KPLABS Course

HashiCorp Certified: Consul Associate

Infrastructure & High-Availability

ISSUED BY
Zeal

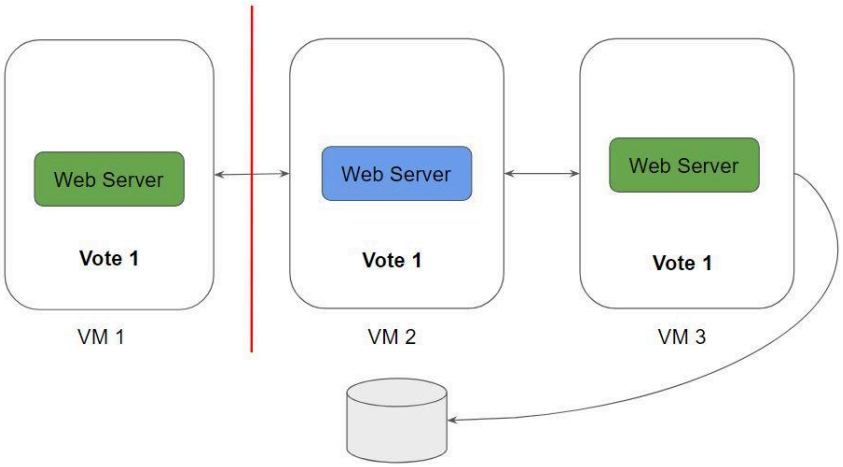
REPRESENTATIVE
instructors@kplabs.in

Module 1: Split-Brain & Importance of Quorum

Swarm manager nodes use the Raft Consensus Algorithm to manage the swarm state. You only need to understand some general concepts of Raft in order to manage a swarm.

There is no limit on the number of manager nodes. The decision about how many manager nodes to implement is a trade-off between performance and fault-tolerance.

Adding manager nodes to a swarm makes the swarm more fault-tolerant.



,

We should maintain an odd number of nodes within the cluster.

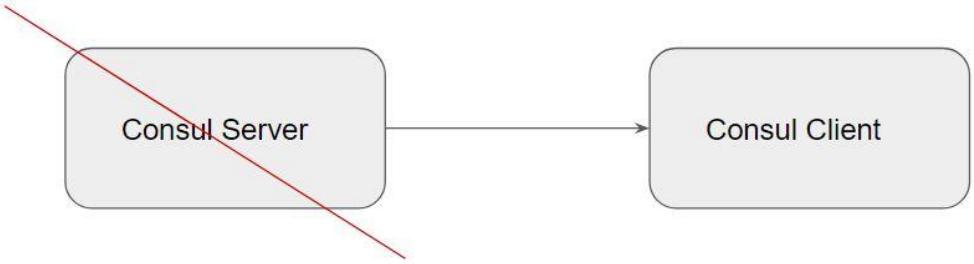
Cluster Size	Majority	Fault Tolerance
1	0	0
2	2	0
3	2	1
4	3	1
5	3	2
6	4	2
7	4	3

Module 2: Implementing High-Availability in Consul

2.1 Understanding the Challenge

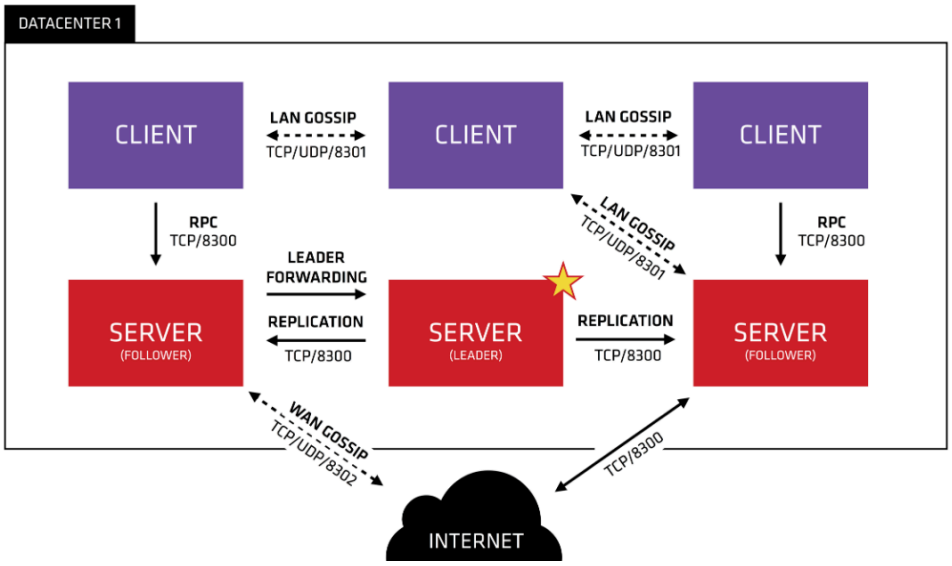
As of now, we have been making use of a single server and a client for our learning.

If the server goes down, then all of your data will be inaccessible and requests would fail.



2.2 Ideal Setup

In a production environment, you should have multiple sets of servers for high-availability.



2.3 Overview of Bootstrap Expect

Bootstrap Expect flag informs Consul of the expected number of server nodes and automatically bootstraps when that many servers are available.

If you have 3 servers that will be part of the cluster, bootstrap-expect should be 3.

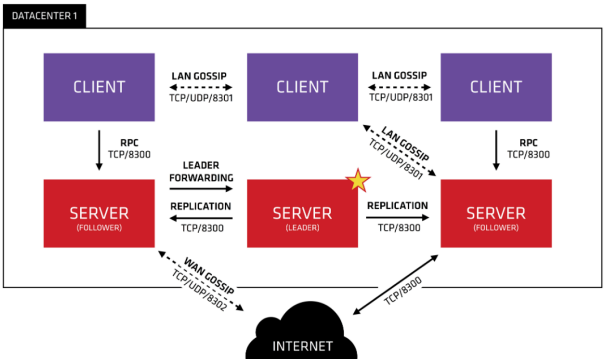
Module 3: Multiple Datacenter in Consul

3.1 Revising Concept of Datacenter

A datacenter is a networking environment that is private, low latency, and high bandwidth.

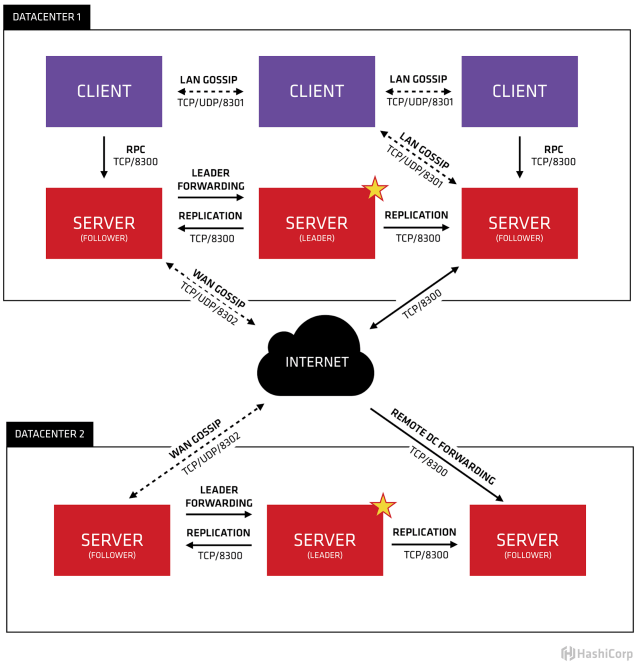
This excludes communication that would traverse the public internet.

Multiple Availability Zone within a single AWS region would be considered part of a single datacenter.



3.2 Multiple Datacenter approach

A large scale organization can be hosting their infrastructure across multiple cloud platforms and even on-premise.



3.3 Important Pointers

All server nodes must be able to talk to each other. Otherwise, the gossip protocol as well as RPC forwarding will not work.

In general, data is not replicated between different Consul datacenters. When a request is made for a resource in another datacenter, the local Consul servers forward an RPC request to the remote Consul servers for that resource and return the results.

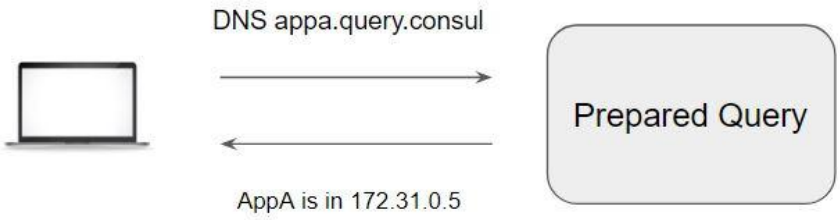
.

Module 4: Overview of Prepared Query

4.1 Understanding Prepared Query

Prepared queries are objects that are defined at the datacenter level.

Once created, prepared queries can then be invoked by applications to perform the query and get the latest results.

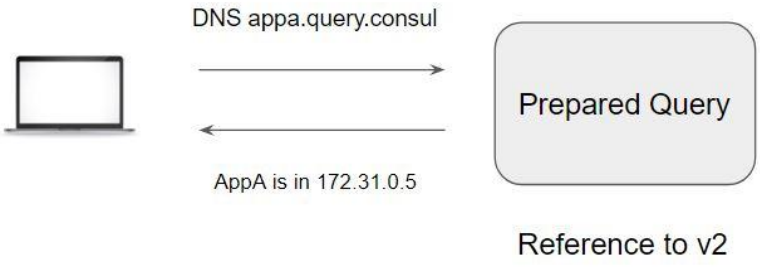


4.2 Use Case 1 - Multiple Versions

Let's assume there are multiple versions of AppA: v1 and v2

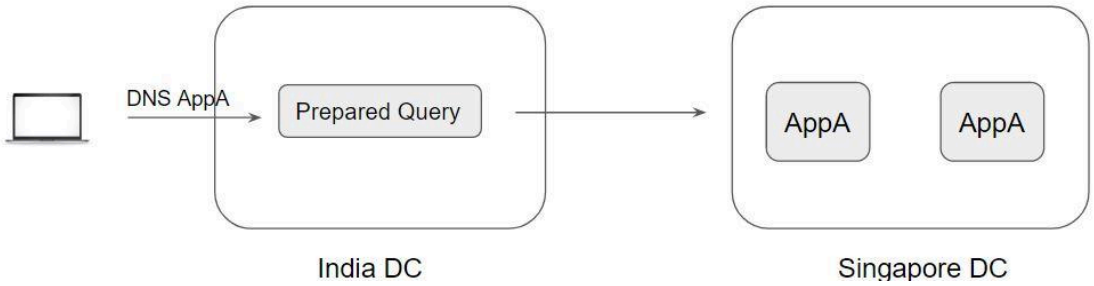
Within the prepared query, we can explicitly specify a version that needs to be returned to the client.

There are no configuration changes required at the client-side.



4.3 Use Case 2 - Failover Policy

You can contact other data centers once there are no healthy instances in the local datacenter.



4.4 Steps Involved in Failover Policy

Consul servers in the local datacenter will attempt to find healthy instances of the "AppA" service within the dc1

If none are available locally, the Consul servers will make an RPC request to the Consul servers in "dc2" to perform the query there.

Finally, an error will be returned if none of these datacenters had any instances available.

Module 5: Backup & Restore

All servers write to the -data-dir before commit on write requests

Consul provides the snapshot command that saves a point-in-time snapshot of the state of the Consul servers. Some of the important data includes:

- Key-Value entries
- the service catalog
- prepared queries
- sessions
- ACLs

Following commands can be used to take backup and restore data from backup.

Commands	Description
consul snapshot save backup.snap	Takes backup and stores it to backup.snap file.
consul snapshot restore backup.snap	Restore data from the backup.

Important Note:

Snapshots will not be saved if the datacenter is degraded or if no leader is available.

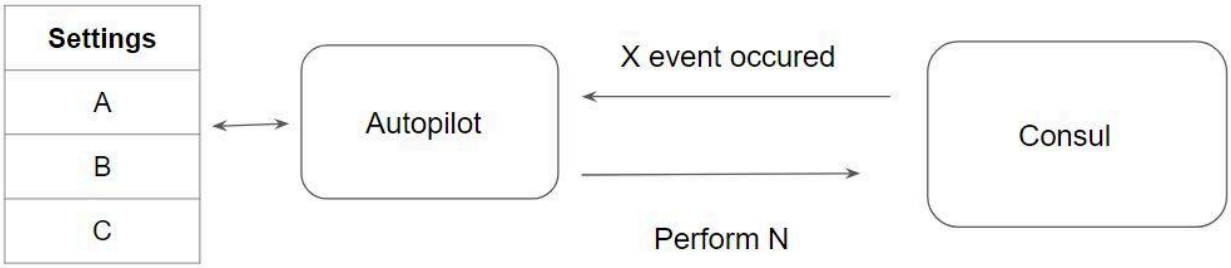
It is possible to run the snapshot on any non-leader server using stale consistency mode. This means that a very small number of recent writes may not be included.

It is recommended to regularly take a backup in an automated way.

Module 6: Overview of AutoPilot

6.1 Overview of AutoPilot Pattern

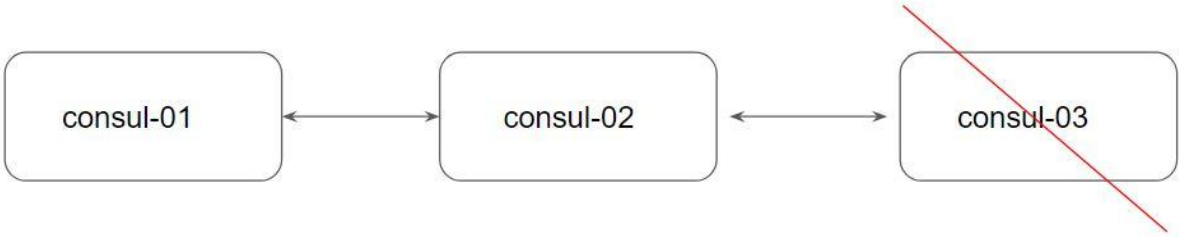
The autopilot pattern automates in code the repetitive and boring operational tasks of an application, including startup, shutdown, scaling, and recovery from anticipated failure conditions for reliability, ease of use, and improved productivity.



6.2 Use-Case 1 - Dead Servers

It will take 72 hours for dead servers to be automatically reaped or an operator must write a script to consul force-leave

Autopilot helps prevent these kinds of outages by quickly removing failed servers as soon as a replacement Consul server comes online.



6.3 Use-Case 2 - Server Stabilization Time

When a new server is added to the data center, there is a waiting period where it must be healthy and stable for a certain amount of time before being promoted to a full, voting member.

This is defined by the ServerStabilizationTime autopilot's parameter and by default is 10 seconds.

In case your configuration requires a different amount of time for the node to get ready, you can tune the parameter and assign it to a different duration.

6.4 Additional Features for Enterprise

There are some additional features of autopilot that are available in Consul Enterprise

- Redundancy Zones
- Automated upgrades

Module 7: Automatically Joining Servers

7.1 Approaches to Join Servers

There are multiple options for joining the servers.

- Specify a list of servers with -join and start_join options.
- Specify a list of servers with -retry-join option.
- Use automatic joining by tag for supported cloud environments with the -retry-join.

You can choose the method which best suits your environment and specific use case.

7.2 Approaches to Join Servers

We have been using this approach for our practicals.

consul join SERVER-NODE

If the server is not available, this command will fail and you will see your cluster is not created.

7.3 Approach 2 - Retry Join

Retry Join is similar to -join but allows retrying a join until it is successful.

consul agent -retry-join "10.0.4.67"

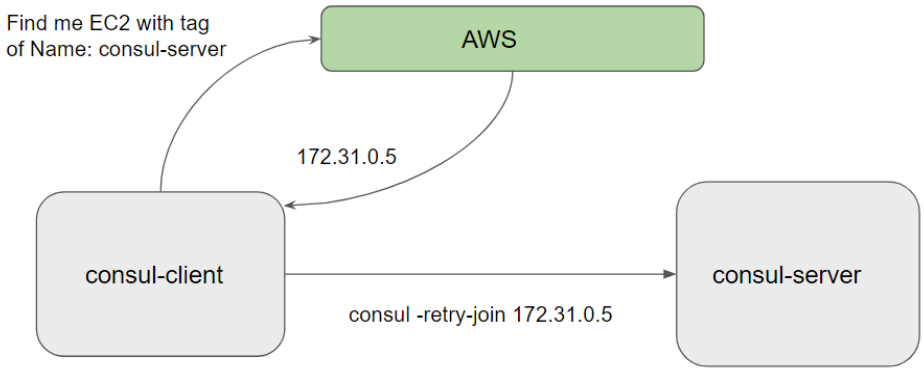
This is useful for cases where you know the address will eventually be available

retry_join could be more appropriate to help mitigate node startup race conditions when automating a Consul cluster deployment.

.

Module 8: Cloud Auto-Join

retry-join accepts a unified interface using the go-discover library for automatically joining a Consul datacenter using cloud metadata.



Working Steps:

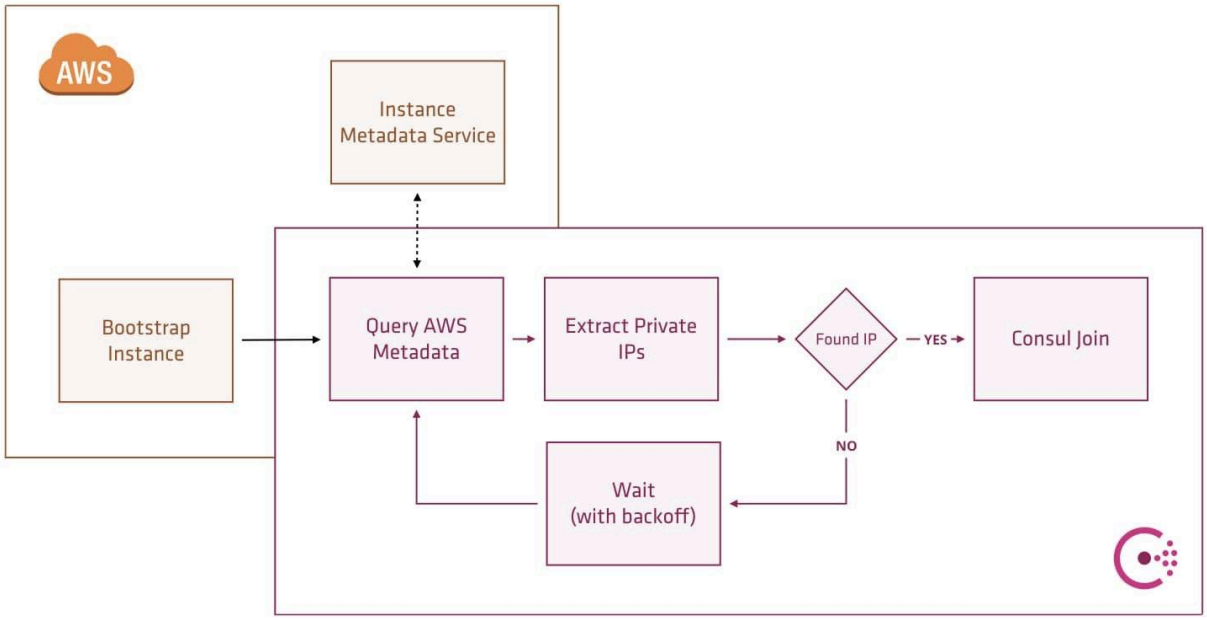
The instance bootstraps and installs consul

Init system starts consul with the configuration to join via EC2 metadata

On start, consul queries the EC2 metadata service with ec2:DescribeInstances to list all instance tags

Consul extracts the private IP addresses of other EC2 instances which have the configured tag name and tag value from the metadata

Consul runs consul join on those private IP addresses



Module 9: Cloud Logs

9.1 Consul Monitor

The monitor command is used to connect and follow the logs of a running Consul agent.

The amount of logged data depends on the overall logging level.

Available log levels are:

- Trace
- Debug
- Info (default)
- Warn
- Err

9.2 Consul Debug

The consul debug command monitors a Consul agent for the specified period of time, recording information about the agent, cluster, and environment to an archive written to the current directory.

```
[root@consul-01 consul.d]# consul debug
[WARN] Unable to capture pprof. Set enable_debug to true on target agent to enable profiling.
==> Starting debugger and capturing static information...
     Agent Version: '1.9.0'
          Interval: '30s'
          Duration: '2m0s'
            Output: 'consul-debug-1606483957.tar.gz'
           Capture: 'metrics, logs, host, agent, cluster'
==> Beginning capture interval 2020-11-27 13:32:37.086429377 +0000 UTC (0)
==> Capture successful 2020-11-27 13:32:37 +0000 UTC (1)
```

Module 10: Reloadable Configuration

Reloading configuration does not reload all configuration items. The items which are reloaded include:

- Services
- ACL Tokens
- Configuration Entry Bootstrap
- Checks
- Discard Check Output
- HTTP Client Address
- Log level
- Node Metadata
- Watches

Join Our Discord Community

We invite you to join our Discord community, where you can interact with our support team for any course-based technical queries and connect with other students who are doing the same course.

Joining URL:

http://kplabs.in/chat