
AWS
re:Invent

CON307

Amazon EKS: To ~~40,000~~ 100,000 CPUs and beyond!

Paul Lewis

Principal Solutions Architect, Container Technologies

AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Objectives

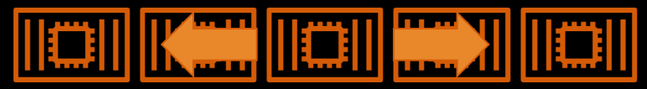
Run batch processing job, at massive scale

- 30,000+ pods
- 2 – 4 CPUs per pod — 120,000 CPUs
- 7 – 15 GB RAM per pod — 450 TB RAM
- Single cluster per job
- Run multiple copies of the job in parallel



Challenges





dropped packets

control plane performance

IP exhaustion

core service issues

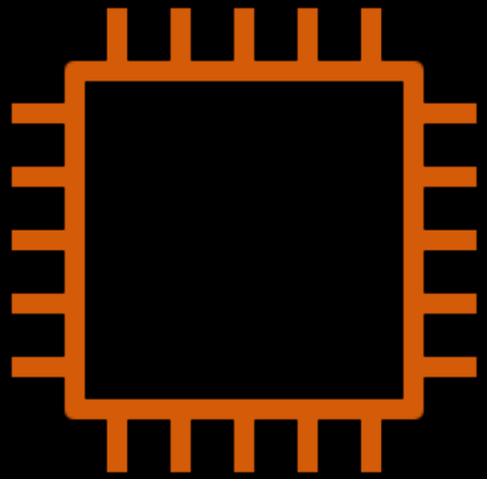
AWS API limits

Scaling to 10,000 pods



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



IP Address 1
IP Address 2
IP Address 3
…
IP Address *n*

IP Address 1
IP Address 2
IP Address 3
…
IP Address *n*

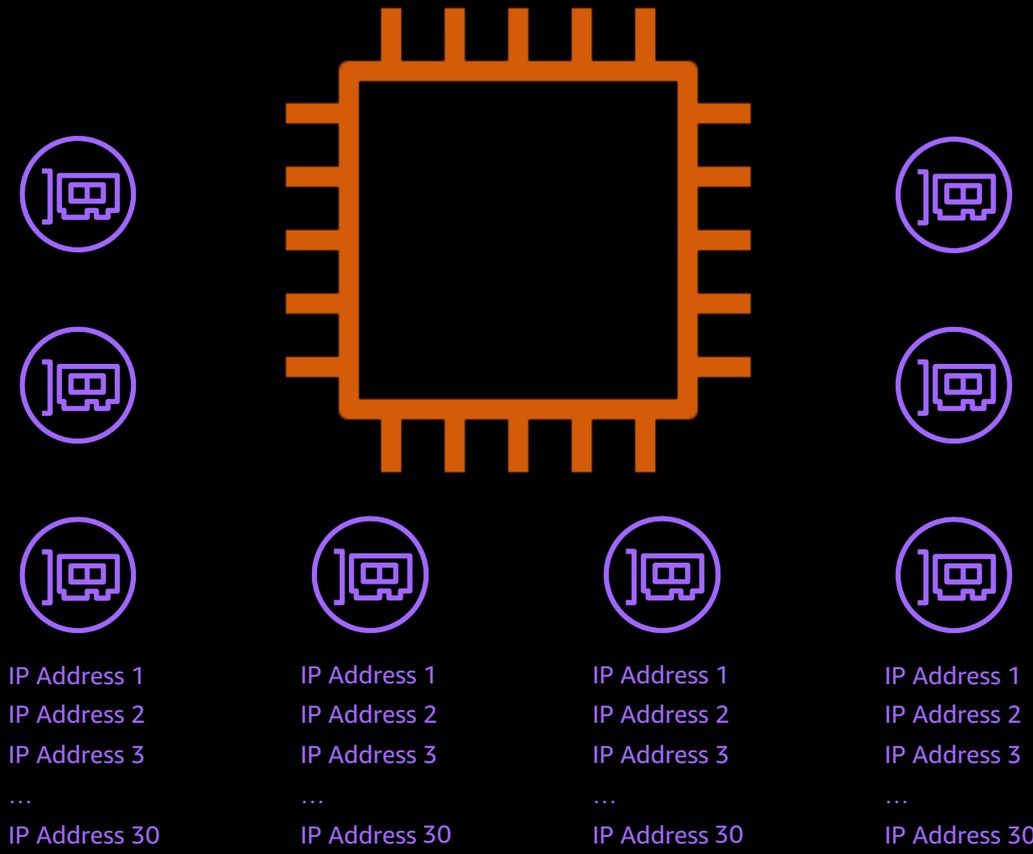
Max pods = (Number of network interfaces for the instance type × (the number of IP addresses per network interface - 1)) + 2

m5.4xlarge

8 ENIs

30 IPs / ENI

234 pods / node (max)



Tuning the Amazon CNI Plugin

Variable	Usage	Example Value	No. of ENIs / IPs
WARM_ENI_TARGET	default	1	2 / 60

Assumptions:

1st IP per ENI for worker node

2 daemonsets running in cluster

no. of IPs is before workloads are running

https://docs.aws.amazon.com/eks/latest/userguide/cni-env-vars.html

Tuning the Amazon CNI Plugin

Variable	Usage	Example Value	No. of ENIs / IPs
WARM_ENI_TARGET	default	1	2 / 60
WARM_IP_TARGET	dynamic scaling	10	1 / 13

Assumptions:

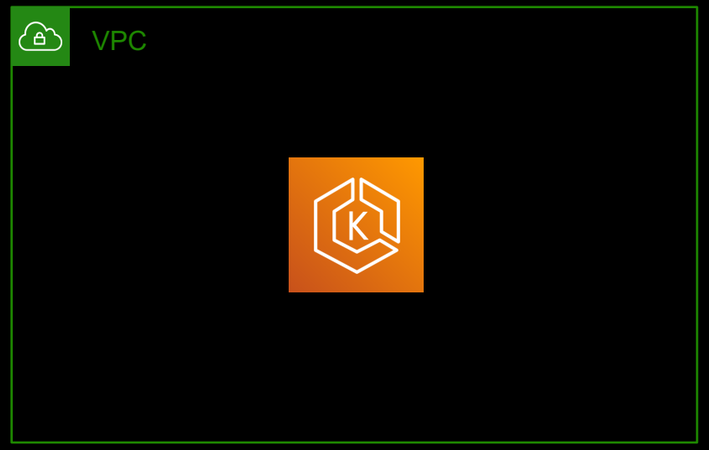
1st IP per ENI for worker node

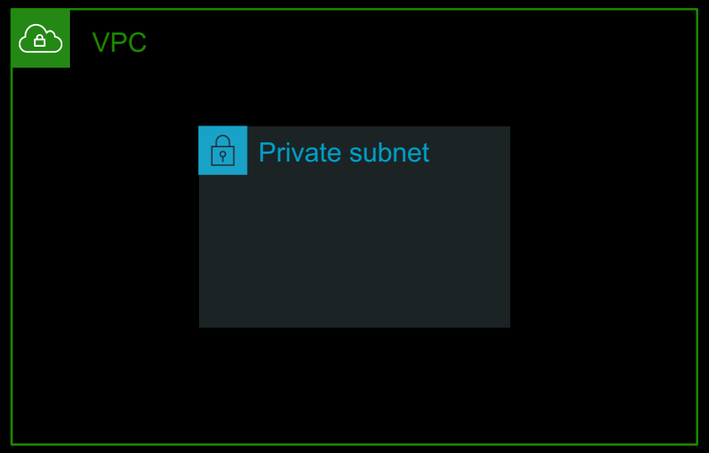
2 daemonsets running in cluster

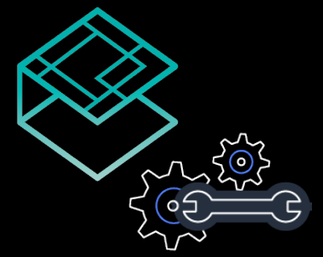
no. of IPs is before workloads are running

https://docs.aws.amazon.com/eks/latest/userguide/cni-env-vars.html

Preventing IP Address Exhaustion







One EKS cluster per VPC

Single consumer of IP addresses

Large VPC, large subnets

/16 VPC => 65,536 IP addresses

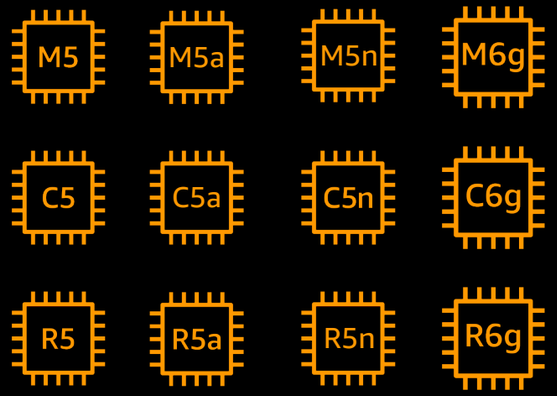
/18, /20, /22 subnet per AZ =>
64,464 IP addresses for pods

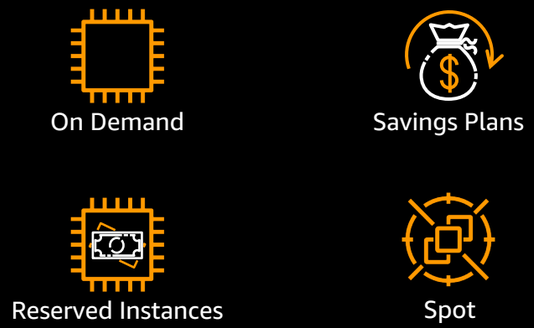
CNI Configuration

Reduce IP address consumption

EKS Deployment Recommendations







One EKS cluster per AWS Account

Single consumer of account-level API limits

Choose your instance types

CA assumes homogeneous node pools

Use heterogeneous node pools for diversification

Choose your pricing models

Blend pricing models to optimize costs

For Spot: diversify, diversify, diversify!

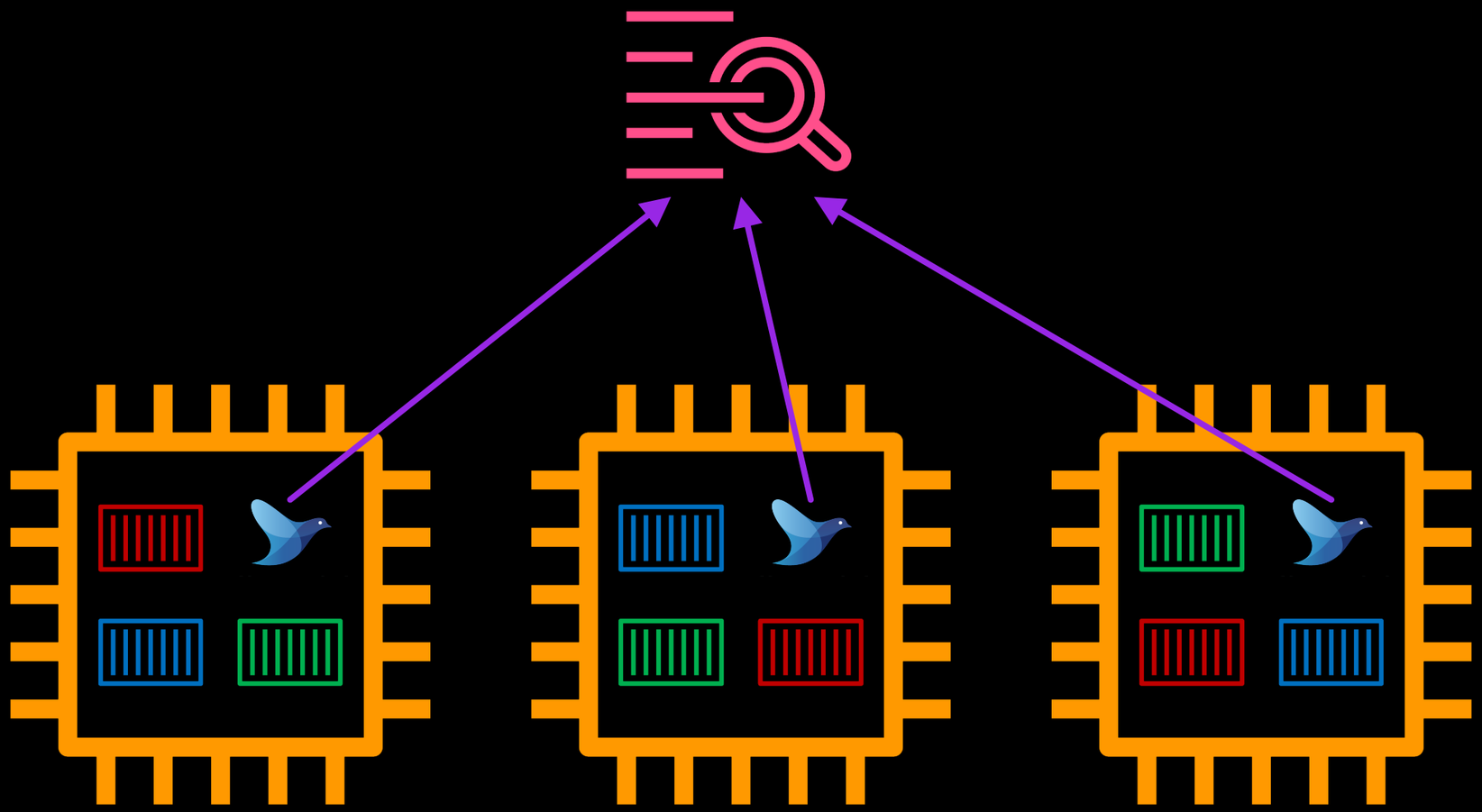
CMP402 - Scaling Kubernetes for less with Amazon EC2 Spot Instances

Scaling to 20,000 pods



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Centralized logging agents



Default fluentd log output

DON’T ATTEMPT TO READ! ☺


```
{
    "log": "2020-11-04 15:37:31 +0000 [info]: #0 [filter_kube_metadata_fluentd] stats namespace_cache_size: 1, pod_cache_size: 1, pod_cache_watch_delete_ignored: 27,
pod_cache_watch_misses: 84, pod_cache_watch_ignored: 1, namespace_cache_api_updates: 1, pod_cache_api_updates: 1, id_cache_miss: 1\n",
    "stream": "stdout",
    "docker": {
        "container_id": "0e69c2a63931035ece82e9fd359ae41a7ff7add1fafaa54940ebf6cea5806dd2"
    },
    "kubernetes": {
        "container_name": "fluentd-cloudwatch",
        "namespace_name": "amazon-cloudwatch",
        "pod_name": "fluentd-cloudwatch-tjc4g",
        "container_image": "fluent/fluentd-kubernetes-daemonset:v1.7.3-debian-cloudwatch-1.0",
        "container_image_id": "docker-pullable://fluent/fluentd-kubernetes-daemonset@sha256:9b8b2f99ea884853205150364eceaac9fff5ea97fc3300cc0080f48c3eac8b8a",
        "pod_id": "463e5f1d-9b2d-4064-89e1-21df6bd8f6d7",
        "host": "ip-192-168-79-253.eu-west-1.compute.internal",
        "labels": {
            "controller-revision-hash": "78b6469d9d",
            "k8s-app": "fluentd-cloudwatch",
            "pod-template-generation": "1"
        },
        "master_url": "https://10.100.0.1:443/api",
        "namespace_id": "6c6ee5b9-6750-47a8-b69b-9cc83da1144a",
        "namespace_labels": {
            "name": "amazon-cloudwatch"
        }
    }
}
```

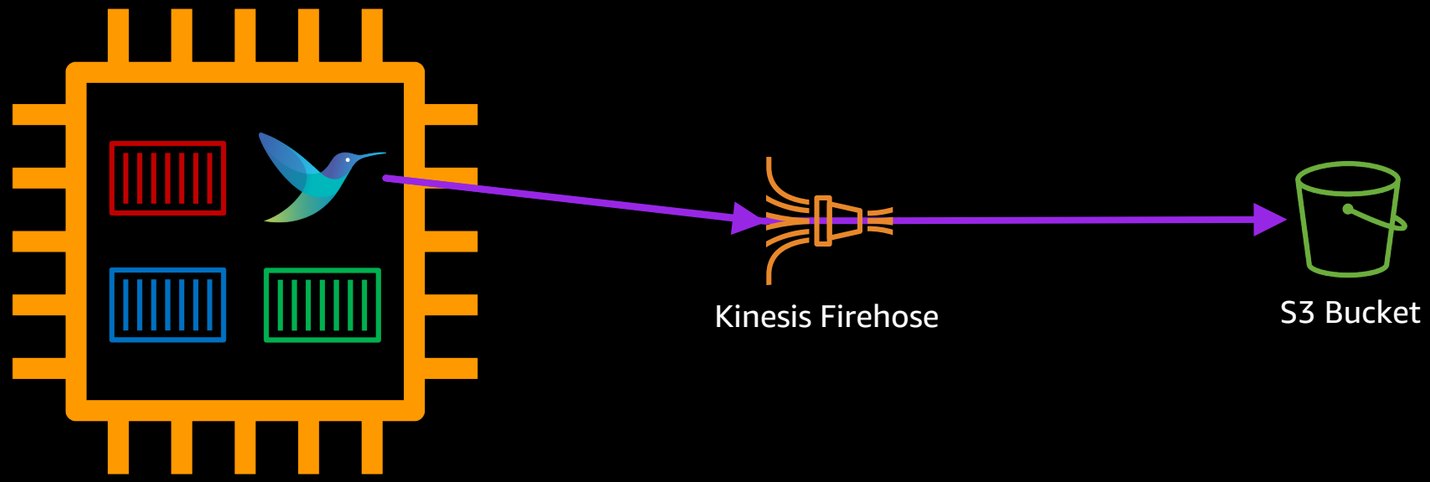
Tuning fluentd configuration

```
<filter kubernetes.var.log.containers.**.log>
  @type kubernetes_metadata
  @id filter_kube_metadata_application
  @log_level "warn"
  skip_labels true
  skip_container_metadata true
  skip_master_url true
  skip_namespace_metadata true
  cache_size 30000
</filter>
```

Tuned fluentd log output


```
{
    "log": "2020-11-04 15:58:53 +0000 [info]: #0 [filter_kube_metadata_fluentd] stats - namespace_cache_size: 0, pod_cache_size: 1, namespace_cache_miss: 12,
pod_cache_api_updates: 1, id_cache_miss: 1\n",
    "stream": "stdout",
    "docker": {
        "container_id": "b2ae57983b3851c3dc8ffec1eed484a1d1214e0efa98f5a5f70f58f228c17d43"
    },
    "kubernetes": {
        "container_name": "fluentd-cloudwatch",
        "namespace_name": "amazon-cloudwatch",
        "pod_name": "fluentd-cloudwatch-qlk78",
        "pod_id": "52482c0f-63f7-4548-a1b7-0431c1b6849f",
        "host": "ip-192-168-174-129.eu-west-1.compute.internal"
    }
}
```

Fluentd → Fluent Bit

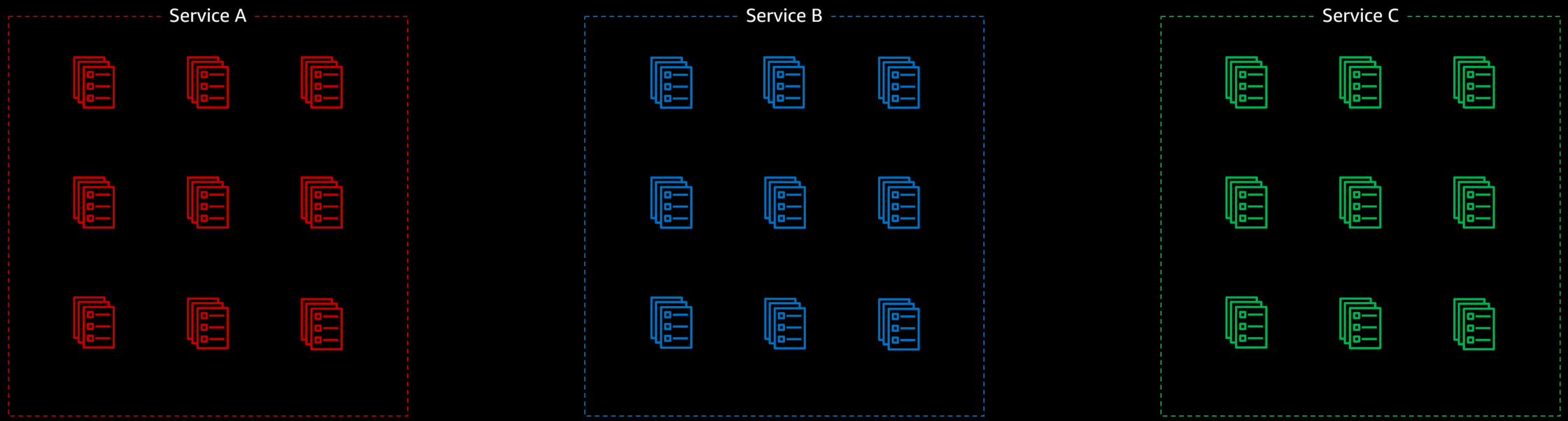


Scaling to 30,000 pods



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Service A Endpoint	Service B Endpoint	Service C Endpoint
192.168.176.45	192.168.177.106	192.168.166.144
192.168.136.173	192.168.136.36	192.168.163.82
192.168.138.167	192.168.161.166	192.168.170.86
192.168.140.56	192.168.178.230	192.168.151.150
192.168.158.186	192.168.185.184	192.168.167.120
192.168.128.186	192.168.176.48	192.168.147.252
192.168.141.182	192.168.156.138	192.168.155.108
192.168.187.139	192.168.190.205	192.168.162.31

Endpoints object



Simple design from the early days of k8s

List of IPs & Ports

Scales linearly with # pods



As # pods → 5,000

Endpoint object → 1.5MB

1.5MB = etcd object limit



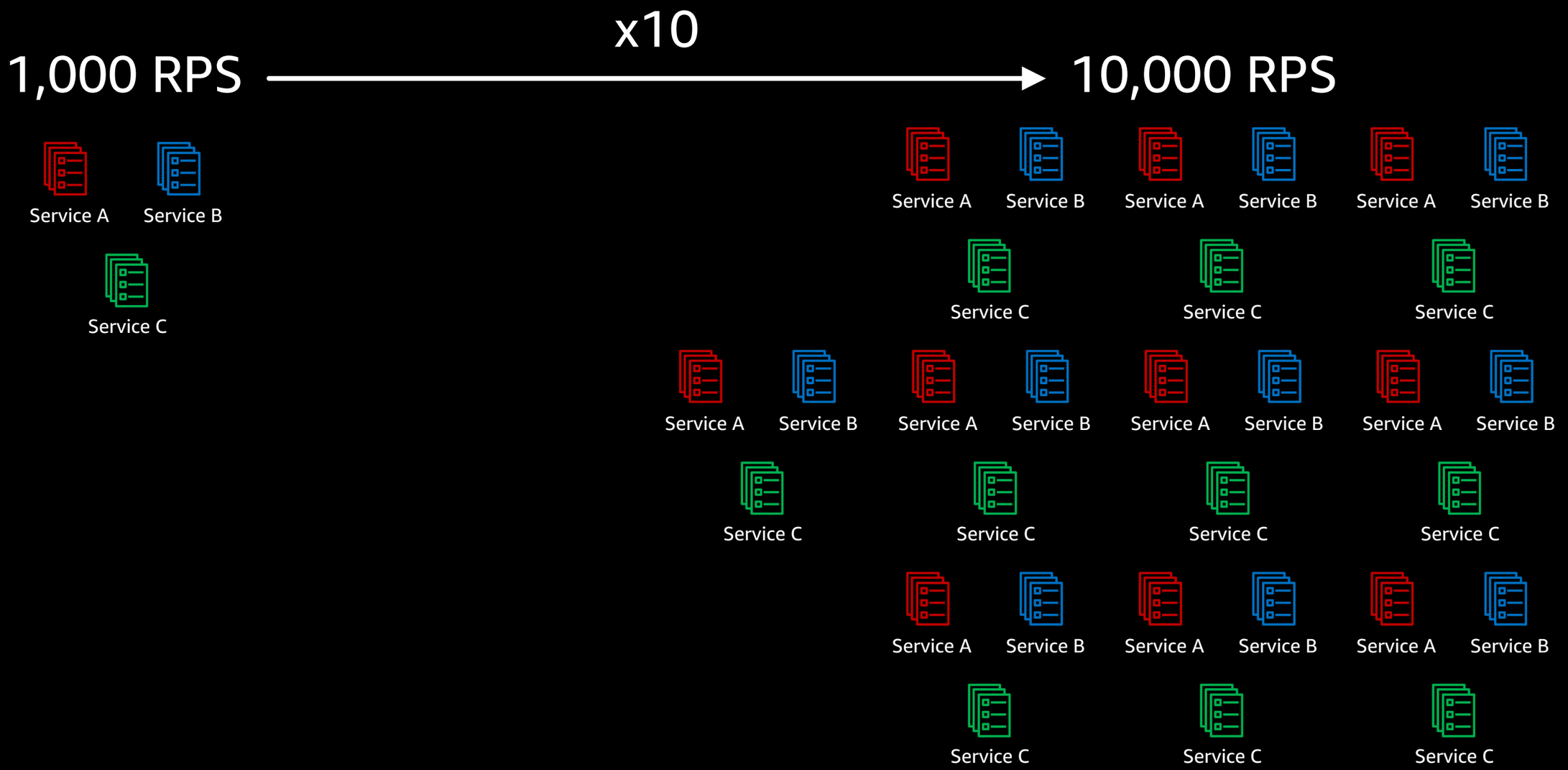
Scalability SIG

Performance degrades in services with >250 pods

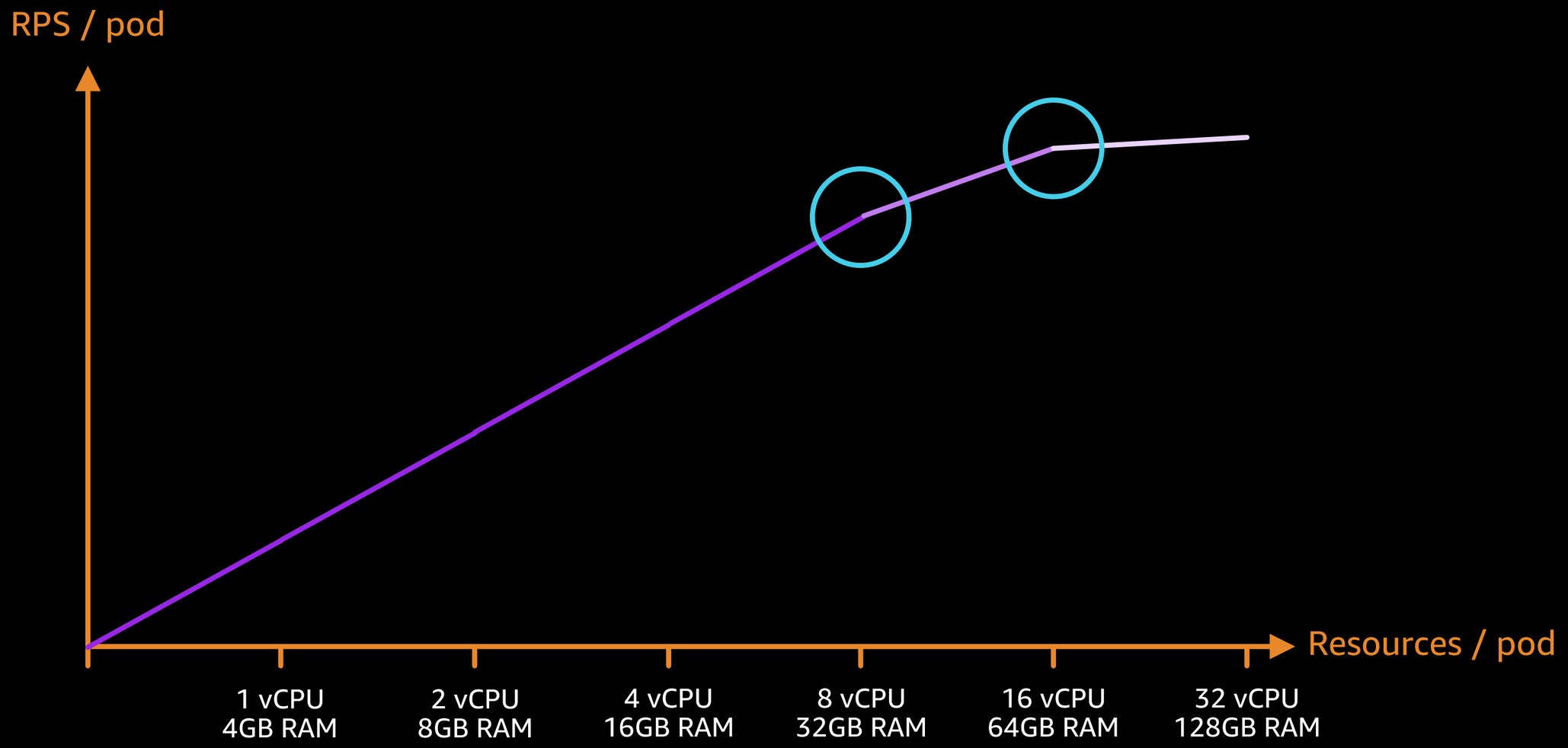
EndpointSlices to the rescue!

- EndpointSlices will automatically chunk endpoint objects to 100 objects or less
  - In beta since 1.17
  - Enabled by default since 1.19
- Until then, if you're running > 250 pods per service:
  - Ensure you have appropriate rate throttling controls in front of your services
  - Consider whether you can vertically scale your pods to handle more requests
  - Consider whether you can further decompose your microservices

Testing at scale

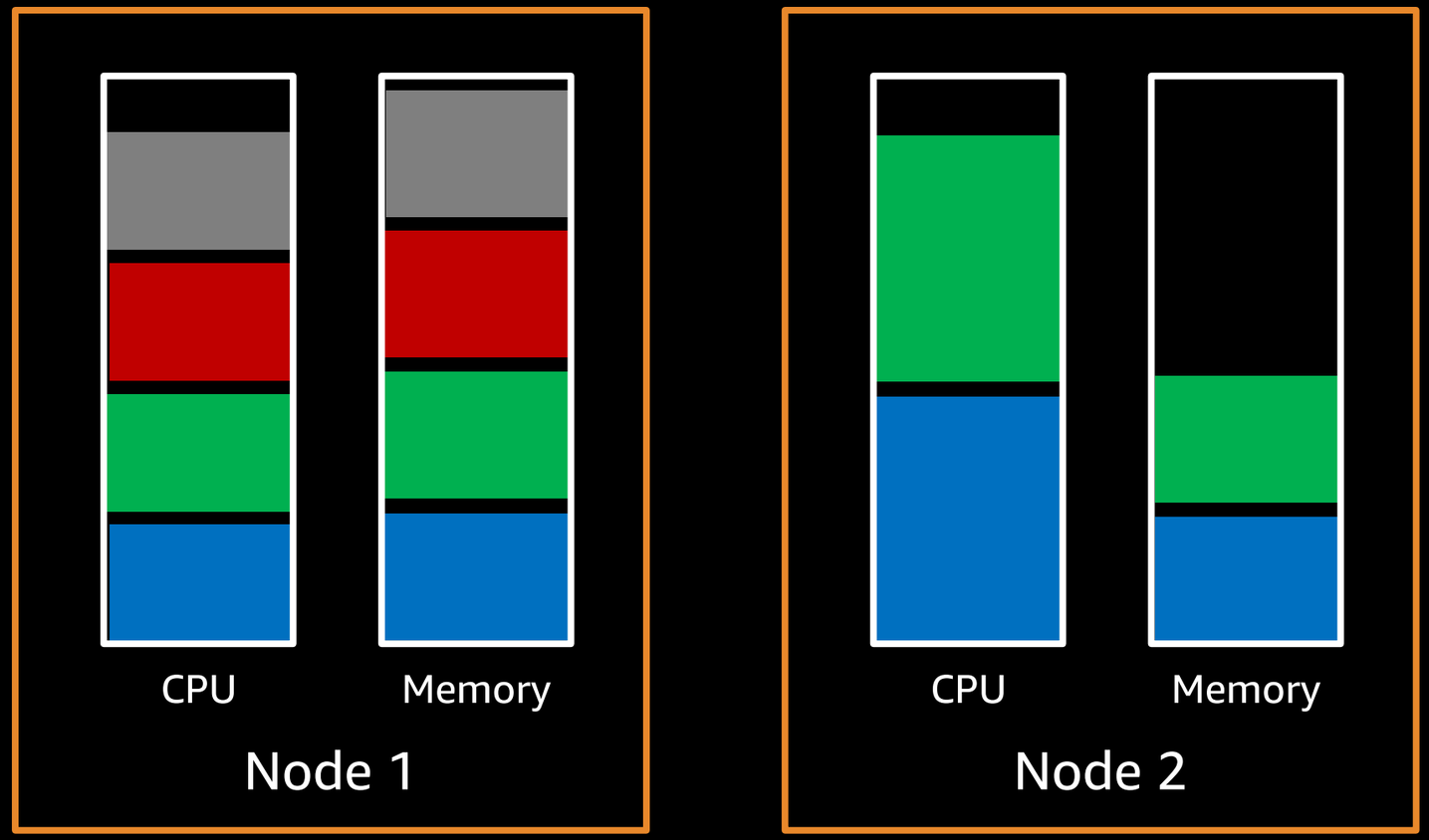


Finding the limits of your containers



Setting your container resources

```
resources:
    limits:
      cpu: 1
      memory: 128Mi
    requests:
      cpu: 250m
      memory: 128Mi
```



HPA considerations

The controller calculates the utilization value as a percentage of the equivalent resource request on the containers in each pod

```
resources:
    limits:
      cpu: 8
      memory: 128Mi
    requests:
      cpu: 8
      memory: 128Mi
```

Summary

- Deploy your EKS clusters with plenty of IP address capacity
- Tune the Amazon VPC CNI plugin if necessary
- Be mindful of how AWS API limits can impact cluster functionality
- Select your worker node instance types and pricing model carefully
- Tune your logging daemon configuration to minimise API calls
- Switch your logging agent if necessary
- Maintain < 250 pods per service if possible
- Test your pods at scale, as well as the application
- Set your resource requests and limits appropriately

Get in touch!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.