# All You Ever Wanted To Know About IP Routing

(By Adrien de Croy – Rev 1.0 25 April 2003)

# 1. Introduction

In IP-based networks, the concept of routing is one of the most commonly misunderstood, yet important issues. It can make or break a network. Routing misconfigurations can have weird, wonderful, obscure and seemingly magical negative effects. It is often considered a black art.

It is however completely unworthy of this amount of deference!

Misunderstandings are invariably the result of lack of knowledge about some basic principles of routing, which if known would have made countless lives easier. This is sometimes not helped either, when most operating systems vendors have (albeit understandably) chosen to avoid this issue, and automate things as much as possible.

So, if a few fundamental pieces of information are kept in mind, **understanding routing can be extremely simple**.

This document aims to provide the necessary pieces of information in a coherent form (and in one location), which have been learned over many years of finding things out the hard way.

If you can understand and keep these principles in mind, then you should have no problems setting up networks and making modifications to system route tables etc.

# 2. Basics

## *2.1 EVERY operating system that supports TCP/IP has a router and a route table.*

This includes every version of Windows (right back to Windows 3.1), linux and other unix systems, etc etc. You name it, there is one.

A common misconception is that if an operating system cannot act as a "router" (cannot relay packets received on one interface out onto another), such as Windows 95 or 98, then it doesn't have a process in it, which routes packets. ***This is not true!***

In any version of Windows, you can go to a command prompt, and type "route print", and be shown a copy of the system route table. Understanding this table and how it is used by the operating system is the key to understanding routing. We'll get on to that soon, but before that you need to know the following **MOST IMPORTANT KEY FACT**:

## *2.2 The route table is used ONLY when it comes to deciding how to DELIVER packets.*

If you remember this and nothing else, you will be 90% of the way to becoming a routing expert. Consider the following:

There are several ways that the operating system can find itself with the task of having to deliver a packet somewhere.

- A user application on that system may have attempted to make a connection, or send some data to another machine. E.g. you may have tried to check your email, or may be browsing a website

- The system may have received a packet from some machine, which it must then forward to another machine.

However, in either of these cases, or any other, the networking subsystem has a packet, and it has to do something with it. It needs to know where to **deliver it TO**.

Because the route table is only used when it comes to delivery of packets, it stands to reason that:

## *2.3 The only piece of information in the packet used when routing, is the DESTINATION address.*

I can't think of any router software that considers for instance the source of a packet in determining where to forward it. The source may be important, yet by that stage the system has already received the packet, and needs to forward it on, so the destination is what the routing software focuses on.

There is one other piece of information of major importance.

## *2.4 Routing MUST be configured correctly on BOTH endpoints in a communication and on every hop in between*

99.9% of any **communications** of note used on the Internet or indeed private networks **are** 2-way or **BIDIRECTIONAL**. In other words a machine talks to one machine, which then talks back to the first one.

Since however routing is performed based only on the destination address of a packet, **Routing is UNIDIRECTIONAL**, and therefore **each** endpoint (machine) in a conversation **must believe it can send a packet** to the other. This requires each machine to have a route entry in its route table that it can use to send a packet to the other machine.

Furthermore, if the packets must go across multiple networks (e.g. the internet) to get from one participant in the communications to the other, then each forwarding agent (router) must be able to forward the packet on. That means routing must be configured properly on every single forwarding agent on the path. It is therefore possible and actually fairly common, due to quirks in routing setup, for the forward path A > B to be quite different to the reverse path back B > A

You can see the path that a packet will take from the machine you are on to another using the program "tracert.exe" which you run from the command line.

So, now we can consider how routing is performed.

# 3 How Routing Happens

OK, so the operating system has this packet, and it needs to know what to do with it. Basically, what it does is as follows.

It takes the **destination IP address** of the packet, and attempts to **match** that destination address with **every entry** in the OS route table until it finds the **best** match. If it doesn't find a match, it simply **drops the packet**, which is then lost, and may return an error message to the sender of the packet (if it believes it can).

So, there are a few new concepts here, which I shall now explain.

## *3.1 Destination IP address:*

Every IP packet, which nowadays is used for pretty much all network communications, has what is known as a header – the envelope for the letter (payload), which gives the information about where the packet is from, and where it is going. The address of where the packet must ultimately be delivered to is its destination address.

## *3.2 Route Table:*

The route table is made up of several entries. These define specific rules for:

- How to match the destination address to this route entry.
- How to forward the packet (whether to another router or directly to the end recipient)
- Which local network interface should be used to send the packet out onto.

OK, so now we are ready to look at a route table. This is the one off my machine. It has one interface, a network card with an IP address of 192.168.4.7, MASK 255.255.255.0, and a default gateway setting of 192.168.4.1. Here it is:

```
c:\>route print
===========================================================================
Interface List
0x1 ........................... MS TCP Loopback interface
0x1000003 ...00 02 b3 33 20 b3 ...... Intel(R) PRO PCI Adapter
===========================================================================
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0      192.168.4.1     192.168.4.7       1
        127.0.0.0        255.0.0.0        127.0.0.1       127.0.0.1       1
      192.168.4.0    255.255.255.0      192.168.4.7     192.168.4.7       1
      192.168.4.7  255.255.255.255        127.0.0.1       127.0.0.1       1
    192.168.4.255  255.255.255.255      192.168.4.7     192.168.4.7       1
        224.0.0.0        224.0.0.0      192.168.4.7     192.168.4.7       1
  255.255.255.255  255.255.255.255      192.168.4.7     192.168.4.7       1
Default Gateway:       192.168.4.1
===========================================================================
Persistent Routes:
  None
```

**Figure A – a typical system route table**

### 3.2.1 How to Match

Matching is done with three bits of information. The Network Destination, the Netmask, and the destination IP address of the packet.

Basically, the destination IP address is bit-wise ANDed with the Netmask (this is the definition of masking – I will explain below), and if the resultant number equals the Network Destination, then we have a match!

**The humble, yet omnipotent Netmask**

It is not by accident that the word "mask" is used in this sense. Just like when you put a facemask on your face, all that others can then see is the mask, and whatever bits of your face that show through (that haven't been masked). When an IP address is masked with the netmask, the pieces you see after the masking relate to the network portion of the address. Therefore **the Netmask for a network adapter must be carefully chosen**, as it defines which part of an address is the network part, and therefore how routing will match destination addresses to route entries.

Bit-wise ANDing is as below.

Firstly, consider that IP addresses, even though they are represented in the form say "192.168.4.7", they are actually a **32 bit number**. A bit is a binary digit. You will be used to seeing decimal digits, but computers work in binary.

Netmasks are also 32 bit numbers, as are Network Destinations. The process of ANDing is applying the logical AND operator. The resultant bit of a bit-wise AND operator is 1 if and only if **both** inputs bits are 1.

**Example.**

Say I want to check my email on our internal company email server, whose address is 192.168.4.100. Ok, so the destination IP address on the packets I send to this server is going to be 192.168.4.100.

From the route table above, if I choose to try and match this destination address with the third route table entry, you get the following question to answer. Does 192.168.4.100 when masked with 255.255.255.0 equal 192.168.4.0? In this case, yes it does, so therefore this route matches.

| | |
|---|---|
| 192.168.4.100 in binary is | 11000000 10101000 00000100 01100100 |
| 255.255.255.0 in binary is | 11111111 11111111 11111111 00000000 |

if you line up each bit vertically, the output bit is 1 only if both inputs (the mask and the address) are 1. So we get

| | |
|---|---|
| Result | 11000000 10101000 00000100 00000000 |

Which equals 192.168.4.0, which is the Network Destination for this route entry.

**Default Route**

If you tried this with all the other entries in the route table, you would find that there are no other matches, except for the **default route**. The default route is the route entry with both Network Address AND Netmask of "0.0.0.0". Since anything masked with all zeros will be all zeros, therefore by definition **the default route matches ALL DESTINATIONS.**

For this reason the default route is only used as a last resort, if no other route entries match.

### Some Shortcuts:

If you look more closely at the columns n the example, you see that wherever the mask was a one, the result is the same as the destination IP address. And where the mask bit is zero, the result is zero. If you then take the shortcut that 255 is just 11111111, then you can quickly see that the bits of the network mask that are 255 will mean that the corresponding **octet** (the number between the dots in the address) in the destination address will not be masked.

## 3.2.2 How to forward the packet

They actually cheat you a bit with what information you are provided with in the output from "route print". There are several other pieces of information, which in some circumstances could be quite useful, but you need to use other tools to find them out. One piece of useful information is the **route type**.

You can however deduce the route type of a route entry based on certain fields in the entry.

**Local segment route**

Basically, if the gateway and interface addresses are the same, that means that if a destination matches this route, then the system expects that the recipient exists locally on the network segment that the interface is directly connected to. That means that the system will try and look up the Ethernet address of the recipient, which if it fails will cause a lookup for that (an ARP request)

If the host is not available, or not responding to ARP requests, then the packets will be dropped. (more on ARP later).

**Gateway route**

If the gateway and interface addresses are different, that means that the packet whose destination address matches the route entry must be forwarded to the gateway, who will then presumably relay the packet onwards towards its destination. The system therefore does not attempt to look up the Ethernet address of the destination, but rather the Ethernet address of the gateway. It then changes the destination Ethernet address (destination MAC address) to the gateway's Ethernet address, and sends the packet on the interface.

**Localhost route**

If the interface address of the route entry is “127.0.0.1”, then any packet destined for an address that matches this route entry, will be received by the machine itself.

**Broadcast route**

You can tell broadcast routes because the network mask is 255.255.255.255, and the network address looks like a normal network address with any zeroes replaced with ones – i.e. 192.168.4.255 is a broadcast route.

What this means, is that if a packet’s destination address matches this route entry, the packet will be BROADCAST on the interface. That means that the Ethernet address will be set to a broadcast Ethernet address, which means that every Ethernet device on the local segment will notify its host machine that the packet has been received. Broadcast traffic is a bit outside the scope of this document. Note however, that the broadcast address is also defined by that humble network mask!

**Multicast route**

Multicast is beyond the scope of this document. Suffice to say that multicast was developed to allow a broadcast-like functionality, where many clients could receive a copy of the same packet, without actually using broadcast.

## *3.3 Best Match*

The concept of best match is not often used, unless you happen (like me) to occasionally write router software! However, the more observant of you might have considered that there are some addresses that will match more than one route entry. For example take the destination address 192.168.4.7.

This will match not only the third route entry (192.168.4.0 mask 255.255.255.0) but also the fourth (192.168.4.7 mask 255.255.255.255).

The intent obviously is that a packet with this destination address is actually bound for the machine itself, so it needs to not be forwarded out any other interface, but presented to the network software on the machine itself.

So, to get around this, there are deemed priorities of routes. The system looks for an exact match with the highest precision mask (255.255.255.255 is more precise than 255.255.255.0, which is more precise than 0.0.0.0).

There are some other obscure cases where you may wish to have a specific route which supercedes a local segment route. In the main these normally only point to an issue somewhere else which would best be solved in another way (like renumbering an adapter or subnet) so I won’t be discussing them here.

## *3.4 Default Route*

The route with the network address 0.0.0.0 and the mask of 0.0.0.0 is the default route.

Since this route matches any destination address, this defines the route that is used if no better route can be found. Typically this is the route used to forward packets to your Internet Service Provider whenever you access the Internet.

In some cases there may be more than one default route on the system (which causes problems, but it does happen).

For instance, on windows, if you have a default route set on your network adapter, then you dial up with a modem to the internet, chances are you will end up with 2 default routes. Normally the operating system deprecates the previous default route by **increasing the metric of the route**.

**Aside: Metric**

The metric is a number associated with a route which conveys the concept of priority of the route. The smaller the metric, the higher the priority. Originally the concept of metric was a type of cost associated with using the route, so routes with lower cost were favoured over those with higher cost. If you have 2 routes that are the same (same destination address and mask), then the one with the lower metric will be used.

**Multiple default routes.**

The problem with multiple default routes is as follows: Basically if you had been using the previous default route (say to access other segments in your corporate LAN through an in-house router), then when you dial up the internet, then you lose access to the rest of your LAN. The way around this has historically been to manually create routes in your route table specifically for the other remote subnetworks that are being accessed through this in-house router, and removing the default gateway setting on the LAN adapter.

## *3.5 What do you mean "drops the packet"???*

It occasionally happens that a destination does not match ANY route entry. But hang on a minute you say, what about the default route – that matches any destination address, right? Correct. On some systems however, there is no default route configured. This is surprisingly common.

So, if there is no default route, and no route entries match, the destination is deemed to be **unreachable**.

If this packet was one being sent by your local system (i.e. you are checking your mail etc, but there is a configuration problem), then the packet is not sent, and an error is instead returned to your application, and you will hopefully get a message like "cannot connect to <blah> destination unreachable". Less helpful software may return an error number, the winsock error for destination unreachable is 10061

If this packet had been received from another machine, and the system was supposed to be forwarding it on to its destination, then a well-behaved (some don't) router will create a special error packet (which uses the ICMP protocol), which it sends back to the originator of the packet if it can. I say if it can, because it is possible that the source address of the original packet is not reachable either! Even though you actually received the packet, and therefore it is physically reachable, **if the system doesn't believe the destination is reachable, it won't attempt to send the packet**.

# 4 Windows Networking

There are a few more things about windows networking that it helps to know.

## *4.1 How are route entries created?*

Route entries can end up in your route table via a number of different paths.

### 4.1.1 System generated routes

By default, when you install a network adapter under windows, and you give it an IP address and a network mask, and a default gateway, there are several routes created.

It is also possible that a DHCP server assigned you automatically with these 3 pieces of configuration information.

In any case, the following routes are created.

- **Loopback route**. This is the route with the destination address of 127.0.0.0. This defines a large number of destination addresses that can be used to send and receive packets from the machine to itself.

- **Localhost**, or local interface route. This is the route that points to the machine itself. The network mask is 255.255.255.255, and the Network Destination is the same as the interface IP.

- **Local subnet route**. This is the route that defines how to get to other machines on the same logical subnet. The Network Destination is made up from the interface IP address and the network mask (interface IP ANDed with the network mask)..

- **Subnet broadcast route**. This defines the addresses that are deemed to be broadcast addresses for the subnet. The Network Destination is also made up from the interface IP address and the network mask, but in a different way (logical OR with the inverse of the network mask).

- **Global broadcast route**. One of these are created for every adapter in the system also. The destination address and network mask are always 255.255.255.255. This prevents the router software from ever forwarding a packet destined to 255.255.255.255 to any other router.

- **Multicast route**. This is always 224.0.0.0, but there is one created for every interface in your system.

- **Default route**. The network address and network mask are always 0.0.0.0, and the interface and gateway are the interface IP and default gateway settings. If you don't specify a default gateway setting in any adapter on your system, or your DHCP server does not allocate you a default gateway, then this route won't be created.

So, going back to my example route table before, you should be able to see which ones are which. It is interesting to note, that **every single entry in the**

**route table in figure a, was created by the operating system from only the interface IP address, network mask, and default gateway setting**.

### 4.1.2 Manually entered routes

You may also create routes manually with the program route.exe, which is a console application (you run it from a command line).

You may need to do this for a variety of reasons.

### 4.1.3 Other software-created routes

Other software can also on occasion create routes. WinGate will create routes in certain circumstances, also the Qbik RIPv2 client will create routes based on updates from RIPv2 servers.

RIP is a protocol designed for routers to advertise routes to each other, and so automate a certain amount of route configuration. It is especially useful for VPNs where perhaps for some reason a VPN gateway is not the default gateway for a network.

## 4.2 Logical vs Physical subnets

It is first useful to consider some of the salient points of an Ethernet network, since these are the most commonly used, and even many other network types simply emulate Ethernet anyway.

### 4.2.1 Ethernet networks.

Ethernet is the name given to the system that provides the basic communications between computers over Ethernet hardware, such as network cards, cable, hubs and switches. There are other systems, such as Token Ring, but these are very small in number compared to Ethernet.

In the early days of Ethernet, the Ethernet adapters were connected together by a single coaxial cable, with T connectors in a daisy-chain configuration.

Nowadays people use twisted pair cable, and hubs and switches to form networks in more of a star configuration. It is important to realize however, that in many respects a hub-based system is the same as the old daisy chain network. Every packet sent by each machine, is received by every other Ethernet device (network card) on that segment.

With a switch things are different, and the switch itself learns which Ethernet addresses are available on each of its ports, and only sends traffic destined for a machine connected to a port out that port.

Many people use hubs however, as they are less expensive than Ethernet switches, so the broadcast nature of Ethernet is applicable.

So, whether or not that packet makes it up to the operating system on the host machine depends on whether the Ethernet destination address matches the

Ethernet address of the network adapter (MAC address) – or is the broadcast Ethernet address of FF-FF-FF-FF-FF-FF. In general (unless special software on that computer which puts the Ethernet card into "promiscuous mode" is in use) the network card only notifies the host computer if a packet is received that matches its address (or is broadcast).

This brings us to:

### 4.2.2 Ethernet Addresses (MAC Addresses)

So, hang on a minute you say… what's all this about Ethernet addresses? I was just getting the hang of IP addresses. How does all this work?

One of the fundamentals of any network, whether it be the telephone network, the postal service, or anything else you can think of, is that the participants in the network, or the devices used to participate, each have a unique address. You have a phone number, a postal address, and your Ethernet card has an Ethernet address.

The Ethernet address is a number that is 48 bits long, and is usually written in the form 00-02-b3-33-20-b3.

Every Ethernet device has a unique address. This is pretty hard to fathom, but actually the manufacturers of Ethernet devices get a chunk of numbers allocated to them, which they program into the firmware on the cards! True!

So, you don't need to worry about allocation of these addresses. You can actually tell a bit about a card from its MAC address as well, since you can find out the manufacturer from the first few numbers of the address.

So, now this is a bit of a shift – now there are Ethernet addresses to think about. How do two machines communicate on an Ethernet segment if you only know the IP address? How does it find out the Ethernet address from that?

The answer is:

### 4.2.3 ARP (Address Resolution Protocol)

There is a method used to discover an Ethernet address that matches an IP address… it is called ARP (Address Resolution Protocol). It uses a broadcast Ethernet packet, with the request. The machine that owns the IP address in the request makes a response packet to identify itself as the owner.

However, since the Ethernet frame is only sent out the local segment, a machine will only respond if it is on that segment.

For this reason, you cannot use ARP to determine the Ethernet address of a machine that is not on a network directly connected to you. That is why we forward packets to routers, and have other protocols such as IP to handle forwarding across multiple networks, and concepts such as subnetworks.

So, hopefully now the mechanism of routing is making more sense. I know when I started considering routing, I thought there must be more fields in the

IP header that specified a router. Nope. In order to forward a packet to a router for delivery, you simply send it to the router's Ethernet address!

One other thing about ARP, is that the results of ARP requests are cached – remembered by the system. It would be no good if you had to do an ARP request for the destination of every packet you wanted to send to a machine.. that could be millions of ARP requests, and anyway, people's IP addresses don't change often.

You can even view the contents of the ARP cache on your machine by going to the command line and typing "arp –a"

### 4.2.4 So What is a logical subnet?

So, now you know a bit about Ethernet (physical) subnets, what is a logical subnet?

There is an important distinction between logical and physical subnets. I remember when I first got my first 2 computers in one place, and got hold of some network cards. I had the full expectation that simply plugging the things in was sufficient to get them talking to each other. I found out the hard way that this is not the case (except nowadays for *Autonet addresses*, which I shall also discuss later).

Being plugged into the same hub, or Ethernet segment means you are on the same physical sub-network (subnet). To be on the same logical subnet, you need to have addresses on the machines that are recognized by each machine in that logical subnet as being locally accessible addresses. In other words, the IP addresses of the machines must be in the same network, where that network is defined by using the netmask to mask the IP address of each machine (note that the network masks may not necessarily all be the same!).

Why this comes about is due to the routing issue again. If a machine does not believe it has a route to another machine, it will report an error instead of trying to contact it (most of the time if it just bothered to try, it would actually work!). Since windows only creates local network addresses by itself, if you wanted to have 2 different ranges of IP addresses in use on the same Ethernet segment, you would have to manually add the correct routes to enable the machines to think they can communicate.

### 4.2.5 Autonet

Starting with Windows98 SE, Microsoft decided to take the pain out of some basic network configuration. The problem they were trying to address was the one I had seen when I first tried to network two computers… I didn't know that I had to allocate IP addresses in the same logical subnet.

To get around this, Microsoft devised a system whereby the machine when turned on, if it didn't have TCP/IP properties set, it would attempt to find a DHCP server to allocate them. If it failed to retrieve a

configuration from a DHCP server, then it would go into Autonet mode, and allocate itself an IP address in the range 169.254.X.Y.

Now, if you presume that another machine on the network encounters the same problem, not finding a DHCP server, and does the same thing, bingo! These two machines have put themselves on the same logical subnet, and can communicate.

Problems arise with this however on networks where for some reason or another, a DHCP server is not available (machine may be down when the client makes its request), and so the machine allocates itself an Autonet address. It is then unable to communicate with other machines on its network, which have had addresses either manually set, or previously allocated by the network's DHCP server.

### 4.2.6 So, what is this DHCP thing?

I have mentioned DHCP a few times, and you may well have seen it in your networking configuration, or other documentation.

Early versions of Windows mention DHCP in the TCP/IP properties of network adapters. Later versions use words such as "Obtain an IP address automatically".

In both these cases, they are talking about the Dynamic Host Configuration Protocol (DHCP).

As its name suggests, DHCP is a method whereby clients (hosts – more Internet geek speak for a machine) are able to have their TCP/IP properties configured by a server centrally.

The DHCP client on your Windows machine uses broadcast packets to discover a DHCP server. The DHCP server then often uses a broadcast packet to respond, since the client machine often doesn't have an IP address when it makes the request.

Broadcast traffic, as I mentioned before, is notified to the operating system of every machine on the physical Ethernet segment.

## 4.3 Multi-homed hosts

A multi-homed host is Internet geek terminology for any machine that has more than one network interface.

This is common in gateway machines, routers etc. Even often some manufacturers provide more than one network interface by default, one of which may never be used (yet still cause troubles).

Whenever you are tasked with setting up a multi homed host, you need to keep in mind a few things, depending on what this machine will be doing.

Firstly,

### 4.3.1 Make sure that each network interface is on a different logical subnet!!!

You can make life difficult for yourself if you say have 2 network cards in a machine, one is given the IP address 192.168.0.1 mask 255.255.255.0, and the other is 192.168.0.2 mask 255.255.255.0. There will be in this case numerous routes created, and if you go to the command line and type "route print", you will see a route table that includes the following two route entries:

| Network Destination | Netmask | Gateway | Interface | Metric |
|---|---|---|---|---|
| 192.168.0.0 | 255.255.255.0 | 192.168.0.1 | 192.168.0.1 | 1 |
| 192.168.0.0 | 255.255.255.0 | 192.168.0.2 | 192.168.0.2 | 1 |

The immediate thing that should be ringing warning bells, is that the Network Destination is the same!!!!

So, what happens now if you want to send a packet to 192.168.0.3, which is on the segment connected to the adapter with the address 192.168.0.2?

Well, there is immediately a problem, since the destination 192.168.0.3 will match both of these route entries, and so which interface should the OS send the packet out of? There are only 2 options, both, or one or the other. Neither of these options is fantastic. One is wasteful of network resources, and the other may not work.

Some operating systems may actually send the packet out both, but most will simply pick the first best match, which will then depend on what order the routes are in in the route table.

### 4.3.2 Make sure only one of the interfaces has a default gateway set

In general, any more than one default route is a problem. If you have a default gateway set on two adapters that is different for each one, you create a dilemma for the routing software… which one to choose?

# APPENDIX A - Importance of the netmask

The humble network mask that you typed into the TCP/IP properties for your network adapter, or which was assigned to you (perhaps automatically) can have vast and grave consequences! In fact, random or inappropriate choice of network mask for an adapter can have really wacky results! I sure wish they had told me this when I first started setting up networks!

For instance, say your finger slips when you type in the netmask, and instead of typing in an address of say 192.168.0.1, and mask 255.255.255.0, you say type in a mask of 225.255.255.0

Looks close enough doesn't it, in fact you might not even notice. But now instead of having a nice mask of

11111111 11111111 11111111 00000000

you now have

11000000 11111111 11111111 00000000

That means that any address that matches on this will be sent out this interface. Consider the address 202.168.4.37. This is a public address, and may well exist on the internet. Do the math on this, and you will find that it matches the route that will have been created, and so will be sent out your LAN interface. If you ever wanted to access this machine on the internet, you would not be able to. There would be many other addresses that would also match this route entry, in fact any address that starts with a number greater than or equal to 192, and has a middle two octets of 168.4 and ends in any number.

This is why the network mask is so important. The network mask is also used to determine what is considered to be a broadcast address.