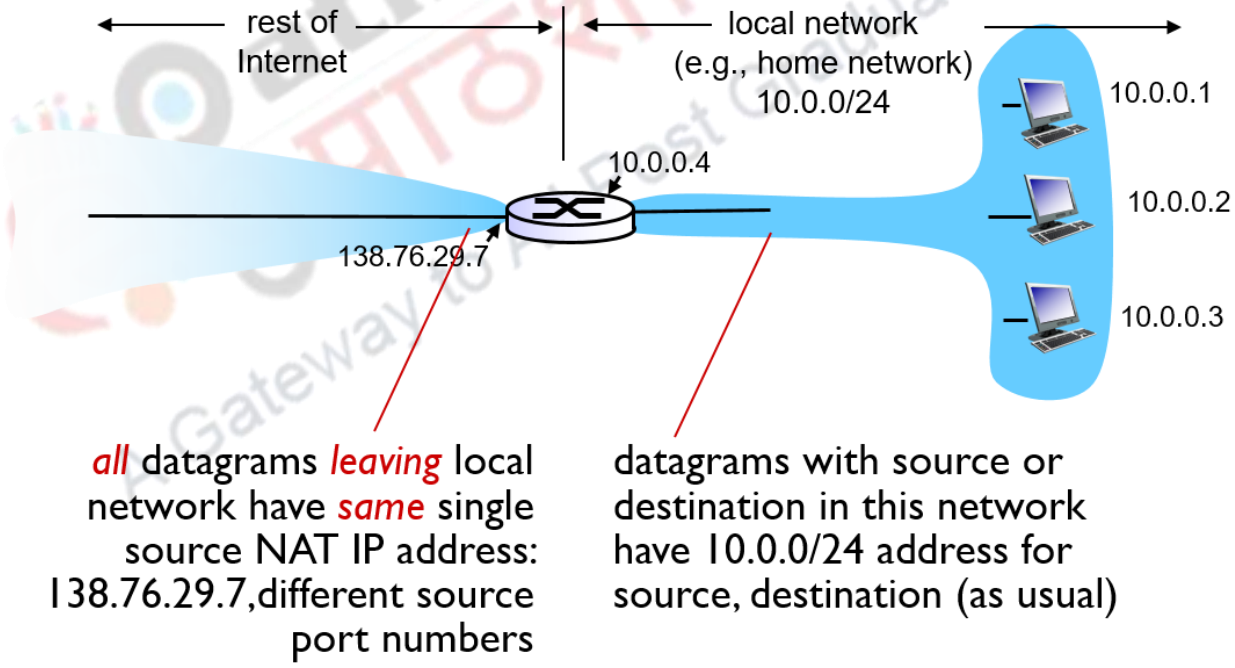


Figure 15.6 NAT with port address translation

The steps to be followed at the NAT router are as follows:

replace (source IP address, port #) of *every outgoing datagram* to (NAT IP address, new port #)

remember (in NAT translation table) every (source IP address, port #) to (NAT IP address, new port #) translation pair

replace (NAT IP address, new port #) in dest fields of *every incoming datagram* with corresponding (source IP address, port #) stored in NAT table.

These steps are shown for the same example network in Fig. 15.7.

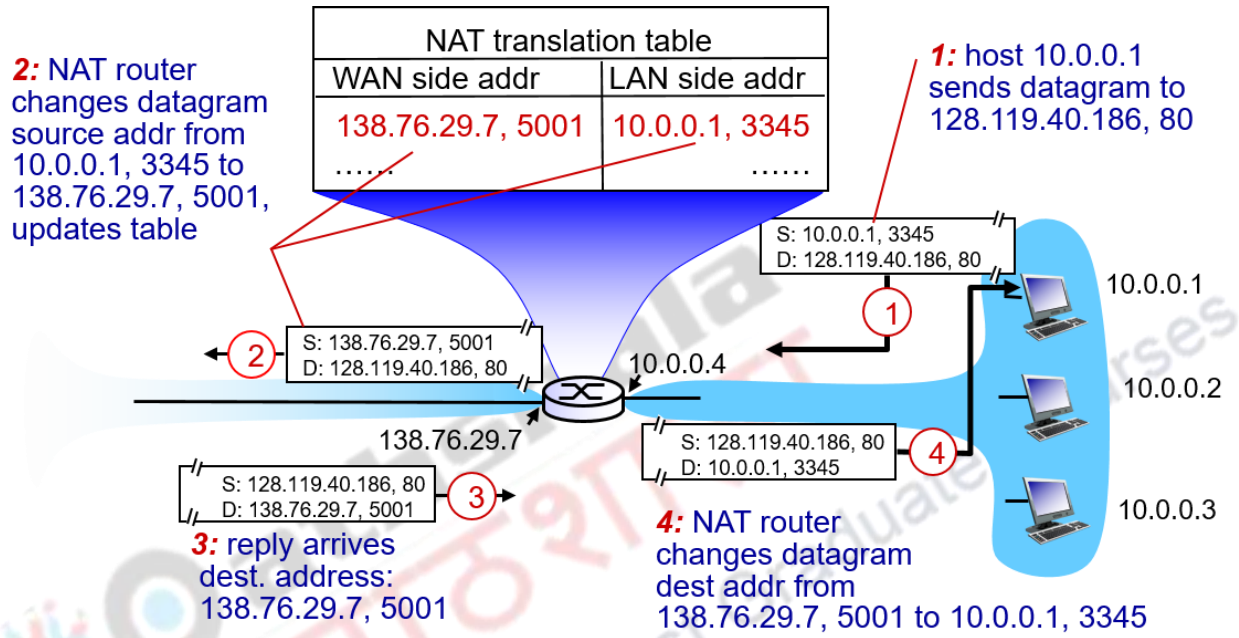


Figure 15.7 NAT port address translation – steps

Note the information stored in the NAT translation table, and the translations done when a packet is leaving the network (steps 1 & 2), and when the response enters the network (steps 3 & 4).

(iii) Port redirection

In this type of NAT, all WWW services are redirected to a specific IP address and private port number. For example DNS may point the service to 140.123.101.38. NAT will redirect this to a private IP address & port number 10.2.2.2:8080.

DNS: www.cs.ccu.edu.tw ==> 140.123.101.38

NAT: 140.123.101.38:80 ==> 10.2.2.2:8080

(iv) Transparent proxy

In this NAT, all www traffic are forced to a proxy with cache.

140.123.101.38:80 ==> internal www proxy (10.1.1.1)

Thus, all HTTP requests go to the internal proxy.

Thus NAT helps to locally solve the IP address requirement problem. However, it is a controversial idea since a NAT router goes beyond layer 3, and tampers with layer 4 information. Remember that routers only process up to layer 3. Thus it interferes with the end-to-end argument, and can be problematic for some applications. Nevertheless, it is one of the common techniques employed by many organizations.

15.5 Summary

To summarize, in this module, we have examined the IP addressing related techniques, challenges and solutions. We have looked at the challenges of class-based addressing. As solutions to that we have looked at subnets, and the use of subnet masks. We have then looked at classless addressing, CIDR and its details. Finally we have had an overview of NAT.

Acknowledgements & References :

1. *Computer Networking: A Top Down Approach Featuring the Internet*, 6th edition. Jim Kurose, Keith Ross Addison-Wesley, 2012.
2. Computer Networks: A systems Approach, 5th edition, David Peterson, Davie, Morgan Kauffman, 2012.
3. Computer Networks An Open Source Approach, Ying-Dar Lin, Ren-Hung Hwang, Fred Baker, McGraw Hill, 2012.