

# TEMPORÄR HAUS

**Foolproof Networking**

Automate your Hackspace Network

# Wer bin ich - und warum bin ich heute nicht hier

Johannes Deger:

- Mache Netzwerk an der Uni Ulm.
- War schon ein paar mal hier...

🎩 temporärhaus e.V.



✉️ [johannes.deger@temporaerhaus.de](mailto:johannes.deger@temporaerhaus.de)  
✉️ [hallo@jaydi.de](mailto:hallo@jaydi.de)



# temporärhaus - Ein Hackspace in Neu-Ulm

Meist **zwei** bis **drei** Termine die Woche. Beispielhaft:

- Entwicklung von Projekten, um die Stadt besser zu machen – zum Beispiel selbst gebaute Feinstaubmessstationen
- Veranstaltungen und Workshops rund um die Themen des Internetzeitalters
- Gemeinsamer Arbeitsplatz mit Ausrüstung für alle: **Holzwerkstatt**, **Lötstudio**, **Serverlandschaft**

Wir sind  
Nerds



# Welche Infrastruktur betreiben wir?

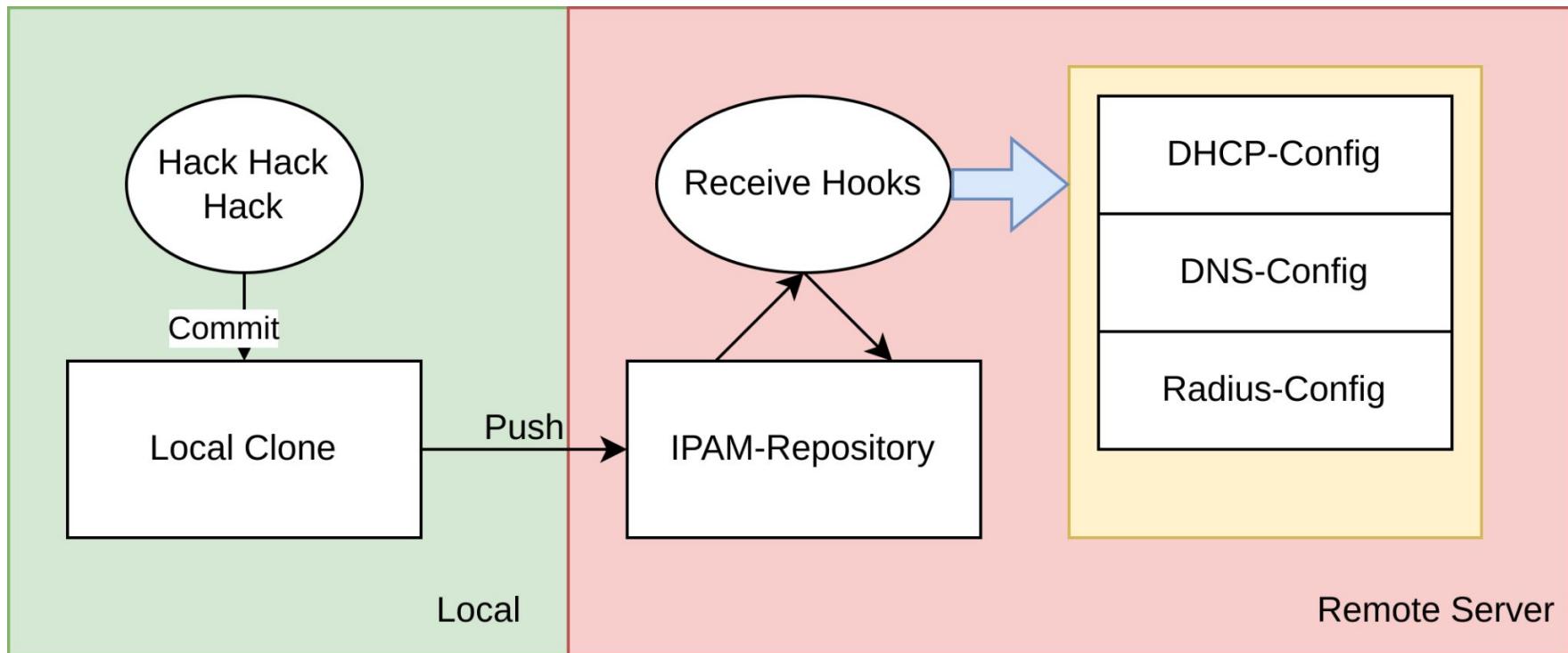
- Netzwerk im Haus. Früher mal **10GBit/s** ans BelWue.
- **Funkbrücken** mit den Bürgernetzen sind geplant.
- **Virtualisierungslandschaft** basierend auf Proxmox.
- Viele, viele **Dienste** für den Betrieb eines Hackspace.
  - Wiki, Pad, Terminfinder, Gitlab, Vereinsverwaltung, Inventarsystem, Asterisk... (~100 VMs)



# Wie managten wir bisher das alles?

Mit dem Management-System der Uni Ulm!





# Management der Switches



## Konfiguration von Hand:

- Vlans auf Switchports.
- Vlan Trunks.
- "Special" Konfigurationen.
- Baseline-Konfiguration.

## Test-Driven Networking:

- Periodische Konsistenzchecks des ganzen Netzes.
- Basierend auf SNMP + REST-API.
- Heuristiken prüfen auch Access-Ports!
- Skaliert sehr gut!

# The old way

## The Good:

- Sehr flexibel.
- Alle Features eines Devices sind nutzbar.
- Vendor-Zoo ist einfach zu bändigen.
- Checks halten die wichtigen Parameter konsistent.

## The Bad:

- Viel manuelle Konfig.
- Man braucht recht viel Know How, um verhältnismäßig einfache Änderungen zu schaffen.

# The Worst: Human Interaction

Leben in nem Hackspace ist nicht wie an der Uni.

- Leute wollen ständig Netzwerkdinge.
- Geräte wandern ständig.
- Leider braucht man immer ein paar “selected few”, die Zugang zum Managementsystem haben.
- Viele können nicht wirklich eine Switch CLI konfigurieren.

**So geht das nicht.**



# Kurzes Brainstorming - Was wollen wir denn?

So viel Automatisierung wie möglich!

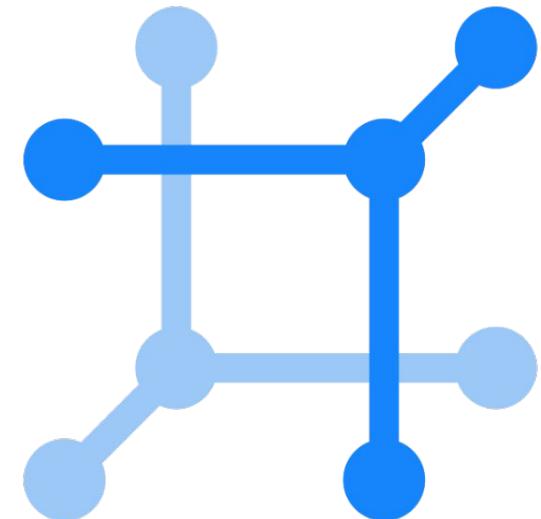
- Source of Truth.
- Automatisches Konfigurieren von Netzwerkgeräten.
- Zentrale Registrierung von Devices.
- Daraus: Generieren von allen relevanten Configs.
- Automatische Zuordnung von Devices in Vlans (LAN + WLAN).

# Schritt 1: Source of Truth!

Haben wir doch?! IPAM? 😱

Puh - vielleicht was mit GUI?!

Schritt 1: Source of Truth!



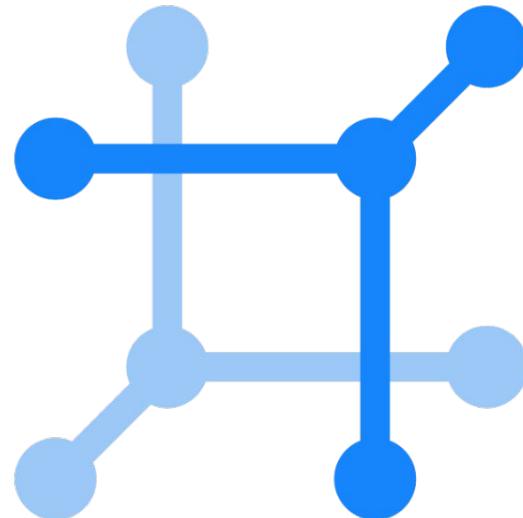
**netbox**

# Schritt 1: Source of Truth!

Netbox it is!

- Entwickelt von Digital Ocean.
- Als Source of Truth zur **Dokumentation** entwickelt.
- Bildet alles mögliche ab!
- Im Prinzip: **Datenstruktur mit WebGUI**
- Kann OIDC!
- Hat ein Rechtesystem.
- Umfangreiche API (REST + GraphQL)
- Custom Fields!

WEB UI 😊



# Schritt 1: Source of Truth!

The screenshot shows the NetBox dashboard interface. On the left is a sidebar with navigation links: Organization, Devices, Connections, Wireless, IPAM, Overlay, Virtualization, Circuits, Power, Provisioning, Customization, Operations, NetBox DNS, and Plugins. The main area contains several data tables:

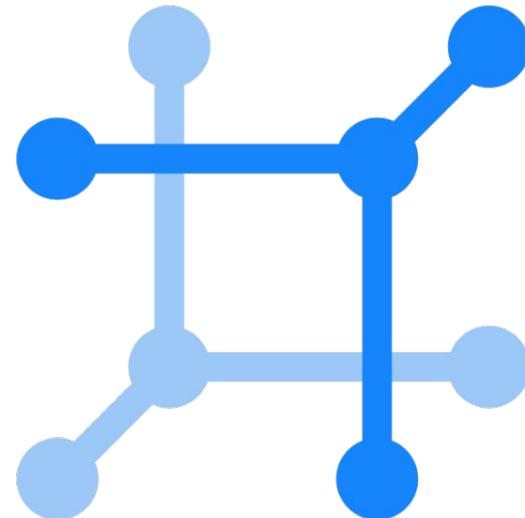
- Organization**: Shows 2 Sites, 1 Tenant, and 4 Contacts.
- Circuits**: Shows 0 Providers, 0 Circuits, and 0 Provider Networks.
- Virtualization**: Shows 1 Cluster and 1 Virtual Machine.
- IPAM**: Shows 0 VRFs, 0 Aggregates, 11 Prefixes, 1 IP Range, 22 IP Addresses, and 5 VLANs.
- DCIM**: Shows 2 Sites, 3 Racks, 791 Device Types, 13 Devices, and 87 Cables.

A central modal window displays a "New Release Available" message for NetBox v3.7.3, with a link to "Upgrade Instructions". To the right, a "Welcome!" panel says: "This is your personal dashboard. Feel free to customize it by rearranging, resizing, or removing widgets. You can also add new widgets using the 'add widget' button below. Any changes affect only your dashboard, so feel free to experiment!". Below it, a "NetBox News" panel features a headline about ServiceNow integrations and a link to "How to Auto Provision Devices with NetBox and Cisco PnP Provision".

# Schritt 1: Source of Truth!

## Design-Paradigmen

- **Stock-Funktionen, wo es nur geht.**
- Bei Bedarf: **Datenmodell mit Bordmitteln erweitern.**



## wendelrack-gs1

Created 2023-10-18 14:03 · Updated 1 month, 1 week ago

[+ Add Components](#) [Bookmark](#) [Clone](#) [Edit](#) [Delete](#)

Device

Interfaces 53

Console Ports 2

Power Ports 1

Config Context

Render Config

Status

LLDP Neighbors

Config

Contacts

Journal

Changelog

Quick search



Configure Table

<input type="checkbox"/>	Name	Label	Enabled	Type	Parent	LAG	MTU	Mode	Description	IP Addresses	Cable	Connection	Untagged VLAN	Tagged VLANs	Mac Auth Bypass	Link Peers	Tags
<input type="checkbox"/>	tph-atelier-ap1		✓	1000BASE-T (1GE)	—	—	—	—	Tagged	—	#104	—	mgmt (124)	access (100) iot (172)	✗	wendeltreppe_patchpanel1 > 1 (1.OG Besprecher:A1)	wifi-uplink
<input type="checkbox"/>	—		✓	1000BASE-T (1GE)	—	—	—	—	Access	—	#46	—	access (100)	—	✓	wendeltreppe_patchpanel2 > 1 (Nest Hinten B1)	
<input type="checkbox"/>	—		✓	1000BASE-T (1GE)	—	—	—	—	Access	—	#105	—	access (100)	—	✓	wendeltreppe_patchpanel1 > 2 (3.OG:A2)	
<input type="checkbox"/>	—		✓	1000BASE-T (1GE)	—	—	—	—	Access	—	#47	—	access (100)	—	✓	wendeltreppe_patchpanel2 > 2 (Nest Hinten B2)	
<input type="checkbox"/>	—		✓	1000BASE-T (1GE)	—	—	—	—	Access	—	#106	—	access (100)	—	✓	wendeltreppe_patchpanel1 > 3 (2.OG:A3)	
<input type="checkbox"/>	—		✓	1000BASE-T (1GE)	—	—	—	—	Access	—	#48	—	access (100)	—	✓	wendeltreppe_patchpanel2 > 3 (Nest Hinten B3)	
<input type="checkbox"/>	tph-theke-ap1		✓	1000BASE-T (1GE)	—	—	—	—	Tagged	—	#107	—	mgmt (124)	access (100) iot (172)	✗	wendeltreppe_patchpanel1 > 4 (EG Theke:A4)	wifi-uplink
<input type="checkbox"/>	—		✓	1000BASE-T (1GE)	—	—	—	—	Access	—	#49	—	access (100)	—	✓	wendeltreppe_patchpanel2 > 4 (Nest Hinten B4)	
<input type="checkbox"/>	—		✓	1000BASE-T (1GE)	—	—	—	—	Access	—	#108	—	access (100)	—	✓	wendeltreppe_patchpanel1 > 5 (2.OG Büro:A9)	
<input type="checkbox"/>	—		✓	1000BASE-T (1GE)	—	—	—	—	Access	—	#50	—	access (100)	—	✓	wendeltreppe_patchpanel2 > 5 (Nest Hinten B5)	

802.1Q Switching

802.1Q Mode Tagged

IEEE 802.1Q tagging strategy

Untagged VLAN access (100)

Tagged VLANs Select Tagged VLANs +

**Wireless**

Wireless role -----

Wireless channel -----

Channel frequency (MHz) Channel frequency (MHz)  
Populated by selected channel (if set)

Channel width (MHz) Channel width (MHz)  
Populated by selected channel (if set)

Wireless LAN group -----

Wireless LANs Select Wireless LANs +

**Custom Fields**

Auto-Negotiation auto

Mac Auth Bypass \* True  
Mac Auth is only allowed for untagged interfaces!  
Is Mac-Auth enabled for the port?

# Netbox: Custom Fields und Konsistenzchecks

## Custom Fields:

- Einfache Erweiterung des Datenmodells durch Custom Fields.

## Konsistenzchecks:

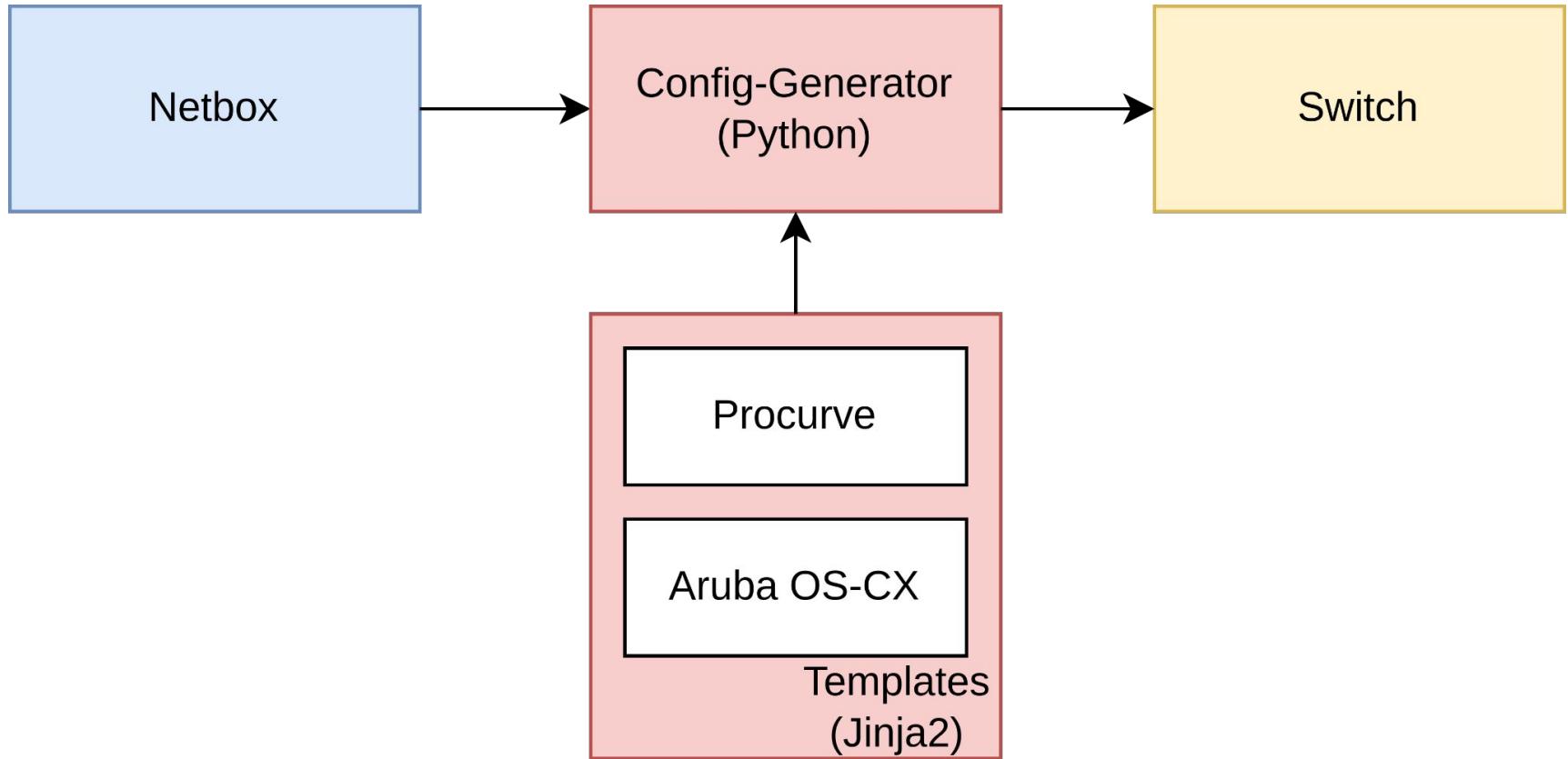
- Verhindern Fehlkonfigurationen.
- Netbox liefert eigene Klasse mit, um Checks durchzuführen.
- Einfache Erweiterung!

# Kurzes Brainstorming - Was wollen wir denn?

So viel Automatisierung wie möglich!

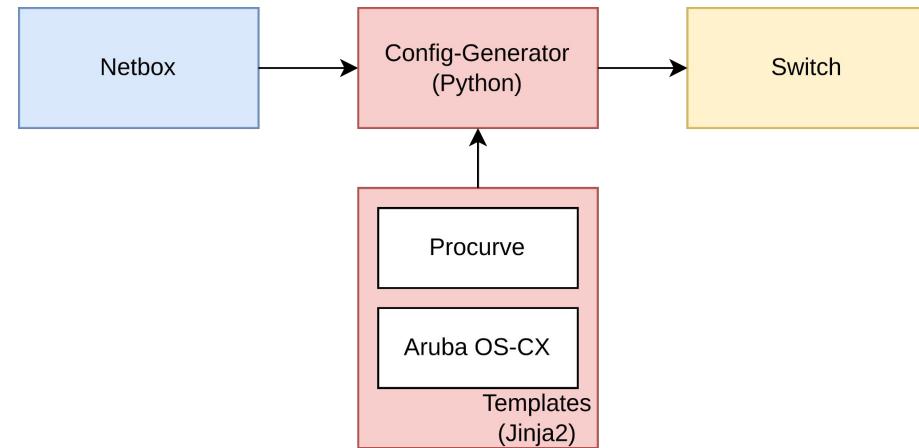
- Source of Truth 
- Automatisches Konfigurieren von Netzwerkgeräten.
- Zentrale Registrierung von Devices.
- Daraus: Generieren von allen relevanten Configs.
- Automatische Zuordnung von Devices in Vlans (LAN + WLAN).

## Schritt 2: Switches automatisch konfigurieren.



## Schritt 2: Switches automatisch konfigurieren.

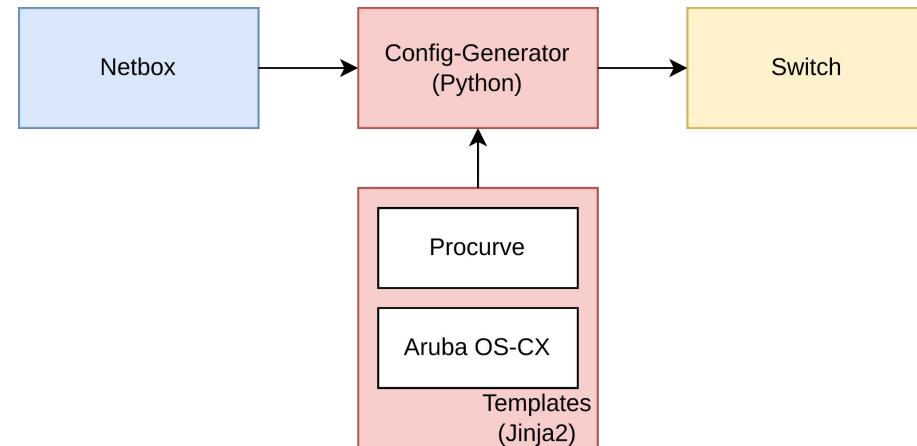
- Python-Script holt alle relevanten Daten aus Netbox.
- Vlans, IPs, Portbeschriftungen, MGMT usw.
- Aus einem Jinja2-Template wird dann eine Switch-Config generiert.
- "Push" auf den Switch.
- **Config Replace.**



# Schritt 2: Switches automatisch konfigurieren.

## Lessons learned:

- Netbox natives Templating ist limitiert. Viel Logik im Jinja2.
- Deswegen: besser externes Script via API ankoppeln.
- Bringt nur, wenn Switches ein sinnvolles Config-Replace können.
- Procurve, Cisco: Top.
- ArubaOS-CX:  
Exzessiv cursed. 🔥🔥🔥



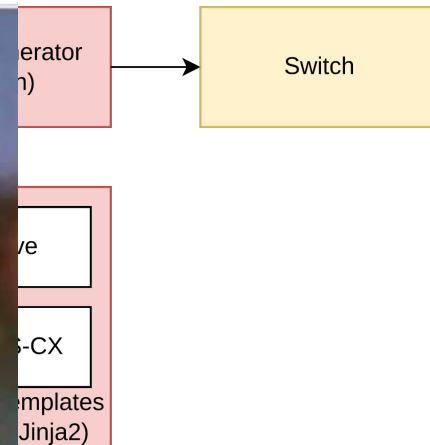
**Doctor: Are you feeling any Pain now?**

Schritt 2: **Network Engineer: Just the normal amount.**

**Doctor: The normal amount is zero.**

Lessons learned: **Network Engineer:**

- Netbox nicht limitiert. Viele Möglichkeiten.
- Deswegen kann man via API anpassen.
- Bringt nichts, was sinnvoll sein könnte.
- Procurve ist zu teuer.
- ArubaOS ist zu exzessiv.



# Kurzes Brainstorming - Was wollen wir denn?

So viel Automatisierung wie möglich!

- Source of Truth 
- Automatisches Konfigurieren von Netzwerkgeräten. 
- Zentrale Registrierung von Devices.
- Daraus: Generieren von allen relevanten Configs.
- Automatische Zuordnung von Devices in Vlans (LAN + WLAN).

# Schritt 3: Zentrale Registrierung von Devices.

How could we do that? 

**Eintragung in Netbox! (Potentiell Self-Service)**

# Kurzes Brainstorming - Was wollen wir denn?

So viel Automatisierung wie möglich!

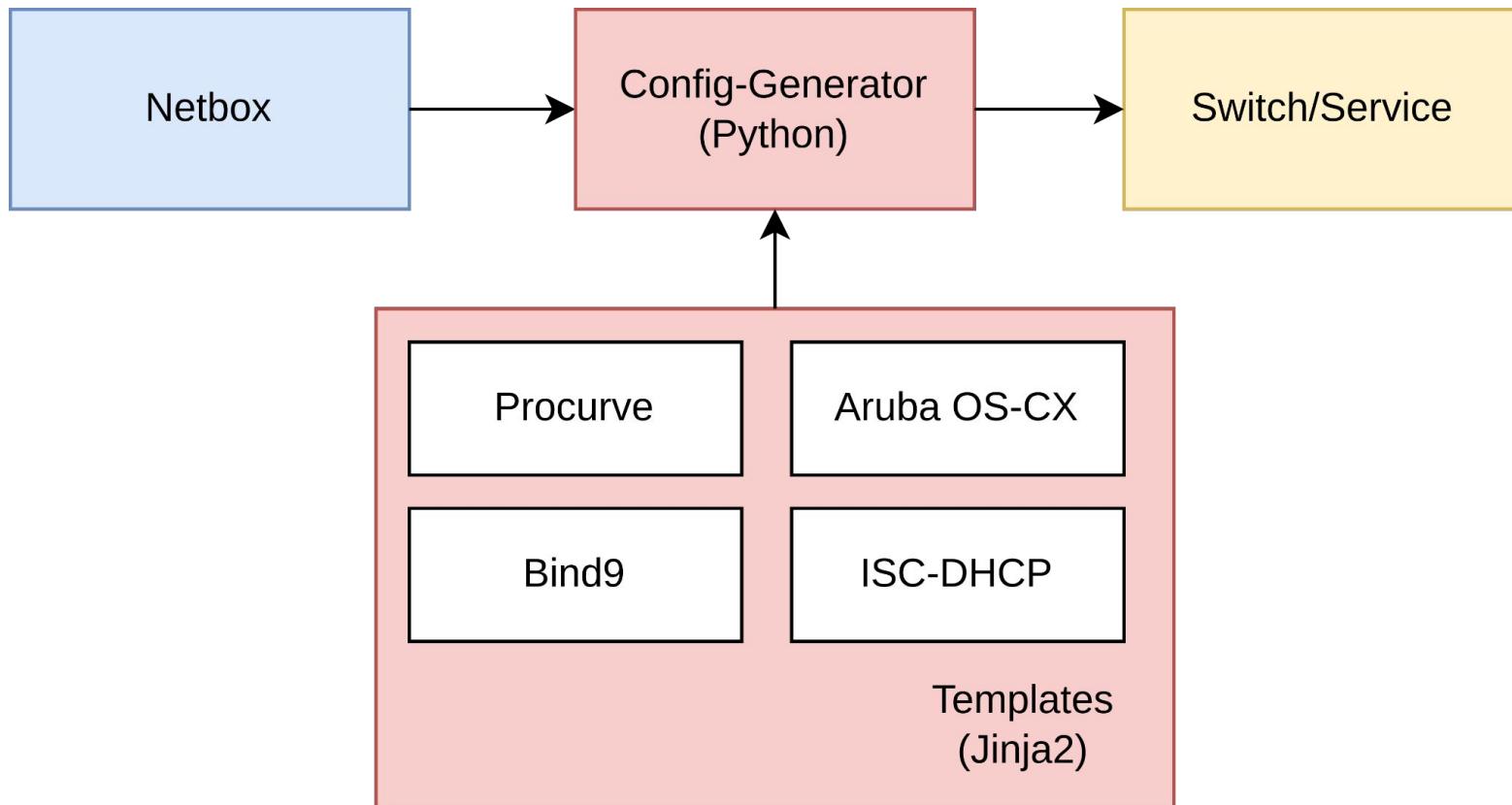
- Source of Truth ✓
- Automatisches Konfigurieren von Netzwerkgeräten. ✓
- Zentrale Registrierung von Devices. ✓
- Daraus: Generieren von allen relevanten Configs.
- Automatische Zuordnung von Devices in Vlans (LAN + WLAN).

# Schritt 4: Generieren von DHCP/DNS-Config

How could we do that?



## Schritt 4: Generieren von DHCP/DNS-Config

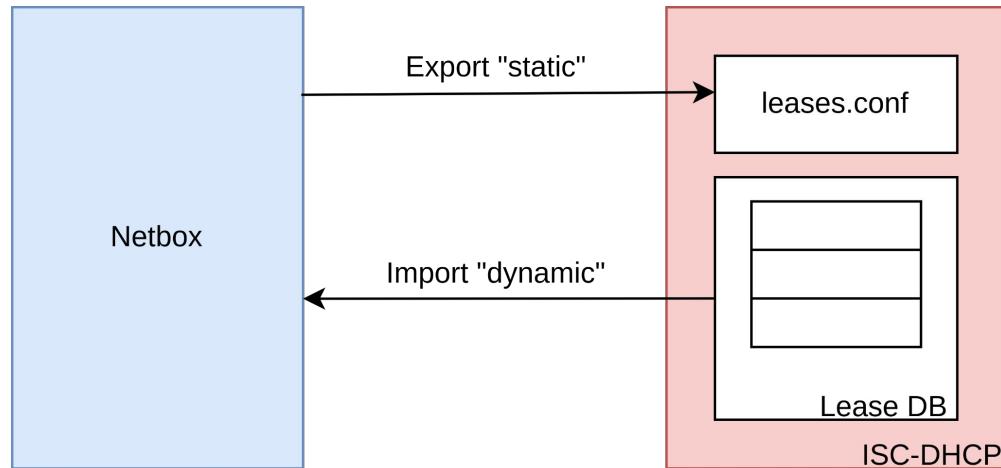


# Schritt 4: Catch bei DHCP

**Es gibt nicht nur Push, sondern auch Pull!**

- DHCP-Server vergibt auch dynamische Leases. Die wollen wir im IPAM sehen.
- Gleichzeitig pushen wir statische Leases auf den DHCP.
- Wie vermeidet man einen Loop?

**TAGS!** 



# Schritt 4: DHCP - Templates

**Nicht alle Clients brauchen gleiche DHCP-Options.**

(Netboot.xyz, Unifi-Gedöns usw.)

Switches bekommen Initialconfig.

**Templates Custom-Field to the rescue.**

# Schritt 4: Catch bei DHCP

IP Addresses

Results 22 Filters Quick search Configure Table

IP Address	Status	Role	Tenant	Assigned	DNS name	Description	Parent	DHCP Template	MAC-Address	Tags	Action
10.72.40.152/16	Active	—	—	—	zer	—	—	—	42:a6:e3:b5:6:d2	dhcp dynamic imported	
10.72.1.21	Active	—	—	—	chromecast-audio	—	—	—	54:60:09:eb:c0:9c	dhcp dynamic imported	
10.72.3.198/21	Active	—	—	—	sep001c58f9dfec	—	—	—	00:1c:58:f9:df:ec	dhcp dynamic imported	
10.72.4.79/21	Active	—	—	—	android-3086f3b800329942	—	—	—	44:ef:bf:be:46:e3	dhcp dynamic imported	
10.72.4.115/21	Active	—	—	—	tph-macbookpro	—	—	—	f8:4d:89:64:28:fa	dhcp dynamic imported	
10.72.6.90/21	Active	—	—	—	win-osjh5uk5qcl	—	—	—	0c:9a:42:2f:2b:fe	dhcp dynamic imported	
10.72.6.126/21	Active	—	—	—	pixel-4a	—	—	—	9a:ec:86:fb:da:ab	dhcp dynamic imported	
10.72.6.129/21	Active	—	—	—	—	—	—	—	02:e1:c3:f6:a1:66	dhcp dynamic imported	
10.72.6.164/21	Active	—	Fiber Garden	—	iphone-bro.fiber.garden	—	—	unifi	f4:be:ec:03:10:50	dhcp static	
10.72.6.249/21	Active	—	—	—	leihladen	—	—	—	ac:fd:ce:f2:ad:a5	dhcp dynamic imported	
10.72.6.250/21	Active	—	—	—	leihladen	—	—	—	ac:fd:ce:f2:ad:a5	dhcp dynamic imported	
10.72.7.100/21	Active	—	—	—	l-xd000885	—	—	—	dc:4f:22:9c:5c:42	dhcp dynamic imported	
10.72.7.190/21	Active	—	—	—	jurek	—	—	—	d0:3c:1f:12:3a:95	dhcp dynamic imported	
10.72.7.232/21	Active	—	—	—	gl-mv1000-a2d	—	—	—	94:83:c4:09:da:2d	dhcp dynamic imported	
10.72.7.250/21	Active	—	—	—	blackbox	—	—	—	00:c2:c6:7e:f0:81	dhcp dynamic imported	
10.72.1.3/32	Active	—	—	—	robbi5-tmbp-2.fiber.garden	—	—	std	8c:85:90:47:d7:f6	dhcp static	
10.72.24.49/21	Active	—	—	✓	wendelrack-gs2.mgmt.fiber.garden.	—	wendelrack-gs2	std	—	—	
10.72.24.56/21	Active	—	—	—	—	—	—	std	—	—	

# Kurzes Brainstorming - Was wollen wir denn?

So viel Automatisierung wie möglich!

- Source of Truth ✓
- Automatisches Konfigurieren von Netzwerkgeräten. ✓
- Zentrale Registrierung von Devices. ✓
- Daraus: Generieren von allen relevanten Configs. ✓
- Automatische Zuordnung von Devices in Vlans (LAN + WLAN).

# Schritt 5: Automatische Vlan Zuweisung

Bis jetzt: Vlan Zuordnung von Switchports in Netbox gemanaged.

Was, wenn ein Device den Switchport wechselt?

Was, wenn es ein reines Wlan-Device ist? Zig SSIDs?

**MAB to the rescue!**

# Schritt 5: Mac-Authentication Bypass

**Vielen bekannt: 802.1x:**

- Passwort/Zertifikatsbasierte Authentifizierung am Switchport/Wlan.
- Auth meist über Radius Server.
- Radius-Server kann Meta-Informationen bei Auth mitgeben. z.B Vlan.

**Das ist ja Passwort-basiert?!**

**Und braucht extra Software auf dem Client?!**

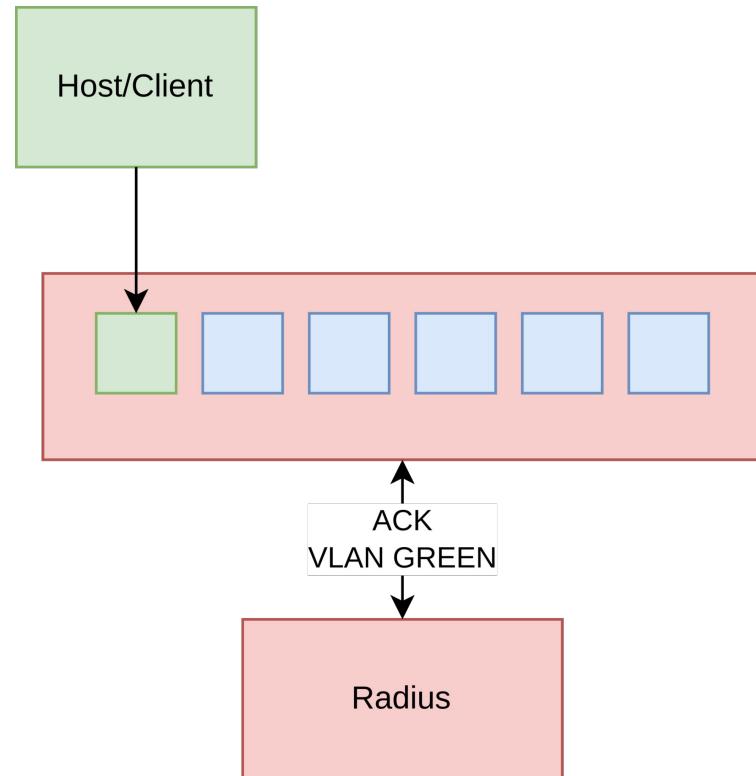


# Schritt 5: Mac-Authentication Bypass

## Besser MAB:

- “Bypass” Authentication based on Mac-Address.
- Switchport lernt Mac-Adresse bei erstem empfangenen Frame.
- Übermittelt diese als Username/Passwort an Radius-Server.
- Radius-Server sagt “Okay + Vlan-ID”.
- Funktioniert im LAN + WLAN!

**Client-agnostisch!**



# Schritt 5: Mac-Authentication Bypass

## Constraints:

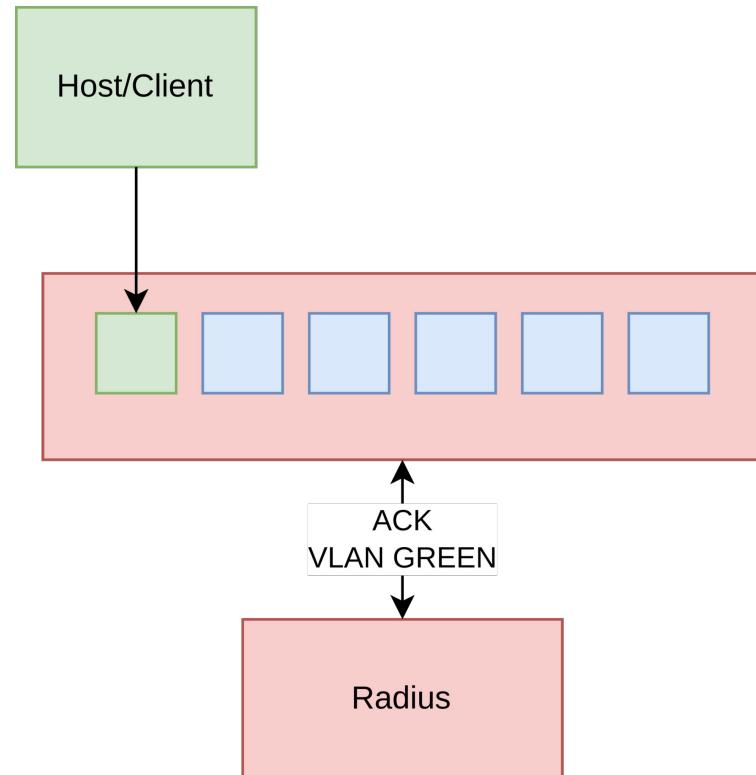
- Wlan: Einfach. Immer ein Client.
- Was macht man, wenn mehrere Clients an einem Port hängen?
- Was ist mit Trunk-Ports?

## Trunk-Ports:

- Geht auch mit MAB! Aber: Es muss die richtige Mac-Adresse im ersten Frame sein. Schwierig bei Switches. Sonst LLDP.

## Dumme Switches an Ports:

- Annahme: Alle nutzen Default-Vlan, außer Ausnahmen.



# Schritt 5: Radius Config

**Prinzipiell recht einfach:**

- Clean up der Mac-Adresse!
- Dann Wifi Auth-Flow durchlaufen.
- Dann Switch Auth-Flow durchlaufen.
- Wenn nicht gefunden, setze default-Vlan und akzeptiere.

**Aber:**

Nicht alle Vendors machen alles gleich 🔥🔥🔥  
Deswegen auch zwei auth-flows...

```
authorize {
    preprocess

    # If cleaning up the Calling-Station-Id...
    rewrite_calling_station_id

    # always check against the authorized_macs file first
    authorized_macs_wifi
    # if found, all is well. Only update Auth-Type
    if (ok) {
        update control {
            Auth-Type := Accept
        }
        update reply {
            Reply-Message := "Hello! You have a static DHCP-Lease :)"
        }
        return
    }
    authorized_macs_switch
    if (ok) {
        update control {
            Auth-Type := Accept
        }
        return
    }

    update reply {
        Tunnel-Type = 13,
        Tunnel-Medium-Type = 6,
        Tunnel-Private-Group-ID = 100
    }
    update control {
        Auth-Type := Accept
    }

    # If this is NOT 802.1x, mac-auth
    if (EAP-Message) {
        reject
    }
}
```

# Schritt 5: Nachworte

## Security - Naja 🤷

- Mac ist einfach spoofbar!
- Sollte nur für Vlans verwendet werden, bei denen es eher unkritisch ist.

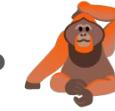
## Lessons learned:

- Timeouts!

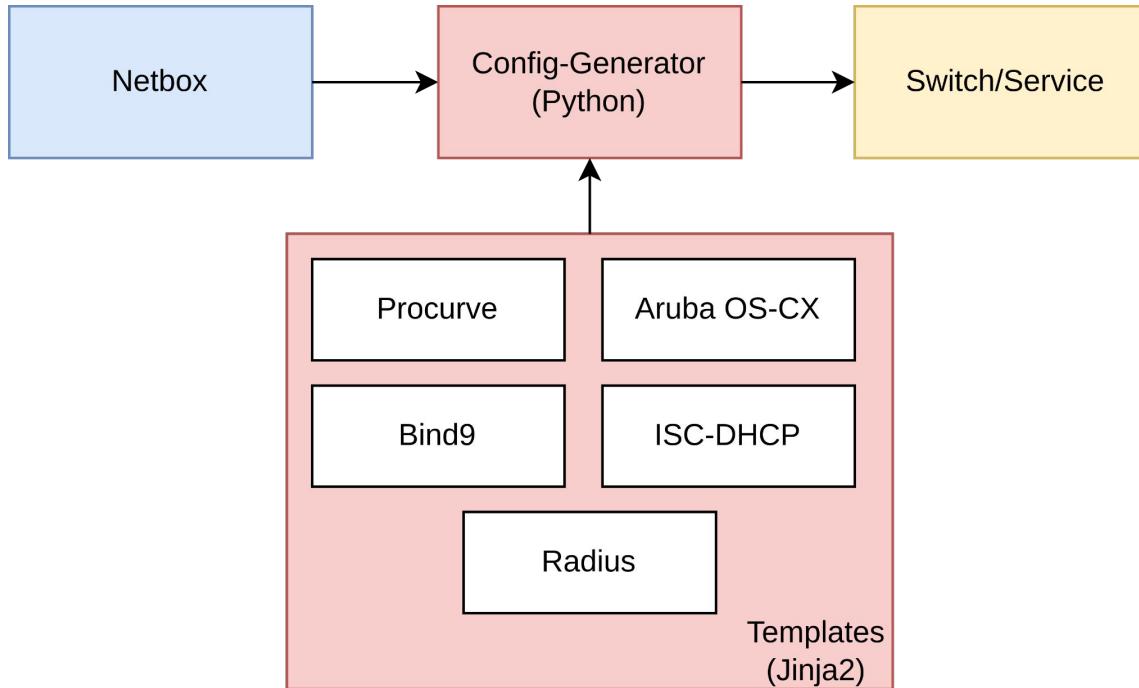
```
#phone-21.voip.fiber.garden.  
00-1E-7A-C3-A9-AD  
    Tunnel-Type = 13,  
    Tunnel-Medium-Type = 6,  
    Tunnel-Private-Group-ID = 132  
#inulm.fiber.garden.  
FE-62-F4-4E-8B-4B  
    Tunnel-Type = 13,  
    Tunnel-Medium-Type = 6,  
    Tunnel-Private-Group-ID = 1000  
#phone-26.voip.fiber.garden.  
FC-FB-FB-CB-EF-C8  
    Tunnel-Type = 13,  
    Tunnel-Medium-Type = 6,  
    Tunnel-Private-Group-ID = 132  
#phone-24.voip.fiber.garden.  
00-1E-BE-91-02-3B  
    Tunnel-Type = 13,  
    Tunnel-Medium-Type = 6,  
    Tunnel-Private-Group-ID = 132  
#brrring.voip.fiber.garden.  
C6-25-C8-FE-EE-A7  
    Tunnel-Type = 13,  
    Tunnel-Medium-Type = 6,  
    Tunnel-Private-Group-ID = 132
```

# Schritt 5: Radius-Config

Woher bekommt der Radius seine Config?



# Schritt 5: Radius-Config



# Kurzes Brainstorming - Was wollen wir denn?

So viel Automatisierung wie möglich!

- Source of Truth 
- Automatisches Konfigurieren von Netzwerkgeräten. 
- Zentrale Registrierung von Devices. 
- Daraus: Generieren von allen relevanten Configs. 
- Automatische Zuordnung von Devices in Vlans (LAN + WLAN). 

**Fertig.**

# Was hat es uns gebracht?

- Gerade die dynamische Zuweisung von Vlans war ein "Quality of Life"-Update!
- Unser Netzwerk ist vollständig dokumentiert ( 4 Switches... )
- Time To Market für neues Features recht gering.
- Wir können auch Laien ran lassen! [citation needed]

# IPAM/DDI: Old vs New?

## Kiz IPAM

- + IPAM-Funktionalität
- + DDI
- + Sehr flexibel.
- Keine Dokumentation darüber hinaus.
- Keine Management der Switches.
- Perl.
- “Spezialisten-Software”

## Netbox

- + IPAM
- + DDI
- + Web-GUI
- + Vollständige Source of Truth.
- + Umfangreiche API.
- + Time To Market bei Anpassungen gering.
- Dual Stack ist Gefrickel.
- Manchmal etwas langsam.
- Viel Frickelei.

# Würde ich NetBox an der Uni Ulm einführen?

😊 JA 😊

Aber...

# Würde ich NetBox an der Uni Ulm einführen?

## **Netbox verschiebt Komplexität:**

- Überwachung des States vs. Konsistenzcheck vor Rollout.
- Verlassen darauf, dass Soll = Ist eher schwierig.
- Kostet teilweise Flexibilität: Neue Features müssen in Templates abgebildet werden.
- Vorteile überwiegen aber ganz klar.

**Open-Source ftw!**

**Andere maintainen!**

 Danke fürs Zusehen! 

Kommt gerne mal vorbei!

**temporärhaus**

 Augsburger Straße 23-25  
89231 Neu-Ulm



johannes.deger@temporaerhaus.de  
hallo@jaydi.de