CHEF™

CODE CAN

Be More Pushy

Michael Ducy - Goat Father - Chef

Why

- Sometimes pull sucks
- Need to “instantly” initiate action
- Need to empower others to perform actions

…..Enter, Chef’s Push Jobs Server

Push jobs
in a command line

- `knife job start -quorum 90% 'chef-client' --search 'role:webapp'`
- Finds all nodes with role webapp
- Submits a job with quorum of 90% to the pushy server.
  - Checks quorum
  - Starts job on available nodes
  - Gathers success and failures
- And will do this for ten nodes...or a thousand

Push jobs
Why not use X?

- We wanted to build a tool that could be deeply integrated into chef.
- Integrated with authentication model
  - Clients use their client key to authenticate to the server
  - Users use their keys to send commands to the api
- Integrated with the authorization model
  - Groups control access now
  - Eventually there will be fine grained ACLs
- Integrated with search and other Chef features
- Scalability

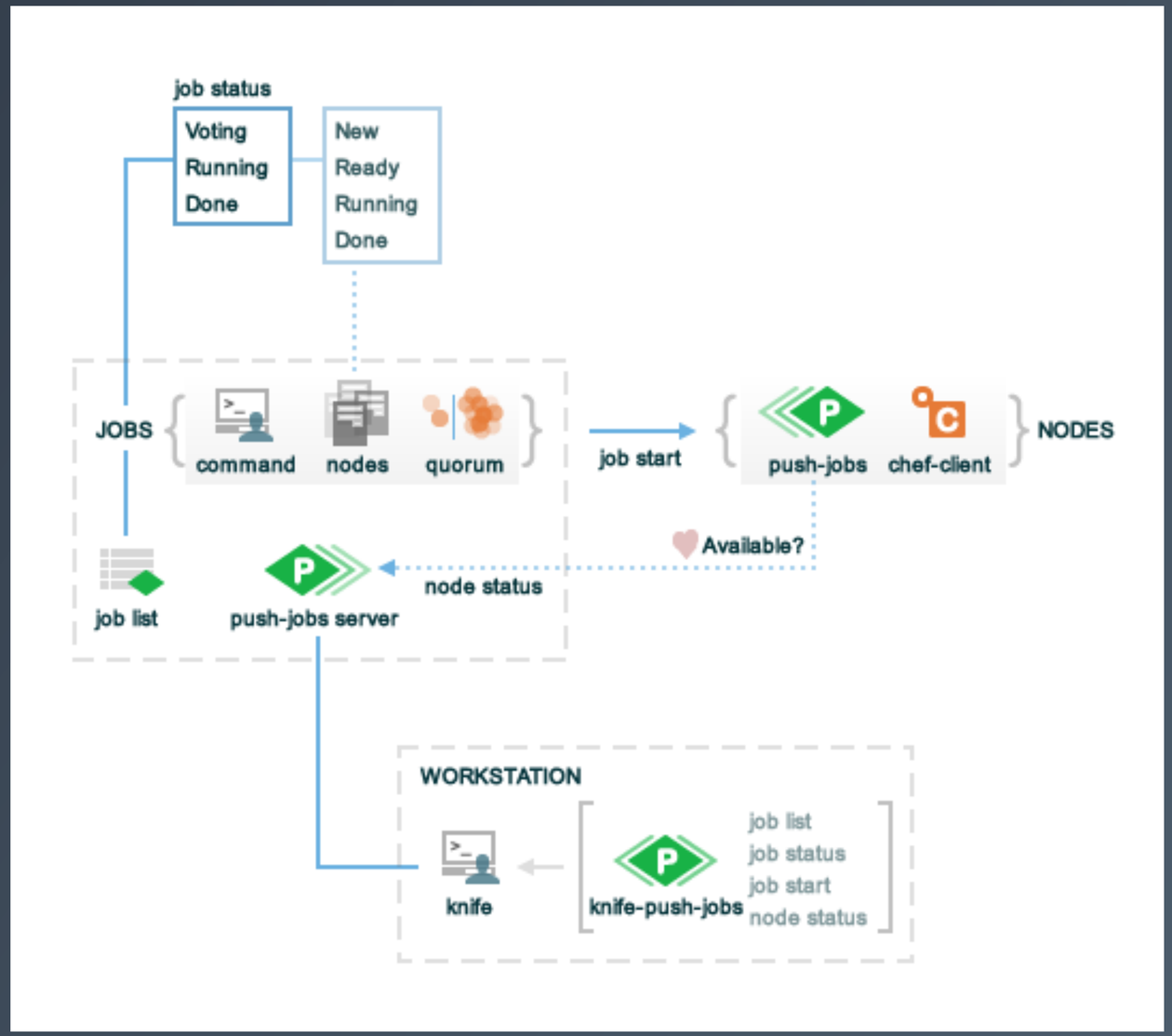
Push jobs Server

- Erlang service
- Extends the Chef REST API
  - Job creation and tracking
  - Push client configuration
- Controls the clients via ZeroMQ
  - Heartbeating to track node availability
  - Command execution
- Modular

Push jobs Client

- Simple ruby client
  - Receives heartbeats from the server
  - Sends back heartbeats to the server
  - Executes commands
- Configuration requirements are minimal
- The client initiates all connections to the server
  - Most configuration is via chef API call using the client key
  - Opens ZeroMQ connections to server for all other communication
- Requires 4 extra ports (10000-10003) open to server

The lifecycle of a job



Push jobs
Knife extension

- All control for pushy jobs is via extensions to the chef API
  - Node status
  - Job control
    - start
    - stop
    - status
  - Job listing

Whitelist

- Whitelist in client config for jobs that can be executed
- Set via attributes

```
"default_attributes": {
    "push_jobs": {
        "whitelist": {
            "chef-client": "chef-client",
            "apt-get-update": "apt-get update",
            "tomcat6_restart": "service tomcat6 restart"
}
```

Pushy Demo

Internals:
Client server interaction

- The client initiates all connections to the server
- The client authenticates to the server and receives
  - A session key and TTL
  - ZeroMQ connection information (ports, heartbeat rate, etc)
- Subscribes via ZeroMQ to server heartbeats (1 to many)
- Connects via ZeroMQ to the server (1-1)
  - Sends heartbeats to the server as long as it receives server heartbeats
  - Awaits commands from the server

Security

- Protocol security
  - We leverage the existing API signing mechanism to exchange session keys
  - All ZeroMQ messages are signed
    - HMAC SHA256 signing protocol protects point to point messages
    - RSA 2048/SHA1 protects broadcast messages (just like the chef API)
- Relies on the SSL chain of trust to the server.

Access control

- Access rights controlled by groups
  - ‘push_job_writers’ group controls job creation and deletion
  - ‘push_job_readers’ group controls read access to job status and results
- Whitelist for commands
  - The client rejects commands that aren’t on the whitelist
- In the future we’d like to do finer grained access control
  - Perhaps persistent job templates with their own access rights and commands

Performance and scalability results

- We can run a job over 2000 nodes
  - 15 sec heartbeats
  - c1.medium
- Bottlenecks
  - Heartbeats consume a lot of resources
  - Everything goes through router process for zeromq messages

More Use Cases

- Orchestration of work via a push_jobs resource
  - Example:
    - Bring up new nodes
    - Install Tomcat
    - Notify HAProxy node to rebuild config and restart
- Continuous Delivery

More Use Cases

```
pushy "chef-client-delay" do
  action :run
  nodes pool_members.uniq
end
```

Push Jobs Walkthrough

Ideas for next steps

- Job templates with ACLs
- Scheduled jobs
- Scaling for Hosted Chef
- Passing job parameters (Reg Exs supported currently)

Availability

- Enterprise Chef only for now
- Hosted Chef deferred
  - Scalability: hosted has more than 2k nodes
  - Security: ZeroMQ messages aren’t encrypted
- Open Source: Really, Really Soon

More Info

- Documentation
  - http://docs.opscode.com/push_jobs.html
- Installing
  - http://docs.opscode.com/install_push_jobs.html
- Walkthrough blog post:
  - http://www.getchef.com/blog/2013/12/16/getting-pushy-with-chef/

Questions?

- First level bullets go here
  - Secondary bullets go here
    - Tertiary bullets go here