**01. Introduction to Ingress Groups**

--- in this section, we are going to learn about ingress groups and also implement a practical

demo for ingress groups. So, without ingress groups, we used to define our configurations in a single ingress resource.

**Without ingress groups**



--- So, let's understand what is the drawbacks of not using ingress groups in a larger configuration.

So, example, like in our current use cases, we have app1, app2 and app3 applications and we have their only deployment and then node port service Kubernetes Manifest.

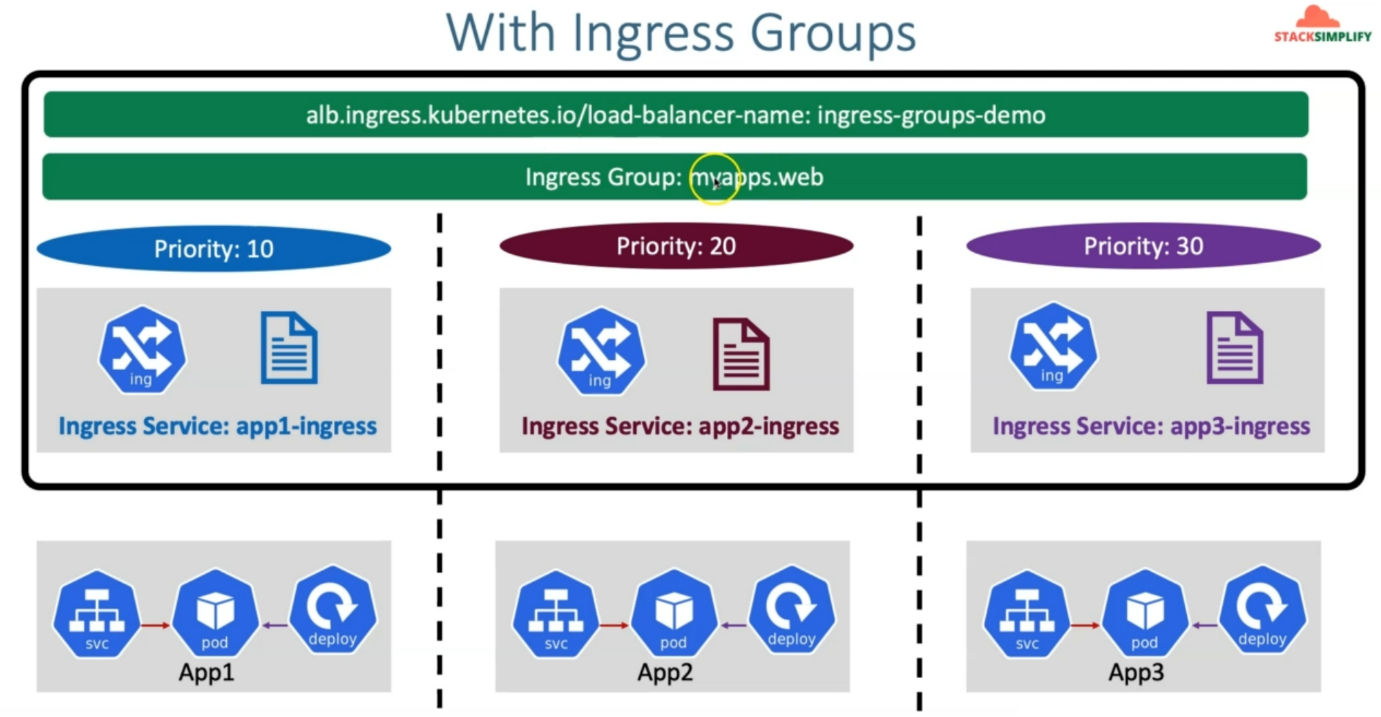
--- for all these three apps, we used to have a single ingress resource. Maybe my-app is the ingress resource name and for these three apps, the routing rules we have defined in a single file, so single ingress resource for all three apps, we have created so far and any changes to any of these apps, you need to go into that respective ingress resource,

--- for all the three apps, you have only one ingress resource and you need to go inside that ingress resource, update that and then deploy those changes.

--- the important thing here is If you have three apps, it is fine, but if we have 50 apps or 100 apps and all these 100 apps needs different rules embedded in your application load balancer and those rules need to be managed in a single ingress Kubernetes manifest.

--- So, if that is the case, then maintaining that single file becomes tedious and also confusing. So, the Ingress Group's concept helps us in segregating the Conflict in different files with the different ingress resource names, but you can merge them together to create a single application load balancer.

**With ingress groups**



--- So, with ingress groups, we can simplify our Kubernetes ingress manifest when dealing with multiple apps requiring a single application load balancer.

--- So, with the ingress groups, what happens is so you can see you have app1, app2 and app3 related deployment and nodeport service-related Kubernetes manifest.

--- So, for **app1**, you will create the **app1-ingress** resource and you define the **app1** related rules

only inside of it and for **app2**, you will define the **app2-ingress** resource, and you will define the app2 related rules inside that and for **app3**, you will do the same thing.

--- So, these three are, three different Kubernetes manifests and all together Using the ingress group concept and using the Ingress Group annotation.

--- you will group them together with **myapps.web** whatever the name you want. So, this ingress resources from myapps.web or whatever it is. I just give the name as myapps.web for my group.

--- which means I'm going to have an Ingress group annotation in Ingress Service app1-ingress, app2-ingress and app3-ingress.and I'm going to say all these three ingresses belong to **myapps.web** Ingress Group.

--- in addition to that, I am also going to define the priority for this ingress rules. I'm going to say app1-ingress is going to have Priority 10, app2-ingress is going to have Priority 20 and app3-ingress is going to have Priority 30.

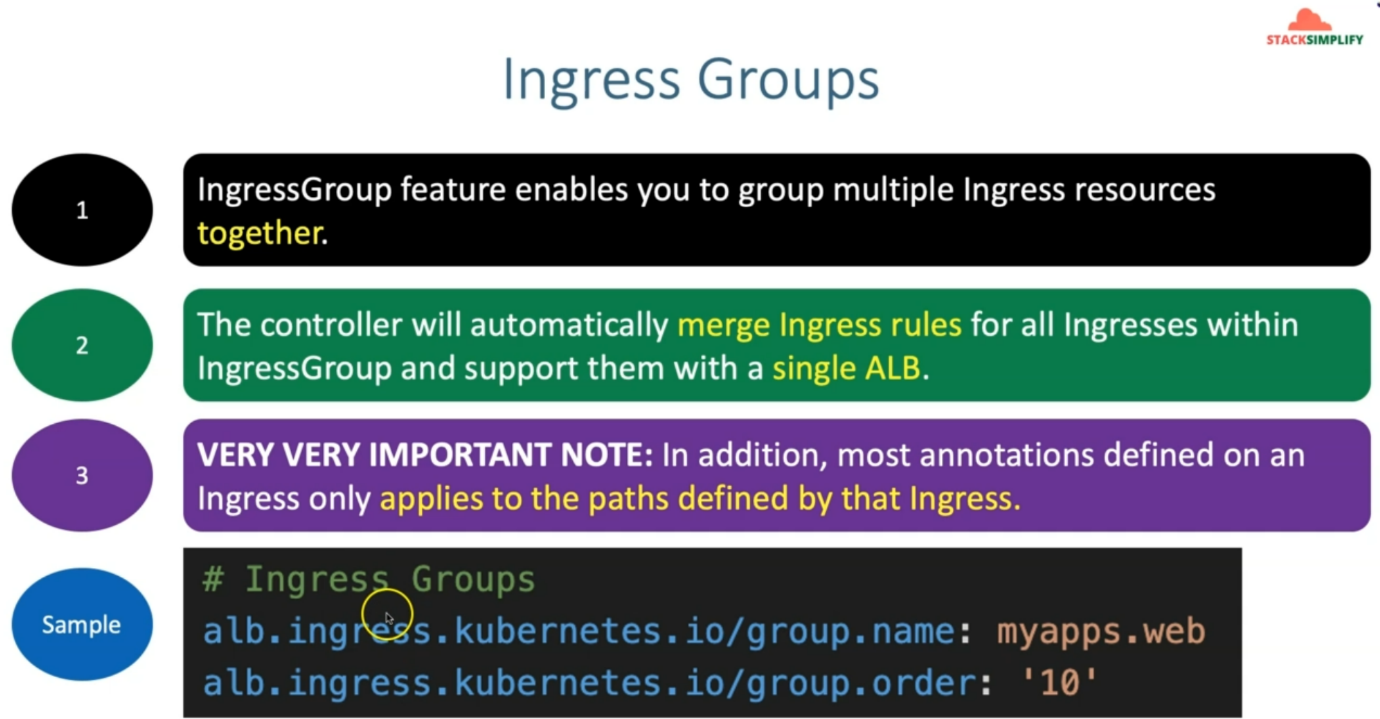
--- at this point, what happens is, in the application load balancer. it is going to Put the rules in that same order.

--- in addition to that. So, you are calling from application load balancer perspective this as a priority, but from ingress perspective, we call it as ingress group order.

--- for all these three things is going to be same as Ingress Group's demo. So, which means this load balancer name related annotation will be there in all three files and you will have the names as ingress groups demo but the name of your ingress service right, in the metadata, you provide the name right.

--- So that can be **app1-ingress**, **app2-ingress** and **app3-ingress**, whatever based on your ingress service name but all these things will be connected to a single load balancer, which is **Ingress Group's demo** and all these three things will have the same ingress group, which is **myapps.web**.

**Ingress groups**

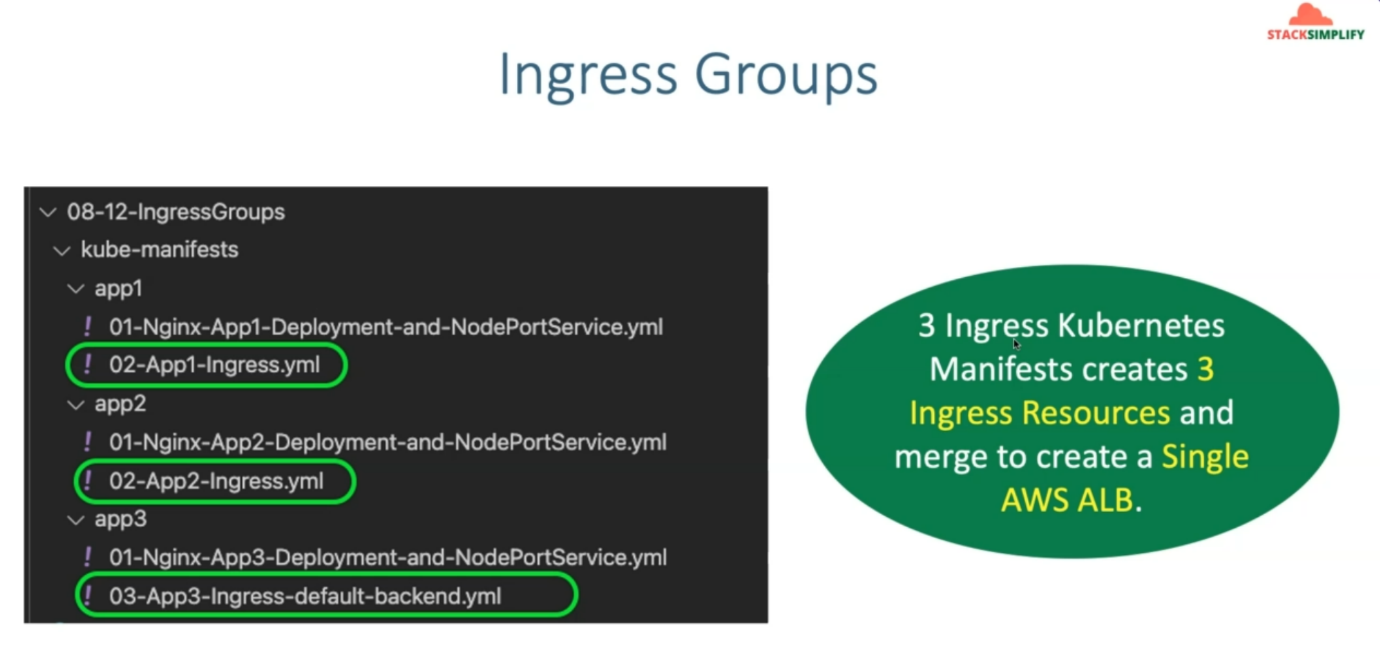


--- So, let's understand a little bit theory, this group feature enables you to group multiple ingress resources together. the controller will automatically merge ingress rules for all ingress within Ingress Group and support them with their single application load balancer.

--- very important note here is, in addition, most annotations defined on an ingress only applies to the paths defined by that ingress.

--- so, this is the sample. for the ingress groups, you are going to use the group name. in every ingress resource. You are going to add these two annotations and provide that ingress resource belongs to which group and also what is the priority of that ingress resource.

--- So, in our case, in 08 ingress group section will include Manifest. we are going to have app1 folder, app3 folder and app3 folder.



--- you will find above files in those folders.

--- So, three Ingress Kubernetes Manifest creates three ingress resources in Kubernetes, and all three will be merged to create a single AWS application load balancer.