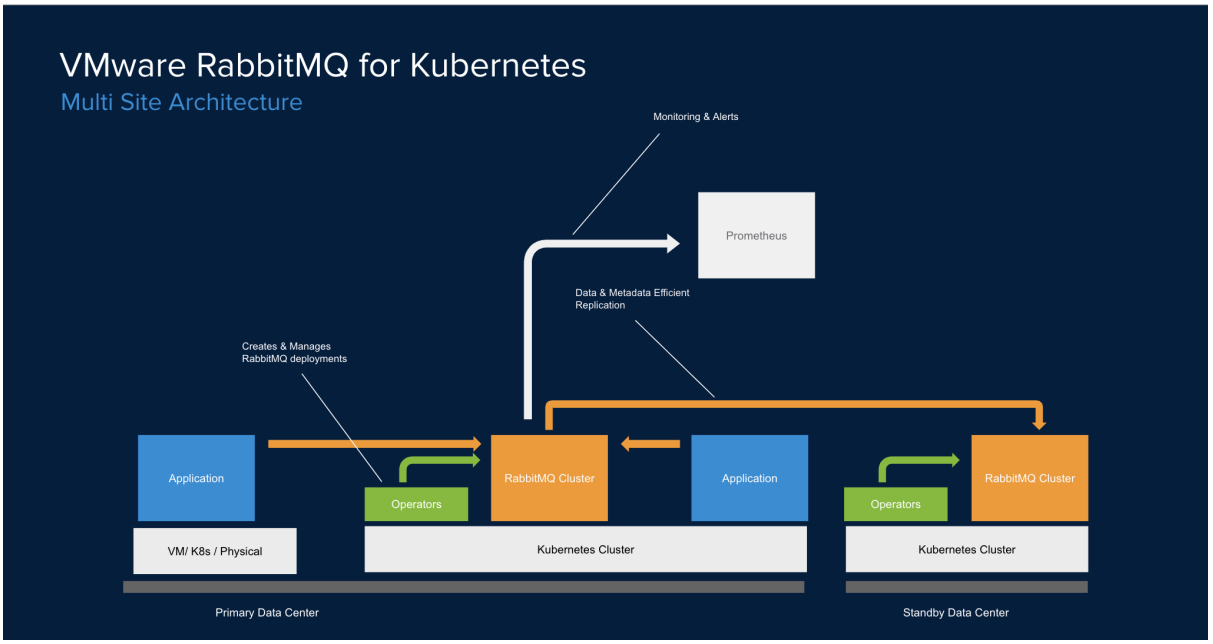
Overall Architecture



INSTALLATION

https://docs.vmware.com/en/VMware-RabbitMQ-for-Kubernetes/1/rmq/installation.html

kapp: https://carvel.dev/kapp-controller/docs/v0.41.0/install/

RBAC Configuration Required for the VMware RabbitMQ (Kubernetes üzerinde gerekli izinler): Page: 162.

Resource Requests:
For production:

- 12cpu (3 replicas, 4 cpu each) and
- 30GB memory (3 replicas, 10Gb memory each)
- Persistent storage 500GB

Documentation Link

https://docs.vmware.com/en/VMware-RabbitMQ-for-Kubernetes/1.4/rmq.pdf

PREREQUISITES

- VMware Tanzu Network account and its access (username and password) (https://network.tanzu.vmware.com/) to install tanzu-cluster-essentials. Required because it installs the kapp-controller and secretgen-controller controllers on your Kubernetes cluster which are needed for rabbitmq installation. Internet connection for installing these packages from https://network.tanzu.vmware.com will be needed.
- Kubernetes clusters up and running for stage, production and DR environment.
- Suitable compute resources (CPU, memory) and storage on the Kubernetes nodes. Existing infra uses 8 core, 16gb ram vm's for each rabbitmq node and 600GB storage for production. Based on that, recommended resource and storage allocation for each rabbitmq cluster on the specific environments listed below;

Stage (3 nodes)	Production (3 nodes)	DR (3 nodes)
6 cpu, 6gb ram (2cpu, 2gb ram for each node)	12 cpu, 30gb ram (4cpu, 10gb ram for each node)	12 cpu, 30gb ram (4cpu, 10gb ram for each node)
160GB Storage	500GB Storage	500GB Storage

- Ports will need to be allowed (accessible) for VMware RabbitMQ Cluster and Monitoring tools.

Inter-node and CLI tools	AMQP 0.9.1	Management UI	Prometheus & Grafana	epmd
25672 35672-35682	5672, 5671	15672, 15671 If there is any specific port wanted for accessing the RabbitMQ management panel, it must be also allowed. For exp: 8080	Prometheus: 9090, Grafana: 3000 Prometheus metrics from rabbitmq clusters: 15692, 15691	4369

- Production and DR clusters should be able to talk to each other over the network and that the right rules are set up for them to communicate on port 5672 and 5671. DR and production clusters will need to know the external_ip's of the rabbitmq clusters for Standby Replication and Synchronization.
- Access information and instructions for accessing each environment (staging, production, disaster recovery) Kubernetes clusters.
- Service Account creation with necessary permissions (RBAC Configurations, cluster level) for VMware RabbitMQ.

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: tanzu-rabbitmq-crd-install
rules:
- apiGroups:
  - admissionregistration.k8s.io
  resources:
  - validatingwebhookconfigurations
  - mutatingwebhookconfigurations
  verbs:
  - "*"
- apiGroups:
  - apiextensions.k8s.io
  resources:
  - customresourcedefinitions
  verbs:
  - "*"
- apiGroups:
  - apps
  resources:
  - deployments
  verbs:
  - "*"
- apiGroups:
  - cert-manager.io
  resources:
  - certificates
  - issuers
  verbs:
  - "*"
- apiGroups:
  - ""
  resources:
  - configmaps
  - namespaces
  - secrets
  - serviceaccounts
  - services
  verbs:
  - "*"
- apiGroups:
  - rbac.authorization.k8s.io
  resources:
  - clusterrolebindings
  - clusterroles
```

```
  - rolebindings
  - roles
  verbs:
  - "*"
- apiGroups:
  - coordination.k8s.io
  resources:
  - leases
  verbs:
  - "*"
- apiGroups:
  - ""
  resources:
  - events
  verbs:
  - create
  - get
  - patch
- apiGroups:
  - ""
  resources:
  - endpoints
  verbs:
  - get
  - list
  - watch
- apiGroups:
  - ""
  resources:
  - persistentvolumeclaims
  verbs:
  - create
  - get
  - list
  - update
  - watch
- apiGroups:
  - ""
  resources:
  - pods
  verbs:
  - get
  - list
  - update
  - watch
- apiGroups:
```

```
  - ""
  resources:
  - pods/exec
  verbs:
  - create
- apiGroups:
  - apps
  resources:
  - statefulsets
  verbs:
  - create
  - delete
  - get
  - list
  - update
  - watch
- apiGroups:
  - rabbitmq.com
  - rabbitmq.tanzu.vmware.com
  resources:
  - "*"
  verbs:
  - "*"
---
apiVersion: v1
kind: ServiceAccount
metadata:
  name: tanzu-rabbitmq
  namespace: default
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: tanzu-rabbitmq-crd-install-binding
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: tanzu-rabbitmq-crd-install
subjects:
- kind: ServiceAccount
  name: tanzu-rabbitmq
  namespace: default
```

- SSL/TLS certifications and details if there is any. (*.crt , *.key)
- Internet connection to download required images and packages for VMware RabbitMQ from the registry: registry.tanzu.vmware.com
- If **kapp** command not found installation link: (Carvel - kapp - Install)

- If you need to **delete PackageInstall** you can do with kubectl delete.

What can VMware RabbitMQ for Kubernetes do?

When VMware RabbitMQ is deploying a RabbitMQ cluster **it also creates a Kubernetes service** that allows other pods to use the cluster **without the need for additional load balancer.** If you want to expose the RabbitMQ cluster to applications outside the Kubernetes cluster, VMware RabbitMQ configures this service to have a routable endpoint on the Kubernetes external load balancer.

Warm Standby Replication for Disaster Recovery:

It is an automated disaster recovery solution, a strategy which replicates or copies data from an upstream (primary) RabbitMQ cluster to a downstream (standby) cluster. If a disaster occurs on the upstream (primary) cluster, **an administrator can quickly start** the recovery process with minimal downtime or data loss.

Requirements:

- The **Standby Replication Operator** should be installed. If the upstream and downstream clusters are in different kubernetes clusters, the operator should be installed on both clusters. The Standby Replication Operator is used to configure the Continuous Schema Replication and Standby Message Replication plugins.
- Know the credentials (username and password) that you want to use for Warm Standby Replication.

Setting up the Upstream and Downstream RabbitMQ Clusters

Note: If it is not specified, the cluster size (initial-cluster-size) should always be set to a **minimum value of 3.**

Example setup can be found here:
Documentation**:**https://docs.vmware.com/en/VMware-RabbitMQ-for-Kubernetes/1.4/rmq.pdf
Page:70 - 86

KNOWLEDGE

Basic Concepts

Data sync between clusters?

Quorum Queues, like Mirror Queues, are primarily designed for replicating data within a single RabbitMQ cluster for high availability and fault tolerance. They ensure that messages are replicated across multiple nodes (replicas) **within the same cluster**, so that if one node fails, the data is still accessible from other replicas.
However, if you want to **replicate data between RabbitMQ clusters deployed in separate Kubernetes environments** (i.e., between clusters), you would typically use other mechanisms such as **federation or custom solutions**. **Quorum Queues alone do not provide native support for replicating data between clusters.**
For replicating data between clusters, you might set up a federated exchange in each cluster, where messages published to one cluster are federated (replicated) to the other cluster. Alternatively, you might develop custom applications or scripts to handle data replication between clusters, depending on your specific requirements and constraints.

Note: Quorum queues can not be set as default queues. Default queue is classic. Therefore, the argument **x-queue-type** should be set to 'quorum' when the queue is created. However, **after VMWare RabbitMQ version >=1.4 we can set the default queue for the vhosts**.

Note: Before a queue can be used it has to be declared. Declaring a queue will cause it to be created if it does not already exist. The declaration will have no effect if the queue does already exist and its attributes are the same as those in the declaration. When the existing queue attributes are not the same as those in the declaration a channel-level exception with code 406 (PRECONDITION_FAILED) will be raised.

Queue Attributes/Properties

Queues have properties that define how they behave. There is a set of mandatory properties and a map of optional ones:

- Name
- Durable (the queue will survive a broker restart)
- Exclusive (used by only one connection and the queue will be deleted when that connection closes)

- Auto-delete (queue that has had at least one consumer is deleted when last consumer unsubscribes)
- Arguments (optional; used by plugins and broker-specific features such as message TTL, queue length limit, etc)

Optional queue arguments, also known as **"x-arguments"** are key/value pairs that can be provided by clients when a queue is declared or to groups of queues **using policies (recommended)**.

- Queue type (exp: quorum, classic)
- Message and queue TTL (Time-to-live)
- Queue Length Limit (the number of messages ready for delivery)
- Priorities (Number between 1 - 255) For queues
- Consumer Priorities

and so on.

Note: queue type (**x-queue-type**) and max number of queue priorities (**x-max-priority**) **must be set at queue declaration time** which means **they can only be provided by clients not with policies** and cannot be changed after that.

Operator Policy for queue arguments:

Operator policy will **take priority** over the client-provided arguments.

Use operator policies to introduce **guardrails for application-controlled parameters** related to resource use (e.g. peak disk space usage).

Durability:

Transient queues will be deleted on node/cluster boot. They therefore will not survive a node restart, by design. Messages in transient queues will also be discarded. **The plan is to remove transient queues in future RabbitMQ releases.**

Durable queues will be recovered on node boot, including messages in them published as persistent. Messages published as transient will be discarded during recovery, even if they were stored in durable queues. **Quorum queues must be durable.**

Priorities:

Queues can have **0 or more priorities**. This feature is opt-in: only queues that have a maximum number of priorities configured via an optional argument will do prioritization. Publishers specify message priority using the **priority field in message properties**. If priority queues are desired, **we recommend using between 1 and 10**. Currently **using more priorities will consume more resources** (Erlang processes).

Quorum queues support consumer priorities, but not message priorities. **To prioritize messages with Quorum Queues, use multiple queues;** one for each priority.

Poison Message Handling

Quorum queues support poison message handling via a **redelivery limit**. This feature is currently **unique to quorum queues**. (delivery-limit)

Quorum queues keep track of the number of unsuccessful delivery attempts and expose it in the **"x-delivery-count" header** that is included with any redelivered message.

"Quorum queues", bu tür **hatalı mesajları yönetmek için zehirli (poison) mesaj işleme** özelliğini destekler. Bu özellik, bir mesajın belirli bir yeniden iletim sınırına ulaştığında, yani belirli bir sayıda yeniden işlenme girişiminden sonra, mesajın işlenemeyeceği ve kuyruktan kaldırılacağı anlamına gelir. Bu, uygulamanın sürekli olarak aynı hatalı mesajları işlemeye çalışmasını önler ve sistem performansını korur.

Queue Leader Location

To avoid some nodes in a cluster hosting the majority of queue leader replicas and thus handling most of the load, queue leaders should be reasonably evenly distributed across cluster nodes.

Documentation**:**https://docs.vmware.com/en/VMware-RabbitMQ-for-Kubernetes/1.4/rmq.pdf
Page:106

Bunun ne olduğuna bakılması lazım: rabbitmq-sharding (trade off message ordering for parallelism (better CPU core utilisation))

Acknowledgements:

Delivered messages can be acknowledged by consumers explicitly or automatically as soon as a delivery is written to the connection socket.

Automatic acknowledgement mode generally will provide **higher throughput rate** and uses **less network bandwidth**. However, it offers the **least number of guarantees** when it comes to **failures**. As a rule of thumb, consider using manual acknowledgement mode first.

Manual acknowledgement mode provides a way to set a limit on the number of outstanding (unconfirmed) deliveries: channel QoS (prefetch). henüz onaylanmamış olan (doğrulanmamış) teslimatların sayısını sınırlayan bir ayar olarak düşünülebilir.

VHOSTS

Think of vhosts as individual, uniquely named containers. Inside each vhost container is a logical group of exchanges, connections, queues, bindings, user permissions, and other

system resources. Different users can have different permissions to different vhost and queues and exchanges can be created, so they only exist in one vhost. When a client establishes a connection to the RabbitMQ server, it specifies the vhost within which it will operate, as in the following example: amqp://myuser:mypassword@localhost:5672/myvhost

Example Usage: Imagine you have a RabbitMQ setup for a company that provides multiple services, such as a blog platform and an e-commerce website. Each service has its own messaging needs and data, and you want to keep them isolated for better management and security.

In RabbitMQ, vhosts are available across all nodes within a cluster by default. When you create a vhost on one node in the cluster, it is automatically replicated and made available on all other nodes within the cluster.

Dead Lettering

Dead letter strategies for quorum queues: **at-least-once, at-most-once.** Read docs to learn how to enable it.
Documentation**:**https://docs.vmware.com/en/VMware-RabbitMQ-for-Kubernetes/1.4/rmq.pdf
Page:101

When Not to Use Quorum Queues

- Temporary nature of queues: transient or exclusive queues, high queue churn (declaration and deletion rates)
- Lowest possible latency: the underlying consensus algorithm has an inherently higher latency due to its data safety features
- When data safety is not a priority (e.g. applications do not use manual acknowledgements and publisher confirms are not used)
- Very long queue backlogs (streams are likely to be a better fit)

Streams

For cases that would benefit from

- **replication and repeatable reads,**
- **large fan-outs,**

streams may be a better option than quorum queues.

Exchange? Routing Key? Purpose:

Publishing to **queues** lets you only implement basic publish-subscribe scenarios, where the producer and consumer use the exact queue. In case of multiple consumers a single queue of messages is distributed between multiple consumers.

Publishing to **exchanges** lets you create complicated scenarios, because of **routing** between exchanges and queues.

For example, a **fanout exchange** routes messages to all bound queues. This way, you can have one producer and multiple consumers and each message is **copied** to all bound queues independently and received independently.

Another example of exchange, a **topic exchange** routes messages to bound queues based on the routing *key* in a message and a pattern on a queue. This introduces an interesting possibility of *tagging* messages and delivering them conditionally.

Two option:

```
publish(
    exchange: string,
    routingKey: string,
    content: Buffer,
    options?: Options.Publish,
    callback?: (err: any, ok: Replies.Empty) => void,
): boolean;
sendToQueue(
    queue: string,
    content: Buffer,
    options?: Options.Publish,
    callback?: (err: any, ok: Replies.Empty) => void,
): boolean;
```

Approaches (Who creates what?):
https://gigi.nullneuron.net/gigilabs/rabbitmq-who-creates-the-queues-and-exchanges/

Networking

Connections:

- **Basic:**
  - All protocols supported by RabbitMQ are TCP-based and assume long-lived connections (a new connection is not opened per protocol operation) for efficiency.
  - Connection churns when a new connection is created and closed. Long-lived connections should be used if possible.
  - For clients that cannot establish a long-lived connection, connection churn can be reduced by using a special poxy (Proxy Github Link)
- **Protocol Differences:**
  - Different protocols use different ports. (Port List)
  - AMQP 0-9-1: After successfully opening a connection and authenticating, applications open one or more channels and use them to perform protocol operations, e.g. define topology, consume and publish messages.
  - AMQP 1.0: After successfully opening a connection and authenticating, an application opens one or more sessions. It then attaches links to the session in order to publish and consume messages.
- **Error Handling and Protocol Exceptions:**
  - AMQP 0-9-1: For example, if a channel with the same ID (number) is opened more than once. After sending an error to the client, the connection is closed. Errors that can be corrected and retried are communicated using channel exceptions ("soft errors").
  - AMQP 1.0: In AMQP 1.0, most errors fall into either session errors or link errors. A session error is unrecoverable and leads to all operations received by the peer that detected the error to be discarded until session termination.

Networking:

- Stop new client connection command: *rabbitmqctl suspend_listeners*
- What is epmd? epmd (for Erlang Port Mapping Daemon) is a small additional daemon that runs alongside every RabbitMQ node and is used by the runtime to discover what port a particular node listens on for inter-node communication.
  - The default epmd port is 4369, but this can be changed using the *ERL_EPMD_PORT* environment variable
  - All hosts in a cluster must use the same port.
  - When *ERL_EPMD_PORT* is changed, both RabbitMQ node and epmd on the host must be stopped.

- Routingler icin video: (RabbitMQ in Depth | Message Lifecycle | RabbitMQ Architecture | Networking Concepts | Part 2 (youtube.com)) (Dk: 10-video sonuna kadar)
- Deadletter exchange nedir? Neden kullanilir? (Dead Letter Exchange)

Using TLS and IPSEC for security.

Management Plugin

The RabbitMQ management plugin provides an HTTP-based API for management and monitoring of RabbitMQ nodes and clusters, along with a browser-based UI and a command line tool, rabbitmqadmin.

Example setup can be found here:
Documentation**:https://docs.vmware.com/en/VMware-RabbitMQ-for-Kubernetes/1.4/rmq.pdf**
Page:216

Monitoring

Note: For every cluster node to have its metrics collected, it is still required that the **rabbitmq-management-agent plugin** is enabled on each node, otherwise the metrics from the node won't be available on the UI.

What Are Health Checks?

- Health checks involve executing commands or scripts to collect essential metrics over time.
- Health checks focus on specific metrics at a particular moment in time.
- Monitoring with Prometheus-compatible Tools (Page: 533-537). VMware RabbitMQ for Kubernetes Documentation

RabbitMQ Metrics:
Cluster-wide Metrics (537):

- GET /api/overview

Metric	JSON field name
Cluster name	`cluster_name`
Cluster-wide message rates	`message_stats`
Total number of connections	`object_totals.connections`
Total number of channels	`object_totals.channels`
Total number of queues	`object_totals.queues`
Total number of consumers	`object_totals.consumers`
Total number of messages (ready plus unacknowledged)	`queue_totals.messages`
Number of messages ready for delivery	`queue_totals.messages_ready`
Number of unacknowledged messages	`queue_totals.messages_unacknowledged`
Messages published recently	`message_stats.publish`
Message publish rate	`message_stats.publish_details.rate`
Messages delivered to consumers recently	`message_stats.deliver_get`

Node-wide Metrics (538):

- GET /api/nodes/{node} returns stats for a single node
- GET /api/nodes returns stats for all cluster members

Metric	JSON field name
Total amount of memory used	`mem_used`
Memory usage high watermark	`mem_limit`
Is a memory alarm in effect?	`mem_alarm`
Free disk space low watermark	`disk_free_limit`
Is a disk alarm in effect?	`disk_free_alarm`
File descriptors available	`fd_total`
File descriptors used	`fd_used`
File descriptor open attempts	`io_file_handle_open_attempt_count`
Sockets available	`sockets_total`
Sockets used	`sockets_used`
Message store disk reads	`message_stats.disk_reads`
Message store disk writes	`message_stats.disk_writes`
Inter-node communication links	cluster_links
GC runs	`gc_num`
Bytes reclaimed by GC	`gc_bytes_reclaimed`
Erlang process limit	`proc_total`
Erlang processes used	`proc_used`
Runtime run queue	`run_queue`

Individual Queue Metrics (546):

- GET /api/queues/{vhost}/{qname}

Metric	JSON field name
Memory	`memory`
Total number of messages (ready plus unacknowledged)	`messages`
Number of messages ready for delivery	`messages_ready`
Number of unacknowledged messages	`messages_unacknowledged`
Messages published recently	`message_stats.publish`
Message publishing rate	`message_stats.publish_details.rate`
Messages delivered recently	`message_stats.deliver_get`
Message delivery rate	`message_stats.deliver_get_details.rate`
Other message stats	`message_stats.*` (see HTTP API reference)

Application Level Metrics (539):

Some client libraries and frameworks provide means of registering metrics collectors or collect metrics out of the box. **RabbitMQ Java client, Spring AMQP, and NServiceBus** are some examples.

- Connection opening rate
- Channel opening rate
- Connection failure (recovery) rate
- Publishing rate Delivery rate
- Positive delivery acknowledgement rate
- Negative delivery acknowledgement rate
- Mean/95th percentile delivery processing latency

Individual Node Checks: Several examples of node health check (540-542)
Health Check for Readiness Probe: During node **restarts**, health checks designed for nodes awaiting peer synchronization can prevent the completion of a cluster-wide restart by **assuming** that the **node has fully booted.**
Monitoring of Clusters: There are 2 ways to do this. Prometheus or Management Plugin
Monitoring Tools (545):

Monitoring Tool	Online Resource(s)
AppDynamics	AppDynamics, GitHub
AWS CloudWatch	GitHub
collectd	GitHub
DataDog	DataDog RabbitMQ integration, GitHub
Dynatrace	Dynatrace RabbitMQ monitoring
Ganglia	GitHub
Graphite	Tools that work with Graphite
Munin	Munin docs, GitHub
Nagios	GitHub
Nastel AutoPilot	Nastel RabbitMQ Solutions
New Relic	New Relic RabbitMQ monitoring
Prometheus	Prometheus guide, GitHub
Sematext	Sematext RabbitMQ monitoring integration, Sematext RabbitMQ logs integration

Troubleshooting

Ref Pages: 197-201

Cluster Kubernetes Operator

Needed for management and creation of the rabbitmq cluster.
Page: 170

Messaging Topology Operator

- Can declare queues and policies in a RabbitMQ cluster.
- Can manage exchanges and bindings.
- Can be used to create RabbitMQ users and assign user permissions.
- Can create virtual hosts.
- Can define Federation upstreams.
- Can declare dynamic Shovels.

Page: 201-212

Note: It's **not recommended** setting optional **queue arguments on queues directly**. Once set, queue properties cannot be changed. **Use policies instead**.

Authentication and Authorization

Page: 402

Note: Production environments typically need to pre-configure (seed) a number of virtual hosts, users and user permissions.

Authentication using Client TLS (x.509) Certificate Data:
To authenticate client connections using X.509 certificate a built-in plugin, rabbitmq-authmechanism-ssl, must be enabled and clients must be configured to use the EXTERNAL mechanism. With this mechanism, any client-provided password will be ignored.

This plugin allows RabbitMQ clients to authenticate using x509 certificates and TLS (PKI) peer verification mechanism instead of credentials (username/password pairs).

User permissions:
 # First ".*" for **configure permission** on every entity
 # Second ".*" for **write permission** on every entity
 # Third ".*" for **read permission** on every entity

To grant permissions to a user in a virtual host:
rabbitmqctl set_permissions -p "custom-vhost" "username" ".*" ".*" ".*"

Clear permissions in a virtual host:
rabbitmqctl clear_permissions -p "custom-vhost" "username"

Note:

If the RabbitMQ management UI is configured to require username and password authentication, enabling TLS certificate authentication for clients doesn't bypass this requirement.
In such a setup, even if TLS certificate authentication is enabled for client connections to RabbitMQ, accessing the management UI would still prompt for a username and password. The **TLS certificate authentication would be for the purposes of other types of client connections, such as those made programmatically or through other client applications**, but it wouldn't affect the authentication requirements for accessing the management UI.

Therefore, to access the management UI, you would still need to provide a valid username and password, even if TLS certificate authentication is enabled for other types of connections.

Schema Definition Export and Import

Definitions **can be exported to a file** and then imported into another cluster or used for **schema backup** or **data seeding.**
Page: 391

DEPLOYMENT STEPS

1) Designing the Architecture:

Start by designing a highly available architecture that spans across both data centers. This may involve deploying multiple RabbitMQ instances in a clustered configuration within each data center.
Consider using Kubernetes native features such as StatefulSets to manage the RabbitMQ instances, ensuring stability and persistence of data.

2) Multi-Data Center Setup:
Configure Kubernetes clusters in both data centers, ensuring they are interconnected and can communicate with each other.
Implement replication mechanisms between the RabbitMQ clusters in different data centers to synchronize data and ensure consistency across environments.

3) High Availability Configuration:
Configure RabbitMQ clusters with redundancy and failover mechanisms to ensure high availability within each data center. This may involve deploying multiple RabbitMQ nodes with mirrored queues and automatic failover.

4) Disaster Recovery Planning:
Implement disaster recovery strategies to handle failures and disruptions, including data center outages. This may involve setting up asynchronous replication between RabbitMQ clusters in different data centers and implementing failover procedures.
Test the disaster recovery setup regularly to ensure readiness and effectiveness in case of emergencies.

5) Security Considerations:
Implement security measures such as network segmentation, encryption, and access controls to protect RabbitMQ clusters and data transmitted between data centers. Ensure compliance with relevant security standards and regulations.

6) Monitoring and Alerting:
Set up monitoring and alerting systems to monitor the health and performance of RabbitMQ clusters across both data centers. Use tools such as Prometheus and Grafana to collect metrics and visualize cluster status.
Configure alerts to notify administrators of any anomalies or potential issues that require attention.

7) Deployment Automation:
Use Kubernetes manifests or Helm charts to automate the deployment and configuration of RabbitMQ clusters across both development and production environments.
Implement CI/CD pipelines to streamline the deployment process and ensure consistency between environments.

8) Documentation and Training:
Document the deployment architecture, configuration settings, and disaster recovery procedures to serve as a reference for administrators and operators.
Provide training sessions for the operations team to familiarize them with managing RabbitMQ on Kubernetes and handling high availability and disaster recovery scenarios.

Low Level Design

- Rabbitmq nodelar açılacak.
- kapp-controller, secretgen-controller,tanzu cluster essentials, tanzu package repo? Bunları indirdik. Designda anlatmaya belki gerek yoktur ama imagelerin nereden geldiğini göstermek ya da bunların ne olduğunu ve neden indirdiğimizi açıklamak gerekebilir.
-

HELPFUL LINKS

- General: https://www.cloudamqp.com/docs/index.html
- Configuration Examples:https://github.com/rabbitmq/cluster-operator/blob/main/docs/examples
- Message Topology Operator: https://github.com/rabbitmq/messaging-topology-operator/tree/main/docs/examples
- Logging:https://docs.vmware.com/en/VMware-RabbitMQ-for-Kubernetes/1/rmq/logging.html
- Best Practices: RabbitMQ Best Practices
- Testing: https://github.com/rabbitmq/rabbitmq-perf-test
- SSL/TLS auth: https://www.cshelton.co.uk/blog/2019/12/18/rabbitmq-client-certificate-authentication/
- https://stackoverflow.com/questions/61127677/send-messages-to-rabbitmq-from-outside-kubernetes-cluster-using-nginx-ingress
- https://groups.google.com/g/rabbitmq-users
-

TO BE CLARIFIED

audit-logger: bunu kurmak gerekicek mi acaba. Optional yazıyor ama isterler belki, bu burada kalsın. Ref: dökümanda sayfa: 159
Federation Plugin: Lazım gibi geldi de emin olamadım. sayfa: 293

shovel plugin: bir clusterdan başka bir clustera taşıma işlemini yapıyomuş. Yani sanki buna gerek kalmayacak. sayfa: 299

REQUIREMENTS GATHERING

Session 1: Existing Infrastructure

Subject

Understanding of the existing RabbitMQ infrastructure, including hardware, software, and networking components. Evaluation on the number of queues, exchanges as well as the current connections.

Questions

1. Do you have the license agreements for the VMWare RabbitMQ for Kubernetes product and Cluster Essentials for VMware Tanzu? (Login to https://network.tanzu.vmware.com/ and search for products and accept the EULA's(end user licence agreement)). +
2. Is there any specific Vmware RabbitMQ for Kubernetes version that you prefer or should we go for the latest version which is 1.5? +
3. Could you please share the architecture diagram of the existing Open Source Rabbit MQ implementation? +
4. Can you provide any details/documentation around your current RabbitMQ deployment, including the configuration, version, messaging process and management? +
5. What is the count of the number of policies,virtualhosts,queues, exchanges, bindings, connections etc in the MQ system? (Approve it is correct or not: 960 queues, producers/consumers-470, 700-1200 messages/second? , 3-10 MB)+ (`rabbitmqctl list_queues -p <vhostName>`,`rabbitmqctl list_exchanges --vhost foo`, `rabbitmqctl list_bindings --vhost foo`) +
    a. How do you use classic Queues as of today? +
6. Please share the list of systems sending (Producers) and Receivers (Consumers) of the RabbitMQ messages - Current and future +
7. How much CPU and memory is used by the existing rabbitMQ? Have you encountered any problems related to resources? +
8. How much persistent storage is used by the existing RabbitMQ? +
9. Which plugins are installed in the OpenSource RabbitMQ? (`rabbitmq-plugins list -v`) +
10. Which monitoring tools are currently configured to monitor the existing RabbitMQ? For the EnterpriseMQ which monitoring tool integration is in scope? +

11. How many nodes are there in the existing RabbitMQ cluster? How many nodes should the cluster include for the enterprise version?+
12. Are we gonna deploy Rabbitmq for which environments, like dev, qa, production? Based on my notes: Staging 1 cluster, Production 1 cluster, DR 1 cluster. +
13. Are you currently utilizing vhosts within your messaging infrastructure? If yes, could you please give more details?
    a. Would you prefer that we handle the setup and management of vhosts in the new RabbitMQ cluster, or are you planning to manage them internally?
    b. For the upcoming deployment of the new RabbitMQ cluster, do you anticipate needing the same vhosts as in your current setup, or are there any changes planned? (`rabbitmqctl list_vhosts`) +
14. What are your authentication & authorization strategy requirements for the RabbitMQ cluster?
    a. What built-in authentication mechanism is enabled for the current RabbitMQ installation (Plain,AMQPlain, External etc) , What will be the future state authentication and authorization mechanism?
    b. What authentication and authorization backend mechanisms (LDAP, HTTP, Oauth2, AMQP etc) are enabled currently for the current RabbitMQ installation, What will be the Enterprise RabbitMQ authentication and authorization mechanism?
15. Do you have any running Kubernetes environment currently (needed for start), if yes
    a. What is the kubernetes cluster version?
    b. Do you have a TLS certificate already in place and the certificate manager is already installed on the kubernetes cluster? +
    c. Is the existing RabbitMQ enabled with TLS? +
    d. Is the Kubernetes cluster deployed in an air gapped network, or does it have connectivity to the internet and external networks?
16. Do you have any existing network policies or firewall rules in place that we need to consider? +
17. Do you have any load balancers in your infrastructure? And do you want to use this RabbitMQ cluster? +
18. Which messaging protocols are currently implemented in the OpenSource MQ, Will there be any new protocols for the enterprise MQ implementation? (AMQP 0-9-1, AMQP 1.0, MQTT 3.1, STOMP etc.) +

Session 2: VMware RabbitMQ solution

Subject

Identifying the specific features and functionality required from the VMware RabbitMQ solution, including message queuing, routing, clustering capabilities, High Availability (HA), and Disaster Recovery (DR) features.

Questions

1. How many (estimated) queues do you anticipate needing for your messaging system?

2. What are the characteristics and configurations of the queues, such as durability, message expiration, and maximum queue length?
3. Are there any specific Security, performance, scalability, or reliability requirements that VMware RabbitMQ needs to meet?
4. Are there any specific message attributes or headers that need to be considered for message routing or processing?
5. Do we need to create some exchanges, queues and the bindings (predefined)? Or are you gonna manage them on the client side?
6. What are you planning for a Disaster Recovery setup? Are you looking to maximize resource utilization by having both data centers actively serving traffic (active-active)? Or do you prefer a setup where one data center remains idle until needed as a backup (active-passive) Primary-standby?
7. What are your desired Recovery Time Objective (RTO) and Recovery Point Objective (RPO) for Disaster Recovery? ?
8. Are you gonna use Quorum queues only or both streams and quorum? Is there any specific use case for classic-queues? (Note: Mirroring for classic queues will be deprecated)
9. What is the plan for the monitoring and alerting system? (The rabbitmq default user interface (management plugin)? or Prometheus & grafana implementation?(recommended) or both?) If yes, is there any specific configuration for management plugin?
10. Which protocols do your applications use? (AMQP 0-9-1, AMQP 1.0, MQTT 3.1, STOMP) +
11. Are there any cases where the default replication factor for quorum queues doesn't apply? For instance, situations such as high-priority data needing more redundancy, low-priority data with relaxed durability needs specific replication configurations.
12. What are the data persistence requirements for VMWare RabbitMQ?
13. Any user management and vhost configuration needed? If yes, provide the details.

Session 3: Integration with Kubernetes

Subject

Discussion on the specifications, including integration with Kubernetes, compatibility with existing infrastructure, security considerations and scalability needs.

Questions

1. Service Account creation with RBAC for RabbitMQ on kubernetes cluster? +
2. We will need storageClass, so it should be provisioned for the kubernetes cluster. For production; maybe 500GB. +
3. The communication between kubernetes clusters on different data centers already set? +
4. Where is the stage Kubernetes cluster placed? In the same place as the production cluster. +

5. How often the downstream (standby) node initiates the schema to be synchronized? (There is no interval for message synchronization because messages are synchronized continuously)
    a. Replicates, Schema definitions such as vhosts, quorum queues, users, exchanges, bindings, runtime parameters, and so on.
    b. Replicates, Messages that are published to quorum queues.
6. Is there any specific port needed for management ui or prometheus etc?
7. How do you manage the connections (tcp) on port 5672 and 5671? Cause Ingress nginx controller only supports http(s) based routings. Which means 15672 (management ui) will be ok, but 5672? +

8. Are the applications that will use VMWare RabbitMQ located both inside and outside of Kubernetes? +
9. What testing procedures will be in place to validate the integration of VMWare RabbitMQ with Kubernetes?
10. Are there any existing Kubernetes resources or configurations that we need to align with during the deployment? +
11. VMware Tanzu Network Account credentials? +
12. Do you need any specific kubernetes namespace for the rabbitmq cluster to be placed. +
13. Do you have any used solution currently for secret management in the kubernetes cluster? Is it ok to go with secretMaps directly? +
14. Is there any specific threshold wanted for free disk space on the nodes? (currently set at 50 MB)? +

Session 4: Review and approve

Subject

Review and approve the final requirements.

Questions

…

Dummy Questions

General Questions?

1. Do you have any running Kubernetes environment currently (needed for start), if yes what is the cluster version? +
2. Do you have a TLS certificate already in place and the certificate manager is already installed on the kubernetes cluster? +
3. Do you have a VMware Tanzu Network account already in place? +

4. Do you have the license agreements for the VMWare RabbitMQ for Kubernetes product and Cluster Essentials for VMware Tanzu? (Login to https://network.tanzu.vmware.com/ and search for products and accept the EULA's). +
5. Are the applications that will use VMWare RabbitMQ located both inside and outside of Kubernetes? +
6. Are there any existing Kubernetes resources or configurations that we need to align with during the deployment? +
7. What are the data persistence requirements for VMWare RabbitMQ? Do we need to configure storage solutions for durable message storage and recovery in case of failures? +
8. What testing procedures will be in place to validate the integration of VMWare RabbitMQ with Kubernetes? How do we plan to test message delivery, fault tolerance, and scalability under different scenarios? +?
9. What are you planning for a Disaster Recovery setup? Are you looking to maximize resource utilization by having both data centers actively serving traffic (active-active)? Or do you prefer a setup where one data center remains idle until needed as a backup (active-passive) Primary-standby? +
10. What are your desired Recovery Time Objective (RTO) and Recovery Point Objective (RPO) for Disaster Recovery? +
11. Is there any specific vmware RabbitMQ version that you prefer or should we go for the latest which is 1.5? +
12. Is the Kubernetes cluster deployed in an air gapped network, or does it have connectivity to the internet and external networks? +
13. Service Account creation with RBAC for RabbitMQ on kubernetes cluster? +
14. Are you gonna use Quorum queues only or both streams and quorum? +
15. How many nodes the cluster should include. (And cluster count for each data center) +
16. What is the plan for the monitoring and alerting system? (The rabbitmq default user interface (management plugin)? or Prometheus & grafana implementation?(recommended) or both?) If yes, is there any specific configuration for management plugin etc.? +
17. Do you have any load balancers in your infrastructure? +
18. We will need storageClass, so it should be provisioned for the kubernetes cluster. For production; maybe 500GB. +
19. How to communicate kubernetes clusters? +
20. Which protocols do your applications use? (AMQP 0-9-1, AMQP 1.0, MQTT 3.1, STOMP) +
21. What types of exchanges do you require (e.g., direct, fanout, topic, headers)?
22. Is there any specific scenario or queue where it's imperative that none of its messages are lost, not even a single one?
23. Are there any cases where the default replication factor for quorum queues doesn't apply? For instance, situations such as high-priority data needing more redundancy, low-priority data with relaxed durability needs specific replication configurations. +
24. Do you have any existing network policies or firewall rules in place that we need to consider or integrate with? +
25. Any user management needed, do I need to create users? +

26. Any identity Provider Integration? Are we gonna use any authentication mechanism for the clients and management UI. It supports oAuth 2 (Keycloak,Auth0,Azure,UAA).
27. How do you envision VMware RabbitMQ integrating with your existing VMware infrastructure?
28. Are there any specific Security, performance, scalability, or reliability requirements that VMware RabbitMQ needs to meet? +
29. Which monitoring tools are currently configured to monitor the existing RabbitMQ for availability, performance and security? For the EnterpriseMQ which monitoring tool integration is in scope? +
30. Please list the ports enabled for the existing RabbitMQ installation - External and Internal communication? ??(Bu ne demek)

Queue Questions?

1. How many (estimated) queues do you anticipate needing for your messaging system? +
2. What are the characteristics of each queue, such as durability, message expiration, and maximum queue length?
3. What types of messages will be sent to each queue, and what processing logic is associated with them?
4. Are there any specific message attributes or headers that need to be considered for message routing or processing? +
5. How will messages be routed to queues from the exchanges?

VHOST Questions?

1. Are you currently utilizing vhosts within your messaging infrastructure? How many vhosts are currently in use? +
2. For the upcoming deployment of the new RabbitMQ cluster, do you anticipate needing the same vhosts as in your current setup, or are there any changes planned? +
3. Would you prefer that we handle the setup and management of vhosts in the new RabbitMQ cluster, or are you planning to manage them internally? +
4. Are there any specific requirements or considerations regarding vhosts that we should be aware of for the new RabbitMQ cluster?

Load balancer Questions?

1. Do you currently have a load balancer in your infrastructure that you want to use for this RabbitMQ cluster? +
2. If yes, what type of load balancer is it? (e.g., hardware-based, software-based, cloud-based)
3. What are the specifications and capabilities of the existing load balancer?

4. What are your specific requirements for load balancing in terms of traffic distribution, scalability, and high availability?
5. Do you require Layer 4 (TCP/UDP) or Layer 7 (HTTP) load balancing for your RabbitMQ cluster?
6. Are you using any specific Kubernetes-native load balancing solutions (e.g., Kubernetes Services, Ingress Controllers)?
7. Do you have any preferences for integrating the load balancer with Kubernetes?
8. Are you open to using cloud-based load balancers (if applicable)?
9. Do you prefer managing the load balancer yourself or utilizing managed load balancing services provided by the cloud provider?
10. Are there any security requirements or protocols that the load balancer needs to adhere to, such as SSL termination or firewall rules?
11. Do you require any additional security features like Web Application Firewall (WAF) or DDoS protection?
12. How do you envision scaling the load balancer as traffic increases?
13. Do you require active-active or active-passive redundancy for the load balancer?

Network & Security Questions?

Current Network Setup:

- Can you provide details about your current network setup within your Kubernetes environment?
- Do you have any existing network policies or firewall rules in place that we need to consider or integrate with? +

Security Requirements:

- What are your specific security requirements for the RabbitMQ cluster? +
- Are there any compliance standards or regulations that need to be adhered to (e.g., GDPR, HIPAA)?

Inter-Pod Communication:

- Do you have any specific requirements for restricting inter-node communication within the RabbitMQ cluster?
- Should communication between RabbitMQ nodes be limited to only the necessary ports and protocols?

Client Access Control:

- How do you plan to manage access control for client Pods connecting to the RabbitMQ cluster? +
- Are there specific Pods or services that should be allowed or denied access to the messaging traffic?

Trusted Sources:

- Are there specific IP addresses or ranges that should be allowed to access the RabbitMQ cluster?
- Do you have a list of trusted sources that we should whitelist for communication with the RabbitMQ nodes?

Sample Network Policies:

- Would you like us to provide sample NetworkPolicies based on best practices and recommendations?
- Are there any specific scenarios or use cases you'd like us to address in the Network Policies?

Review and Update Process:

- How frequently do you anticipate reviewing and updating the Network Policies for the RabbitMQ cluster?
- What process will be in place for making changes to the network security configuration as needed?

Monitoring and Enforcement:

- What tools or mechanisms will be used to monitor and enforce network policies within the Kubernetes environment?
- How do you plan to detect and respond to any potential security incidents related to network traffic?

Backup and Disaster Recovery:

- How are you planning to incorporate network security considerations into your backup and disaster recovery strategies for the RabbitMQ cluster?
- Do you have any specific requirements for ensuring network security during failover or recovery scenarios?

Documentation and Training:

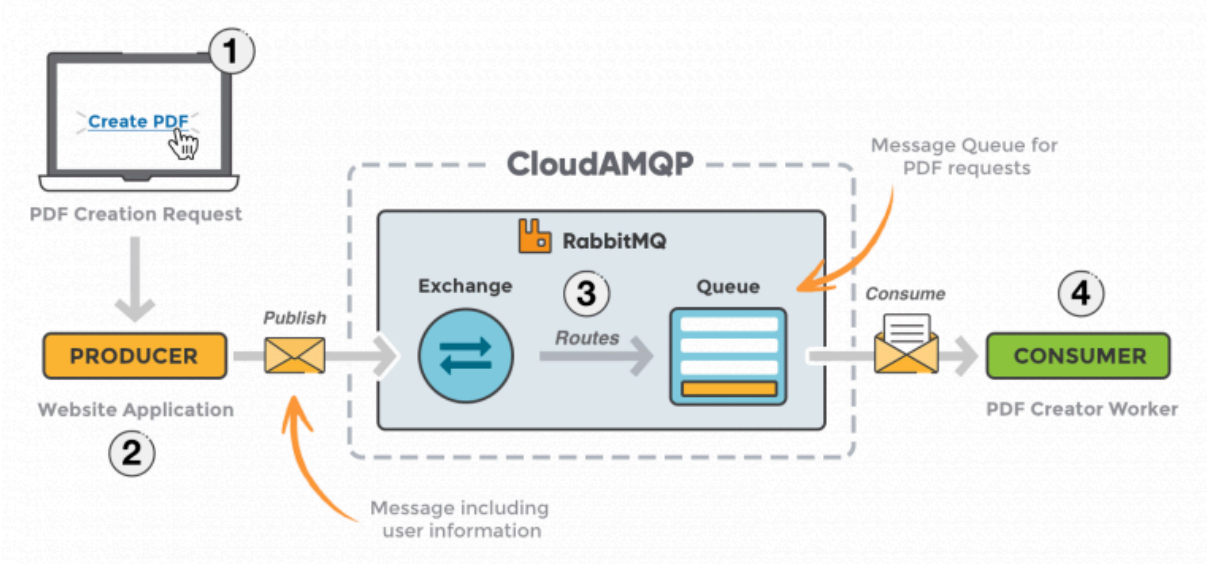
- Do you have any documentation or training materials related to network security best practices that we should review or incorporate into the deployment process?
- Are there any specific requirements for documenting the implemented Network Policies for future reference?

RANDOM INFOS

- VMware RabbitMQ 1.5.3 on Erlang 25.3.2.7
- kapp bulamamamizin sebebi **carvelin** yuklu olmamasiymis.
    - Carvel: Uygulama oluşturma, yapılandırma ve Kubernetes'e dağıtımı destekleyen güvenilir, tek amaçlı, birleştirilebilir araçlar sunan açık kaynaklı bir projedir.
- high available yapabilmek icin policy (bunun sayesinde ha oldugunu soyleyebiliriz) (Building a High Availability RabbitMQ Cluster | by Krzysztof Słomka | Medium)
    - Install RabbitMQ on each node
    - Configure RabbitMQ on each node
    - Set up the RabbitMQ cluster:
        - join islemi

    - **Create a high-availability policy**: Create a RabbitMQ policy to enable queue mirroring for the desired queues
        - `rabbitmqctl set_policy ha-all ".*" '{"ha-mode":"all","ha-sync-mode":"automatic"}'`

- security konusunda rabbitmq ustunde yapilacabilecek tek sey tls onun disindakiler kubernetes ustunde:
    - authentication and authorization: RBAC
    - encryption: TLS/SSL
        - tls 1.3 destekliyor
        - AMQP: 5671 / RabbitMQ Stream Protocol: 5551 / MQTT: 8883 / HTTP API: 15617 (TLS ports)
        - AMQP: 5672 / RabbitMQ Stream Protocol: 5552 / inter-node and CLI communication: 25672 (without TLS ports)
    - network polices: control traffic between rabbitmq clusters / pods / endpoints
    - audit?
    - secrets for credentials
    - persistent volumes
- guzel semalar var bunlara bakabiliriz: Chapter 7. Scaling RabbitMQ with clusters - RabbitMQ in Depth (manning.com) (payment yeri gelince ufak bir refresh atinca duzeliyor)
- rabbitmq ha proxy ve ayakta durmasi icin shell scriptler: RabbitMQ cluster with HAProxy & Keepalived for high availability (github.com)
- kuberneteste deploy edilmis rabbitmq: nanit/kubernetes-rabbitmq-cluster: Deploy-ready rabbitmq cluster for kubernetes (github.com)
- k8s ha rabbitmq: vishnudxb/k8s-rabbitmq-clusters: RabbitMQ cluster running in Kubernetes with high availability (github.com)



- ingress-nginx tcp port forwarding: ingress-nginx/docs/user-guide/exposing-tcp-udp-services.md at main · kubernetes/ingress-nginx (github.com)
    - bunu yapmayi unutma:

```
 args:
    - /nginx-ingress-controller
    - --tcp-services-configmap=ingress-nginx/tcp-services
```

- prometheus rabbiitmq icinde yuklu halde geliyor. (ornek bir kurulum quick overview: Monitoring with Prometheus & Grafana (vmware.com)) (detayli kurulum: Monitoring with Prometheus & Grafana (vmware.com))
- prometheusu aktif hale getirmek icin:
    - rabbitmq-plugins enable rabbitmq_prometheus
- rabbitmq_prometheus default port: 15692
- scrape_config example:

```
scrape_configs:
  - job_name: 'rabbitmq'
    static_configs:
      - targets: ['<rabbitmq-host>:15692']
```

- The Prometheus metrics can be secured with TLS similar to the other listeners. For example, in the configuration file

```
prometheus.ssl.port       = 15691
prometheus.ssl.cacertfile = /full/path/to/ca_certificate.pem
prometheus.ssl.certfile   = /full/path/to/server_certificate.pem
prometheus.ssl.keyfile    = /full/path/to/server_key.pem
prometheus.ssl.password   = password-if-keyfile-is-encrypted
## To enforce TLS (disable the non-TLS port):
# prometheus.tcp.listener = none
```

- To enable TLS with peer verification, use a config similar to

```
prometheus.ssl.port       = 15691
prometheus.ssl.cacertfile = /full/path/to/ca_certificate.pem
prometheus.ssl.certfile   = /full/path/to/server_certificate.pem
prometheus.ssl.keyfile    = /full/path/to/server_key.pem
prometheus.ssl.password   = password-if-keyfile-is-encrypted
prometheus.ssl.verify     = verify_peer
prometheus.ssl.depth      = 2
prometheus.ssl.fail_if_no_peer_cert = true
## To enforce TLS (disable the non-TLS port):
# prometheus.tcp.listener = none
```

- Alert Manager Docs: (Alerting overview | Prometheus)
- Alert Manager example yaml: In this example, if any of the instances are going to be down (up == 0) for one minute, then the alert will be firing.

```
groups:
- name: AllInstances
  rules:
  - alert: InstanceDown
    expr: up == 0
```

```
    for: 1m
    annotations:
      title: 'Instance {{ $labels.instance }} down'
      description: '{{ $labels.instance }} of job {{ $labels.job }} has
been down for more than 1 minute.'
    labels:
      severity: 'critical'
```

- More examples: [Notification template examples | Prometheus]