MARCH 1993

NOVELL® RESEARCH

Installing and Configuring NetWare TCP/IP on a NetWare 3.11 Server

Auston Erwin
Technical Information Services
Connectivity Products Division

Radhika Padmanabhan
Product Support Engineer
Connectivity Products Division

Julie Dickinson
Product Support Engineer
Connectivity Products Division

The NetWare 3.11 operating system comes with a set of NLMs (NetWare Loadable Modules) which allow NetWare servers to participate in TCP/IP networks. This Application Note summarizes how to load and configure the NetWare TCP/IP software, including a detailed discussion of IP addresses and subnet masking. It also gives some troubleshooting techniques for testing an IP routing configuration, and outlines the extra steps necessary to route IP packets through IBM source routing bridges.

Disclaimer

Novell, Inc. makes no representations or warranties with respect to the contents or use of these Application Notes (AppNotes) or of any of the third-party products discussed in the AppNotes. Novell reserves the right to revise these AppNotes and to make changes in their content at any time, without obligation to notify any person or entity of such revisions or changes. These AppNotes do not constitute an endorsement of the third-party product or products that were tested. Configuration(s) tested or described may or may not be the only available solution. Any test is not a determination of product quality or correctness, nor does it ensure compliance with any federal, state, or local requirements. Novell does not warranty products except as stated in applicable Novell product warranties or license agreements.

Contents

Introduction

Each copy of the NetWare 3.11 operating system includes software which allows NetWare file servers to participate in networks that use the Transmission Control Protocol/Internet Protocol (TCP/IP) suite of protocols. This software, known as NetWare TCP/IP, is a set of NetWare Loadable Modules (NLMs) that provide IP router and IP tunneling functions, as well as SNMP management capabilities, on a NetWare 3.11 file server. (NetWare TCP/IP does not provide TCP/IP applications such as Telnet, FTP, and so on. These services are provided by products such as Novell's LAN WorkPlace for DOS.)

This Application Note deals mainly with the IP router function, which allows IP packets to be forwarded through a NetWare file server to remote TCP/IP networks. The AppNote first gives brief instructions for installing and configuring the NetWare TCP/IP software specifically for IP routing. It discusses IP addressing and provides guidelines for subdividing a network by using subnet masking. The AppNote then presents a three-step method for troubleshooting an IP router configuration. It also tells how to configure the file server and the workstation to send IP packets through Token-Ring source routing bridges.

This AppNote does not cover configuration of IP tunneling and network management using SNMP. For information about these topics, refer to the NetWare v3.11 TCP/IP Transport Supervisor's Guide (Appendix

A for IP Tunneling and Chapter 6 for SNMP).

The information in this AppNote is intended to clarify and expand on portions of the NetWare 3.11 documentation. Many of the sections contain references to specific pages in various Novell manuals. In particular, you should have the following manuals close by when reading this AppNote:

- NetWare v3.11 System Administration
- NetWare v3.11 TCP/IP Transport Supervisor's Guide

Loading and Configuring NetWare TCP/IP

The installation and configuration of the NetWare TCP/IP software is accomplished when the NetWare operating system is installed. Before you begin, make sure your server meets the following prerequisites for running NetWare TCP/IP:

Computer	80386- or 80486-based PC running NetWare 3.11 or higher
RAM	At least 4MB
LAN Adapter	Ethernet, Token-Ring, or Arcnet adapter with corresponding driver
LAN Driver	Must be certified for NetWare 3.11

(For more information, refer to page 2-1 of the NetWare TCP/IP v3.11 Transport Supervisor's Guide.)

Basic Steps

The basic steps for loading and configuring NetWare TCP/IP are outlined below. More detailed instructions for each step are given in the sections that follow.

1. If necessary, set the maximum physical receive packet size and the maximum number of packet receive buffers for the server. These settings are made via SET commands in the STARTUP.NCF file and the AUTOEXEC.NCF file.
2. Use the LOAD TCPIP command at the server console to load the NetWare TCP/IP software.
3. Use the LOAD LAN_Driver command to load the LAN driver(s).
4. Use BIND commands to bind IPX and IP to the appropriate LAN drivers.

If this is a first-time installation, you may want to test these commands by entering each one individually at the NetWare console prompt before placing them in the AUTOEXEC.NCF file.

Note: To get on-line help with the SET commands, at the NetWare console prompt type "SET <Enter>", select option "1" for Communications, select "ADVANCED", and then select the specific SET option you want information about.

Setting the Maximum Physical Packet Size

To set the maximum size of packet that the NetWare file server can receive, enter the following setting in the STARTUP.NCF file:

SET MAXIMUM PHYSICAL RECEIVE PACKET SIZE = n

- For Ethernet or Arcnet networks, the value for n should be 1514.
- For Token-Ring networks, the value for n should be 4202.

- If the file server is connecting an Ethernet or Arcnet network to a Token-Ring network, the value for n should be the largest packet size--which is 4202.

(For more information on this SET command, refer to page 259 of the NetWare v3.11 System Administration manual.)

Setting the Maximum Number of Receive Buffers

Depending on the TCP/IP applications and their usage, you may need to increase the value of the SET Maximum Packet Receive Buffers parameter. The default value is 100 buffers, which may be insufficient for some configurations. Supported values are 50 through 2000.

To change the maximum number of packet receive buffers for the NetWare file server, enter the following setting in the AUTOEXEC.NCF file:

SET MAXIMUM PACKET RECEIVE BUFFERS = n

where n is a number greater than 100. A suggested number to start out with is 200.

To determine whether the current maximum is sufficient, use MONITOR.NLM to observe the number of Packet Receive Buffers currently allocated. When many TCP/IP applications are concurrently active, the buffering needs are greater.

(For more information on determining what value to set for this parameter, refer to page 260 of the NetWare v3.11 System Administration manual and page 2-3 of the NetWare v3.11 TCP/IP Transport Supervisor's Guide.)

Loading the NetWare TCP/IP Module

The LOAD TCPIP command loads TCPIP.NLM and related NLMs on the file server. The options you use with this command determine whether the TCP/IP module will be recognized as a TCP/IP host on the network or as an IP router between two or more networks (either Ethernet, Token-Ring, or Arcnet). This command is usually placed in the AUTOEXEC.NCF file.

You can specify three options with the LOAD TCPIP command:

LOAD TCPIP [FORWARD = YES|NO] [RIP = YES|NO]
[TRAP = ip_address]

FORWARD=	configures the server as an IP router; default is NO.
RIP=	enables or disables the Routing Information Protocol; default is YES.
TRAP=	specifies the IP address to which the local system should send SNMP trap messages.

To make a NetWare 3.11 file server visible as a TCP/IP host on the network, enter the command as follows:

LOAD TCPIP

If the NetWare 3.11 file server is to route IP packets, the command should be:

LOAD TCPIP FORWARD=YES

(For more information, see pages 3-2 through 3-4 in the NetWare v3.11 TCP/IP Transport Supervisor's Guide.)

Loading LAN Drivers

To load a LAN driver, use the LOAD LANDriver command. The general syntax for this command is:

LOAD LANDriverName INT=n PORT=xxx FRAME=FrameType
NAME=Name

Here are some examples for various types of LAN adapters.

- For an NE3200 Ethernet LAN driver:

  LOAD NE3200 INT=3 PORT=300 FRAME=ETHERNET_II

  NAME=ip_enet

- For an IBM Token-Ring LAN driver:

  LOAD TOKEN FRAME=TOKEN-RING_SNAP NAME=ip_token

- For an SMC-Plus Arcnet LAN driver:

  LOAD SMCPLUSSV INT=5 PORT=320 FRAME=NOVELL_RX-NET

  NAME=ip_arc

The NAME specified at the end of each command is a unique name that you assign when loading the LAN driver re-entrantly with more than one frame type. For example, if you are loading both IPX and IP for the same LAN adapter, the LOAD statements might appear as follows:

LOAD NE3200 INT=3 PORT=300 FRAME=ETHERNET_802.3
NAME=IPX_LAN

LOAD NE3200 INT=3 PORT=300 FRAME=ETHERNET_II
NAME=IP_LAN

(For more information, refer to Chapter 3 of the NetWare v3.11 TCP/IP Transport Supervisor's Guide or to pages 122 through 154 in the NetWare v3.11 System Administration manual.)

Binding Protocols to the LAN Drivers

After loading the LAN driver(s), you need to use the BIND command to bind the protocols to the appropriate LAN driver, referenced by logical board name. For the IPX and IP protocols, the syntax for the BIND commands is as follows:

BIND IPX TO name NET=#
BIND IP TO name ADDR=#.#.#.# [MASK= subnet mask]

name	the identifying string assigned in the LOAD statement as the NAME parameter
NET=	specifies the IPX network number
ADDR=	specifies the IP address to be assigned to this network adapter
MASK=	used if a subnet is specified on the network (optional)
#	indicates a numerical value which can be either decimal or hexadecimal (for example, 255.255.255.0 or FF.FF.FF.0)

Here is an example of the BIND commands for IPX and IP:

```
BIND IPX to IPX_LAN NET=2
        BIND IP to IP_LAN ADDR=130.2.1.254 MASK=FF.FF.FF.0
```

(The BIND command has several other options, which are explained on pages 3-4 through 3-8 in the NetWare v3.11 TCP/IP Transport Supervisor's Guide. For more information about IP addressing, refer to Appendix B of the NetWare v3.11 TCP/IP Transport Supervisor's Guide.)

Example AUTOEXEC.NCF File

The following sample AUTOEXEC.NCF file contains the entries necessary for a NetWare 3.11 server to provide routing of IPX and IP packets between Ethernet, Token-Ring, and Arcnet networks:

```
SET MAXIMUM PACKET RECEIVE BUFFERS = 200
        LOAD TCPIP FORWARD=YES
        LOAD NE3200 int=3 port=300 frame=ETHERNET_802.3
          name=IPX_LAN
        LOAD NE3200 int=3 port=300 frame=ETHERNET_II name=IP_LAN
        LOAD TOKEN frame=TOKEN-RING name=IPX_TOKEN
        LOAD TOKEN frame=TOKEN-RING_SNAP name=IP_TOKEN
        LOAD SMCPLUSSV int=5 port=320 frame=NOVELL_RX-NET
          name=ARCNET
        BIND IPX to IPX_LAN  net=1
        BIND IP to IP_LAN addr=130.2.1.254  mask=FF.FF.FF.0
        BIND IPX to IPX_TOKEN net=2
        BIND IP to IP_TOKEN addr=130.2.2.254  mask=FF.FF.FF.0
        BIND IPX to ARCNET net=3
        BIND IP to ARCNET addr=130.2.3.254 mask=255.255.255.0
```

Note that there is only one LOAD statement for Arcnet. For any topology (Ethernet, Token Ring, or Arcnet) when the same frame type is used for both IPX and IP, only one LOAD statement is required. Since Arcnet uses the same frame type (NOVELL_RX-NET) for IPX and IP, you only need one LOAD statement.

Note: If all of the servers on your internetwork are NetWare 3.11 servers, Ethernet adapters can have both IPX and IP bound to the LAN driver with Ethernet_II frame types for easier configuration (one LOAD statement with the frame type ETHERNET_II and two BIND statementsþone for IP and the other for IPX). If other NetWare servers on the internetwork are using the Ethernet_802.3 frame type, you will need to load both frame types (two LOAD statementsþETHERNET_802.3 for IPX and ETHERNET_II for IP).

(For additional sample IP network configurations, refer to Chapter 5 of the NetWare v3.11 TCP/IP Transport Supervisor's Guide.)

Configuring NetWare TCP/IP as an IP Router

This section discusses some important points to consider when configuring the NetWare TCP/IP software as an IP router. It summarizes some of the key concepts to understand about IP addressing. A good reference for further study is Internetworking with TCP/IP, Volume I by Douglas E. Comer, published by Prentice Hall.

IP Addresses

IP addresses are 4-byte (32-bit) numbers that identify both a network and a local host or node on that network. They are usually represented in dotted decimal notation, where each byte is a decimal number and dots separate the bytes (for example, 129.47.6.17). IP addresses can also be represented with hexadecimal numbers.

An IP address consists of two parts: the network address (which identifies the network) and the host

address (which identifies the node). IP addresses are differentiated into three classes based on the most significant bits of the first byte (see Figure 1). This is done so that routers can efficiently extract the network portion of the address.

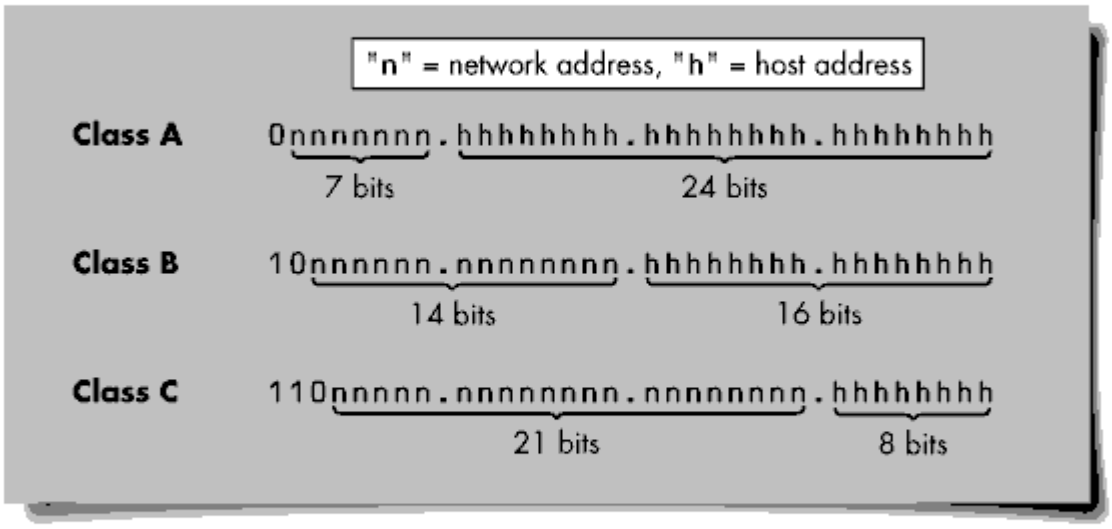


Figure 1: IP addresses are divided into three classes.

The first byte of an IP address fits in the ranges listed below and tells you which of the three network classes the address belongs to.

Class A:	1 to 127	(1.h.h.h to 127.h.h.h)
Class B:	128 to 191	(128.n.h.h to 191.n.h.h)
Class C:	192 to 223	(192.n.n.h to 223.n.n.h)

An IP address beginning with 154 would be a Class B address, with the first two bytes of the address representing the network portion of the address and the last two representing the host portion. For example, in the IP address 154.1.0.3, the IP network address is 154.1.0.0 and the host address on that network is #.#.0.3.

The network portion of an IP address should be the same for all nodes on that network. So, for example, the LAN adapter in a file server that connects to IP network 89.0.0.0 needs to have a unique IP host address assigned to it, such as 89.0.0.254.

The key to selecting a number for the host portion of the IP address is to make sure that the number selected is unique--in other words, no other host on the network has the same IP address.

(For more information on IP addressing, refer to pages B-7 through B-9 of the NetWare v3.11 TCP/IP Transport Supervisor's Guide.)

IP Routing Between Different Network Segments

If packets must travel across different physical networks, routing becomes necessary. When the NetWare server is used as an IP router between two or more networks, each network must have a unique IP network address. The different LAN adapters in the NetWare 3.11 server should have different IP network addresses corresponding to the networks they are connected to (as defined in the BIND IP . . . statement in the AUTOEXEC.NCF file).

For example, if one LAN adapter in the file server is connected to network 89.0.0.0, the second adapter in the file server needs to be connected to a different network, such as 192.1.1.0. The adapter connected to IP network 89.0.0.0 needs an IP address of 89.#.#.# in order to be recognized on this network. The second adapter that is connected to network 192.1.1.0 will need an address of 192.1.1.#.

Subnet Masking

On pages B-18 and B-19 of the NetWare v3.11 TCP/IP Transport Supervisor's Guide, the following reasons are given for subdividing a network:

- To use multiple network media (cabling segments)
- To reduce traffic between nodes
- To reduce CPU usage from broadcast packets being received
- To isolate a network segment
- To improve security

If your organization has been assigned a single IP network address, you must use subnets to accomplish any of these goals. That is, you must partition the host address space by assigning subnet numbers to the LANs. If you can acquire a new, unique network number for each LAN, subnets are not necessary.

With subnetting, 4-byte IP addresses can be interpreted as:

<network> <subnet> <host>

The network part is defined by the IP network address, while the subnet and host parts are defined by the subnet mask.

The subnet mask indicates how the host portion of the IP address will be divided into the subnet and host portions. It is a 32-bit number with all ones for the net and subnet portions of the complete IP address, and all zeros for the host portion (see Figure 2).

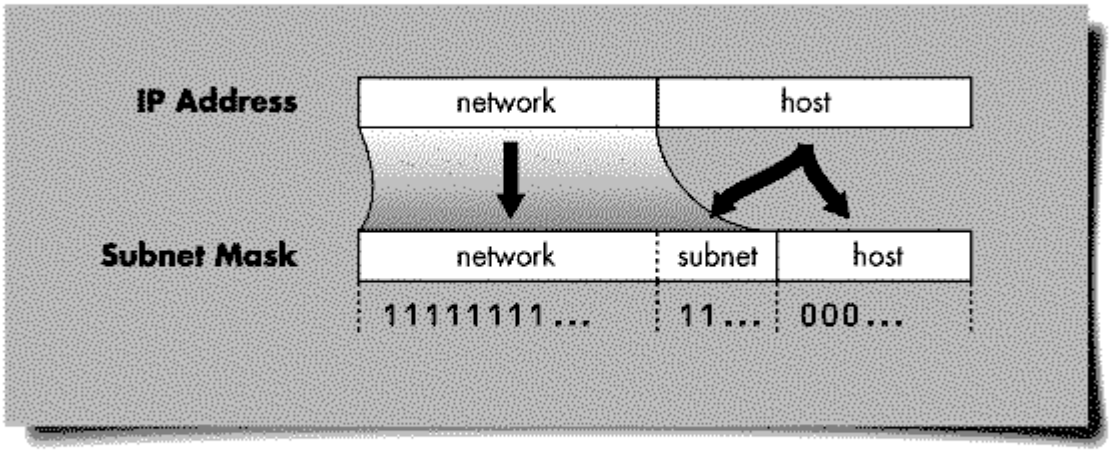


Figure 2: The subnet mask indicates how the host portion of an IP address is divided into the subnet and host portions.

For example, suppose a company has been assigned the Class B IP network address of 154.4.0.0, and wants to implement subnet masking. Since Class B addresses use the first two bytes for the network address, the company can subdivide the address space by using a mask that maps off all or part of the third byte.

With a subnet mask of 255.255.255.0 (FF.FF.FF.0), the first three bytes of the IP address would be recognized as the network portion of the address, because the first three bytes in the mask are all ones. Figure 3 gives the binary and decimal representation of the third byte of subnet mask 255.255.255.0.

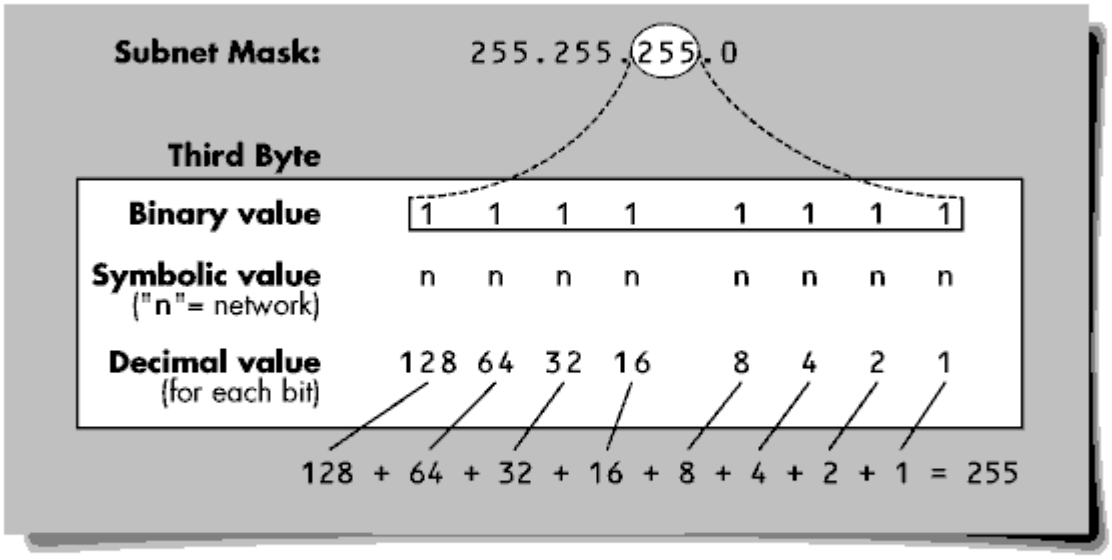


Figure 3: Binary and decimal representation of the third byte of subnet mask 255.255.255.0.

With this subnet mask, the network portion of the address is 154.4.1.0 instead of 154.4.0.0. A total of 254 subnets can be assigned: from 154.4.1.0 to 154.4.254.0.

(For more information regarding methods of partitioning your network and creating subnets, refer to pages B-19 through B-22 in the NetWare v3.11 TCP/IP Transport Supervisor's Guide.)

Invalid Subnet Masks. A subnet mask of all zeros is not recommended by RFC 950 and is not supported by Novell. Having such a mask defeats the purpose of creating subnets in the first place. To illustrate, suppose a company is using the Class B network address of 154.4.0.0 with a mask of 255.255.255.0 (FF.FF.FF.0). If they then assign 0 as the value for the third byte, it still appears as the original address (154.4.0.0), as if the subnet mask was not being implemented.

RFC 950 also recommends that the values of all ones in the subnet field should not be assigned in the subnet portion of an IP address. Addresses in which the network portion is set to all ones are reserved.

Default Subnet Masks. If no subnet mask is assigned, the subnet mask is the default IP network mask assigned to each class of IP network:

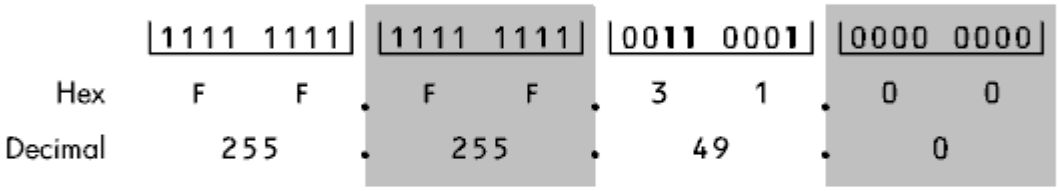
Class	Default Subnet Mask
Class A (1 to 127)	255.0.0.0 (FF.0.0.0)
Class B (128 to 191)	255.255.0.0 (FF.FF.0.0)

Class C (192 to 223) 255.255.255.0 (FF.FF.FF.0)

Deciding on a Subnet Mask. The simplest method of subnet masking is to mask off an entire byte, as described in the example above (Figure 2). However, you can mask off only a portion of a byte as the network address. To do this, you need to recognize which bits in the byte represent the network and the host portions, then calculate the appropriate subnet mask.

In assigning the subnet mask, it is best to make the subnet bits contiguous and located as the most significant bits of the IP address. It is not illegal to have non-contiguous bits, but to avoid confusion it is recommended that they be contiguous.

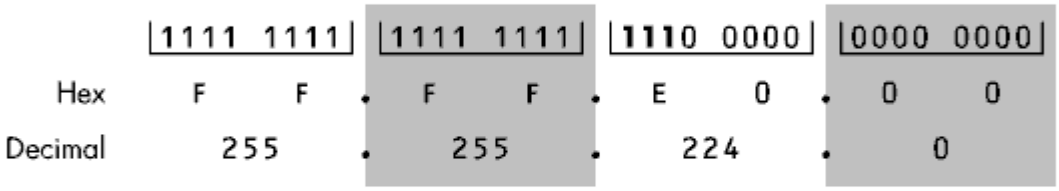
For example, the binary representation of a subnet mask of 255.255.49.0 (FF.FF.31.0) is:



Figure

The bits of this subnet mask are non-contiguous, and they are not located as the most significant bits of the IP address. As you can see, it becomes more difficult to determine which portion of the IP address is the subnet (network) portion and which bits represent the host address.

A more desirable approach is to use a subnet mask of 255.255.224.0 (FF.FF.E0.0), with the binary representation:



Figure

By following these guidelines, you can create a number of different subnet masks. The number of valid subnets you'll end up with depends on the number of bits you use in the mask (see Figure 4).

Figure 4: Chart of possible subnet masks for Class B and Class C IP addresses.

Number of Bits	Subnet Mask	Number of Subnets Recommended	Number of Hosts (Nodes) per Subnet
CLASS B			
2	255.255.192.0	2	16,382
3	255.255.224.0	6	8,190
4	255.255.240.0	14	4,094
5	255.255.248.0	30	2,046
6	255.255.252.0	62	1,022
7	255.255.254.0	126	510
8	255.255.255.0	254	254
9	255.255.255.128	510	126

10	255.255.255.192	1,022	62
11	255.255.255.224	2,046	30
12	255.255.255.240	4,094	14
13	255.255.255.248	8,190	6
14	255.255.255.252	16,382	2

CLASS C

2	255.255.255.192	2	62
3	255.255.255.224	6	30
4	255.255.255.240	14	14
5	255.255.255.248	30	6
6	255.255.255.252	62	2

(For more information on subnet masking, read pages B-18 through B-22 in the Novell TCP/IP Transport Supervisor's Guide and RFC 950.)

Configuring NetWare TCP/IP with Subnets

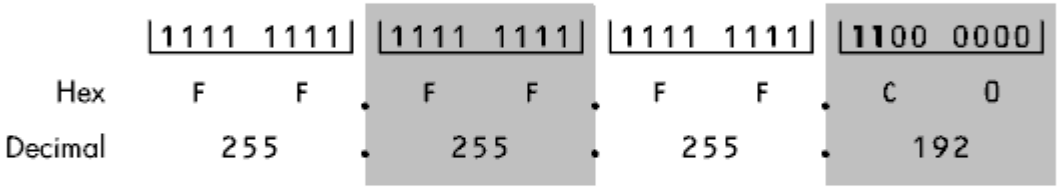
If it is determined that subnet masking is needed on a network, all hosts on that network need to be configured with that mask. On the NetWare TCP/IP file server, the mask is configured with the BIND command:

BIND IP TO LAN_Driver ADDR=#.#.#.# MASK= #.#.#.#

Note: The subnet mask can be assigned using either hexadecimal values of 0 through F, or decimal values of 0 through 255. For example, the mask values of FF.FF.0.0 and 255.255.0.0 are the same, and either can be loaded in the AUTOEXEC.NCF file.

Following are some examples that illustrate the use of various subnet mask values for an IP address. The examples use the Class C IP address of 192.1.1.0, which has a default mask of 255.255.255.0 (FF.FF.FF.0).

Example 1. To allow subnetting, you could decide to mask off the first two bits (bits 8 and 7) of the fourth byte. The corresponding IP subnet mask would have the following binary representation:



Figure

This subnet mask translates to FF.FF.FF.C0 hexadecimal, or 255.255.255.192 decimal.

By using all possible binary combinations for the first two bits of the fourth byte, you get the following network addresses:

Binary Values (4th byte)		Decimal Address (full)
bits 8 and 7 off	**00**00 0000	192.1.1.**0**
bit 7 turned on	**01**00 0000	192.1.1.**64**
bit 8 turned on	**10**00 0000	192.1.1.**128**
bits 7 and 8 on	**11**00 0000	192.1.1.**192**

Since you can't have a subnet of all zeros (192.1.1.0) or a subnet of all ones (192.1.1.192), the subnet mask 255.255.255.192 (FF.FF.FF.C0) yields two valid subnet networks: 192.1.1.64 and 192.1.1.128. So using this subnet mask with the single IP network address 192.1.1.0 provides two functional IP networks.

Example 1a. Now let's see how many host (node) addresses you can have on each of these networks, starting with 192.1.1.64.

Figure 5 gives the binary and decimal representation of the fourth byte with the subnet mask of 255.255.255.192 (FF.FF.FF.C0).

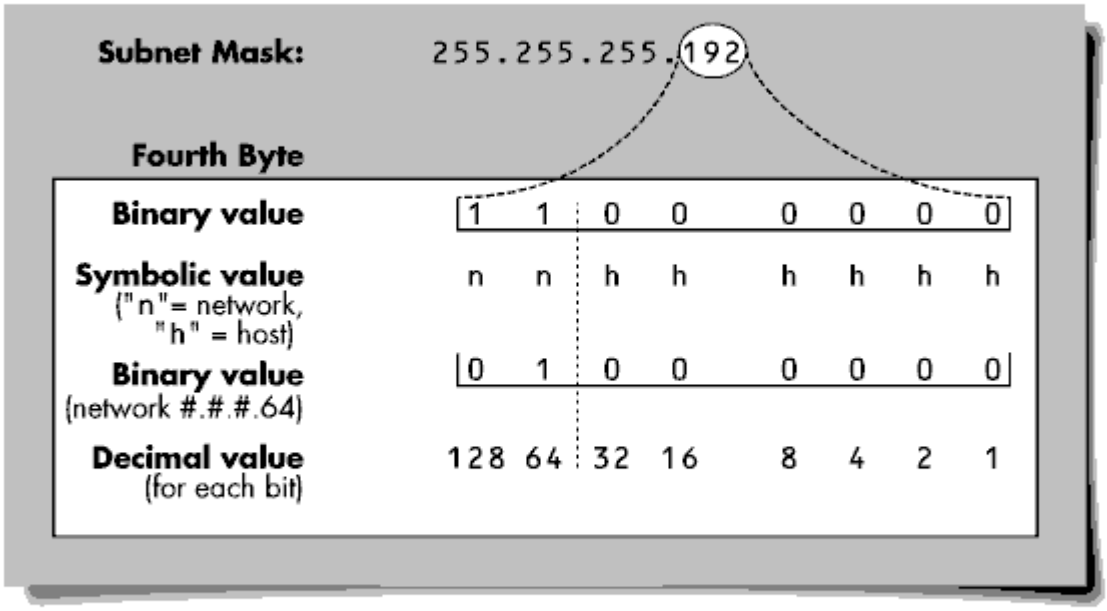


Figure 5: Binary, symbolic, and decimal representation of the fourth byte of IP address 192.1.1.64.

The fourth byte determines both the subnet network number and the host portion of the full IP address. In this example, the two highest bits (bits 8 and 7) indicate the subnet portion, while the other six bits are used for the host portion.

The value of the fourth byte is made up of the combination of the subnet network number (#.#.#.64) plus any combination of the last six bits (32, 16, 8, 4, 2, 1) that are the host address portion of the fourth byte. For example, a value of #.#.#.116 breaks down into the subnet network number of 64 and a host number of 52 116 - 64 = 52), as shown in Figure 6.

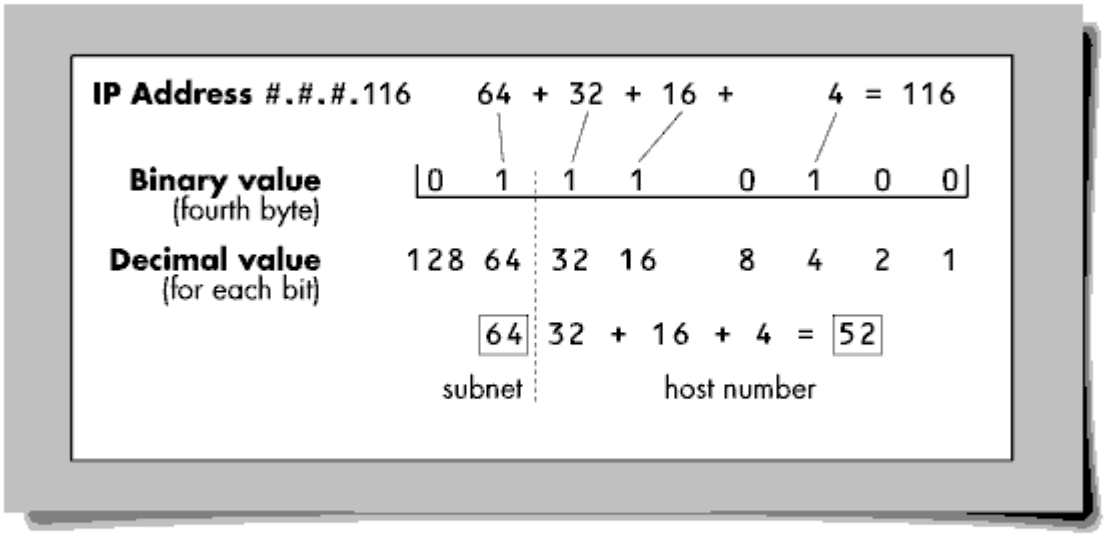


Figure 6: The value of the fourth byte determines the subnet and host number.

Figure 7 shows the binary, symbolic, and decimal representation of the fourth byte of the subnet mask.

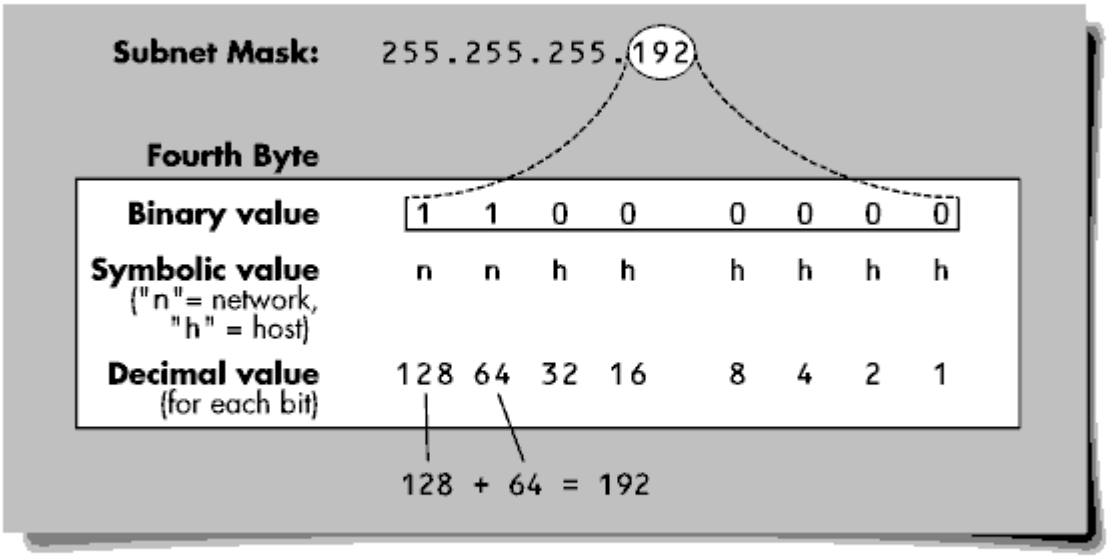


Figure 7: Binary, symbolic, and decimal representation of the fourth byte of subnet mask 255.255.255.192.

So for network address 192.1.1.64, with the subnet mask of 255.255.255.192, the fourth byte can be assigned any decimal value between 65 and 127 inclusive. As shown in Figure 7, this provides 62 available host addresses, from 191.1.1.65 and 191.1.1.127 inclusive.

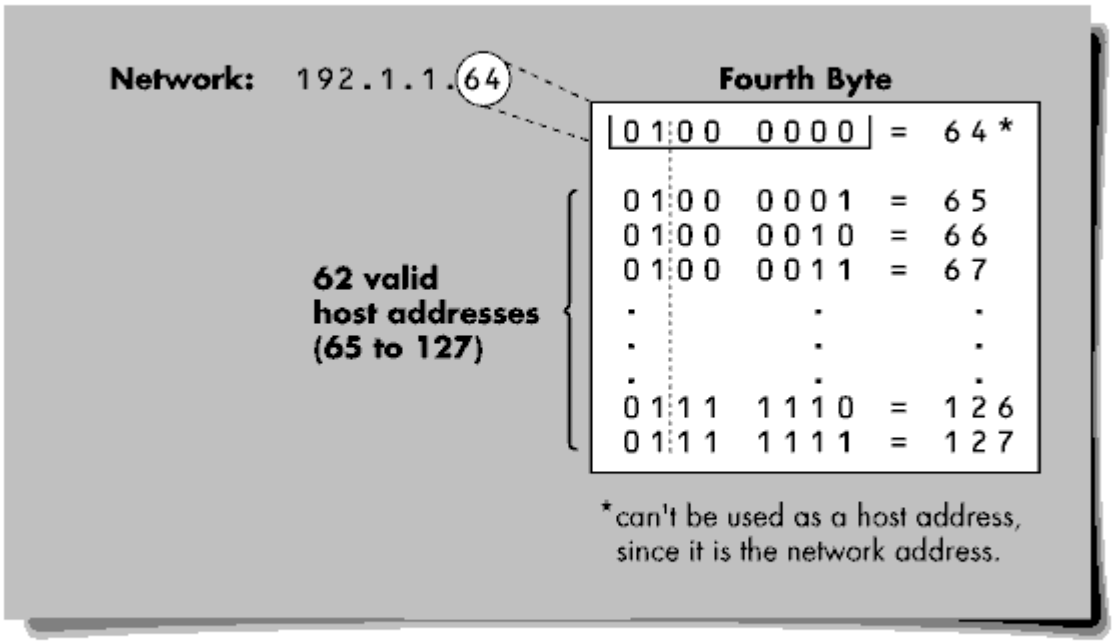


Figure 8: With subnet mask 255.255.255.192, IP address 192.1.1.64 provides 62 available host addresses.

Note: The value of 64 cannot be used in the fourth byte. Because of the subnet mask 255.255.255.192 (FF.FF.FF.C0), the bits 7 and 8 of the fourth byte (with the values of 64 and 128 respectively) are recognized as the network portion of the address. Since this example is using 192.1.1.64 as the network address, the value of 64 cannot also be used as a host address.

Example 1b. Now let's look at network 192.1.1.128. Figure 9 gives the binary and decimal representation of the fourth byte of IP address 192.1.1.128 with a subnet mask of 255.255.255.192 (FF.FF.FF.C0).

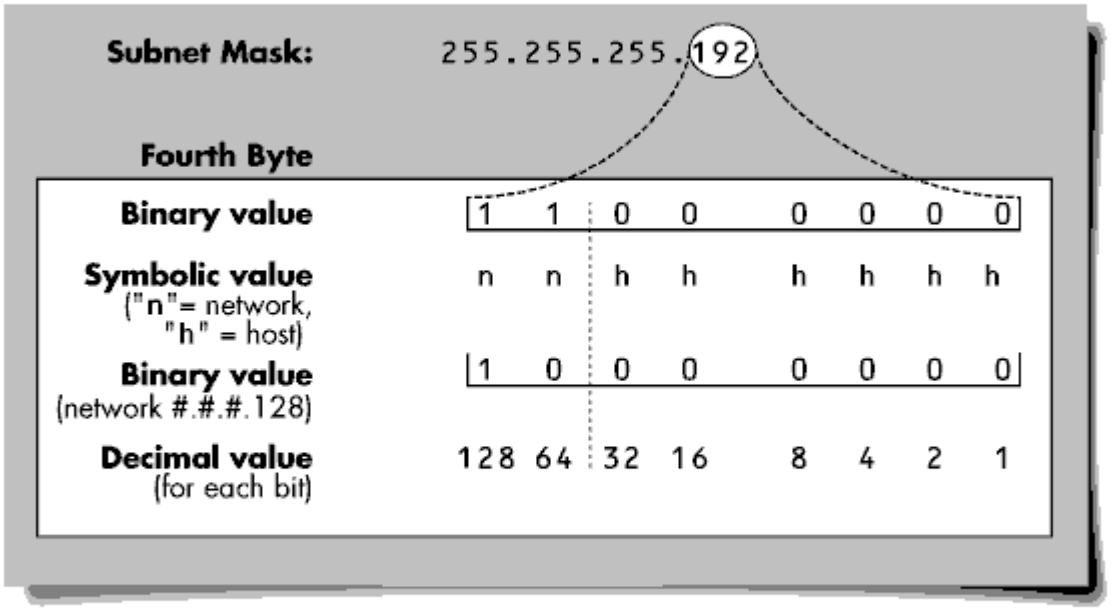


Figure 9: Binary and decimal representation of the fourth byte with the subnet mask of 255.255.255.192 (FF.FF.FF.C0).

Again, the fourth byte determines both the subnet network number and the host portion of the full IP address. In this example, the two highest bits (bits 8 and 7) indicate the subnet portion, while the other six bits are used for the host portion.

The value of the fourth byte is made up of the combination of the subnet network number (#.#.#.128) plus any combination of the last six bits (32, 16, 8, 4, 2, 1) that are the host address portion of the fourth byte. For example, a value of #.#.#.139 breaks down to 128 as the subnet network number, leaving a host number of 11 (139 - 128 = 11), as shown in Figure 10.

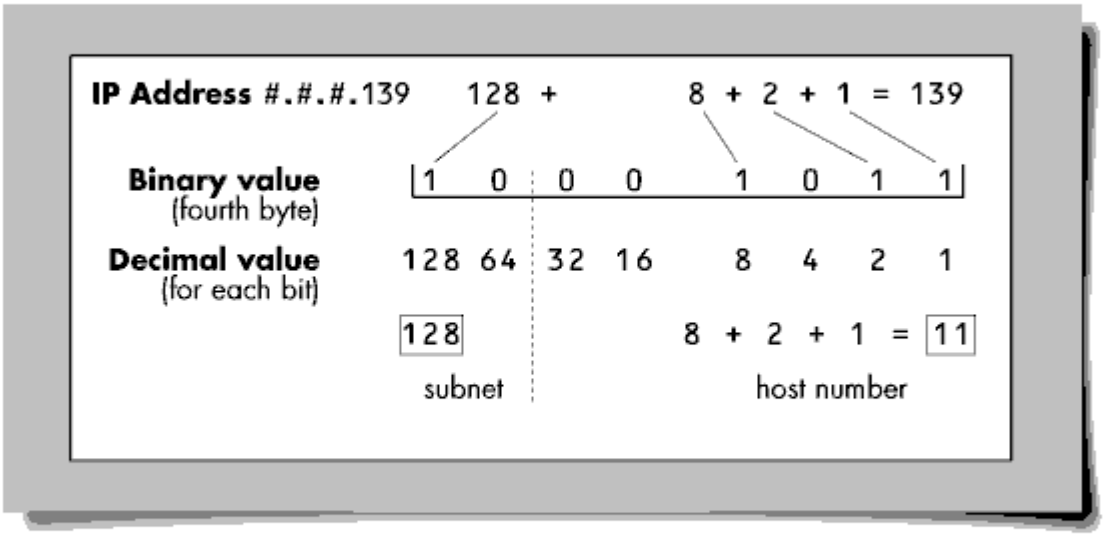


Figure 10: The value of the fourth byte determines the subnet and host number.

So for network address 192.1.1.128, with the subnet mask of 255.255.255.192, the fourth byte can be assigned any decimal value between 129 and 191 inclusive. This provides 62 available host addresses from 192.1.1.129 and 192.1.1.191 inclusive.

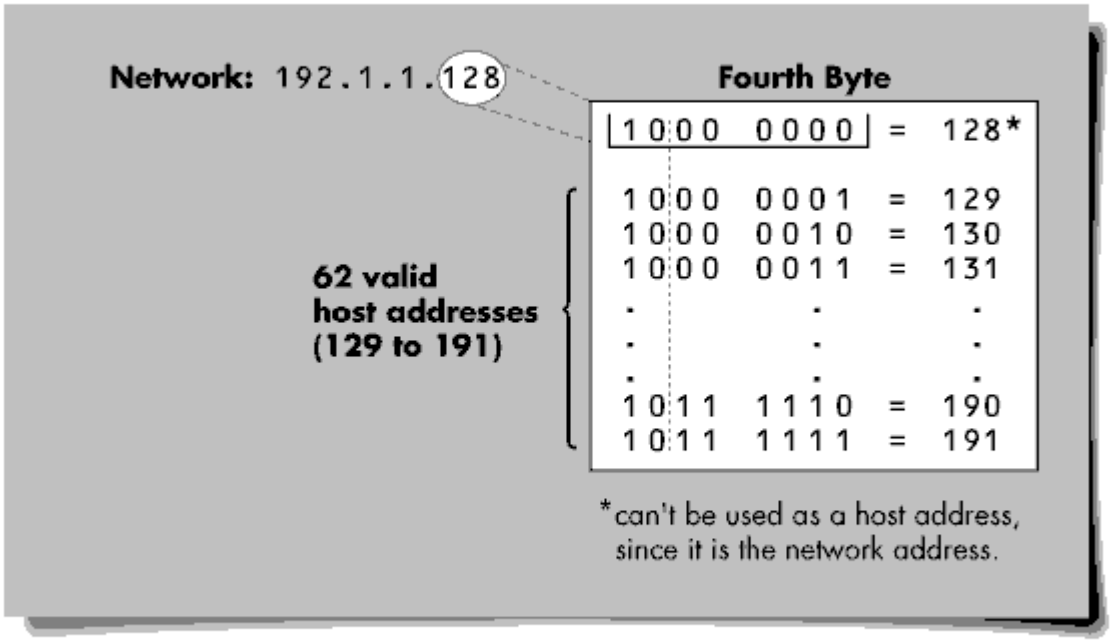


Figure 11: With subnet mask 255.255.255.192, Ip address 192.1.1.128 provides 62 available host addresses.

Note: The value of 128 cannot be used in the fourth byte. Because of the subnet mask 255.255.255.192 (FF.FF.FF.C0), bits 7 and 8 of the fourth byte (with the values of 64 and 128 respectively) are recognized as the network portion of the address. Since this example is using 192.1.1.128 as the network address, the value of 128 cannot also be used as a host address.

Example 1c. In the AUTOEXEC.SYS file for a NetWare file server acting as an IP router, the BIND statements for the two LAN adapters could be:

BIND IP to IPNET1 ADDR=192.1.1.116 MASK=255.255.255.192
BIND IP to IPNET2 ADDR=192.1.1.139 MASK=255.255.255.192

Note: As shown above, the IP network addresses of 192.1.1.128 and 192.1.1.64 (with the subnet mask of 255.255.255.192) are network addresses and should not be assigned to TCP/IP hosts. For example, the following BIND commands would be illegal:

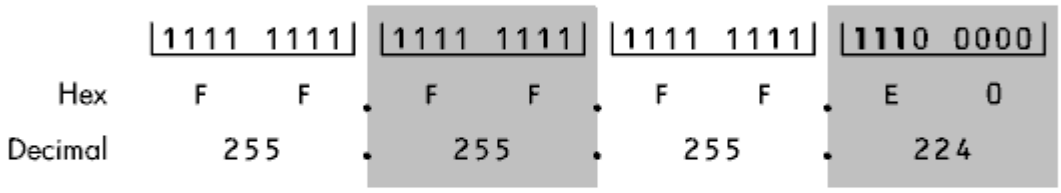
BIND IP to IPNET1 ADDR=192.1.1.128

MASK=255.255.255.192 (illegal address)

BIND IP to IPNET2 ADDR=192.1.1.64

MASK=255.255.255.192 (illegal address)

Example 2. This example uses the same Class C IP address of 192.1.1.0, which has a default mask of 255.255.255.0 (FF.FF.FF.0). To allow subnetting, you decide to mask off the first three bits of the fourth byte. The corresponding IP subnet mask would have the following binary representation:



Figure

This subnet mask translates to FF.FF.FF.E0 hexadecimal, or 255.255.255.224 decimal.

By using all possible binary combinations for the first three bits of the fourth byte, you get the following network addresses:

Binary Values (4th byte)		Decimal Address (full)
bits 6, 7, 8 off	**000**0 0000	192.1.1.**0**
bit 6 turned on	**001**0 0000	192.1.1.**32**
bit 7 turned on	**010**0 0000	192.1.1.**64**
bits 6 and 7 on	**011**0 0000	192.1.1.**96**
bit 8 turned on	**100**0 0000	192.1.1.**128**
bits 6 and 8 on	**101**0 0000	192.1.1.**160**
bits 7 and 8 on	**110**0 0000	192.1.1.**192**
bits 6, 7, 8 on	**111**0 0000	192.1.1.**224**

Again, you can't have a subnet of all zeros (192.1.1.0) or a subnet of all ones (192.1.1.224). So when

masking off the first three bits in the last byte, the recommended six network addresses are:

```
   192.1.1.32              192.1.1.128
192.1.1.64              192.1.1.160
192.1.1.96              192.1.1.192
```

Figure 12 gives the binary and decimal representation of the fourth byte of network address 192.1.1.128 with subnet mask 255.255.255.192 (FF.FF.FF.E0).

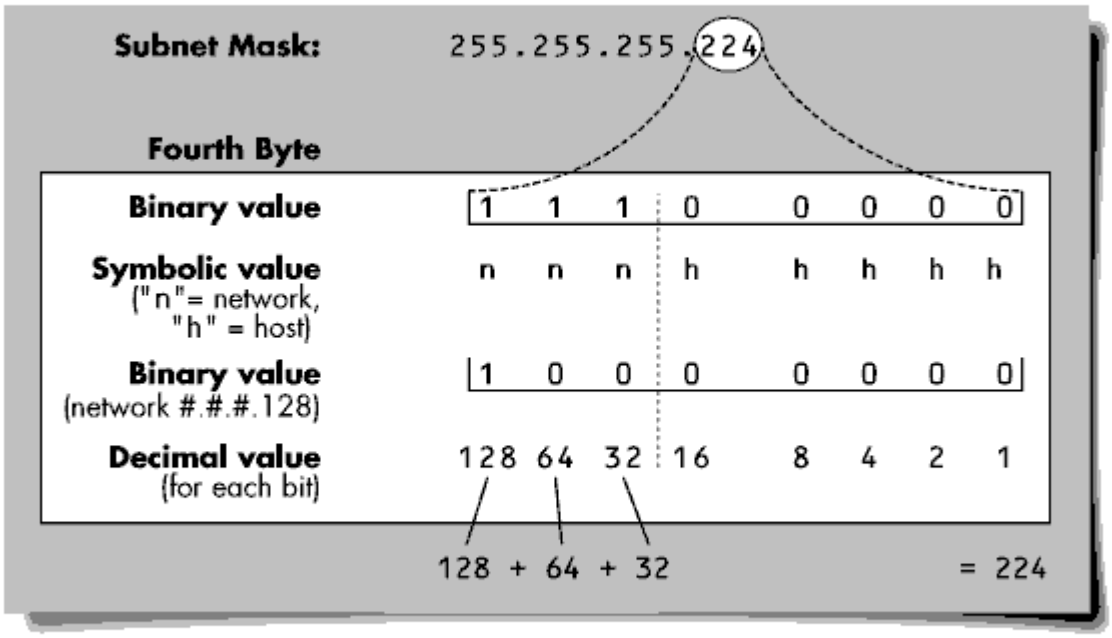


Figure 12: Binary, symbolic, and decimal values of the fourth byte of address 192.1.1.128 with subnet mask 255.255.255.192.

By following the same procedure as in Example 1, you will find that with a subnet mask of 255.255.255.192 (FF.FF.FF.E0), each of the six subnet network addresses provides 30 available host addresses.

Troubleshooting

Once the IP addressing has been configured, you are ready to test the configuration to see if IP packets can be routed between the two networks.

For purposes of illustration, assume that the network configuration we are going to test consists a DOS workstation running LAN WorkPlace for DOS (LWPD) that is attempting to connect to a UNIX host on another physical network. A NetWare 3.11 file server is routing IP packets between the two networks. This sample network is illustrated in Figure 13.

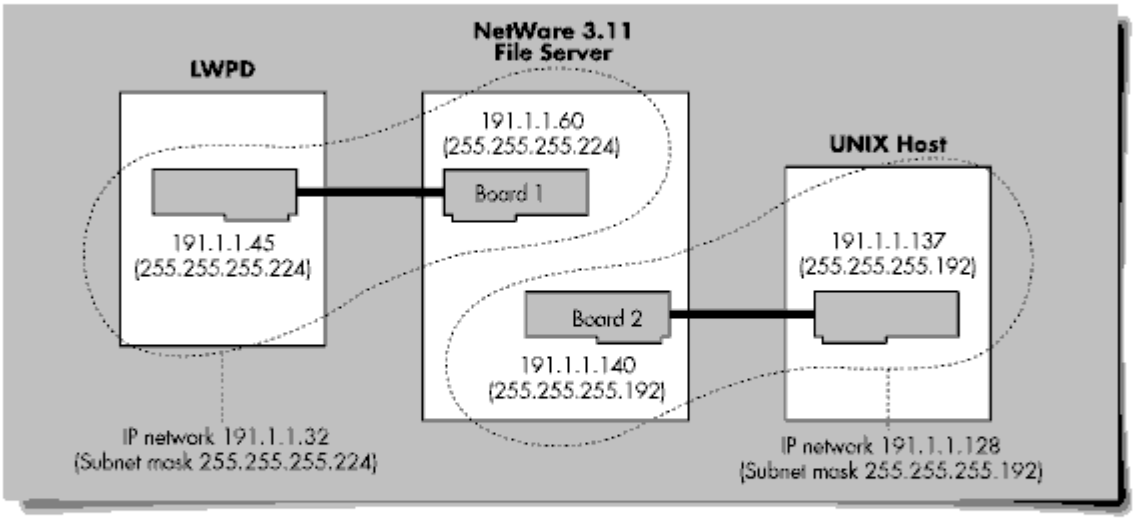


Figure 13: Sample network for troubleshooting IP routing configuration.

The easiest way to test the routing configuration is with the UNIX ping command. You can initiate testing from either the LWPD workstation or from the UNIX system. In this example, we will start testing from the LWPD workstation.

Step One: Test LWPD-to-Local Network Connection

To test the IP connection between the workstation and the file server, enter the following command from the LWPD workstation:

ping 191.1.1.60

If the response that is returned is 191.1.1.60 is alive, the IP packet was successfully sent to the remote address and the remote host (the file server) was able to return the packet successfully.

If you do not receive the confirmation that the IP packet was returned correctly, there is a problem.

At this point, the board in the NetWare file server should appear as another host on the local network, so there should not be any routing issues. Here are some other things to check.

- The network portion of the IP addresses on both the workstation and the file server should be the same.

- The workstation and the file server should use the same subnet mask (which could be the default mask).
- The physical connection between the two hosts should exist and should be intact.

Step Two: Test LWPD-to-Remote Network Connection

The next step in the testing process is to ping from the LWPD workstation, through the file server, to the second board that is connected to the remote network. The command is:

ping 191.1.1.140

If the test is not successful, do the following.

1. Verify that the AUTOEXEC.NCF file on the server contains the command LOAD TCPIP FORWARD=YES so IP packets can be routed through the file server.
2. Verify that the LWPD workstation has the following entries in its NET.CFG file:

   IP_ROUTER 191.1.1.60

   IP_NETMASK 255.255.255.224

   These entries indicate that the host at 191.1.1.60 (the NetWare file server) is the router to route packets from network 191.1.1.32 to network 191.1.1.128.

   (For an example of other TCP/IP entries that need to be in the NET.CFG file, refer to page 2-10 of the LAN WorkPlace for DOS Administrator's Guide.)
3. Verify the TCP/IP configuration on the file server by typing the CONFIG command at the console prompt. Check to see if the IP address and the subnet mask are being recognized as the values you configured in the AUTOEXEC.NCF file.
4. If you are still not able to resolve the problem, you may want to obtain technical support. See the section on Obtaining Technical Support at the end of this AppNote.

Once you are successfully able to ping the second board in the file server, you know that the routing on the LWPD workstation (the local TCP/IP host) is set up correctly and the NetWare file server is routing packets correctly.

Step Three: Test LWPD-to-UNIX Host Connection

The final step in testing the configuration is to ping from the LWPD workstation to the UNIX TCP/IP host on the remote network. The command is:

ping 191.1.1.137

If the test is not successful, do the following:

1. Ping from the UNIX host to the NetWare file server:

   ping 191.1.1.140

   If this doesn't work, go to Step 2 to resolve the problem. After the UNIX host has successfully pinged 191.1.140, go to Step 3.
2. Check the addressing on the file server. Verify that the correct network address (191.1.1.140) has been assigned to the second board in the file server. (If the second board is physically attached to network

191.1.1.128 with a subnet mask of 255.255.255.192 (FF.FF.FF.C0), the address of the board in the file server must have an IP address and subnet mask that is legal for the 191.1.1.128 network.)

3. Ping from the UNIX host to the first board in the file server, which is attached to network 191.1.1.32:

   ping 191.1.1.60

   If this doesn't work, proceed to Step 4.

4. Verify that the routing is configured correctly on the UNIX host by making sure that IP packets can be routed to the NetWare file server. In other words, check to see if the host at 191.1.1.140 (the NetWare file server) is recognized as the router to route packets from network 191.1.1.128 to network 191.1.1.32.

   On a UNIX system, the general command is:

   ROUTE ADD <destination network> <gateway address>

   <metric>

   (Consult the manual for the UNIX host you are working with for the exact syntax for adding routes.)

   In this example, the command would be:

   ROUTE ADD 191.1.1.32 191.1.1.140 1

   This route command says that if there is a packet that is to be routed to network 191.1.1.32, send it to the router at address 191.1.1.140 which is only one hop away (the packet has to be forwarded through one router to get to its remote network destination).

   To assign the NetWare server (IP address of 191.1.1.140) as the default router for the UNIX system (in order to route packets destined for any network other than the local), the command syntax would be:

   ROUTE ADD 0.0.0.0 191.1.1.140 1

   If there are other non-NetWare TCP/IP routers on the network, verify that those routers support RIP. The NetWare TCP/IP router supports only RIP. If the other routers do not support RIP, static routes have to be added at all routers. (Configuring static routes for the NetWare TCP/IP router is explained on page 6-27 of the NetWare v3.11 TCP/IP Supervisor's Guide.)

Note: If routing is configured correctly on one network but not on the other, packets may be sent to a host on the remote network but the remote host will not be able to route the packets back. In this example, the LWPD workstation may have the correct routing entries in the NET.CFG file (resulting in the ping packet being correctly sent to the UNIX host on the remote network), but the UNIX host does not know how to route the packet back because it does not have a ROUTE ADD command set up.

After the above steps have been completed, and routing has been configured correctly on both networks, hosts on one network should be able to ping to hosts on the remote network. You have now finished testing the NetWare TCP/IP configuration.

Token-Ring Source Routing

This section is for those who need to route IP packets through IBM source routing bridges. It briefly outlines the commands you need in the AUTOEXEC.NCF file at the server, and in the AUTOEXEC.BAT file at the workstation.

Source Routing Configuration on the NetWare File Server

To route IP packets through IBM source routing bridges, you need to enter the following commands in the AUTOEXEC.NCF file:

```
LOAD ROUTE BOARD=01
LOAD ROUTE BOARD=02
```

Note: These LOAD statements need to be entered in the AUTOEXEC.NCF file after the LOAD and BIND commands that are required to configure NetWare TCP/IP.

The board numbers listed in the LOAD statements are logical board numbers. These statements will enable IPX (logical board 1) and IP (logical board 2) frames to go through the IBM source-routing bridge.

To determine the value to assign for BOARD= parameters, do the following:

1. At the NetWare console prompt, type LOAD TCPCON <Enter>.
2. Select <INTERFACE TABLE>.
3. Locate the entry or entries for the Token-Ring LAN driver and note the logical board numbers. These are the numbers to add to the LOAD ROUTE command.

(For more information, refer to pages 224 through 229 in the NetWare v3.11 System Administration manual.)

Source Routing Configuration on a Workstation

If you are using source routing on a LWPD workstation, specify the following commands in the AUTOEXEC.BAT file:

```
LSL
LANSUP    (or TOKEN)
IPXODI
ROUTE BOARD=1
ROUTE BOARD=2
NETX
```

The ROUTE.COM module is loaded twice because both frame types (IPX and IP) require source routing and each appears as a separate board to the LSL.

The BOARD= values for the ROUTE.COM commands in the AUTOEXEC.BAT file are determined by the values in the AUTOEXEC.NCF file on the NetWare v3.11 file server. For example, if the AUTOEXEC.NCF file contains LOAD ROUTE BOARD=4 and LOAD ROUTE BOARD=5, then the entries in the AUTOEXEC.BAT file on the LWPD workstation would be "ROUTE BOARD=4" and "ROUTE BOARD=5".

(For more information, refer to the NetWare ODI Shell for DOS manual, Appendix A, "Using the IBM Token-Ring Source Routing Driver," pages 37 through 44.)

Obtaining Technical Support

To receive technical support, contact your local Novell reseller and request support. Your Novell reseller may want to work with Novell directly, since the reseller may not incur a fee.

For support on NetWire, post questions in NOVA section 8 (NetWare NFS/TCP). If you do not have a CompuServe account, call CompuServe at 800-524-3388 or 614-457-0802 and ask for operator 200.

To place a call with Novell Technical Support, call 800-NETWARE (800-638-9273) or 801-429-5588 for international calls. If the issue is determined to be a software bug not previously reported, there is no

charge for the support call.

When calling your Novell reseller, Novell Technical Support, or posting a NetWire message, please provide the following information:

- The IP address of your local host.
- The IP address of a remote host.
- The IP addresses of the two (or more) LAN adapters in the NetWare 3.11 file server.
- The subnet mask for both the local and remote networks.
- Information from the NET.CFG file (LWPD workstation).
- Information from the AUTOEXEC.NCF file (NetWare 3.11 file server).
- Information from the NetWare CONFIG console command.
- Test results from the ping command (from the tests outlined in the Troubleshooting section above).
- root access to any UNIX systems that might be involved in the configuration.
- SUPERVISOR access to the NetWare file server.