

# Networking design approaches for Landing Zone-based organizations on AWS

15 September 2023 - 8 min. read

[Advanced Networking](#)

[Amazon VPC](#)

[governance and compliance](#)

[Landing Zone](#)

## Introduction

When we are faced with designing a Landing Zone in the Cloud, one of the most salient issues is networking since our networking configuration choice will affect our employees' daily operations. Also, all the decisions we will make will also relate to the Cloud-to-on-prem communication in case of hybrid Cloud environments.

(New to the Landing Zone concept? Start from [here!](#))

## Networking design

For designing the networking of our organization, AWS provides us with several possible solutions and approaches, each with particular benefits and specific use cases.

Remember that despite the multiple available possibilities, not all will always work for any use case. We should always **start by figuring out which will best suit our needs and requirements while also meeting the main pillars of our underlying infrastructure.**

Designing networking doesn't always mean reinventing the wheel; the perfect solution could also be an upgraded version of our current infrastructure and not a new implementation. This depends on considering the strengths and weaknesses of our past experience.

This article will look at the **best solutions for implementing a shared, centralized network infrastructure on AWS.** As we will see, there are two main roads for the network infrastructure in a multi-account approach: **single VPC VS multiple VPCs.**

## Single VPC with AWS RAM

Single-VPC approach is hard to adapt to a multi-account context, BUT, thanks to the AWS RAM service, this solution can be efficient and very functional.

This approach is one of the less-known ones; it consists of using **a single centralized VPC to deliver connectivity to the entire AWS organization**. As we will see in a moment, despite its simplicity, this solution will provide us with many benefits that are often overshadowed but can sometimes make all the difference.

First, however, let us understand how to implement a centralized VPC and how to adapt it to our particular requirements.

## Implementation

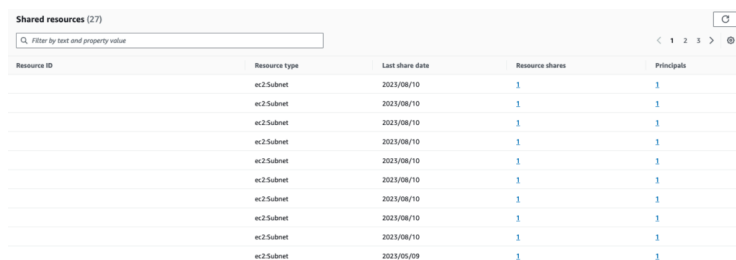
For the implementation of this infrastructure, we will need only a few services: a single VPC and the AWS RAM service.

As a small tip, **we recommend using a VPC with a wide CIDR address**. This will allow us to create numerous subnets without having to resort to extending the CIDR associated with the VPC.

To begin, we will only need to configure an account used for managing the network, from which we will later share all subnets to accounts in our organization (we will call it a *"network-account"*).

Once we create our VPC and a small subset of subnets, we can proceed with sharing them to the desired account, through the use of AWS RAM,

As a final step, we will just need to connect to the target account and accept the sharing of subnets.



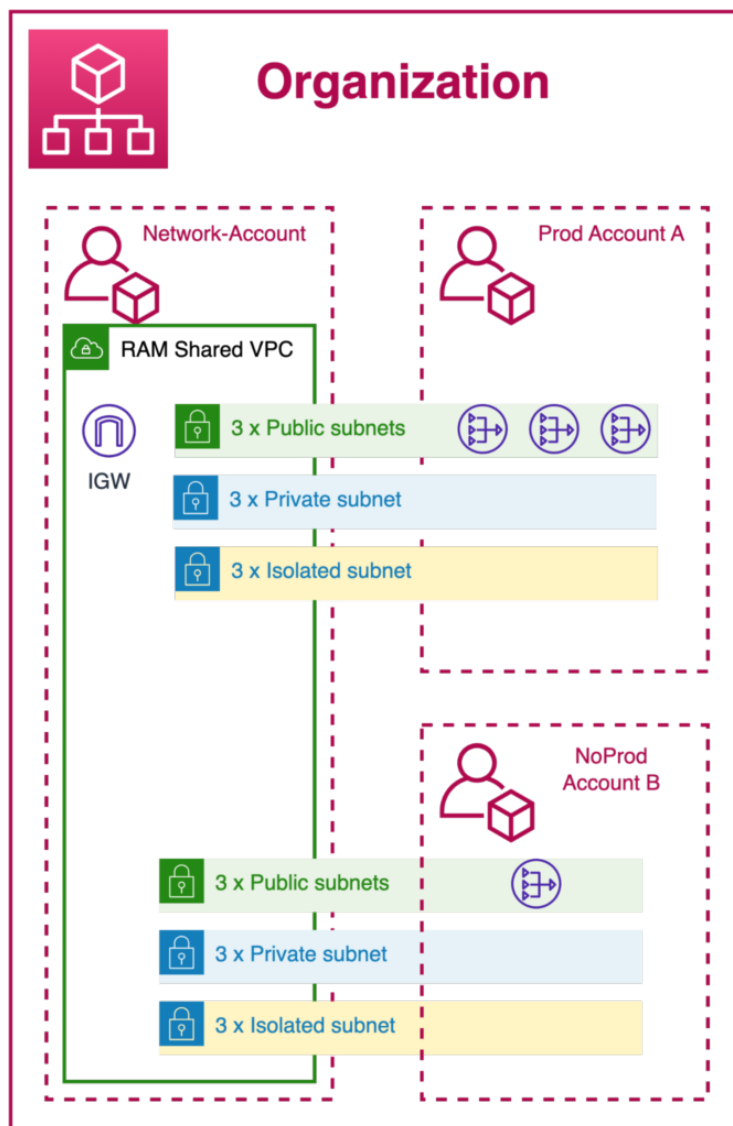
| Resource ID | Resource type | Last share date | Resource shares | Principals |
|-------------|---------------|-----------------|-----------------|------------|
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/06/10      | 1               | 1          |
|             | ec2:Subnet    | 2023/05/09      | 1               | 1          |

For everything related to sharing, we will use the AWS RAM service. It also allows us to configure certain parameters to differentiate how our resources will be shared, and therefore used.

If all has gone well, we should see the subnets we have just shared and their VPC in the application account.

Don't worry if the VPC will be visible, too since, anyway, each account will only have access to the subnets that you have decided to share with it.

Now that we have the subnets on the target account we can create the resources using shared networks while not being the owners of said VPC. Obviously, since we don't have the ownership of these subnets we will not be able to create routes or delete others as we are not the owners of the various routing tables. For all these activities it will be necessary to have a **network administrator with 'network-account' access**, making us also guess one of the advantages we will see later.



In the diagram, we can see an example of Organization structure. In this example, the *Network-Account* will share 3 subnets of each type with the various accounts (production and non-production). The only difference we see between the environments will be the number of Nat Gateways. For the production account, it will be necessary to have at least 2 NATs in order to maintain high availability. For the non-production account, we will only

have one so as to reduce costs.

To optimize costs further, we could consider creating 3 centralized NAT Gateways that will be used by all accounts instead of having a dedicated one for each non-production environment.

These configurations make us realize that the solution turns out to be really versatile and lends itself well for evolving along with our idea of Landing Zone and Business.

## Multi-VPC Approaches

After focusing on the use of a single VPC, it is also necessary to understand what other possibilities are made available to us by AWS.

In a multi-VPC approach, we will have at least **one VPC for each account**. Like its counterpart, it has PROs and CONs to be evaluated to decide if it fits our needs.

Multi-VPC approaches involve either TransitGateway or VPC Peering.

## Transit Gateway

In recent years, the use of the TransitGateway has become the "standard" as far as designing a landing zone is concerned.

(If you are interested in the topic, we already covered the use of AWS TransitGateway on some [preview blog posts](#))

To set up an infrastructure that leverages Transit Gateway, we will only need: a Transit Gateway, Transit Gateway Route table, and Transit Gateway Attachment.

We can imagine **TransitGateway as a virtual router that connects all the VPCs in our Landing Zone in a simple and effective way.**

Also, we will need the so-called *network account* on which the Transit Gateway will actually reside. Being a multi-VPC approach we will also need at least 2 VPCs, one in account A and one in account B (the network account can also have its own VPC). Unlike the previously proposed solution, in this case, all VPCs will be independent in both their configuration and management, while only the Transit Gateway will be reserved for network administrators.

Remember that since we will have to use numerous VPCs we **will have to avoid as much as possible using CIDRs that overlap each other and, above all, that overlap public internet addresses.**

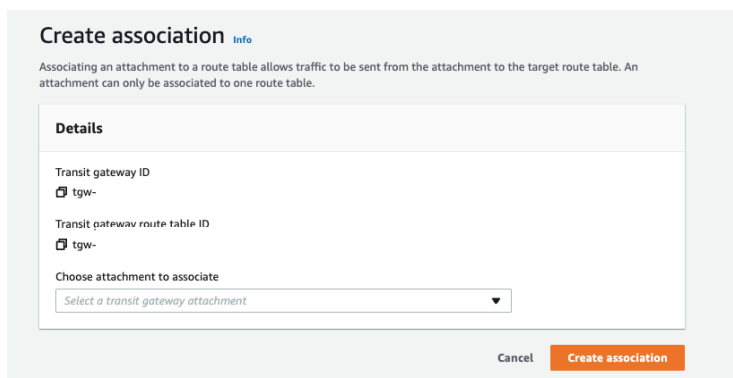
**To attach one of our VPCs to the TransitGateway we will have to create a Transit Gateway Attachment** whether this VPC is in the network account or in one of the other accounts in the landing zone. Again we will make use of the AWS RAM service so that we can share our newly created TransitGateway with one or more accounts in our LandingZone (each account will have its own attachment). Once we have accepted the sharing we can proceed, from our account A, with the creation of our TransitGateway attachment and then connect it to our VPC. By doing so, we create a "virtual" link between our VPC and the centralized TransitGateway.

If you wish to use Infrastructure-as-Code (IaC), you can automate these steps directly in CloudFormation.

In the current state, we would have one or more VPCs that are afferent to the same Transit Gateway but without the possibility of communicating with each other.

Like any router, to manage the path of our network packets, we will need to configure the routes, which in our case reside in the Transit Gateway Route Tables. In general, we could have a routing table for each attachment so that we can better divide the various routes, and segregate the VPCs of specific and particularly critical accounts.

To attach a Route Table to an Attachment, simply create an association directly from the AWS console, specifying the required IDs.



The screenshot shows the 'Create association' dialog box in the AWS console. At the top, it says 'Create association' with an 'Info' link. Below this is a descriptive sentence: 'Associating an attachment to a route table allows traffic to be sent from the attachment to the target route table. An attachment can only be associated to one route table.' The main area is titled 'Details' and contains three fields: 'Transit gateway ID' with a search icon and 'tgw-' text, 'Transit gateway route table ID' with a search icon and 'tgw-' text, and 'Choose attachment to associate' with a dropdown menu showing 'Select a transit gateway attachment'. At the bottom right, there are 'Cancel' and 'Create association' buttons.

## Route Segregation

Unfortunately, like any other multi-VPC solution, this one will require a significant amount of management effort, too to ensure that with the proliferation of VPCs, a good level of security isolation is still ensured. It's important to remember that while we must maintain a certain level of security, we also need to allow our applications to run optimally without impacting their normal operational flows.

Much of the effort will need to be invested in the precise and punctual configuration of routes in each individual Routing Table. Effective management of all VPCs associated with

our Transit Gateway will protect us even if one of our AWS instances is compromised by malicious actors.

## VPC Peering

The approach that best contrasts with the Transit Gateway is VPC Peering.

This is nothing more than a virtual "bridge" created to connect two different VPCs, whether in one account or different accounts. Usually, this approach is good when you have few accounts and do not want to invest too much effort in network management.

By linking VPCs, we will also be able to use security groups that do not belong to the VPC we own, helping us make our infrastructure more secure with ad hoc and well-targeted rules. Remember that if we reference a security group from another VPC, it will not appear as a suggestion directly from the console, but we will have to enter it manually, and if we are cross-region, it is not possible to reference security groups from different VPCs.

One important thing to remember when using VPC peering is that it is not transitive. If we enable the peering between VPC-A and VPC-B and then between VPC-B and VPC-C, VPC-B will be able to communicate with all the other VPCs, but VPC-A and VPC-C will not be able to communicate with each other since VPC-B cannot rotate traffic to the "next hop." This immediately makes us realize that the number of peering we must create will grow exponentially as we create new VPCs.

## Conclusion

As we often hear, the answer to which approach is best suited to our needs is "It depends."

It all depends on what architecture we plan to adopt, how large our organization will be, and - most importantly - how much effort we will want to put into network management alone.

Always remember that a good infrastructure rests its foundation on a well-structured network that will be with us throughout the growth of the organization.

To better understand the advantages and disadvantages of the various possibilities introduced in this article, we will go into detail about their use in a dedicated blog post. We will take some of the most common use cases as an example, outlining a map of suitable solutions depending on the case.

So follow us so you don't miss the second part of our journey into centralizing networking in multi-account Cloud environments on AWS!

---

## About Proud2beCloud

**Proud2beCloud** is a blog by **beSharp**, an Italian APN Premier Consulting Partner expert in designing, implementing, and managing complex Cloud infrastructures and advanced services on AWS. Before being writers, we are Cloud Experts working daily with AWS services since 2007. We are hungry readers, innovative builders, and gem-seekers. On Proud2beCloud, we regularly share our best AWS pro tips, configuration insights, in-depth news, tips&tricks, how-tos, and many other resources. Take part in the discussion!

---



### Riccardo Fragnelli

DevOps @ beSharpI was born on-prem as a Dev before landing on the “Cloud side of IT”. With AWS I discovered a whole new branch of IT that fascinates me more and more; I’m always ready for the next big thing! I’m the fussiest man I know on earth and quite lazy. I like spending my free time jumping between video games and RPGs.

---