Salt

Deploy, Monitor, React.

Nabarun Chatterjee

July 9, 2016

Overview

1. Why Salt
2. Master Minion Mode
3. Main Components
4. Demo1 - States,grains,pillars,reactors,modules
5. Demo2 - Salt ssh
6. Conclusion

Salt

- Agent vs Agentless

Salt

- Agent vs Agentless – Salt has both

Salt

- Agent vs Agentless – Salt has both
    - Master Minion Mode

Salt

- Agent vs Agentless – Salt has both
    - Master Minion Mode
    - Salt ssh mode

Salt

- Agent vs Agentless – Salt has both
    - Master Minion Mode
    - Salt ssh mode
- It is fast

Salt

- Agent vs Agentless – Salt has both
  - Master Minion Mode
  - Salt ssh mode
- It is fast – Uses ZMQ

Salt

- Agent vs Agentless – Salt has both
  - Master Minion Mode
  - Salt ssh mode
- It is fast – Uses ZMQ
- Secure

Salt

- Agent vs Agentless – Salt has both
  - Master Minion Mode
  - Salt ssh mode
- It is fast – Uses ZMQ
- Secure
  - public keys for authentication with the master daemon

Salt

- Agent vs Agentless – Salt has both
  - Master Minion Mode
  - Salt ssh mode
- It is fast – Uses ZMQ
- Secure
  - public keys for authentication with the master daemon
  - AES encryption for payload communication

Salt

- Agent vs Agentless – Salt has both
    - Master Minion Mode
    - Salt ssh mode
- It is fast – Uses ZMQ
- Secure
    - public keys for authentication with the master daemon
    - AES encryption for payload communication
- Open Source

Components

- States

Components

- States
  - Express the state of a host using small, easy to read, easy to understand configuration files

Components

- States
  - Express the state of a host using small, easy to read, easy to understand configuration files
- Grains

Components

- States
  - Express the state of a host using small, easy to read, easy to understand configuration files
- Grains
  - Unit information that helps in targeting minions

Components

- States
  - Express the state of a host using small, easy to read, easy to understand configuration files
- Grains
  - Unit information that helps in targeting minions
- Pillars

Components

- States
  - Express the state of a host using small, easy to read, easy to understand configuration files
- Grains
  - Unit information that helps in targeting minions
- Pillars
  - Pillars are tree-like structures of data defined on the Salt Master and passed through to minions

Components

- States
    - Express the state of a host using small, easy to read, easy to understand configuration files
- Grains
    - Unit information that helps in targeting minions
- Pillars
    - Pillars are tree-like structures of data defined on the Salt Master and passed through to minions
- Reactors

Components

- States
  - Express the state of a host using small, easy to read, easy to understand configuration files
- Grains
  - Unit information that helps in targeting minions
- Pillars
  - Pillars are tree-like structures of data defined on the Salt Master and passed through to minions
- Reactors
  - Trigger actions in response to an event

Components

- States
  - Express the state of a host using small, easy to read, easy to understand configuration files
- Grains
  - Unit information that helps in targeting minions
- Pillars
  - Pillars are tree-like structures of data defined on the Salt Master and passed through to minions
- Reactors
  - Trigger actions in response to an event
- Beacons

Components

- States
    - Express the state of a host using small, easy to read, easy to understand configuration files
- Grains
    - Unit information that helps in targeting minions
- Pillars
    - Pillars are tree-like structures of data defined on the Salt Master and passed through to minions
- Reactors
    - Trigger actions in response to an event
- Beacons
    - Monitor non-Salt processes

All the slides and related material will be updated at https://github.com/nabarunchatterjee/salt-tutorial

Thanks for being patient.

Questions