DevOps

Automatisierung

Prof. Dr.-Ing. Andreas Heil

v1.0.0

Skripte

- Insbesondere Shell-Skripte (Unix)
- Aber auch Batch, PowerShell unter Windows
- Eigentlich schon immer da...

Brainstorming: Was machen Sie eigentlich alles am Rechner von Hand...?

Vorteile von Skripten

- Einfach zu verstehen
- Straight-forward
- Wird interpretier

Nachteile von Skripten

- Wird interpretiert
- Kompatibilität (z.B. bei unterschiedlichen Versionen und Shell-Varianten wie *bash* und *sh*
- Nur lokale Ausführung
- Keine Parallelität
- Langsam (pro Kommando ein Prozess vgl. Vorlesung Betriebssysteme)

Ziele

- Was wollen wir speziell im Betriebsumfeld skripten?
  - Installationen
  - Updates
  - Konfigurationen

➡ *Configuration Management*

- Was wollen wir speziell als Entwickler?
  - Versionskontrolle

Logische Konsequenz

- Wir schreiben wieder und wieder die gleichen Shell- und Batch-Skripte
- Wir fassen die Skripte, ergänzen diese mit div. Programmen und bauen ein Framework
- Wir generalisieren das Framework so weit, dass es andere verwenden können...

Frameworks

- Salt / SaltStack
- Puppet
- Chef
- Ansible

Was ist Ansible

- Framework zur Automatisierung administrativer Tätigkeiten
- Leicht verständlich
- Deklarativ
- Idempotent
- Modular

Was wird benötigt

- Ansible verwendet keinen zentralen Server
- Alle Aufgaben werden in *Playbooks* gespeichert
- Control Node
  - Python (2.7/3.x)
  - Ansible
- Nodes
  - ssh-server
  - Python (2.7/3.x)

Playbook

- Beschreibt den Zielzustand des Nodes
- Deklarativ
- Nutzt YAML

Variabeln

- Ermöglichen die Wiederverwendung von Playbooks
- Verschlüsselung (z.B. für Passwörter)
- Können für Tasks, Kommandozeile, Dateien etc. verwendet werden

Inventory

- Liste von Servern auf die ein Playbook angewendet werden kann

Referenzen