Supernetting for exact match by hand is rare, only method is to manually calculate each subnet’s range (network ID, next network). Supernetting to a single network is more common.

We can use just the network ID and broadcast IP to solve supernetting problems. We will solve these 6:

A number with numbers on it

AI-generated content may be incorrect.

We need to find the smallest IP address and the largest IP address. The smallest IP address here is the network ID of the lowest numerical subnet which is 9.9.9.0. The largest IP address is the last IP address in the highest numerical subnet.

In the image above the last network ID is 9.9.9.112, since we know that /28’s go up by 16 we need to find the next network, so we increment by 16 which gives us 9.9.9.128 then we -1 because we want the broadcast IP (which is the last IP address) which is 9.9.9.127.

Now we need to find a group size increment that includes both the smallest and IP address in the relevant octet:

A number written on a white board

AI-generated content may be incorrect.

So, if we do 127-0, we get 127, so we know we need an increment that is at least 127.

A white paper with black numbers and pink marker

AI-generated content may be incorrect.

The increment here that is at least 127 is 128 which is the CIDR /25. And since there is a /25 at 9.9.9.0, this gives us our answer. The supernet of these 6 /28’s is 9.9.9.0/25.

The thing is, if we find a subnet that includes the smallest IP address and the largest IP address, it will definitely include all the IP addresses in between

Now let’s try supernetting for these:

A number with numbers on it

AI-generated content may be incorrect.

First, we will start by identifying the smallest IP and the largest IP. The smallest IP is 9.9.9.64 and the largest IP is the broadcast IP address of the highest numerical subnet which for us is 9.9.9.112. so, the broadcast IP be that but an increment of 16 (because /28’s increment by 16) then -1. So, if we increment 16, we get .128, so we -1 and the broadcast IP would be 9.9.9.127.

A number written in blue ink

AI-generated content may be incorrect.

So, if we do 127-64, we get 63, and on the chart:

A white rectangular object with black numbers and pink x marks

AI-generated content may be incorrect.

We get to the increment of 64 which the CIDR is /26. And since the /26 happens to start at 9.9.9.64, we have the answer of 9.9.9.64/26.

Let’s solve for this:

A number with numbers on it

AI-generated content may be incorrect.

This time we are solving for mixed sized networks. We need to find the smallest IP address which would be the network ID of the lowest numerical subnet. In our case it would be 9.9.9.128/26. So, the smallest IP address is 9.9.9.128. The largest IP address would be the broadcast IP of the highest numerical subnet. The highest numerical subnet is 9.9.9.240/29. We can see that a /29 has an increment of 8 so we do +8 which is 9.9.9.28 then -1 which is 9.9.9.247.

NEXT PAGE

A number written on a white board

AI-generated content may be incorrect.

Then we need to find a group sized increment that includes both the smallest and largest IP address, so we do 247-128 which gives us 119.

A close up of numbers

AI-generated content may be incorrect.

128 is the smallest group size increment that includes both IP addresses which is the CIDR /25. And since there happens to be a /25 that starts on the smallest IP address (9.9.9.128), we know the answer is 9.9.9.128/25.

Each supernetting problem we have done has been done using subnets, but there are questions that involve summarising individual IP addresses. Let’s solve for this:

A number with numbers on it

AI-generated content may be incorrect.

Lets get the smallest IP addresses which is 9.9.9.140, the largest IP addresses is 9.9.9.170, since we have been given the IP addresses, we don’t need to calculate network ID’s or broadcast IP’s.

So, if we do 170-140, we get 30. So, we need a jump of at least 30 (group size)

A white paper with black numbers and pink lines

AI-generated content may be incorrect.

But we have to be careful, if we take the /27 and we list out the increments in sets of 32 (we can start from 128 because of the speed tip, the smallest IP address is above 128):

NEXT PAGE

A white board with blue writing on it

AI-generated content may be incorrect.

We can see that the IP addresses we are trying to solve, 140 through to 170, start and end in different subnet blocks, so even though a /27 is enough IP addresses to account for this jump, these 2 addresses happen to exist in 2 different /27’s.

A screenshot of a graph

AI-generated content may be incorrect.

This means that a /27 isn’t big enough to include the entire set of IP addresses, so we have to go one step further and go to a /26:

A screenshot of a graph

AI-generated content may be incorrect.

The increment of /26 is 64, so if we start at .128 and increment by 64 each time:

A close up of numbers

AI-generated content may be incorrect.

In between these 2 values, all 6 IP addresses can fit in there, meaning a /26 accounts for all IP addresses. So, the answer would be 9.9.9.128/26.

So far, we have been solving supernetting problems from the fourth octet, we will now solve for different third octets:

A number with numbers on it

AI-generated content may be incorrect.

Fort this we also need to extend the cheat sheet:

A number with numbers on a white background

AI-generated content may be incorrect.

The smallest IP address is 9.9.44.146, the largest IP address is 9.9.47.209.

A number written on a white board

AI-generated content may be incorrect.

Now we need to find a group size increment that includes both IP addresses IN THE RELEVANT OCTET, in this case it is the third octet. So, 47-44 gives us 3.

A screenshot of a computer screen

AI-generated content may be incorrect.

4 is the smallest increment size that accounts for both the largest and smallest IP addresses. Lets see if an increment of 4 works, lets rule out all the increments, we can multiply 4 by 10 (from speed tip), then increment from there:

A white board with blue writing on it

AI-generated content may be incorrect.

As we can see that have found a jump that includes the IP addresses we need to summarise. Which means the answer here is 9.9.44.0/22.

Now lets solve for a problem which uses the third octet, but we will be solving for mixed networks that span the third and fourth octet.

A number and numbers on a white background

AI-generated content may be incorrect.

The smallest IP address is the network ID of the lowest numerical network which is 9.9.162.0/23, then the largest IP address is broadcast IP of the highest numerical network which is 9.9.170.160/27. Since a /27 is for the fourth octet, and it has a jump of 32 increments, we increment the fourth octet with 32 then -1. So, 160+32 is 192 then -1 which gives us 191. So, the broadcast IP is 9.9.170.191. Now we need to find a group sized increment for the relevant octet which includes both.

A close-up of numbers

AI-generated content may be incorrect.

But we have to be careful here because if we do 170-168, that gives us the increment jump of 8 which is the CIDR /21. However, this jump takes us from 9.9.162.0 to 9.9.170.0, we also need to account for the .191 so for this reason we need to find a jump that includes 8 in the third octet + the .191 to account for the fourth octet. So, we have to go a step further and go to the next increment jump value, which is 16, which the CIDR is /20.

So, we can now list out the jumps but instead of starting from 0, we can multiple 16 by 10 to start from 160 then increment (speed tip), so:

A close-up of numbers

AI-generated content may be incorrect.

As we can see, we have found a jump that now includes both these:

A close-up of numbers

AI-generated content may be incorrect.

So, our answer would be 9.9.162.0/20