SaltStack

Orchestration & Application Deployment

Arnold Bechtoldt
Oberhausen, 07.11.2015
Arnold Bechtoldt
Systems Engineer @ inovex GmbH

› Platform Engineering
› System Automation & Development
› DevOps Support & Consulting
› Open Source Software Contributions
Agenda

1. Orchestration in General
2. SaltStack Orchestration Basics
3. Hands on: Demo
4. Q&A
Assets & Links

github.com/bechtoldt/talk-salt-orchestration  inovex.de  arnoldbechtoldt.com
Orchestration in General
Highway To Heaven

- Asset-Management, Accounting
- Bare-Metal Deployment
- Configuration Management/Containerization
- Orchestration
- Auto-Scaling + Elasticity

Cloud
Who needs Orchestration?

› Lazy Admins: „Don’t repeat yourself“ (DRY)

› Busy Admins: „I have a lot of other problems and […]“

› Small Teams: „Ask X, but she/he is out of office until next week“
Orchestration...

› ... uses \textit{remote execution} to distribute system commands
› ... should be more than just executing commands (most of the tools end here)
› ... should be able to respect internal & external dependencies/relations
› ... should happen deterministic, imperative and \textit{fast}
› ... must \texttt{[RFC2119]} be \textit{easy} to understand/learn (YMMV)
A not-so-cool Example

#!/bin/bash -e
for node in $(mco find --np -C roles::node)
  do
echo "restarting ssh on $node"
mco service --np sshd restart -I $node >/dev/null
sleep 2
mco rpc service status service=sshd -I $node -j | \\
jgrep data.status=running -s data.status >/dev/null
echo "ssh is up on $node"
done

MCollective + Shell Script
For instance, if you use a single-purpose deployment tool [...] to trigger some legacy configuration management, that might be called “orchestrating X with Y”. In reality, it’s usually just replacing a non-scaling [...] server solution X with a stand-alone implementation of Y.
SaltStack Orchestration Basics
SaltStack Orchestration

SaltStack implements the same techniques that other tools do:

› send commands to servers
› send commands to servers that have different operating systems
› send commands only to a subset of servers
› run command A, then B, then C

Some people call this orchestration and wrap this shell-oneliner with thousands LoC.
SaltStack Orchestration

What we really want to have:

› scale across thousands of servers

› easy configuration (less software programming)

› an interface to implement internal/external relationships (pre/post tasks)

› fully automated workflows/procedures (no manual interaction)
SaltStack Orchestration

What we really want to have:

› scale across thousands of servers
  
  **SOA – Message Bus Architecture**

› easy configuration (less software programming)
  
  **YAML – JSON – Python DSL**

› an interface to implement internal/external relationships (pre/post tasks)
  
  **State – Execution – Pillar – Returner – Beacon – Cloud (~ 600 modules)**

› fully automated workflows/procedures (no manual interaction)
  
  **Event System – Reactor – Runner – APIs**
## SaltStack Terminology

<table>
<thead>
<tr>
<th>Component</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>State</td>
<td>A resource should look like <code>{…}</code></td>
</tr>
<tr>
<td>Pillar</td>
<td>Database/CMDB (Files, RDBMS, NoSQL)</td>
</tr>
<tr>
<td>Execution</td>
<td>Execute command X on a server. Used by other Salt components.</td>
</tr>
<tr>
<td>Returner</td>
<td>Log store of job results (DB, Elasticsearch, Syslog, Monitoring)</td>
</tr>
<tr>
<td>Beacon</td>
<td>Special event triggers (inotify, load, procs, network/user activity)</td>
</tr>
<tr>
<td>Reactor</td>
<td>Reacts to events by triggering subsequent procedures (execution modules)</td>
</tr>
<tr>
<td>Cloud</td>
<td>IaaS provisioning (EC2, OpenStack, Digital Ocean, Linode, GCE, VMware)</td>
</tr>
</tbody>
</table>

...
Event-Driven System Automation

<table>
<thead>
<tr>
<th>salt/key</th>
<th>salt/job/new</th>
<th>node/stonith/new</th>
<th>update/application</th>
</tr>
</thead>
<tbody>
<tr>
<td>salt/auth</td>
<td>salt/job/return</td>
<td>scale/out</td>
<td>loadbalancer/config/update</td>
</tr>
<tr>
<td>salt/minion/start</td>
<td>salt/presence/present</td>
<td>monitoring/hosts/new</td>
<td>firewall/config/update</td>
</tr>
<tr>
<td>salt/minion/stop</td>
<td>salt/presence/change</td>
<td>monitoring/hosts/remove</td>
<td>coffee/new</td>
</tr>
</tbody>
</table>
Event Structure

salt/job/20151104191820394966/new {
    "_stamp": "2015-11-04T18:18:20.512126",
    "arg": [
        "orchestration.bootstrap",
        {
            "__kwarg__": true,
            "test": false
        }
    ],
    "fun": "state.sls",
    "jid": "20151104191820394966",
    "minions": ["mw42"],
    "tgt": "mw42",
    "tgt_type": "glob",
    "user": "root"
}

custom/minion/haste_server_started {
    "_stamp": "2015-11-04T18:33:54.650568",
    "cmd": "_minion_event",
    "data": {
        "custom": {
            "onchanges": [],
            "foo": "bar",
            "num": 42,
        },
        "sfun": "wait"
    },
    "id": "mw2",
    "pretag": null,
    "tag": "custom/minion/haste_server_started"
}
Event-Driven System Automation

MySQL
Foreman
etcd
master

minion
minion
minion

PostgreSQL
Docker
AWS

Slack
MySQL
HTTP

SMTP
Elasticsearch

Syslog
Icinga

Appliances
Hands on: Demo
Demo Concept

Salt Master/Reactor | MW hosts | FE hosts
---|---|---
1. Tells MW hosts to install MW | 2. Install Node.JS (MW) | 5. Install Haproxy (FE)
9. Report back to Master
Demo Concept

Salt Master/Reactor

1. Tells MW hosts to install MW
4. Tells FE hosts to install FE
6. Tells MW hosts to deploy App
10. Tells FE to reconfigure FE

MW hosts

2. Install Node.JS (MW)
3. Report back to Master
7. Deploy App
8. Send Notification Mail
9. Report back to Master

FE hosts

5. Install Haproxy (FE)
6. Report back to Master
11. Add/Remove HAProxy backends
12. Report back to Master
Assets & Links

github.com/bechtoldt/talk-salt-orchestration

sh.arbe.io/cloud-provision

youtu.be/9MzeK4u4pkM (demo)
Kolleginnen und Kollegen gesucht!

- Application Development
- Business Development
- Consulting
- Data Management & Analytics
- IT Engineering & Operations

- Hamburg
- Karlsruhe
- Köln
- München
- Pforzheim

inovex.de/jobs
Thank You!

Arnold Bechtoldt
IT Engineering & Operations

inovex GmbH
Ludwig-Erhard-Allee 6
76131 Karlsruhe - Germany

arnold.bechtoldt@inovex.de