



Felix Kubli



Jérôme Gygax

Betreuer: Beat Stettler

Projektpartner: INS Institute for networked solutions

## Ausgangslage

Der klassische Netzwerkadministrator konfiguriert seine Netzwerkgeräte manuell. In den immer grösser werdenden Netzwerken ist dies eine mühsame Aufgabe. Deshalb wurden in der Industrie Anwendungen entwickelt, welche es ermöglichen, Netzwerke automatisiert zu verwalten.

Netzwerkautomatisierungs-Lösungen bauen oft auf dem «Network as Code»-Prinzip auf. Das bedeutet, dass ein Netzwerkadministrator seine Geräte nicht wie gewohnt über eine Benutzeroberfläche oder die Shell konfiguriert, sondern Code entwickelt, welcher in seiner Logik die entsprechenden Aktionen zur Konfiguration des Netzwerks beinhaltet.

Der «klassische» Netzwerkadministrator hat häufig nicht viel mit der reinen Softwareprogrammierung zu tun, weshalb diese «Network as Code»-Lösungen eine hohe Einstiegshürde für die Administratoren in die Automatisierung darstellen.

Ziel dieser Arbeit war es, für das beliebte

«Network as Code»-Tool «Nornir» eine Weboberfläche zu entwickeln, so dass auch Netzwerk-administratoren ohne Programmierkenntnisse ihre Netzwerke mit automatisierten Aufgaben verwalten können.

## Ergebnisse

In dieser Arbeit wurde eine Webanwendung mit React.js erstellt, welche über ein RESTful HTTP API mit dem Python Backend kommuniziert. Das Backend stellt mit der API eine Art «Wrapper» für Nornir dar.

In der Webanwendung können Geräte gesucht, gefiltert oder sortiert werden und anschliessend auf eine Auswahl ein sogenannter Task ausgeführt werden. Um in seinem Netzwerk die Downtime so gering wie möglich zu halten, können Tasks zu bestimmten Zeiten geplant werden, etwa während des offiziellen Wartungsfensters.

Das Backend stellt bereits eine kleine Auswahl an Automatisierungstasks zur Verfügung. Versierte Netzwerkadministratoren können diese um eigene Routinen erweitern.

Man stelle sich eine Firma vor, welche zehn Netzwerkadministratoren beschäftigt. Nur einer oder zwei muss dabei Python können und für die Anwendung Tasks entwickeln, aber alle des Teams können von dem enormen Effizienzgewinn durch die Automatisierung profitieren.

## Ausblick

Gewisse Komponenten der Anwendung sind noch statisch, so muss das Inventar an Netzwerkgeräten manuell aktualisiert werden sowie die Job Templates manuell als Skript gespeichert und in der Datenbank eingetragen werden.

Als weitere Ausbaustufen könnte für Anwendung Konnektoren zu bekannten IPAM-Systemen entwickelt werden, um so stehts ein aktuelles Inventar zu besitzen.

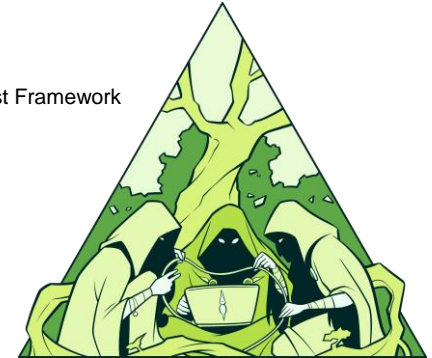
Ebenfalls kann der Anwendung ein «Task-Build» hinzugefügt werden, welcher es unerfahrenen Programmieren erlaubt, mit vorgefertigten «Aktions-Blöcken» neue Automatisierungstasks zu basteln, ohne dabei Python zu programmieren.

## Programmiersprachen

- Python
- JavaScript
- JSX

## Frameworks

- Django
- Django Rest Framework
- Material-UI
- Nornir
- React.js
- Redux



**nornir**

INVENTORY JOB TEMPLATES TASKS DASHBOARD PRECONFIGURED TASKS CONFIGURATION LOGOUT

### Inventory

CREATE TASK WITH SELECTION

Inventory: INS Lab

FILTER SEARCH Search Field

Friendly Name	Hostname	Port	Groups	Platform	Detail View
<input checked="" type="checkbox"/> spine1	10.20.0.201	22	spine	ios	
<input checked="" type="checkbox"/> spine2	10.20.0.202	22	spine	ios	
<input type="checkbox"/> leaf1	10.20.0.203	22	leaf	ios	
<input type="checkbox"/> leaf2	10.20.0.204	22	leaf	ios	
<input type="checkbox"/> leaf3	10.20.0.205	22	leaf	ios	

Rows per page: 25 1 of 5

**nornir**

INVENTORY JOB TEMPLATES TASKS DASHBOARD PRECONFIGURED TASKS CONFIGURATION LOGOUT

### Task Wizard

Select Inventory Select Template Set Variables Finish

BACK SELECT

#	Name	Description	Creator	Detail View
1	hello_world	This prints a hello world	thomastest	
2	Get CDP neighbors	Lists all CDP neighbors	thomastest	
3	Get interfaces	Gets brief information about all interfaces, sh ip int br	thomastest	
4	Ping device	Pings a chosen network device and reports if reachable	thomastest	
5	Get Configuration	Gets all configuration from device	thomastest	

Rows per page: 25 1 of 5

**nornir**

INVENTORY JOB TEMPLATES TASKS DASHBOARD PRECONFIGURED TASKS CONFIGURATION LOGOUT

### Task Dashboard

NEW TASK FILTER SEARCH Search Field

#	Name	Status	Scheduled	Started	Finished	Creator	Template	About Task	Remove Task	Detail View
14	CDP Neighbors spine1	FINISHED		4.12.2020, 12:01:11	4.12.2020, 12:01:12	thomastest	Get CDP Neighbors			

**Task Details** [RETURN](#)

Result

Status: SUCCESS

spine1 / 10.20.0.201

```

Result: Result: {
  neighbor: {
    leaf1: {
      platform: ciscoeos,
      capability: R 1,
      local_interface: Gig 4,
      neighbor_interface: Gig 2
    }
  }
}
  
```

Execution Parameters

Attribute	Value				
ID	14				
Name	CDP Neighbors spine1				
Status	FINISHED				
Date Scheduled					
Date Started	4.12.2020, 12:01:11				
Date Finished	4.12.2020, 12:01:12				
Variables	<table> <thead> <tr> <th>Attribute</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>name</td> <td>CDP Neighbors spine1</td> </tr> </tbody> </table>	Attribute	Value	name	CDP Neighbors spine1
Attribute	Value				
name	CDP Neighbors spine1				
Result Host Selection					
Filters	<table> <thead> <tr> <th>Attribute</th> <th>Value</th> </tr> </thead> <tbody> <tr> <td>hosts</td> <td>spine1</td> </tr> </tbody> </table>	Attribute	Value	hosts	spine1
Attribute	Value				
hosts	spine1				
Created By	thomastest				
Template	Get CDP Neighbors				
Inventory	INS Lab				

Rows per page: 25 1 of 6