Configuration Management Tools - Übersicht

In diesem Kapitel erfahren Sie

- welche Configuratin Management Tools verwendet werden können
- wie sich diese Tools unterscheiden

Configuration Management Tools

Unterschiedliche Tools verschiedener Anbieter stehen als open-source oder enterprise Lösung zur Verfügung

- Puppet
- Chef
- Ansible
- Saltstack
- Terraform

Diese Tools bieten Möglichenkeiten zum
Configuration Managment

- der zentralisierten, automatisierten Konfiguration von Netzgeräten (Ressourcen), innerhalb einer veränderbaren, existenten Netzwerkinfrastruktur
  - Puppet
  - Ansible
  - Chef
  - Saltstack

und teilweise für die **Orchestration**

- ein Prozess der auch die notwendigen Ressourcen bereitstellt (virtuelle Devices: CLOUD Server, vSwitches, vRouter, ..) und die zentralisierte, automatisierte Konfiguration für die Ressourcen bereitstellt. Also für veränderbare Netzwerkinfrastrukturen eingestetzt werden kann
  - Terraform
  - Puppet (teilweise)

Wichtige Anforderungen an Configuration Management Tools
ACHTUNG: auch zum Configuration Management mit einem Controller via NBI API.

- **Zentrale Sicherung der Gerätekonfiguration**
  → centralized configuration files

- **Versionskontrolle der Gerätekonfiguration**
  → version control.

- **Überwachung und Sicherstellung der Gerätekonfiguration**
  → configuration monitoring, configuration enforcement

- **Bereistellung der Konfiguration**
  → configuration provisioning

  Templates (Konfigurationsvorlagen), die **Variablen** enthalten, können multiple Geräte innerhalb einer Funktionsgruppe effizient administriert werden und bieten folgende Vorteile:
  - nahezu identische Konfiguration auf den Geräten
  - einfache Konfiguration neuer Geräte
  - einfaches Troubleshooting und einfache Lösungen (identische Lösung auf allen Geräten der Gruppe)
  - übersichtlichere Überwachung und Dokumentation

Ansible, Chef, Puppet

Ansible → www.ansible.com

Notwendige Einstellungen/Bezeichnungen

- **Inventory**: Einstellungen für Geräte inkl. Rollenzuweisungen
- **Templates**: Konfigurationvorlagen → Sprache: **Jinja2** (**Python-Template-Engine**)
- **Variables**: individuelle Einstellungen für Templates → Sprache: **YAML**
- **Playbooks**: Definition von Aktionen

Kommunikation mit Geräten über **SSH** → **agentless architecture**
keine proprietäre Funktion auf den Geräten (agents) zur Kommunkation notwendig - Geräte müssen natürlich **SSH** (und Python) unterstützen.

Push Model: ein Playbook wird gestartet und die Aktionen des Playbook werden ausgeführt.
Beispiel: Template X für Geräte der Gruppe Y am Zeitpunkt Z via SSH einspielen und bei Fehlern die Aktion A durchführen.

Ansible stellt unterschiedliche Module bereit, mit denen Cisco Geräte unterstützt werden.

Puppet → www.puppet.com

Notwendige Einstellungen/Bezeichnungen → mit Puppet-proprietären Sprachen

- **Resource**, **Class**, **Module**: Strukturierung der Geräte .. ein Geräte ist eine Resource innerhalb einer Class innerhalb eines Modules
- **Templates**: individuelle Einstellungen für Templates
- **Manifest**: Durchführung von Aktionen

Kommunikation mit den Geräten über eigene Software → **agent-based architecture**

Pull Model: Der Puppet-Agent fordert Manifeste vom Server an, die ihm mitteilen welche Konfiguration seine ideale Konfiguration ist.
Beispiel: Auf dem anfragenden Gerät, die im Manifest für das Gerät festgelegte End-Konfiguration herstellen

Für Geräte die den Puppet-Agent nicht unterstützen, wie einige Cisco IOS, kann ein **Puppet Proxy** (agent external) verwendet werden, der mit den Geräten via SSH kommuniziert

Chef → www.chef.io

Notwendige Einstellungen/Bezeichnungen → mit Chef-proprietären Sprachen

- **Resource**: Konfigurationsobjekte .. im einfachsten Sinne das Gerät
- **Recipe**: Definition von Aktionen für Resources
- **Cookbooks**: Gruppierung von Aktionen (der Rezepte)
- **Runlist**: Durchführung von Aktionen

Kommunikation mit den Geräten über eigene Software → **agent-based architecture**

Pull Model: Der Puppet-Agent fordert Informatinen vom Server an, die ihm mitteilen welche Konfiguration seine ideale Konfiguration ist.

.. wird von Cisco IOS nicht unterstützt

Übersicht Configuration Tools

Puppet	**Chef**	**Ansible**	**Saltstack**
configuration management/ orchestration	configuration management	configuration management	configuration management
agent-based Puppet Bold **agentless**	**agent-based**	**agentless**	**agent-based** Saltstack SSH **agentless**
Kommunikation **puppet console** Bold: SSH|WinRM	Kommunikation **knife tool**	Kommunikation **SSH** (WinRM)	Kommunikation **GUI** SSH: SSH
declarative	procedural	procedural	declarative
Ruby DSL Bold: JSON	**Ruby** DSL	**Python**, YAML YAML Playbook	**Python**, YAML
pull model	**pull** model	**push** model	**push** model
Devices **masters**/**agents** (**multimaster** achitecture)	Devices **server**/**clients**	Devices **control station**/ **remote hosts** (**prim./second.** architecture)	Devices **master**/**minions**
Configuration **modules & manifests**	Configuration **cookbooks & recipes**	Configuration **playbooks & plays**	Configuration **pillars & grains**

Terraform ist ein **agentless Tool** zur **Orchestration**.

Selbstkontrolle – Aufgaben und Übungen

Welche der folgenden Aussagen treffen zu?

- ☐ Ansible verwendet ein push model, Chef ein pull model.
- ☐ Ansible verwendet ein pull model, Chef ein push model.
- ☐ Ansible verwendet ein push model, Puppet ein pull model.
- ☐ Puppet verwendet ein pull model, Saltstack ein push model.

Welches der folgenden Tools sind agentless Tools.

- ☐ Ansible
- ☐ Puppet
- ☐ Chef
- ☐ Saltstack

Welches der folgenden Tools sind agent-based Tools.

- ☐ Ansible
- ☐ Teraform
- ☐ Chef
- ☐ Saltstack

Welche der folgenden Aussagen sind richtig?

- ☐ Chef kommuniziert über das "knife tool"
- ☐ Ansible kommuniziert über SSH.
- ☐ Puppet kommuniziert über SSH.
- ☐ Puppet kommuniziert über "ncclient"

Welche der folgenden Aussagen ist wahr?

- ○ Puppet verwendet Python und YAML
- ○ Ansible verwendet Python und YAML
- ○ Chef verwendet Python und YAML
- ○ Saltstack verwendet Ruby

Welche der folgenden Aussagen ist wahr?

- ○ Puppet arbeitet prozedural, Ansible arbeitet deklarativ
- ○ Chef arbeitet prozedural, Ansible arbeitet prozedural
- ○ Ansible arbeitet prozedural, Puppet arbeitet deklarativ
- ○ Saltstack arbeitet prozedural, Chef arbeitet dekalrativ

Selbstkontrolle – Lösungen

Welche der folgenden Aussagen treffen zu?

- x☐ Ansible verwendet ein push model, Chef ein pull model.
- ☐ Ansible verwendet ein pull model, Chef ein push model.
- x☐ Ansible verwendet ein push model, Puppet ein pull model.
- x☐ Puppet verwendet ein pull model, Saltstack ein push model.

Welches der folgenden Tools sind agentless Tools.

- x☐ Ansible
- ☐ Puppet
- ☐ Chef
- ☐ Saltstack

Welches der folgenden Tools sind agent-based Tools.

- ☐ Ansible
- ☐ Teraform
- x☐ Chef
- x☐ Saltstack

Welche der folgenden Aussagen sind richtig?

- x☐ Chef kommuniziert über das "knife tool"
- x☐ Ansible kommuniziert über SSH.
- ☐ Puppet kommuniziert über SSH.
- ☐ Saltstack kommuniziert über "ncclient"

Welche der folgenden Aussagen ist wahr?

- O Puppet verwendet Python und YAML
- xO Ansible verwendet Python und YAML
- O Chef verwendet Python und YAML
- O Saltstack verwendet Ruby

Welche der folgenden Aussagen ist wahr?

- O Puppet arbeitet prozedural, Ansible arbeitet deklarativ
- O Chef arbeitet prozedural, Ansible arbeitet prozedural
- xO Ansible arbeitet prozedural, Puppet arbeitet deklarativ
- O Saltstack arbeitet prozedural, Chef arbeitet dekalrativ