A screenshot of a computer

AI-generated content may be incorrect.

When it comes to VLSM, we have this 9.9.9.0/24 network, we are given a /24 block of IP addresses which totals to 256 IP addresses to delegate. We will assign sub-networks from the /24 across the 8 segments on the 9.9.9.0/24 network. Each segment would require a certain amount of IP addresses. This is a typical VLSM problem, we are given a starting network of IP addresses and a bunch of segments that require a various amount of host IP addresses.

We can break up the /24 network into a /25 network which makes up 2 blocks of 128 IP addresses each, we can continue subnetting:

A white board with colorful squares

AI-generated content may be incorrect.

Lets start with the first network that requires 25 IP addresses. We need to find the smallest size network that can hold 25 IP addresses. We cant assign one of the /28 sub-networks because 16 IP addresses is too small, a /26 is too big because it has 64 IP addresses. But a /27 is perfect because it can hold 32 IP addresses. So, we will assign a /27 block, meaning on the chart we use up the first /27 block. If we look at the subnetting cheat sheet:

A close up of numbers

AI-generated content may be incorrect.

We can see that the /27 CIDR has a group size of 32. The group size is also the increment number, meaning if we increment by 32 each time, we can map out the next network:

A screenshot of a computer

AI-generated content may be incorrect.

As we can see since the increment is 32, the first /27 block contains 32 IP addresses. So, the network ID for the first /27 block is 9.9.9.0, the network ID for the second /27 block is 9.9.9.32. The network ID of the 5th /27 block is 9.9.9.128. We need to pick one of these blocks to assign to our subnetwork on the 9.9.9.0/24 network (the one that requires 25 IP addresses). For this example, we will choose the first /27 block, this block will be assigned to the first sub-network on the 9.9.9.0/24 network. Our first segment which requires 25 IP addresses is assigned the sub-network 9.9.9.0/27 for its IP space. Now we have to do the same for the next network, the next network requires 50 IP addresses, meaning that a /26 is perfect.

We have to use a /26 for the next network, the cheat sheet tells us about the increment for the CIDR /26 is 64, which means our 4 /26 networks have these network ID’s (check the image below), we just need to assign one of these blocks to the next network which requires 50 IP addresses.

NEXT PAGE

A screenshot of a computer

AI-generated content may be incorrect.

We have to be careful, we cant just grab the first block of the /26, because the /27 is occupying that area. When we assigned the /27 to the first network earlier, we committed the IP addresses 9.9.9.0 to 9.9.9.31 to the first segment:

A screenshot of a graph

AI-generated content may be incorrect.

The first part of the /26 includes those IP addresses assigned to the first segment which are no longer available, which means we cannot assign the first /26 because a portion of it has already been used in our 9.9.9.0/24 network.

NEXT PAGE

Since we cant assign the first /26, we have to assign the next one which is free in this case:

A screenshot of a computer

AI-generated content may be incorrect.

We will assign this sub-network (9.9.9.64/26) to the segment which requires 50 IP addresses. Now lets do the same thing for the next network, this network requires 10 IP addresses so we look for the smallest size network that can hold at least 10 IP addresses, in this case it is the /28 network. We can see that the /28 network has a group size of 16 so here are the increments:

A screenshot of a computer

AI-generated content may be incorrect.

There are many free spaces to use for our /28 network, we will simply choose the first available one. Which means our third segment (which requires 10 IP addresses) will be assigned the 9.9.9.32/28:

NEXT PAGE CHECK IMAGE:

A screenshot of a computer

AI-generated content may be incorrect.

Now lets solve the last 3 segments. The more optimal way to solve this would be to list out the network sizes on each network. So, the first segment requires 19 IP addresses meaning a /27 would be good. The following network requires 12 IP addresses so the smallest size network to accommodate for 12 IP addresses is a /28. The last network requires 31 IP addresses. A /27 network has 32 IP addresses but only 30 are usable so we cannot use a /27, hence we should use a /26 which has 64 IP addresses:

A screenshot of a graph

AI-generated content may be incorrect.

The first step was to list out all the required sub-network sizes based upon the needed amount of IP addresses per network. Now we can allocate the sub-network blocks to each segment, this is the current diagram:

A diagram of a rectangular object

AI-generated content may be incorrect.

However, the best strategy to do this is to start with the biggest size sub-network and allocate that a block then go in a descending order until all networks have been assigned sub-networks. Notice we left an empty /28 when assigning sub-networks to the first 3 segments:

A diagram of a chart

AI-generated content may be incorrect.

We could have avoided this if we started allocating sub-networks to the segments which require more IP addresses first. This time for the last 3 segments, we will assign them sub-networks in order from largest to smallest.

The biggest network size we have now is a /26 which requires 31 IP addresses. The next available /26 block is 9.9.9.128/26:

A screenshot of a computer

AI-generated content may be incorrect.

The next biggest network would be the /27, the increments for a /27 network is 32, this is how the increments would look like:

A screenshot of a graph

AI-generated content may be incorrect.

The next available /27 block (subnet) would be this one: (next page)

A screenshot of a graph

AI-generated content may be incorrect.

So, we allocate that one (9.9.9.192/27) to it:

A screenshot of a computer

AI-generated content may be incorrect.

And the last segment is the /28 which requires 12 IP addresses so the increments would look like this, and we’d assign this block:

A screenshot of a graph

AI-generated content may be incorrect.

So now our diagram looks like this:

A screenshot of a computer

AI-generated content may be incorrect.

As we can see that when we assign the sub-networks starting from the biggest requiring network sizes, we don’t leave any free blocks of IP addresses stranded between allocated blocks. This completes step 2 of solving this VLSM problem.

Now we are left to solving these 2 segments, which only require 2 IP addresses:

A diagram of a diagram

AI-generated content may be incorrect.

Segments that only require 2 IP addresses are referred to as point-to-point networks, the only 2 members of the segment are the routers themselves. The smallest size sub-network you could get to encompass 2 usable IP addresses would be a /30 which is a total of 4 IP addresses:

NEXT PAGE:

A diagram of a diagram

AI-generated content may be incorrect.

Now we need to carve out /30 sub-networks, we only extended our diagram to /28, so lets take that only stranded block of /28 which we left over when solving the first 3 segments (due to not starting from the biggest size network), and break it up into two /29’s (with 8 IP addresses each), then break both /29’s into 4 /30’s containing 4 IP addresses each. We can use the first 2 blocks and assign them to the remaining 2 segments.

A screenshot of a computer

AI-generated content may be incorrect.

However, we did say that the smallest sub-network size that has 2 usable IP addresses is a /30. That was true until an RFC was published that allows the use of /31 sub-networks for point-to-point links:

A close-up of a sign

AI-generated content may be incorrect.

The idea behind this RFC is that if there is only one other node on the network, what’s the point of having a broadcast IP, a /31 has 2 IP addresses and for point-to-point links, we can use both IP addresses.

So, we will break down the four /30 block into eight /31 blocks, and we will take 2 of the /31 blocks and use them for the remaining 2 segments instead:

A screenshot of a computer

AI-generated content may be incorrect.