

DIPLOMARBEIT · WISS · HFWI-2321-DD-40

Einführung von Infrastructure as Code

Analyse, Einführung und erste Umsetzung bei der Netrics AG

Jan Maurice Eymann

Experte: Umberto Annino

Eingereicht: Zürich, 24.08.2026

Zielpublikum & Agenda

Diese Präsentation richtet sich an:

Entscheidungsträger und Team- leitung des Infrastructure Services Teams der Netrics AG

Ziel: Aufzeigen, ob und wie sich die Bereitstellung von Infrastruktur sinnvoll automatisieren lässt.

- 01** Ausgangslage & Problemstellung
- 02** Fragestellung & Zielsetzung
- 03** Methodisches Vorgehen
- 04** Analyse & Werkzeugauswahl
- 05** Pilot und Ergebnisse
- 06** Empfehlung, Reflexion & Ausblick

Das Problem: Infrastruktur entsteht heute von Hand

Manueller Aufbau

- Kundenumgebungen manuell aufgebaut

Inkonsistenz

- Unterschiedlich aufgebaut
- Anleitungen werden nicht gelebt

Fehlende Nachvollziehbarkeit

- Änderungen sind schwer Nachvollziehbar

Bus-Faktor

- Wissen ist an einzelne Personen gebunden
- Probleme bei Abwesenheit und Personalwechsel

Fragestellung & Zielsetzung

1

Adressierbarkeit

- Welche Schwachstellen lassen sich durch IaC adressieren?

2

Umsetzung & Messbarkeit

- Wie kann ein konkretes Umsetzungskonzept aussehen?
- Welche Verbesserungen sind praktisch messbar?

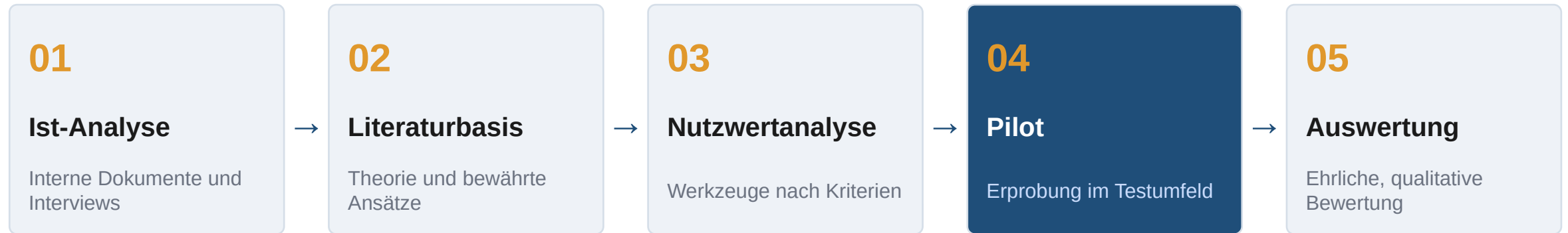
3

Voraussetzungen

- Welche Voraussetzungen müssen im Netrics Umfeld erfüllt werden?

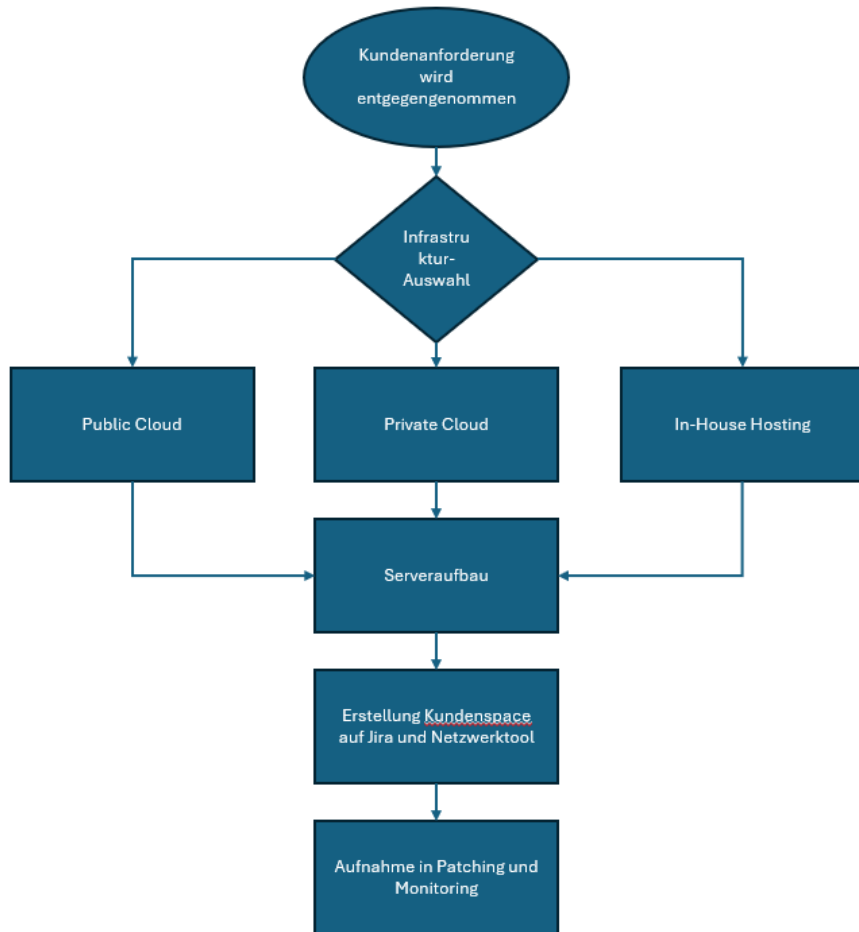
Zielsetzung: bestehende Prozesse strukturiert aufnehmen und bewerten, Potenzial und Voraussetzungen von IaC analysieren, ein Umsetzungskonzept erstellen und in einem Piloten praktisch erproben.

Methodisches Vorgehen



Fallstudienansatz im konkreten betrieblichen Kontext des Teams Infrastructure Services.

14 Schwachstellen entlang von drei Kriterien



Kriterien: Standardisierung · Aufwand · Nachvollziehbarkeit

9

vollständig durch IaC adressierbar

4

teilweise adressierbar

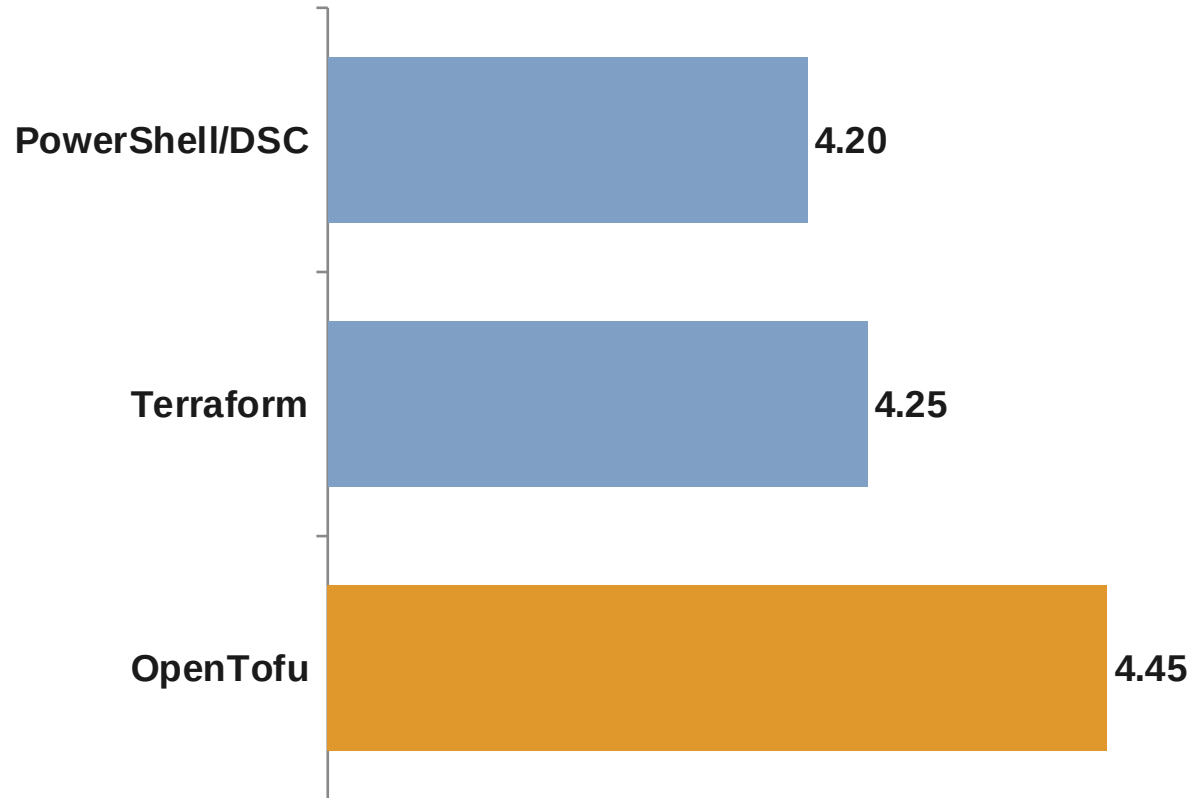
1

gar nicht (S6: Netzwerktool ohne API)

Abbildung 1: Ist-Prozess (Eigene Darstellung, 2026)

Werkzeugauswahl mittels Nutzwertanalyse

7 Kriterien · 5 Werkzeuge · gewichteter Vergleich (Zangemeister, 1970)



Gewinner: OpenTofu

Ehrlich eingeordnet:

- PowerShell/DSC bei Team-Kompetenz und Security am besten
- OpenTofu allgemein jedoch besser

Zielbild: Infrastruktur als Code

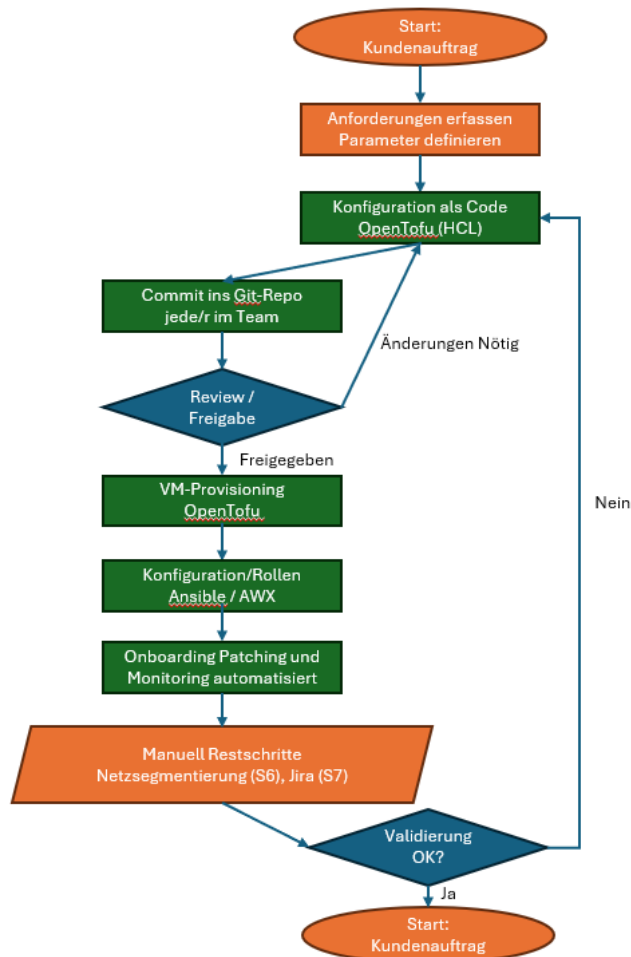


Abbildung 3: Soll-Prozess (Eigene Darstellung, 2026)

● Codestruktur & Repository

- Konfiguration als Code
- Versionierung im Git-Repo

● 4-Augen-Prinzip

- Review und Freigabe vor dem Provisioning

● Zustand & Geheimnisse

- Sorgfältiger Umgang mit State und Zugangsdaten

● Werkzeugkette

- OpenTofu provisioniert
- Ansible/AWX konfiguriert

Pilot: Aufbau des Testfalls

Testumgebung

- Isolierter Test-Host
- Abgegrenzt vom Produktivbetrieb

Werkzeug

- OpenTofu mit Community-Provider taliesins/hyperv

Anbindung

- Hyper-V über WinRM
- Negotiate-Authentifizierung

Anwendungsfall

- Bereitstellung einer ersten VM

Ergebnis: kein Zeitgewinn im Einzelfall – aber echter Mehrwert

~2–3 Min.

- Manueller VM-Bau
- Aktive Handarbeit

3 m 42 s

- Befehl “tofu apply”
- Unbeaufsichtigte Laufzeit

Der Mehrwert liegt nicht in der Geschwindigkeit, sondern in:

Unbeaufsichtigte
Ausführung

Reproduzier-
barkeit

Lineare
Skalierung

Versionierung &
Nachvollziehbarkeit

Erzwungene
Standards

Die grösste Hürde war organisatorisch, nicht technisch

```
Changes to Outputs:
+ vm_id   = (known after apply)
+ vm_name = "iac-pilot-vm01"

Do you want to perform these actions?
OpenTofu will perform the actions described above.
Only 'yes' will be accepted to approve.

Enter a value: yes

hyperv_machine_instance.pilot: Creating...

Error: checking for existing iac-pilot-vm01: error uploading shell script: http response error: 401 - invalid content type

with hyperv_machine_instance.pilot,
on main.tf line 6, in resource "hyperv_machine_instance" "pilot":
  6: resource "hyperv_machine_instance" "pilot" {

PS C:\Source\IaC\iac-pilot> _
```

Abbildung 7: Fehler HTTP 401 bei der Skriptausführung (Eigene Darstellung, 2026)

Fehlerkette bis zum Erfolg

HTTP 401

- Negotiate-Authentifizierung aktiviert
- Task Scheduler für elevated Shell
- Recht «Log on as a batch job»
- **nur per Domänen-GPO anpassbar**

Weitere Erkenntnis:

Der Community-Provider besitzt keine GPG-Signatur,
Reifegrad-Thema für den Produktiveinsatz

Beantwortung der Fragestellungen

● 1 · Adressierbarkeit

- 9 Schwachstellen sind vollständig adressierbar, 4 teilweise, 1 nicht
- Das Potenzial ist erheblich und klar eingegrenzt.

● 2 · Umsetzung & Messbarkeit

- Die Umgebung lässt sich vollständig automatisiert und wiederholbar bereitstellen
- Messbar primär bei Konsistenz und Skalierung, nicht bei Einzel-Geschwindigkeit.

● 3 · Voraussetzungen

- Technisch machbar
- Einsatz hängt an organisatorischen und sicherheitsbezogenen Voraussetzungen

Empfehlung: schrittweise Einführung

01 Voraussetzungen klären

Berechtigungen (batch-logon via GPO) und Secrets-Handling teamübergreifend vorab regeln.

02 Klein starten

Mit klar abgegrenzten Anwendungsfällen beginnen z. B. VM-Provisioning.

03 Bewähren & ausweiten

Erst nach nachgewiesener Stabilität auf weitere Schwachstellen ausdehnen.

04 Kompetenz & Governance aufbauen

HCL-Know-how im Team, Review-Prozess und Sicherheitsvorgaben etablieren.

Grenzen der Arbeit & nächste Schritte

Bewusst offengelassen

- Onboarding in Patching/Monitoring (S8/S9) nicht im Piloten, setzt ein laufendes Betriebssystem voraus.
- Ein Einzelfall im Testumfeld, noch keine breite quantitative Aussage.
- Community-Provider ohne Signatur, Reifegrad im Produktivbetrieb weiter zu prüfen.

Ausblick

- Ausweitung auf weitere adressierbare Schwachstellen.
- Automatisiertes Onboarding in Patching und Monitoring.
- Sauberes Secrets-Management und CI/CD-Anbindung.
- Prüfung offiziell signierter Provider-Alternativen.

KERNBOTSCHAFT

Infrastructure as Code ist bei Netrics technisch machbar und sinnvoll.

Der Nutzen liegt in Wiederholbarkeit und Skalierung und der Erfolg an den organisatorischen Voraussetzungen.

Vielen Dank für Ihre Aufmerksamkeit.