# Æ-DIR

Paranoide Benutzerverwaltung für
Linux-/Unix-Server mit OpenLDAP

GUUG-Frühjahrsfachgespräch 2016

# Wer?

- Michael Ströder <michael@stroeder.com>
- Freelancer
- Schwerpunkte
  - Verzeichnisdienste (LDAP etc.), IAM
  - Angewandte Verschlüsselung, PKI
- Freie Software
  - https://web2ldap.de
  - https://python-ldap.org

# Warum? (1)

- Infrastruktur wird komplexer
  - Viele Systeme
  - Verschiedene Sicherheitsanforderungen
  - Ständige Veränderungen
- Verschiedene administrative Rollen (DevOps)
  - Admins in Produktionsumgebung
  - Entwickler
  - Management / Auditoren

# Warum? (2)

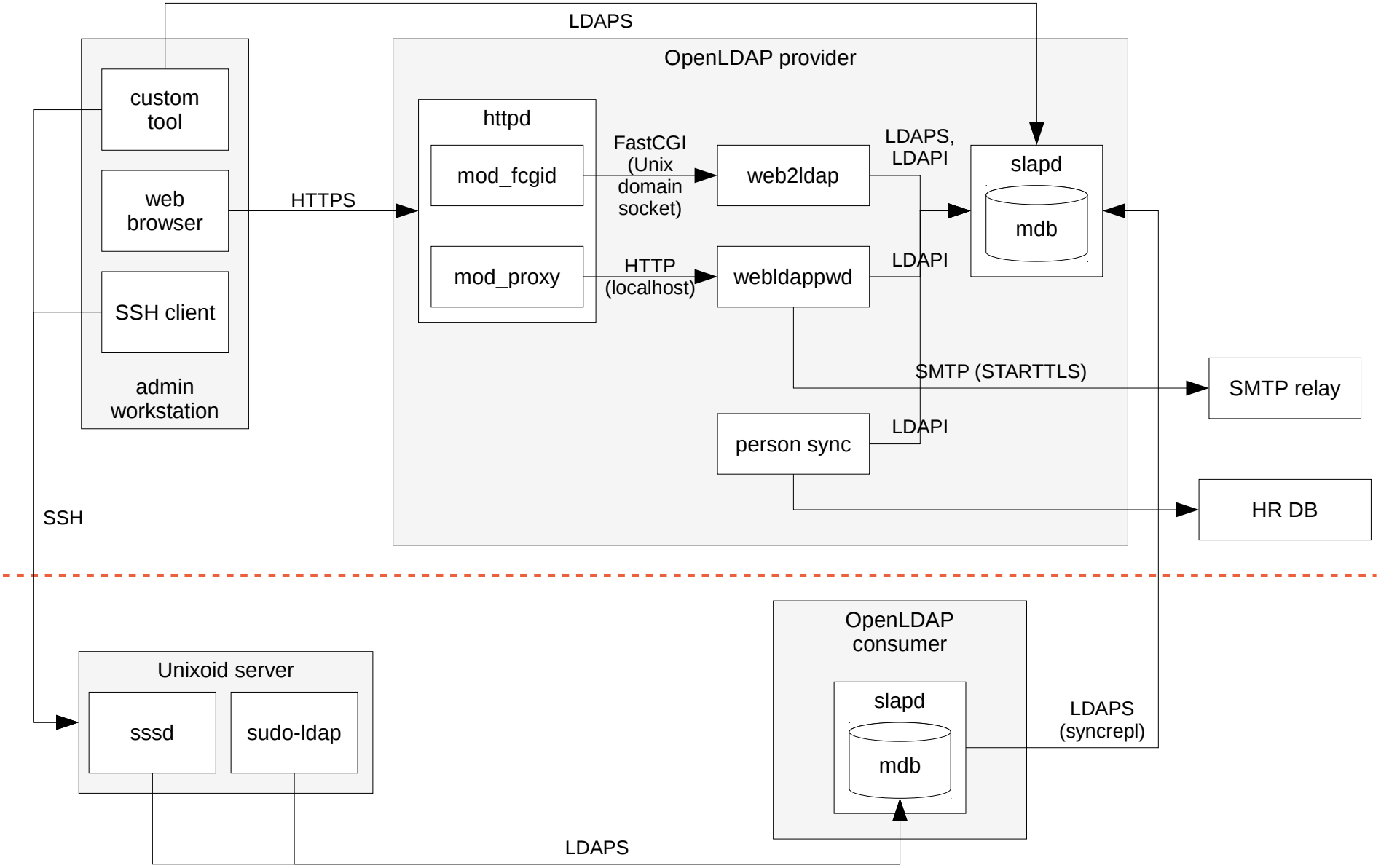
- Striktes Need-to-know-Prinzip!
  - Feingranulierte Autorisierung von Hosts/Diensten auf Benutzer/Gruppen/sudoers etc.
  - Individuelle Authentifizierung der Hosts/Dienste
  - “Views” durch ACLs

- Meines Wissens keine Lösung verfügbar
  => Æ-DIR - Authorized Entities Directory

# Software

- OpenLDAP 2.4.42+
- web2ldap mit HTML/LDIF-Templates & Plugins
- Simple Web-Applikation für Password Self-Service
- Spezielle Admin-Skripte (command-line)
  - Bulk-Initialisierung der Server
  - Reporting
- LDAPS / StartTLS ohne Ausnahme!
- *sssd* und *sudo-ldap* als Client, andere NSS/PAM-Clients möglich

# Architektur



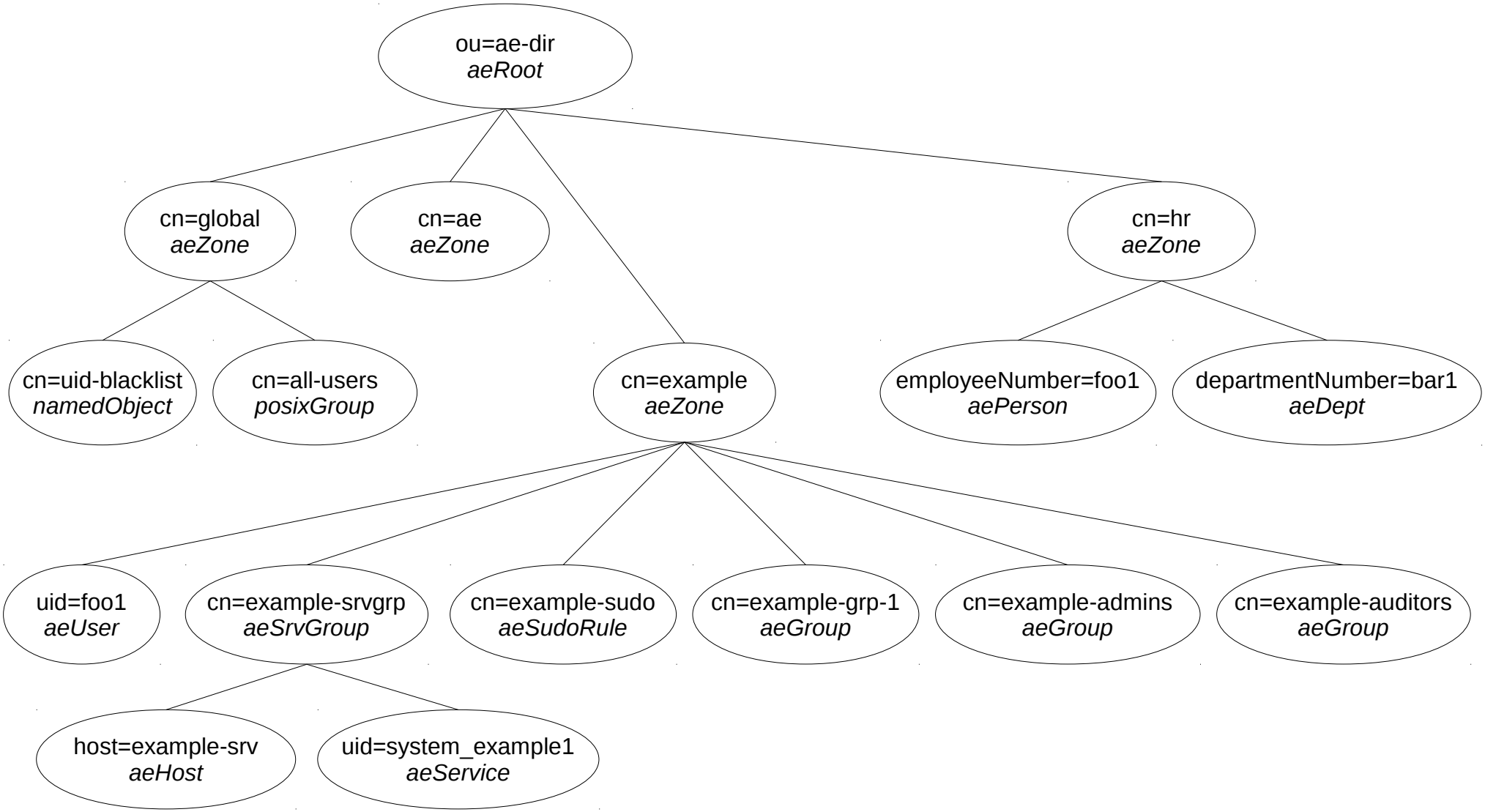
# Rollen

- Kein anonymer Zugriff oder Gastzugang!
- **Æ admins** may *manage* everything within ou=ae-dir and can *read* cn=monitor and cn=config
- **Æ auditors** may *read* everything within ou=ae-dir
- **Zone admins** may *write* anything within a zone
- **Zone auditors** may *read* anything within a zone
- **Setup admins** may *write* aeHost/aeService
- **Users** may *read* own entries, other members of own groups, change own password

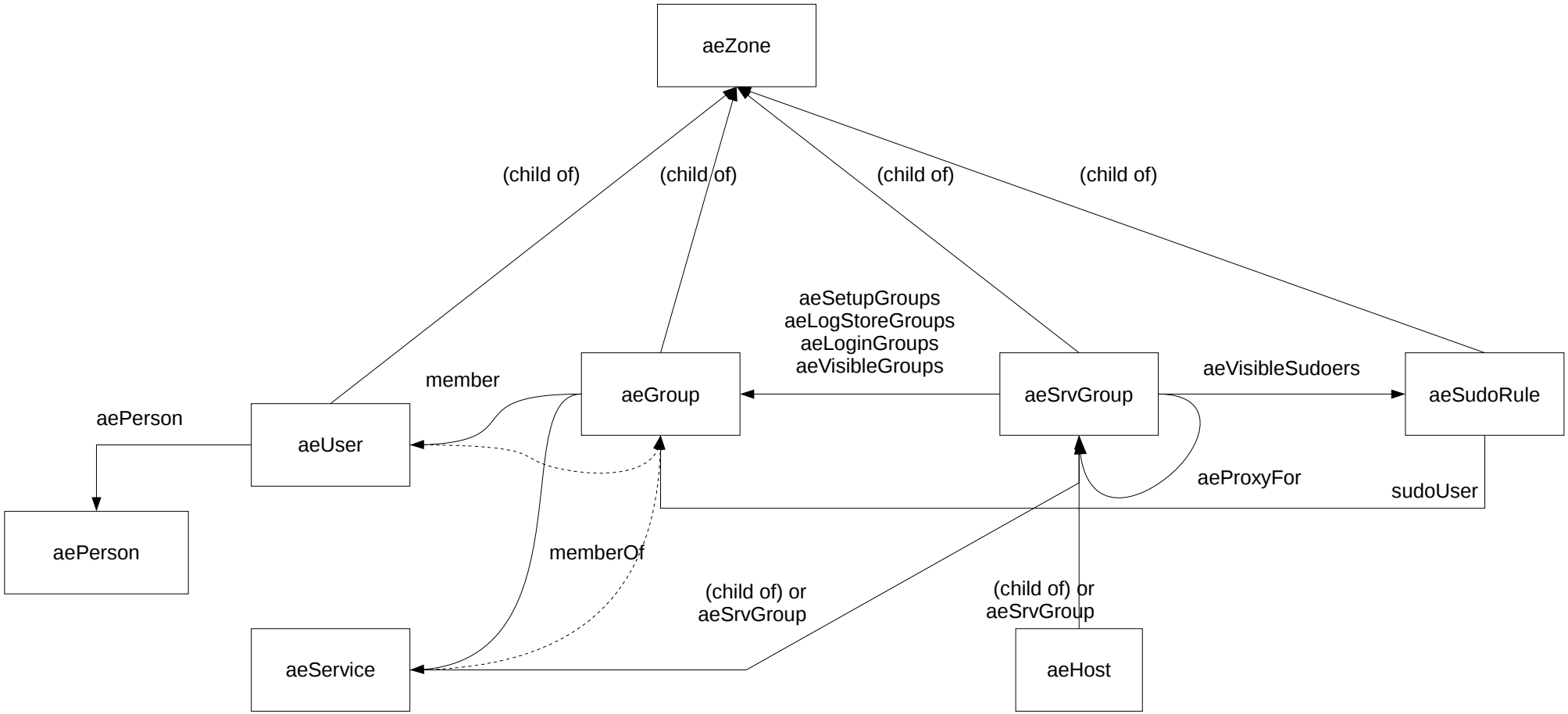
# Anforderungen Schema

- Kompabilität zu NIS-LDAP (RFC 2377 & RFC2377bis)
- Kompabilität zu sudo-ldap Schema
- Unterstützung der gängigen PAM/NSS-Clients
- OpenLDAP-Constraints zur Vermeidung fehlerhafter Dateneingabe
- Meta-Daten (Status, Gültigkeitsdauer, Verwendungszweck, Ticket-Nr.)
- Auditierbarkeit (wer machte was)
  - Eindeutige IDs für alle Entititäten
  - Keine Wiederverwendung von IDs!

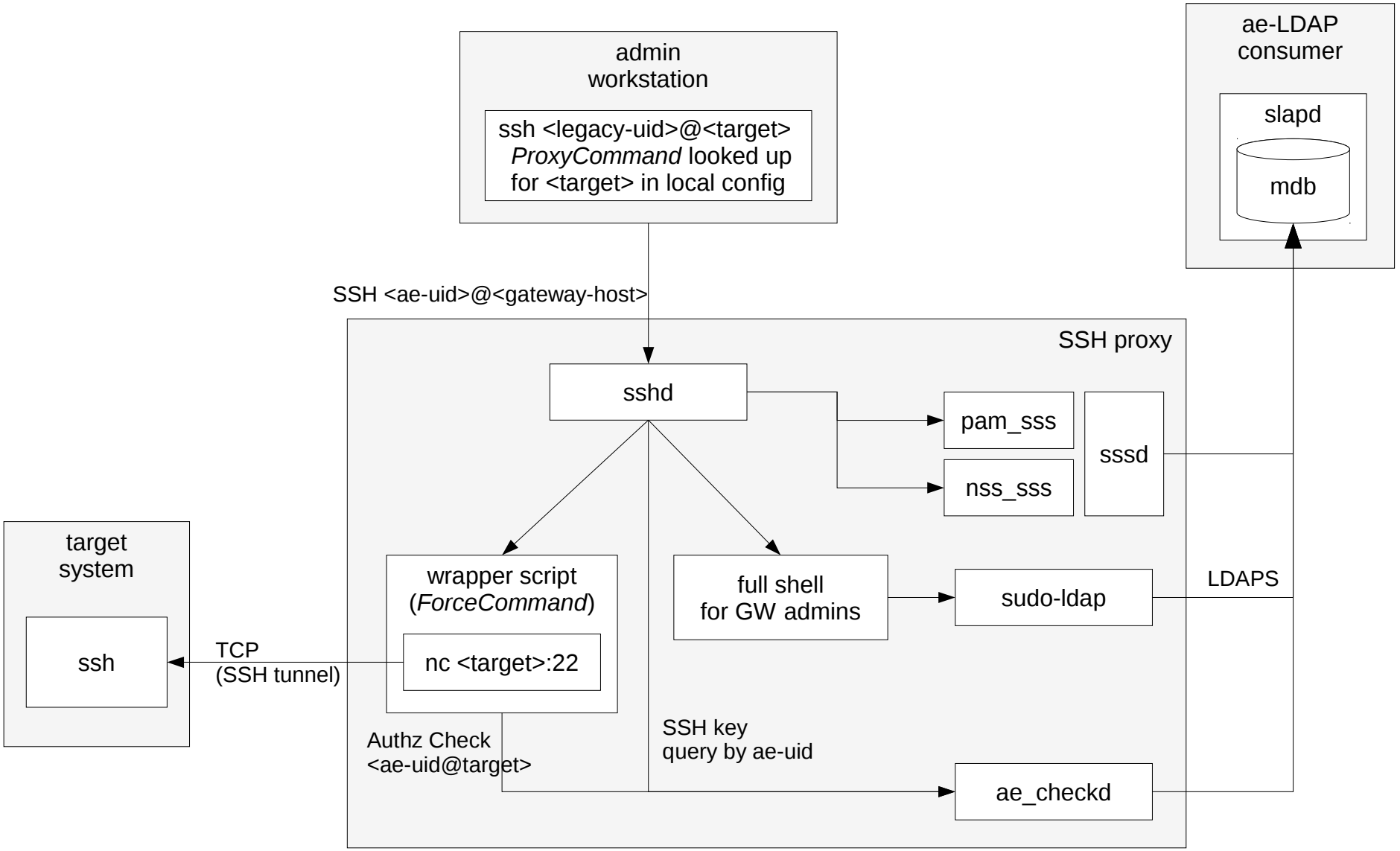
# Directory Information Tree (DIT)



# Entity Relationships



# SSH-Relay mit Autorisierung



# Fazit (1)

- ACLs in OpenLDAP sind zusätzliche Schranke gegen Privilege Escalation in Frontends
- Lokale Komponenten setzen Zugriffskontrolle durch (z.B. Dateizugriffsrechte)
- Separate Passwörter sind im Ernstfall zusätzliche Schranke
- Effektive Sicherheit hängt an Bereitschaft, die Mechanismen wirklich zu nutzen
  => Audits, Schulungen, individuelle Beratung

# Fazit (2)

- Ggf. ist Notfall-Login-Mechanismus notwendig
  => individuelle Prozesse festlegen
- (Set-basierte) ACLs sind
  - sehr komplex
  - langsam (z.Zt. einfach mehr Hardware)
- bei Änderungen
  - Keine Löcher ins System hauen!
  - Neue Ideen sollten immer ins Rollenmodell passen
  - Automatisierte Tests

# Ideen zu mehr Integration

- Machine deployment and network access control: Find out more about existing DHCP/DNS/RADIUS/PXE/TFTP schema mess before
- MIT Kerberos (multiple realms)
- Samba (multiple domains)
- Config management tied to *aeSrgvGroup/aeHost*:
  - *puppet* node declaration
  - *ansible* playbook

# To-Do

- Performance-Verbesserungen!
- Æ-DIR Schema in Internet draft (experimental)
- ae_*demon*
  - Leichtgewichtiger nearly-zero-conf NSS/PAM demon
  - kennt DIT and schema => optimierte Suchen
  - Teilautomatisierte Initialisierung
  - SASL/EXTERNAL mit TLS Client-Zertifikaten (z.B. puppet certs)
- *ae-dir-ui*
- Verwendung *OpenDJ..?*

# Question & Answers

# Zwei-Faktor-Authentifizierung mit OpenLDAP, OATH-HOTP and Yubikey

Axel Hoffmann



# Über mich

- **Axel Hoffmann**, M.Sc.

- Linux System Administrator
- 1&1 Mail & Media Dev. & Tech. GmbH
- axel.hoffmann@1und1.de

# Einführung
## Anforderungen

- Wenig Integrationsaufwand auf den Servern
- Effizienter Umgang mit den Tokens
  - Kein mühseliges Abtippen von Codes
  - Integration als Tastatur
- Bewusstes Auslösen des zweiten Faktors
- Kein Software-Token!
- LDAP-fähige Appliances und Web GUIs einbinden
- Hohe Sicherheitsanforderungen an Token-Rollout
  - Nur der Besitzer sollte den Token nutzen können
  - Keine vorbeschlüsselten Tokens
  - Besitzer&Admins können Schlüssel nicht auslesen

# Einführung

## Yubikey

- Kleines USB-Stick-artiges Gerät
- **Yubikey Standard**: Komponente *YubiKey OTP*
- Zwei Slots für
  - Generieren von OATH HOTPs
  - Generieren von Yubico OTPs
  - Statisches Passwort ausgeben
  - Challenge-Response durchführen
- Arbeitet mit Standard-HID-Treibern/Modulen
- 2 Funktionen können genutzt werden (1 pro Slot)


## Initiative for Open Authentication

- HOTP
  - HMAC-Based One-Time Password[1]
  - $\texttt{TRUNC}(\texttt{SHA1}(\texttt{counter}, \texttt{psk}))$ mod $10^{\texttt{numDigit}}$

[1] numDigit = 6 or 8

# Implementierung

## Überblick

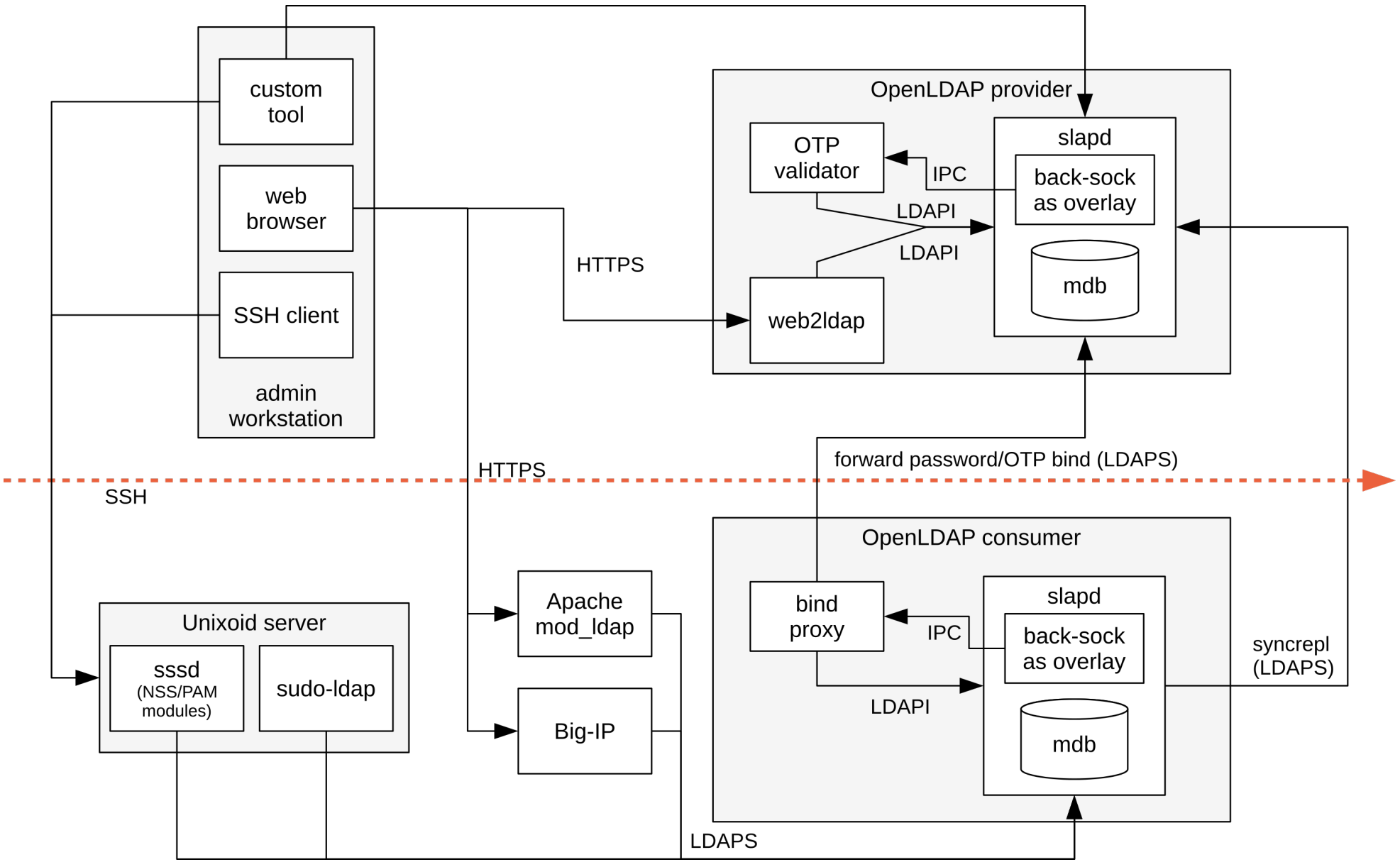



Abbildung : Systemarchitektur mit OTP-Validierung

# Implementierung

## LDAP Structure

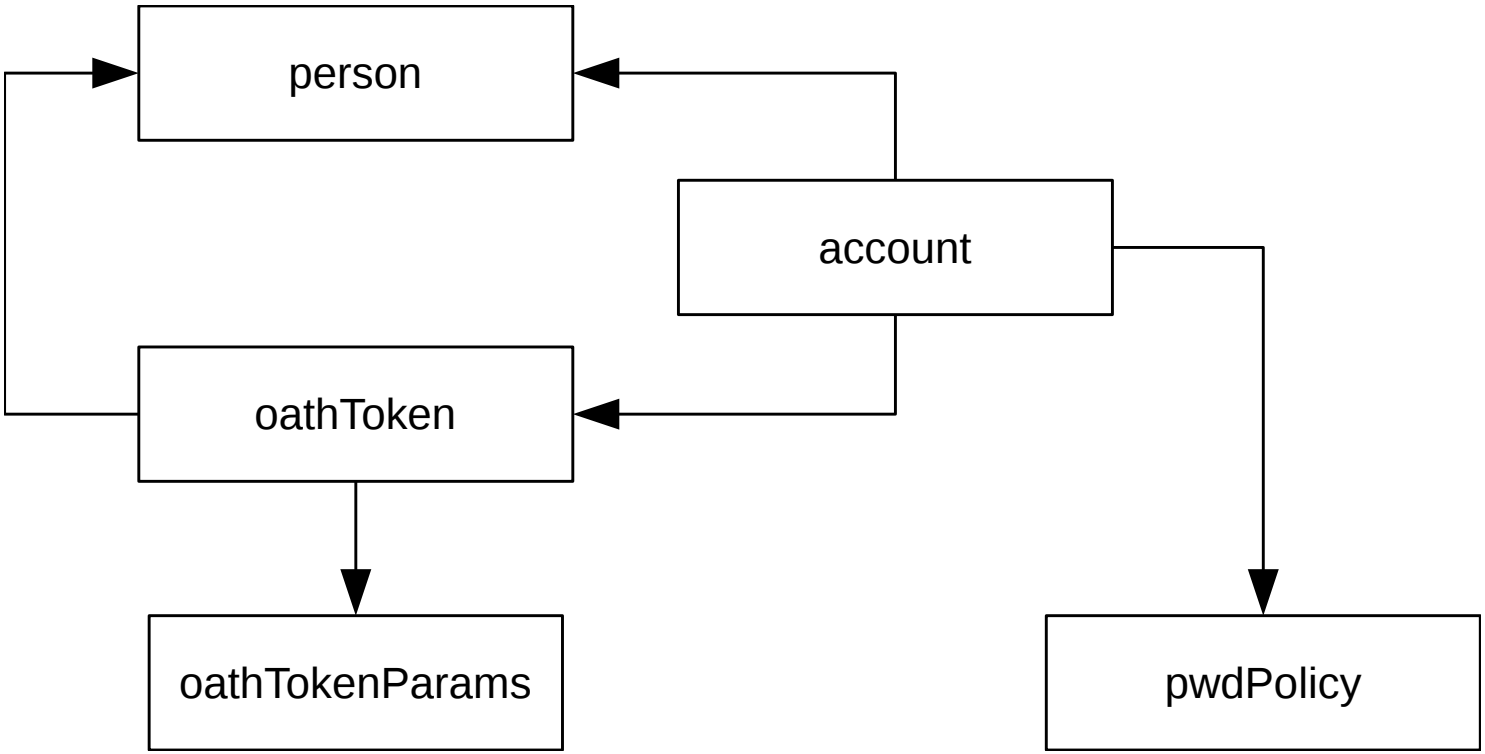



Abbildung : Vereinfachtes LDAP Entity-Relationship-Modell

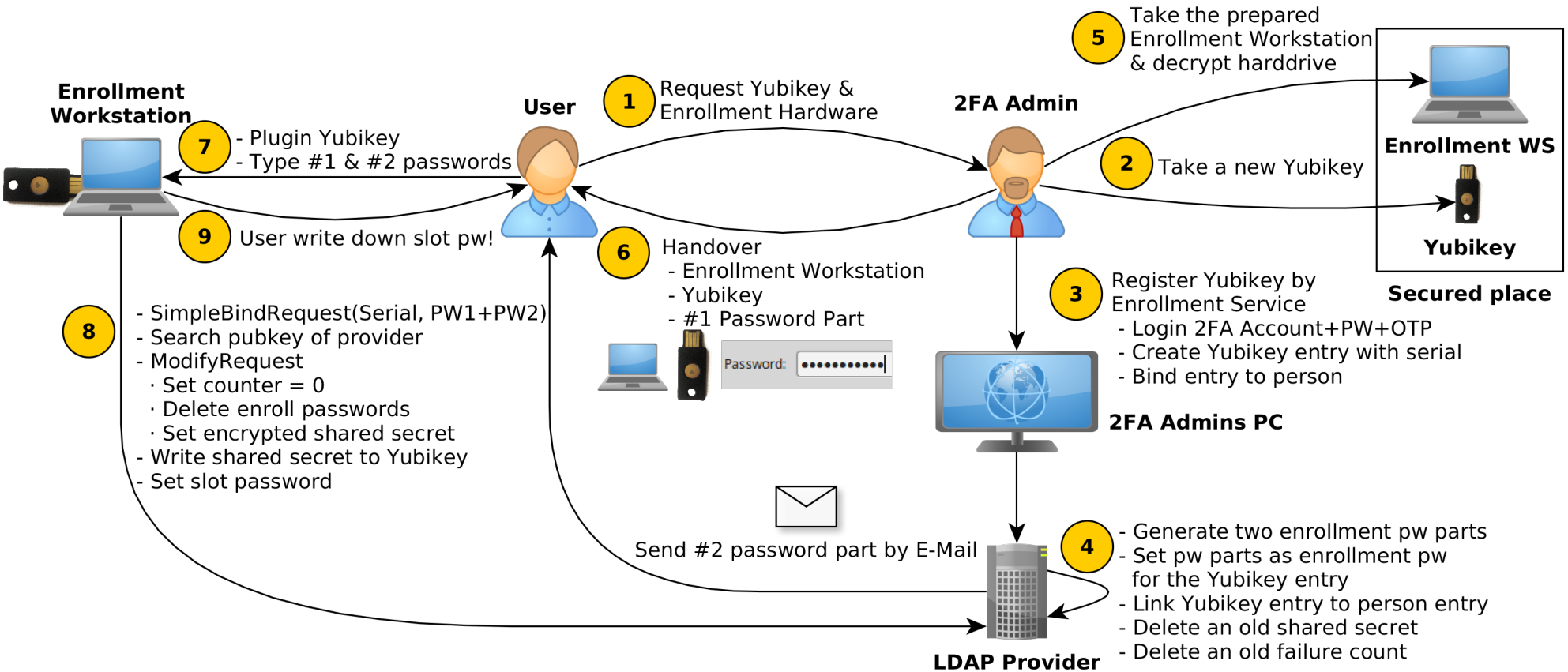
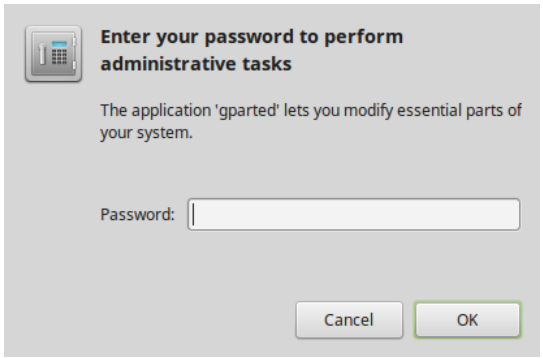



Abbildung : Ablauf des Yubikey-Rollouts

# Gibt es Fragen zum Vortrag?

# Multi-Faktor-Authentifizierung

- Kombination von 2 oder mehr Faktoren:
  - Etwas was du *weißt*:
    - Geheimnis, Passwort, Passphrase, PIN, TAN




  - Etwas was du *besitzt*:
    - Token, SmartCard, Schlüssel

  - Etwas was dich *ausmacht*:
    - Iris, Fingerabdruck, Stimme, Tippgeschwindigkeit/-verhalten

# Rollout Dienst I

- `python-yubico` Bibliothek wurde genutzt
- Kein chaotisches Shell-Script welches CLI Kommandos aufruft
- Läuft auf extra gehärteter Rollout-Hardware
- Bindet Yubikey an HOTP Token Eintrag

# Rollout Dienst II

- Script-Sequenz:
  1. Beide Slots löschen inkl. Passwort
  2. Lese Yubikey Seriennummer
  3. Einloggen in LDAP durch Yubikey Serial und Rollout-Passwort
  4. USB Modus zu nur HID setzen, deaktiviere SmartCard
  5. Setze Modus von Slot 1 zu HOTP und schreibe Schlüssel
  6. Schütze beide Slots mit benutzerdefinierten Passwort
  7. Schreibe Schlüssel ins LDAP und setze Zähler 0
  8. Schalte NFC zum ungenutzten Slot 2

# Zukünftige Erweiterungen

- SmartCard Funktionalität durch `ykneo-openpgp` Applet
  - Erweitere Rollout-Dienst & HW durch PGP-Agent
  - Rollout stößt Generierung eines PGP Keys auf Yubikey an
  - PGP Public Key wird an Token-Eintrag gebunden
  - PGP Key anstelle des SSH Keys auf SSH Client