L2 LoadBalancer

There are three frontends nodes and one worker in the cluster.

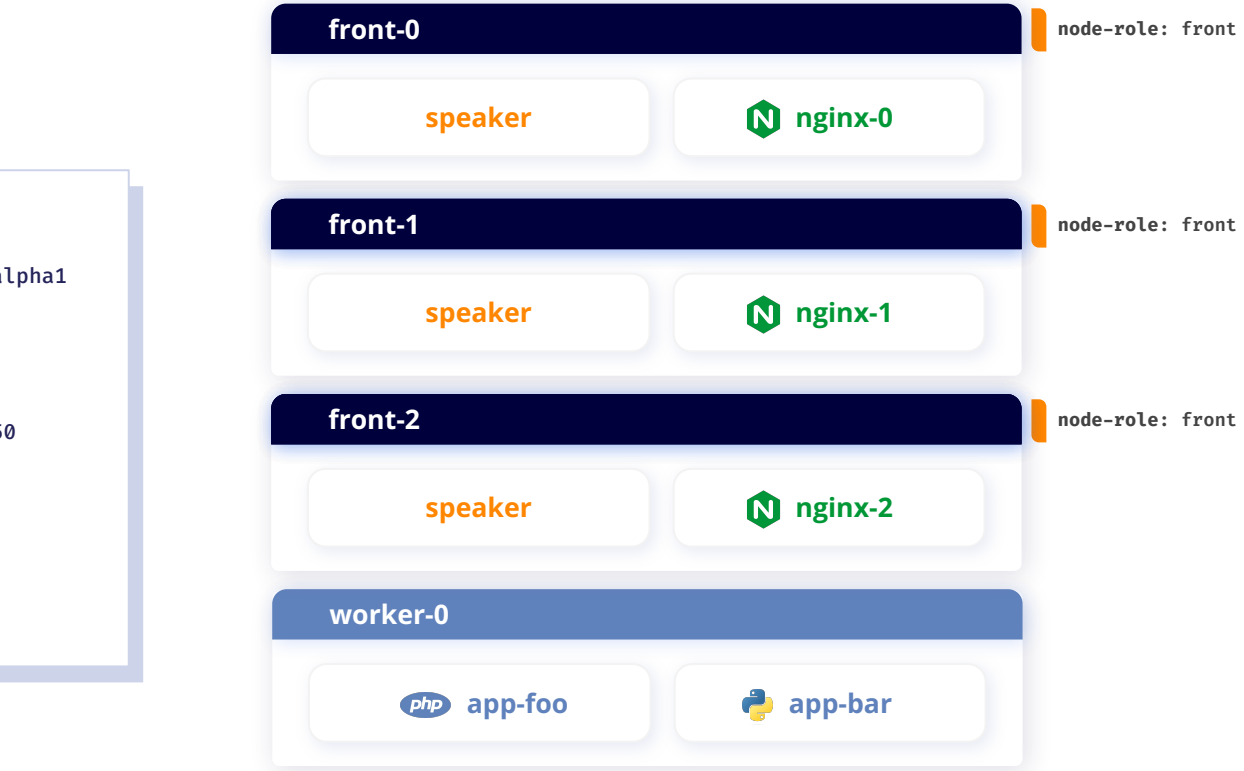
L2 LoadBalancer

```yaml
apiVersion: deckhouse.io/v1alpha1
kind: ModuleConfig
metadata:
  name: l2-load-balancer
spec:
  enabled: true
  version: 1
```

We turn on **MetalLB** module: ModuleConfig version 2.

L2 LoadBalancer

```yaml
apiVersion: network.deckhouse.io/v1alpha1
kind: MetalLoadBalancerClass
metadata:
  name: front
spec:
  addressPool:
    - 192.168.122.100-192.168.122.150
  isDefault: false
  nodeSelector:
    node-role: front
  type: L2
```



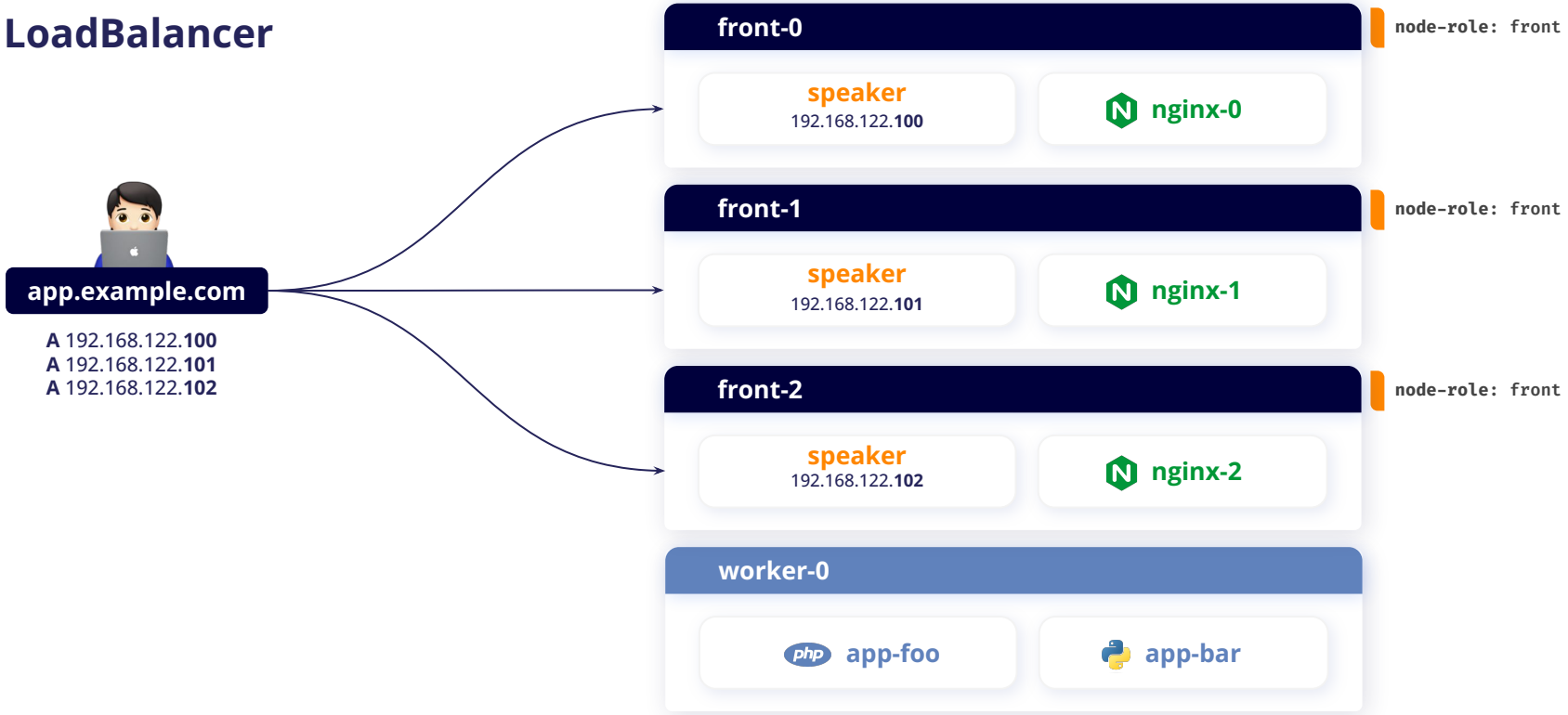
An **MetalLoadBalancerClass** resource has been created specifying front-end nodes and a pool of "public" IP addresses. This allows for easily creating "zones" by associating specific address pools with a group of nodes. Speakers are run on all front-end nodes.

L2 LoadBalancer

```yaml
apiVersion: v1
kind: Service
metadata:
  name: nginx-deployment
  annotations:
    network.deckhouse.io/l2-load-balancer-name: ingress
    network.deckhouse.io/l2-load-balancer-external-ips-count: "3"
spec:
  ports:
  - port: 80
    protocol: TCP
    targetPort: 80
  selector:
    app: nginx
  type: LoadBalancer
  LoadBalancerClass: front
```

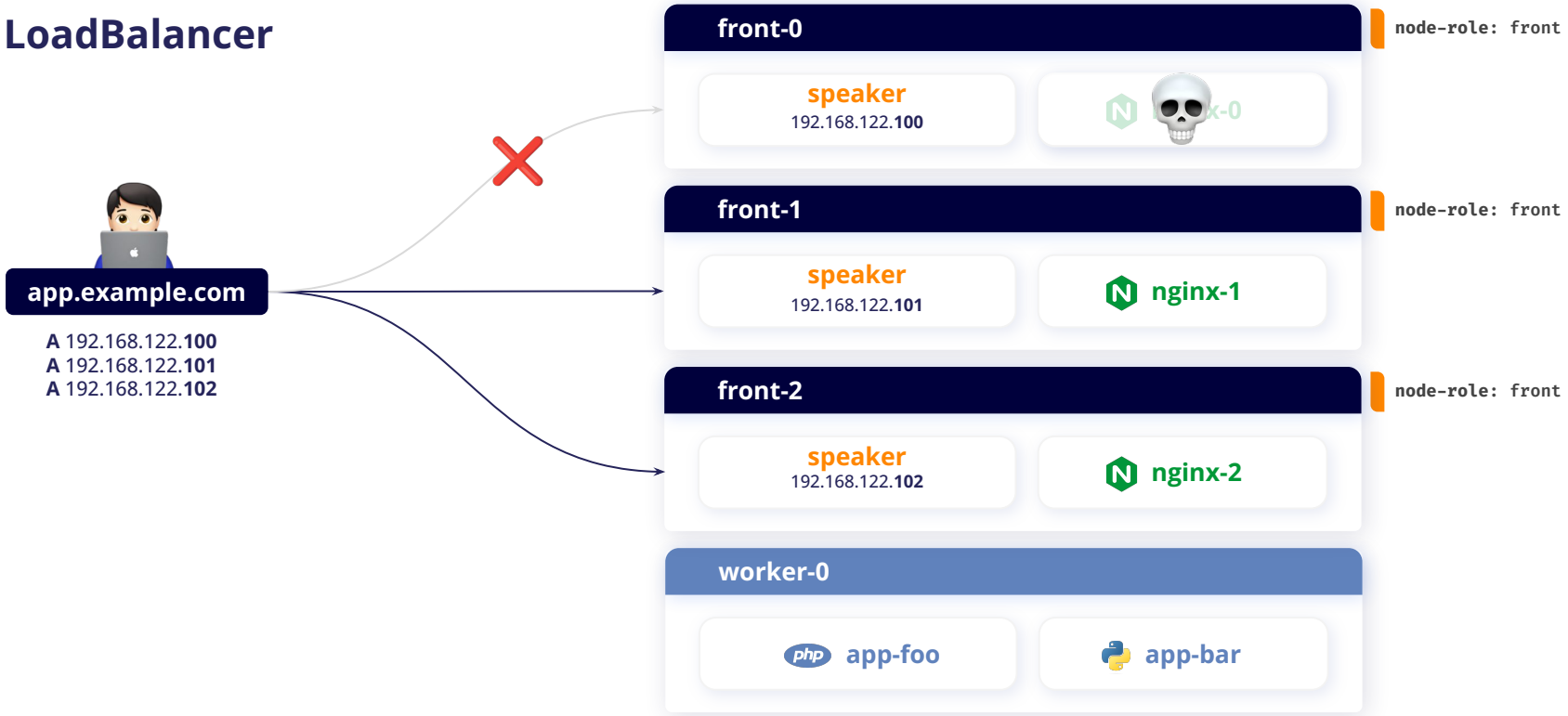
An **Service** resource with type **LoadBalancer** has been created. It contains the LoadBalancerClass name, special annotation with the required number of IP addresses.
Speakers are launched on all frontend nodes, each obtaining one or more addresses from the pool.

L2 LoadBalancer



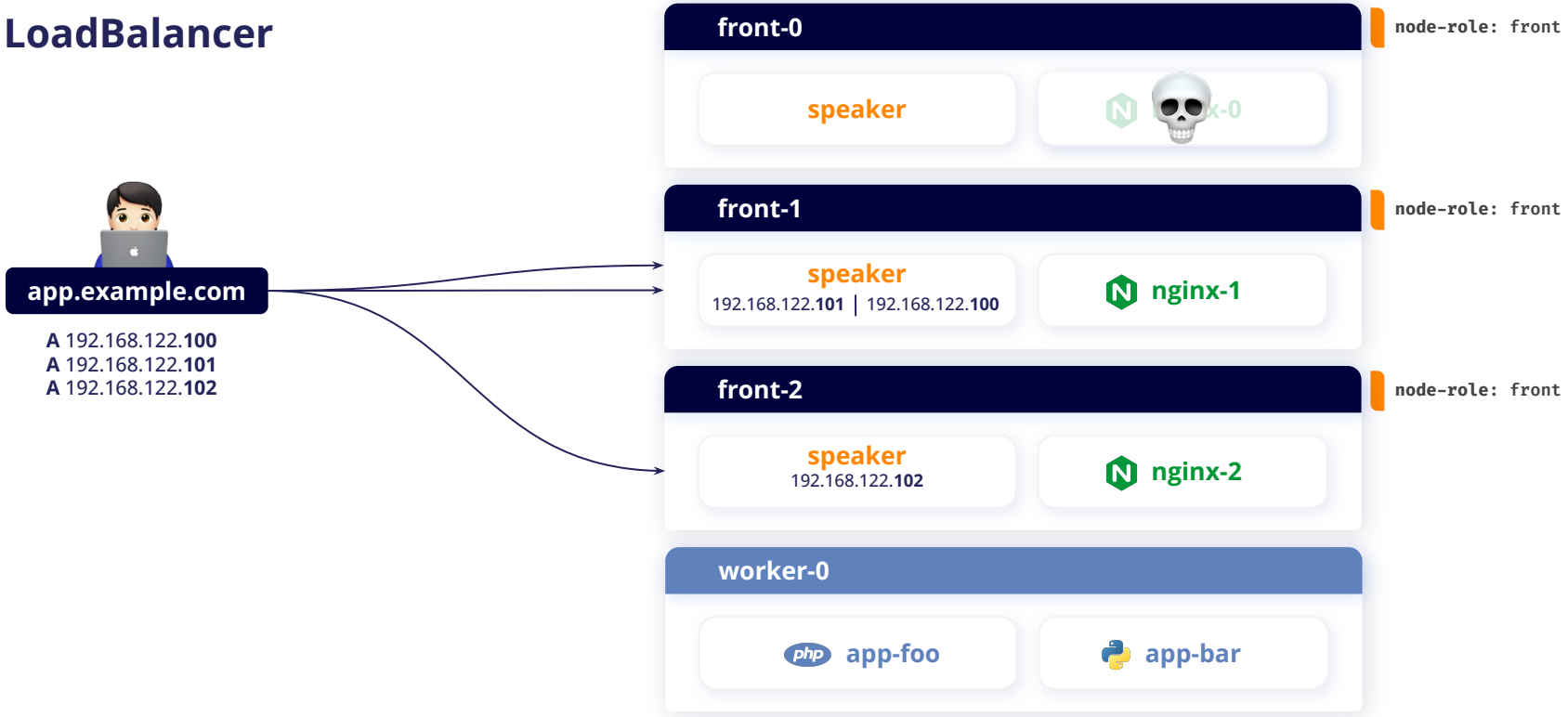
Each front-end node participates in handling application requests.
For this, three A records are specified in the public DNS name of the application.

L2 LoadBalancer



In the event of a failure of the nginx application on one of the front-end nodes or the node itself, a third of the requests will fail,...

L2 LoadBalancer



...and one of the remaining front-end nodes will take over the "problematic" IP address and handle the incoming application requests.