# Consul Service discovery and KV

## CONCEPTS

> Every command talks to a local agent over HTTP (default 127.0.0.1:8500). Point it elsewhere with -http-addr or CONSUL_HTTP_ADDR, and authenticate with CONSUL_HTTP_TOKEN.

**Service** your app
↓ register
**Client agent** on each node
↓ RPC
**Server agents** Raft quorum
↓ query
**Catalog** DNS and HTTP

> Run 3 or 5 servers so Raft keeps a quorum. Losing the majority stops writes; -dev mode is one in-memory server and never for production.

## AGENT

`consul agent -dev`
Run a single in-memory dev server

`consul agent -server -bootstrap-expect=<n> -data-dir=<dir>`
Start a server in a new cluster

`consul agent -data-dir=<dir> -retry-join=<addr>`
Start a client that joins a server

`consul agent -config-dir=<dir>`
Load .json and .hcl config files

`consul validate <file-or-dir>`
Check config before starting

## CLUSTER

`consul members`
List members and their status

`consul leave`
Leave gracefully and shut down

`consul reload`
Re-read config without a restart

## CLUSTER (cont.)

`consul monitor`
Stream logs from the agent

## CATALOG AND SERVICES

`consul catalog services`
List services

`consul catalog nodes`
List nodes

`consul services register <file>`
Register services from HCL or JSON

## KEY-VALUE STORE

`consul kv put <key> <value>`
Set or update a key

`consul kv get <key>`
Read a key

`consul kv get -recurse <prefix>/`
Read all keys under a prefix

`consul kv get -keys <prefix>/`
List key names only

`consul kv delete <key>`
Delete a key

## SNAPSHOTS AND RAFT

`consul snapshot save <file>`
Save a point-in-time snapshot

`consul operator raft list-peers`
Show Raft servers and leader

## ACL

`consul acl token create -policy-name <name> -description <text>`
Create a token for a policy

## DNS

`dig @127.0.0.1 -p 8600 <name>.service.consul SRV`
Get addresses and ports