# Off the Deep End

Using the Python API to Smartly Manage Complex Tasks

Drew Malone

Devops Guy@Optiv Security

@drawsmcgraw

TacticalProgramming.com

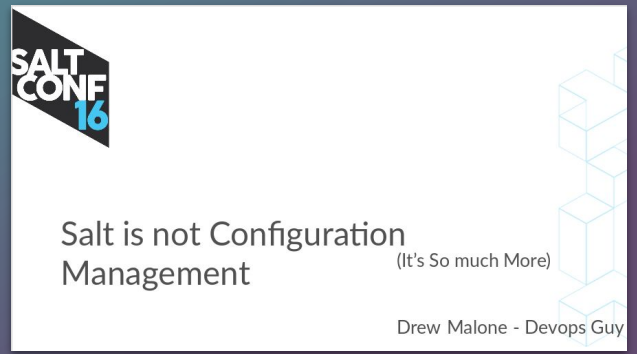
# Agenda

- Salt is not configuration management - It’s a platform
- Why the Python API?
- Command Line vs Python Code
- Handling Job Return Data
- Enterprise API (eAPI)
- Demo
- Security Implications - Today and Long Term

# Salt is a Platform

“Salt is, at its core, a remote execution engine ***upon which*** you can build configuration management.

But if all you’re doing is using Salt for CM, you’re missing out on so much.”



# Why the Python API?

This is for niche, complex tasks (i.e. rolling upgrade of hundreds of compute nodes).

It turns out, *every* task is a niche task.

Bash is limited.

Jinja isn’t made for this much logic.

Reminder: Salt is extremely hackable!

# test.ping

## command line

```
[root@sse-01 ~]# salt '*' test.ping
sse-02.example.local:
    True
sse-03.example.local:
    True
sse-01.example.local:
    True
```

## Python

```
import salt.client
import pprint as pp
client = salt.client.LocalClient()
ret = client.cmd('*',
                 'test.ping',
                 full_return=True)
pp.pprint(ret)
```

# test.ping - return data

```
{'sse-01.example.local': {'jid': '20180820160215728983',
                          'ret': True,
                          'retcode': 0},
 'sse-02.example.local': {'jid': '20180820160215728983',
                          'ret': True,
                          'retcode': 0},
 'sse-03.example.local': {'jid': '20180820160215728983',
                          'ret': True,
                          'retcode': 0}}
```

# SSH Transport - Now More Friendly

`enable_ssh_minions` option on the Master.

Detailed return data in the same format as minion-based transport.

Handle SSH minions as if they were zeromq minions

Available in 2018.3.0

# What about failures?

“No battle plan survives first contact with the enemy”

# test.ping - failure data

```
{'sse-01.example.local': {'jid': '20180820160418573319',
                          'ret': True,
                          'retcode': 0},
 'sse-02.example.local': False,
 'sse-03.example.local': False}
```

# State Runs!

“We haven’t even started yet “

# state.sls

## command line

```
salt sse-01.example.local state.sls haproxy.update_configs
```

# state.sls

## Python

```
ret = client.cmd('sse-01.example.local',
                 'state.sls',
                 ['haproxy.update_configs'],
                 full_return=True)
```

# state.sls (with passion)

## Python

```
nodes = ['sse-01.example.local', 'sse-02.example.local']
pillar = {'foo': 'bar'}
ret = client.cmd(nodes,
                 'state.sls',
                 tgt_type = 'list',
                 kwarg={ 'pillar': pillar },
                 ['haproxy.update_configs'],
                 full_return=True)
```

# What about (state run) failures?

“We purposely broke the state run for a demo.

The results will surprise you!”

# state run - failure data

```
{'sse-01.example.local': {'jid': '20180820180510596065',
                          'out': 'highstate',
                          'ret': ["No matching sls found for 'haproxy.update_config' in env 'base'"],
                          'retcode': 1}}
```

Reminder - this failure is from ‘test.ping’

```
'sse-02.example.local': False,
'sse-03.example.local': False}
```

# I Can’t Trust Anything!?

Don’t write your own code to check the return value.

There’s ~~an app~~ a method for that.

# Checking Return Values

```
import salt.utils
checker = salt.utils.check_state_result
ret = client.cmd(....)
for minion in ret.keys():
  if not checker(ret):
    print “Run failed on minion {0}”.format(minion)
  else:
    print “Run succeeded on minion {0}”.format(minion)
```

## What if I Need More?

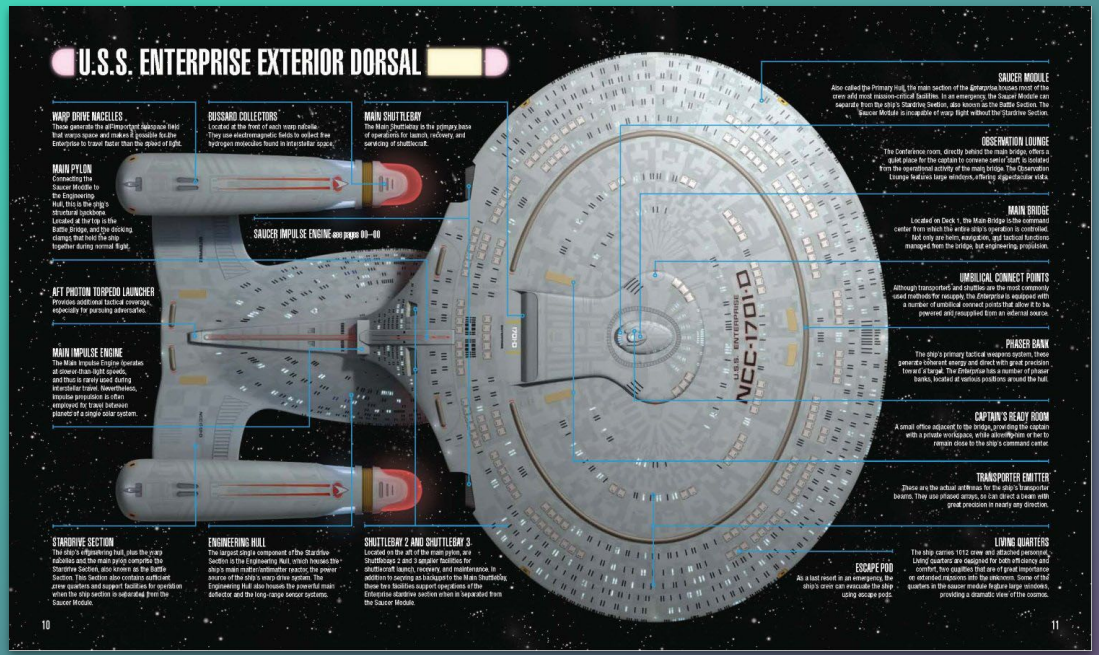
A simple “success” or “failure” doesn’t help much in a post-mortem.

If we’re going to take action, we need to know why things failed.

# Demo

Simple rolling update/restart of haproxy as a demonstration.

# Bonus Round! Saltstack Enterprise API




So many features....

## What can we do with the eAPI?

To name a few:

- Multi-master command and control
- Simple status updates/listings of minions
- Extensive job history details

# How Do I Start?

```
from sseapiclient.tornado import SyncClient
client = SyncClient.connect(server='https://localhost',
                            username='root',
                            password='salt',
                            ssl_validate_cert=False)
```

# Fetch a List of Minions

```
minions = client.api.minions.get_minion_presence()
pp.pprint(minions.ret)
{'sse-01.example.local_master': {
  'sse-01.example.local': {'presence': {'master': ['sse-01.example.local_master'],
                                        'status': 'present',
                                        'timestamp': '2018-08-22T17:37:44.857813'}},

  'sse-02.example.local': {'presence': {'master': ['sse-01.example.local_master'],
                                        'status': 'present',
                                        'timestamp': '2018-08-22T17:37:44.914657'}},

  'sse-03.example.local': {'presence': {'master': ['sse-01.example.local_master'],
                                        'status': 'present',
                                        'timestamp': '2018-08-22T17:37:44.896088'}}
  }
}
```

# Explore the Job Cache

```
returns = client.api.ret.get_returns()
pp.pprint(returns.ret[0])
{u'alter_time': u'2018-08-22T17:37:44.857813',
 u'cmd': u'local',
 u'full_ret': {u'_master_path': [u'sse-01.example.local_master'],
               u'_stamp': u'2018-08-22T17:37:44.838174',
               u'cmd': u'_return',
               u'fun': u'test.ping',
               u'fun_args': [],
               u'id': u'sse-01.example.local',
               u'jid': u'20180822173744949673',
               u'retcode': 0,
               u'return': True,
               u'success': True},
 u'fun': u'test.ping',
<...clipped...>
```

## What does this all mean?

- More control
- Less guesswork
- Easier security
- Less time spent on the mundane

## But what does it mean for *you*?

Two facts:

- Everything is becoming automated
- Security is becoming more important

Put together, what are the consequences of this?

# *This* is what it means for you

Cybersecurity labor crunch to hit 3.5 million unfilled jobs by 2021 - CSO
https://www.csoonline.com/.../cybersecurity-labor-crunch-to-hit-35-million-unfilled-j...
Jun 6, 2017 - ... India alone will need 1 million **cybersecurity professionals** by **2020** to meet the ... Demand for security **professionals** in India will increase in all sectors ... If that's true, then the **cybersecurity** workforce **shortage** is even worse ...

Cybersecurity Has a Serious Talent Shortage. Here's How to Fix It
https://hbr.org/2017/05/cybersecurity-has-a-serious-talent-shortage-heres-how-to-fix-it
May 4, 2017 - ... workforce will have more than 1.5 million unfilled positions by **2020**. ... One way IBM addressing the talent **shortage** is by creating "new collar" jobs, ... Some characteristics of a successfu **cybersecurity professional** simply ...

The Fast-Growing Job With A Huge Skills Gap: Cyber Security - Forbes
https://www.forbes.com/.../the-fast-growing-job-with-a-huge-skills-gap-cyber-security...
Mar 16, 2017 - Some experts predict there will be a global **shortage** of two million **cyber security professionals** by 2019.Shutterstock. Behind every new hack or ...

Global Cybersecurity Workforce Shortage to Reach 1.8 Million as ...
https://www.isc2.org/News-and-Events/Press.../2017-06-07-Workforce-Shortage
Jun 7, 2017 - Global **Cybersecurity** Workforce **Shortage** to Reach 1.8 Million as ... incorporating insights from over 19,000 **cybersecurity professionals**.

Cybersecurity Faces 1.8 Million Worker Shortfall By ... - Dark Reading
https://www.darkreading.com/careers-and.../cybersecurity...18...shortfall.../1329084
Jun 7, 2017 - (ISC)2's Global Information Security Workforce Study, which queried 19,000 **cybersecurity professionals** worldwide, found 66% of survey ...

Global cybersecurity workforce to be short 1.8 million by 2022 ...
https://venturebeat.com/.../global-cybersecurity-workforce-to-be-short-by-1-8-million...
Jun 7, 2017 - Because of the anticipated demand for **cybersecurity professionals** in the ... a $10 million scholarship to tackle **cybersecurity** talent **shortage**.

Cybersecurity world faces 'chronic shortage' of qualified staff • The ...
https://www.theregister.co.uk/2017/.../chronic_shortage_qualified_cybersecurity_bods...
Aug 24, 2017 - **Cybersecurity** world faces 'chronic **shortage**' of qualified staff ... be a 1.5-million worker **shortfall** by **2020**, and then increased it soon after to 1.8 million. ... of them have reported a **shortage** of information security **professionals**.

1.5 million cyber security professionals needed by 2020 - IT Governance

## 1 Day -> 30 minutes

We just reduced the human requirement.

In a time of increasingly scarce security expertise, do you think this is impactful?

# Automating Security

Whether you realize it or not, you (yes, you) are at the forefront of security.

“Who’s patching the cluster this month?”

When you make it easy to do the right thing, people tend to do the right thing.

# Final Thoughts

- Salt is a platform!
- Find new tasking to automate
  Because…
- You are the future of security automation

# Thank You!

Questions?

Resources (slides & code):
https://github.com/drawsmcgraw/saltconf18-demo

Github - @drawsmcgraw

TacticalProgramming.com