kubernetes-faq

This is a random collection of questions and answers I've collected about running and operating Kubernetes clusters (primarily on AWS). New questions and answers are welcome.

#Contents:

Architecture - What happens when a master fails? What happens when a worker fails? - How does DNS work in Kubernetes? - How do I build a High Availability (HA) cluster? - Can I isolate namespaces from each other?

Basic Usage Questions - Should I use Replication Controllers? - How do I determine the status of a Deployment? - How do I update all my pods if the image changed but the tag is the same? - How do I rollback a Deployment? - How do I debug a Pending pod? - How do I debug a CrashLoopBackoff pod? - How do I debug a ContainerCreating pod? - What is a DaemonSet? - What is a PetSet or StatefulSet? - What is an Ingress Controller? - Why do I see 504 errors from my Ingress during deploys? - How does a Kubernetes Service work? - How do I expose a Service to a host outside the cluster? - How do I force a pod to run on a specific node? - How do I force replicas of a pod to split across different nodes? - How can I get the host IP address from inside a pod? - How do I give individual pods DNS names - How do I access the Kubenernetes API from within a pod? - How do I get all the pods on a node? - Can pods mount NFS volumes? - Is it possible to route traffic from outside the Kubernetes cluster directly to pods? - How do I put variables into my pods? - Can I use variables or otherwise parameterize my yaml deployment files? - How do CPU and memory requests and limits work? - Why is my pod pending? - Why is my terminal screwed up? - What monitoring and metrics tools do people use for Kubernetes? - How do I configure credentials to download images from a private docker registry? - Is it possible to run docker inside a pod? - What versions of docker are supported? - Why are some of my pods in an Unknown state?

AWS Questions

- How should I install Kubernetes on AWS?
- How does the default Kubernetes AWS networking work?
- How do I add a node to my AWS Kubernetes cluster?
- How do I set up node auto-scaling
- How do you make a Service create a private ELB in AWS instead of the default public one?
- How do you restrict an AWS ELB to certain source IPs?
- How would I attach an SSL certificate to my AWS HTTPS ELB?
- How would I make a service which listens on both HTTP/80 and HTTPS/443?
- What are some of the AWS limitations?
- Can you run a multi-AZ Kubernetes cluster? What about a multi-region cluster?

- Is there a way to update route53 DNS with a service members?
- Can Kubernetes auto-create an EBS volume?
- When using an EBS PersistentVolume and PersistentVolumeClaim, how does Kubernetes know which AZ to create a pod in?
- Does Kubernetes support the new Amazon Load Balancer (ALB)?
- Is there a way to give a pod a separate IAM role that has different permissions than the default instance IAM policy?
- Is Kubernetes rack aware or can you detect what region or Availability Zone a host is in?
- Is it possible to install Kubernetes into an existing VPC?
- Is it possible to install Kubernetes into a private VPC?

Architecture:

What happens when a master fails? What happens when a worker fails?

Kubernetes is designed to be resilient to any individual node failure, master or worker. When a master fails the nodes of the cluster will keep operating, but there can be no changes including pod creation or service member changes until the master is available. When a worker fails, the master stops receiving messages from the worker. If the master does not receive status updates from the worker the node will be marked as NotReady. If a node is NotReady for 5 minutes, the master reschedules all pods that were running on the dead node to other available nodes.

How does DNS work in Kubernetes?

There is a DNS server called skydns which runs in a pod in the cluster, in the `kube-system` namespace. That DNS server reads from etcd and can serve up dns entries for Kubernetes services to all pods. You can reach any service with the name `<service>.<namespace>.svc.cluster.local`. The resolver automatically searches `<namespace>.svc.cluster.local` dns so that you should be able to call one service to another in the same namespace with just `<service>`.

How do I build a High Availability (HA) cluster?

The only stateful part of a Kubernetes cluster is the etcd. The master server runs the controller manager, scheduler, and the API server and can be run as replicas. The controller manager and scheduler in the master servers use a leader election system, so only one controller manager and scheduler is active for the cluster at any time. So an HA cluster generally consists of an etcd cluster of 3+ nodes and multiple master nodes.

Learn more: http://kubernetes.io/docs/admin/high-availability/#master-elected-components

Can I isolate namespaces from each other?

Yes, network policies allow you to isolate namespaces at the network layer. Full isolation requires use of an overlay network such as Flannel, Calico, Weave, or Romana. http://kubernetes.io/docs/user-guide/networkpolicies/

Basic usage questions:

Should I use Replication Controllers?

Probably not, they are older and have fewer features than the newer Deployment objects.

How do I determine the status of a Deployment?

Use `kubectl get deployment <deployment>`. If the `DESIRED`, `CURRENT`, `UP-TO-DATE` are all equal, then the Deployment has completed.

How do I update all my pods if the image changed but the tag is the same

Make sure your `imagePullPolicy` is set to `Always`(this is the default). That means when a pod is deleted, a new pod will ensure it has the current version of the image. Then refresh all your pods.

The simplest way to refresh all your pods is to just delete them and they will be recreated with the latest image. This immediately destroys all your pods which will cause a service outage. Do this with `kubectl delete pod -l <name>=<value>` where name and value are the label selectors your deployment uses.

A better way is to edit your deployment and modify the deployment pod spec to add or change any annotation. This will cause all your pods to be deleted and rescheduled, but this method will also obey your `rollingUpdate` strategy, meaning no downtime assuming your `rollingUpdate` strategy already behaves properly. Setting a timestamp or a version number is convenient, but any change to pod annotations will cause a rolling update. For a deployment named nginx, this can be done with:

```
PATCH='{"spec":{"template":{"metadata":{"annotations":{"timestamp":"'$(date)'"}}}}}'
kubectl patch deployment nginx -p "$PATCH"
```

It is considered bad practice to rely on the `:latest` docker image tag in your deployments, because using `:latest` there is no way to rollback or specify what version of your image to use. It's better to update the deployment with an exact version of the image and use `--record` so that you can use `kubectl rollout undo deployment <deployment>` or other commands to manage rollouts.

How do I debug a Pending pod?

A `Pending` pod is one that cannot be scheduled onto a node. Doing a `kubectl describe pod <pod>` will usually tell you why. `kubectl logs <pod>` can also be helpful. There are several common reasons for pods stuck in Pending:

** The pod is requesting more resources than are available, a pod has set a `request` for an amount of CPU or memory that is not available anywhere on any node. eg. requesting a 8 CPU cores when all your nodes only have 4 CPU cores. Doing a `kubectl describe node <node>` on each node will also show already requested resources. ** There are `taints` that prevent a pod from scheduling on your nodes. ** The nodes have been marked unschedulable with `kubectl cordon` ** There are no `Ready` nodes. `kubectl get nodes` will display the status of all nodes.

How do I debug a ContainerCreating pod?

A `ContainerCreating` pod is one that has been scheduled on a node, but cannot startup properly. Doing a `kubectl describe pod <pod>` will usually tell you why. Common reasons include:

** Container Networking Interface(CNI) errors are preventing the pod networking from being setup properly. Flannel and weave versions or configuration can sometimes cause this. ** Volume mounts failures are preventing startup. External volumes like EBS or GCE PD sometimes cannot be properly attached to the node.

How do I debug a CrashLoopBackoff pod?

This is the standard error message when a pod fails with an error. `kubectl describe pod <podid>` usually doesn't provide much helpful information, but `kubectl logs <podid>` would show the stdout from the pod during the most recent execution attempt. Another helpful technique is to change the `spec.containers.command` for your pod to `bash -c '<command> || sleep 10d'`. This will start your container and then if it exits with a non-zero error code it will sleep for 10 days. This will enable you then use `kubectl exec -it <podid> -- bash` to enter a shell in the container while it is still running but after the main command has exited so that you can debug it.

Another common reason is that a node is failing its health check and has been killed by Kubernetes. The pod will generally restart itself after some period of time(a backoff time, hence the CrashLoopBackoff), on the same node. A CrashLoopBackoff does not move a pod to a new node.

How do I rollback a Deployment?

If you apply a change to a Deployment with the `--record` flag then Kubernetes stores the previous Deployment in its history. The `kubectl rollout history`

`deployment <deployment>` command will show prior Deployments. The last Deployment can be restored with the `kubectl rollout undo deployment <deployment>` command. In progress Deployments can also be paused and resumed.

When a new version of a Deployment is applied, a new ReplicaSet object is created which is slowly scaled up while the old ReplicaSet is scaled down. You can look at each ReplicaSet that has been rolled out with `kubectl get replicaset`. Each ReplicaSet is named with the format -, so you can also do `kubectl describe replicaset <replicaset>`.

Learn more: http://kubernetes.io/docs/user-guide/kubectl/kubectl_rollout/

What is a DaemonSet?

A DaemonSet is a set of pods that is run only once on a host. It's used for host-layer features, for instance a network, host monitoring or storage plugin or other things which you would never want to run more than once on a host.

Learn more: http://kubernetes.io/docs/admin/daemons/

What is a PetSet or StatefulSet?

In a regular Deployment all the instances of a pod are exactly the same, they are indistinguishable and are thus sometimes referred to as "cattle", these are typically stateless applications that can be easily scaled up and down. In a PetSet, each pod is unique and has an identity that needs to be maintained. This is commonly used for more stateful applications like databases.

Learn more: http://kubernetes.io/docs/user-guide/petset/

In 1.5, PetSets have been renamed to Stateful Sets

Learn more: http://kubernetes.io/docs/tutorials/stateful-application/basic-stateful-set/

What is an Ingress Controller?

An Ingress Controller is a pod that can act as an inbound traffic handler. It is a HTTP reverse proxy that is implemented as a somewhat customizable nginx. Among the features are HTTP path and service based routing and SSL termination.

Learn more: http://kubernetes.io/docs/user-guide/ingress/

Why do I see 504 errors from my Ingress during deploys?

This occurs due to a race condition during pod deletion between the Ingress and the pod. When a pod is deleted, it can shut down before the Ingress knows to

stop sending traffic. So the Ingress may continue to send traffic to a disabled pod.

The simplest way to avoid this is to prevent the pod from shutting down immediately with a `preStop` hook. Adding in a `preStop` hook to the deployment which does `sleep 5` should delay the pod termination long enough to let the Ingress update and remove the disabled pod from its upstream list.

https://github.com/kubernetes/kubernetes/issues/43576 https://github.com/kubernetes/ingress/issues/322 https://github.com/kubernetes/contrib/issues/1140 https://kubernetes.io/docs/concepts/workloads/pods/pod/#
of-pods

How does a Kubernetes service work?

Within the cluster, most Kubernetes services are implemented as a virtual IP called a ClusterIP. A ClusterIP has a list of pods which are in the service, the client sends IP traffic directly to a randomly selected pod in the service, so the ClusterIP isn't actually directly routable even from kubernetes nodes. This is all done with iptables routes. The iptables configuration is managed by the kube-proxy on each node. So only nodes running kube-proxy can talk to ClusterIP members. An alternative to the ClusterIP is to use a "Headless" service by specifying ClusterIP=None, this does not use a virtual IP, but instead just creates a DNS A record for a service that includes all the IP addresses of the pods. The live members of any service are stored in an API object called an `endpoint`. You can see the members of a service by doing a `kubectl get endpoints <service>`

How do I expose a service to a host outside the cluster?

There are two ways:

1. Set the service type to NodePort. This makes every node in the cluster listen on the specified NodePort, then any node will forward traffic from that NodePort to a random pod in the service.
2. Set the service type to LoadBalancer. This provisions a NodePort as above, but then does an additional step to provision a load balancer in your cloud(AWS or GKE) automatically. In AWS it also modifies the Auto-Scaling Group of the cluster so all nodes of that ASG are added to the ELB.

How does a LoadBalancer service work?

A LoadBalancer by default is set up as a TCP Load Balancer with your cloud provider (AWS or GKE). There is no support in bare metal or OpenStack for Load Balancer types. The Kubernetes controller manager provisions a load balancer in your cloud and puts all of your Kubernetes nodes into the load balancer. Because each node is assumed to be running `kube-proxy` it should be

listening on the appropriate NodePort and then it can forward incoming requests to a pod that is available for the service.

Because the LoadBalancer type is by default TCP, not HTTP many higher level features of a LoadBalancer are not available. For instance health checking from the LoadBalancer to the node is done with a TCP check. HTTP X-Forwarded-For information is not available, though it is possible to use proxy protocol in AWS.

http://kubernetes.io/docs/user-guide/services/

How do I force a pod to run on a specific node?

There are two mechanism for this: taints and node selection.

Kubernetes node selection is described here: http://kubernetes.io/docs/user-guide/node-selection/

In kubernetes 1.3 the concept of a `taint` was implemented. A taint is a way of marking a node for a specific purpose. In order to be scheduled onto a tainted node, a pod must "tolerate" the taint.

Use a `taint` to limit the pods that can run on a node, for instance you want to dedicate instances for only a specific kind of pod. On the other hand, use a pod's node selector to limit where a pod can run, for instance if your pod needs specific features on a host like a GPU.

https://github.com/coreos/kubernetes/blob/master/docs/design/taint-toleration-dedicated.md

How do I force replicas of a pod to split across different nodes?

Kubernetes by default does attempt node anti-affinity, but it is not a hard requirement, it is best effort, but will schedule multiple pods on the same node if that is the only way.

Learn more: http://stackoverflow.com/questions/28918056/does-the-kubernetes-scheduler-support-anti-affinity http://kubernetes.io/docs/user-guide/node-selection/

How can I get the host IP address from inside a pod?

In Kubernetes 1.4 the nodename is available in the downward API in the `spec.nodeName` variable.

http://kubernetes.io/docs/user-guide/downward-api/

In AWS specifically, It may be easier to use the AWS metadata API and just `curl 169.254.169.254/1.0/meta-data/local-ipv4`

How do I give individual pods DNS names?

In v1.3, add a `subdomain` field to the pod specification, then create a Headless service which has the same name as the `subdomain`, then each pod gets a DNS record for ...svc.cluster.local

In v1.2, a similar mechanism exists but using annotations.

http://kubernetes.io/docs/admin/dns/#a-records-and-hostname-based-on-pods-hostname-and-subdomain-fields

How do I access the Kubenernetes API from within a pod?

See the above answer on "getting the host IP address from inside a pod" for an example of using the API inside a pod.

How do I get all the pods on a node?

You can use the following command to get all the pods on a node in kubernetes 1.4

```
kubectl get po --all-namespaces  -o jsonpath='{range .items[?(@.spec.nodeName =="nodename")]
```

Can pods mount NFS volumes?

Yes, there's an example here of both an NFS client and server running within pods in the cluster: https://github.com/jsafrane/kubernetes-nfs-example

Is it possible to route traffic from outside the Kubernetes cluster directly to pods?

Yes. But one major downside of that is that ClusterIPs are implemented as iptables rules on cluster clients, so you'd lose the ability to see Cluster IPs and service changes. Because the iptables are managed by kube-proxy you could do this by running a kube-proxy, which is similar to just joining the cluster. You could make all your services Headless(ClusterIP = None), this would give your external servers the ability to talk directly to services if they could use the kubernetes dns. Headless services don't use ClusterIPs, but instead just create a DNS A record for all pods in the service. kube-dns is run inside the cluster as a ClusterIP, so there's a chicken and egg problem with DNS you would have to deal with.

How do I put variables into my pods?

Look at the Downward API: http://kubernetes.io/docs/user-guide/downward-api/ It allows your pods to see labels and annotations and a few other variables using either a mount point or environment variables inside the pod. If those don't contain the information you need, you'll likely need to resort to either using the Kubernetes API from within the pod or something entirely separate.

Can I use variables or otherwise parameterize my yaml deployment files?

There is no built in functionality for this. Helm is a popular third party choice for this. Some people use scripts or templating tools like jinja.

How do CPU and memory requests and limits work?

Think of `request` as a node scheduling unit and a `limit` as a hard limit when it is already running. Kubernetes will attempt to schedule your pods onto nodes based only on the sum of the existing requests on the node plus the new pod request. A request is only used for scheduling which node a pod runs on. A limit is monitored after a pod has been scheduled and is running. Setting a limit enforces a cap and ensures that a pod does not exceed the limit on a node, killing it does.

For simplicity you can just set request and limit to be the same, but you won't be able to pack things tightly onto a node for efficiency. The converse problem is when you set limits which are much larger than requests there is a danger that pods use resources all the way up to their limit and overrun the node or starve other pods on the node. CPU may not hard-capped depending on the version of Kubernetes/Docker, but memory is. Pods exceeding their memory limit will be terminated and rescheduled.

Why is my pod pending?

Pending usually means that a pod cannot be scheduled, because of a resource limitation, most commonly the cluster can't find a node which has the available CPU and memory requests to satisfy the scheduler. `kubectl describe pod <podid>` will show the reason why the pod can't be scheduled. Pods can remain in the Pending state indefinitely until the resources are available or until you reduce the number of required replicas.

Why is my terminal screwed up?

There's an issue with the kubernetes client not handling terminals correctly. I have a script that I use that solves most of these problems. https://github.com/hubt/kubernetes-faq/tree/master/kshell

https://github.com/kubernetes/kubernetes/issues/13585

What monitoring and metrics tools do people use for Kubernetes?

Heapster is included and its metrics are how Kubernetes measures CPU and memory in order to use horizontal pod autoscaling (HPA). Heapster can be queried directly with its REST API. Prometheus is also more full featured and popular.

How can containers within a pod communicate with each other?

Containers within a pod share networking space and can reach other on `localhost`. For instance, if you have two containers within a pod, a MySQL container running on port 3306, and a PHP container running on port 80, the PHP container could access the MySQL one through `localhost:3306`.

Learn more: https://github.com/kubernetes/kubernetes/blob/release-1.4/docs/design/networking.md#container-to-container

How do I configure credentials to download images from a private docker registry?

Create a special secret in a your namespace that provides the registry and credentials to authenticate with. Then use that secret in the `spec.imagePullSecrets` field of your pod specification. http://kubernetes.io/docs/user-guide/production-pods/#authenticating-with-a-private-image-registry http://kubernetes.io/docs/user-guide/images/#using-a-private-registry

Is it possible to run docker inside a pod?

Yes. The two tricks are:

Your pod must run in privileged mode. kubelet must run with `--allow-privileged=true`(this is the default) and the pod must run with `securityContext.privileged: true`. This will allow your pod to mount the host docker socket directly.

You must mount the host docker socket by specifying `volumes.hostPath.path: /var/run/docker.sock` in your pod spec.

Here is a simple alpine docker deployment which can run docker commands:

```
apiVersion: extensions/v1beta1
kind: Deployment
metadata:
  labels:
    app: docker
  name: docker
spec:
  template:
    metadata:
      labels:
        app: docker
    spec:
      containers:
      - command: ["/bin/sleep","365d"]
        image: hubt/alpine-docker
        imagePullPolicy: Always
        name: docker
```

```
    securityContext:
      privileged: true
    terminationMessagePath: /dev/termination-log
    volumeMounts:
    - mountPath: /var/run/docker.sock
      name: docker-socket
  imagePullSecrets:
  - name: registrypullsecret
  volumes:
  - hostPath:
      path: /var/run/docker.sock
    name: docker-socket
```

What versions of docker are supported?

With kubernetes 1.5, Docker versions 1.10.3 - 1.12.3 are supported, with some known issues.

https://github.com/kubernetes/kubernetes/blob/master/CHANGELOG.md#external-dependency-version-information

Why are some of my pods in an Unknown state?

In Kubernetes 1.6, a significant change was made to how the master treats nodes whose state cannot be determined. Pods from the node are put into an `Unknown` state and the master will not attempt to reschedule them.

In Kubernetes 1.5 and earlier, if the node has not sent a heartbeat to the master in 5 minutes, the node is deleted from the masters and the pods are rescheduled automatically.

You can force delete pods with

`kubectl delete pods <pod> --grace-period=0 --force`

https://kubernetes.io/docs/tasks/run-application/force-delete-stateful-set-pod/#force-deletion A longer discussion of the reasoning and the change is here: https://github.com/kubernetes/community/blob/master/contributors/design-proposals/pod-safety.md

AWS Questions:

How should I install Kubernetes on AWS?

`kube-up.sh` is being deprecated, it will work for relatively straightforward installs, but won't be actively developed anymore. Kops is the new recommended deployment tool especially on AWS. It doesn't support other cloud environments yet, but that is planned.

How does the default Kubernetes AWS networking work?

In addition to regular EC2 ip addresses, Kubernetes creates its own cluster internal network. In AWS, each instance in a kubernetes cluster hosts a small subnet(by default a /24), and each pod on that host get its own IP addresses within that node's /24 address space. Whenever a cluster node is added or deleted, the Kubernetes controller updates the route table so that all nodes in the VPC can route pod IPs directly to that node's subnet.

How do I add a node to my AWS Kubernetes cluster?

If you used `kube-up.sh` or `kops` to provision your cluster, then it created an AutoScaling Group automatically. You can re-scale that with kops, or update the ASG directly, to grow/shrink the cluster. New instances are provisioned for you and should join the cluster automatically (my experience has been it takes 5-7 minutes for nodes to join).

With `kops` the recommended process is to edit the InstanceGroup (ig) and then update your cluster. `kops` also supports multiple instance groups per cluster so you can have multiple Auto Scaling Groups to run multiple types of instances within your cluster. Spot instances are also supported.

How do I set up node auto-scaling?

The following project is an autoscaler: https://github.com/kubernetes/contrib/tree/master/cluster-autoscaler

Learn more: https://github.com/kubernetes/kops/blob/master/docs/instance_groups.md

How do you make a service create a private ELB in AWS instead of the default public one?

Add the following metadata annotation to your LoadBalancer service

`service.beta.kubernetes.io/aws-load-balancer-internal: 0.0.0.0/0`

Then delete and re-create the service object and it should be a new private ELB.

How do you restrict an AWS ELB to certain source ips:

Add the following metadata annotation to your LoadBalancer service with a comma separated list of CIDRs:

`service.beta.kubernetes.io/load-balancer-source-ranges`

Each ELB gets its own security group and this annotation will add those CIDR addresses to the allowed source IPs

https://github.com/kubernetes/kubernetes/blob/d95b9238877d5a74895189069121328c16e420f5/pkg/api/service
L34

How would I get my ELB LoadBalancer to be an HTTP/s LoadBalancer instead of the default TCP?

Add the following metadata annotations to your service

```
service.beta.kubernetes.io/aws-load-balancer-backend-protocol: http
```

or

```
service.beta.kubernetes.io/aws-load-balancer-backend-protocol: https
```

https://github.com/kubernetes/kubernetes/pull/23495

How would I attach an SSL certificate to my AWS HTTPS ELB?

Add the following metadata annotations to your service

```
service.beta.kubernetes.io/aws-load-balancer-ssl-cert=arn:aws:acm:us-east-1:123456789012:cer
```

How would I make a service which listens on both HTTP/80 and HTTPS/443?

Create a service which listens on port 80 and 443, attach an SSL certificate as above. Then add the following metadata annotation to your service

```
service.beta.kubernetes.io/aws-load-balancer-ssl-ports: "443"
```

This tells the ELB that the SSL certificate will only be used for 443 and not 80.

What are some of the AWS limitations?

There is a route table entry for every instance in the cluster which allows other nodes to route to the pods on that node. AWS has a soft limit of 50 routes per route table and a hard limit of 100 routes per route table. So with the default VPC networking you will run into one of these limits. Using one of the overlay networks (Flannel, Weave, Romana) can get around this limit. And kops-routing is a planned feature to do this directly from AWS nodes. A planned feature is for all these networking plugins to be drop in additions to an existing cluster.

Can you run a multi-AZ Kubernetes cluster? What about a multi-region cluster?

Yes and "Not out of the box". Provision with kops and specify the AZ's you want and your cluster will be a multi-AZ cluster within a single region. The AutoScalingGroups can add nodes to any region you specify. The planned solution for a multi-region cluster is to build separate clusters in each region and use federation to manage multiple clusters. http://kubernetes.io/docs/admin/federation/. It is also possible to build a multi-region cluster using an overlay network like Flannel, Calico or Weave.

Is there a way to update route53 DNS with a service members?

The new way going forward to do this will be with the external-dns project. https://github.com/kubernetes-incubator/external-dns

Older project with similar functionality: https://github.com/wearemolecule/route53-kubernetes Future work in core kops: https://github.com/kubernetes/kops/tree/master/dns-controller

Can kubernetes auto-create an EBS volume?

Yes. When you declare a PersistentVolumeClaim add the annotation:

```
volume.alpha.kubernetes.io/storage-class: "foo"
```

And the PersistentVolumeClaim will automatically create a volume for you and delete it when the PersistentVolumeClaim is deleted, "foo" is meaningless, the annotation just needs to be set.

When using an EBS PersistentVolume and PersistentVolumeClaim, how does Kubernetes know which AZ to create a pod in?

It just works. EBS volumes are specific to an Availability Zone, and Kubernetes knows which AZ a volume is in. When a new pod needs that volume it the pod is automatically scheduled in the Availability Zone of the volume.

Does Kubernetes support the new Amazon Load Balancer (ALB)?

Not currently.

Is there a way to give a pod a separate IAM role that has different permissions than the default instance IAM policy?

This is a third party tool that enables this: https://github.com/jtblin/kube2iam

Is Kubernetes rack aware or can you detect what region or Availability Zone a host is in?

A Kubernetes node has some useful pieces of information attached as labels here are some examples:

```
beta.kubernetes.io/instance-type: c3.xlarge
failure-domain.beta.kubernetes.io/region: us-east-1
failure-domain.beta.kubernetes.io/zone: us-east-1b
kubernetes.io/hostname: ip-172-31-0-10.us-east-1.compute.internal
```

You can use these for AZ awareness or attach your own labels and use the Downward API for additional flexibility.

Is it possible to install Kubernetes into an existing VPC?

With kops, this is possible. But having more than one Kubernetes cluster in a VPC is not supported.

Is it possible to install Kubernetes into a private VPC?

With kops 1.5, this is possible. There are features like private subnets, NAT Gateways, bastion hosts.