Aufgeschlossen. Zwei-Faktor-Authentifizierung für Linux und mobile Geräte

Dipl.-Inf. Frank Hofmann

efho.de

28. März 2015

Inhalt

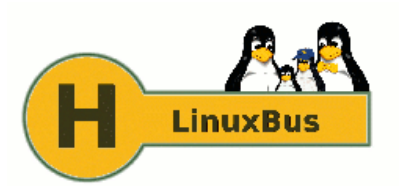
OpenSource-Aktivitäten und Projekte

Chemnitzer Linux-Tage seit 2000

Brandenburger Linux-Info-Tag (BLIT) 2006-2012

seit 2009

Regionales LUG-Treffen Berlin-Brandenburg seit 2008



LinuxBus Berlin-Chemnitz seit 2007

Über Hofmann EDV – Linux, Layout und Satz

Linux, Layout & Satz

WIZARDS OF FOSS
Open Source Schulungen

- Layout und Satz, Druckvorstufe
- Administration und Service
  - Programmierung und Automatisierung auf der Basis von PHP und Python
  - Authentifizierung
- Trainings für IT-Spezialisten

Mitbegründer, Gesellschafter und Trainer

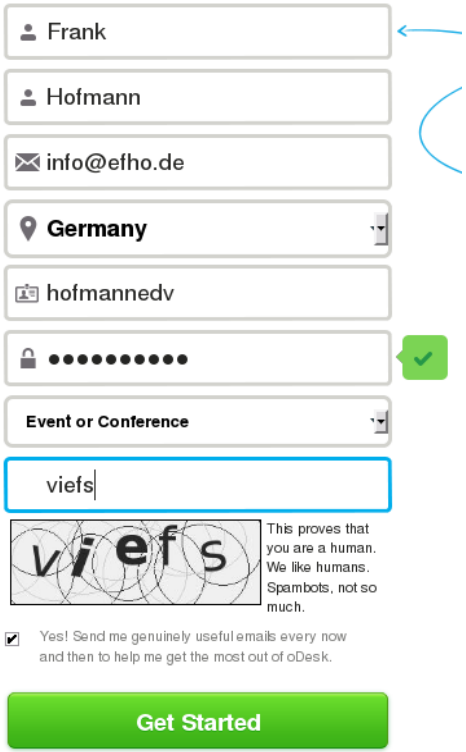
Ziel: Sichere Anmeldung

- falls Zugangsdaten abgefangen werden, soll derjenige möglichst wenig Freude daran haben
- Benutzer soll seinen Account möglichst ohne Einschränkungen weiterbenutzen können
- überschaubarer administrativer Aufwand
- Handhabbarkeit
  - plattformunabhängig
  - übliche Schnittstellen
  - bezahlbar
  - praxistauglich
- arbeiten von wechselnden Standorten aus
  Infrastruktur unbekannt, die bereitsteht
- bestmögliche Absicherung der Zugangswege/Authentifizierung

Klassische Authentifizierung

- Benutzername und statisches Passwort
- Benutzername und dynamisches Passwort (PIN, TAN)
- Zugangsdaten verwaltet jeder Benutzer selbst
- setzt voraus, daß beide Seiten gewissenhaft mit den Zugangsdaten umgehen
  - *Dienstbetreiber*: Verwaltung, regelmäßige Prüfung auf Sicherheit, Zugriffsrechte
  - *Benutzer*: Aufbewahrung über Schlüsselring oder Passwortdienst (LastPass, Keepass)

Begriff: Zwei-Faktor-Authentifizierung



- Ergänzung der Authentifizierung um einen weiteren Faktor
  - Faktor 1: Benutzername und Passwort
  - Faktor 2: weiteres Geheimnis
- Einmalpasswort
  - engl. One Time Password (OTP)
  - Passwort, welches nur ein einziges Mal benutzbar ist
  - Onlinebanking: TAN, mTAN
- Einmalpassworte machen das Abfangen und Wiederverwenden sinnlos (Phishing, Replay-Attacken)

Authentifizierungstoken YubiKey (Teil 1)

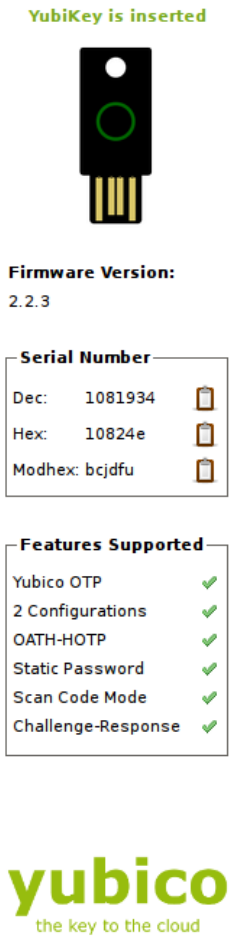
- funktioniert wie eine USB-Tastatur
- YubiKey erzeugt Einmalpassworte

```
ccccccbevgvrcfinkgltgfrnltjevenfhitcfdjevekv

ccccccbevgvrdelcvfgtkifigjidgnliubftchvvvgvj
```

- im Auslieferungszustand benutzt es die YubiCloud – den Authentifizierungsdienst des Herstellers Yubico
- alle benötigten Software-Komponenten stehen unter einer freien Lizenz
- Hersteller Yubico pflegt passende Debian-Pakete dazu

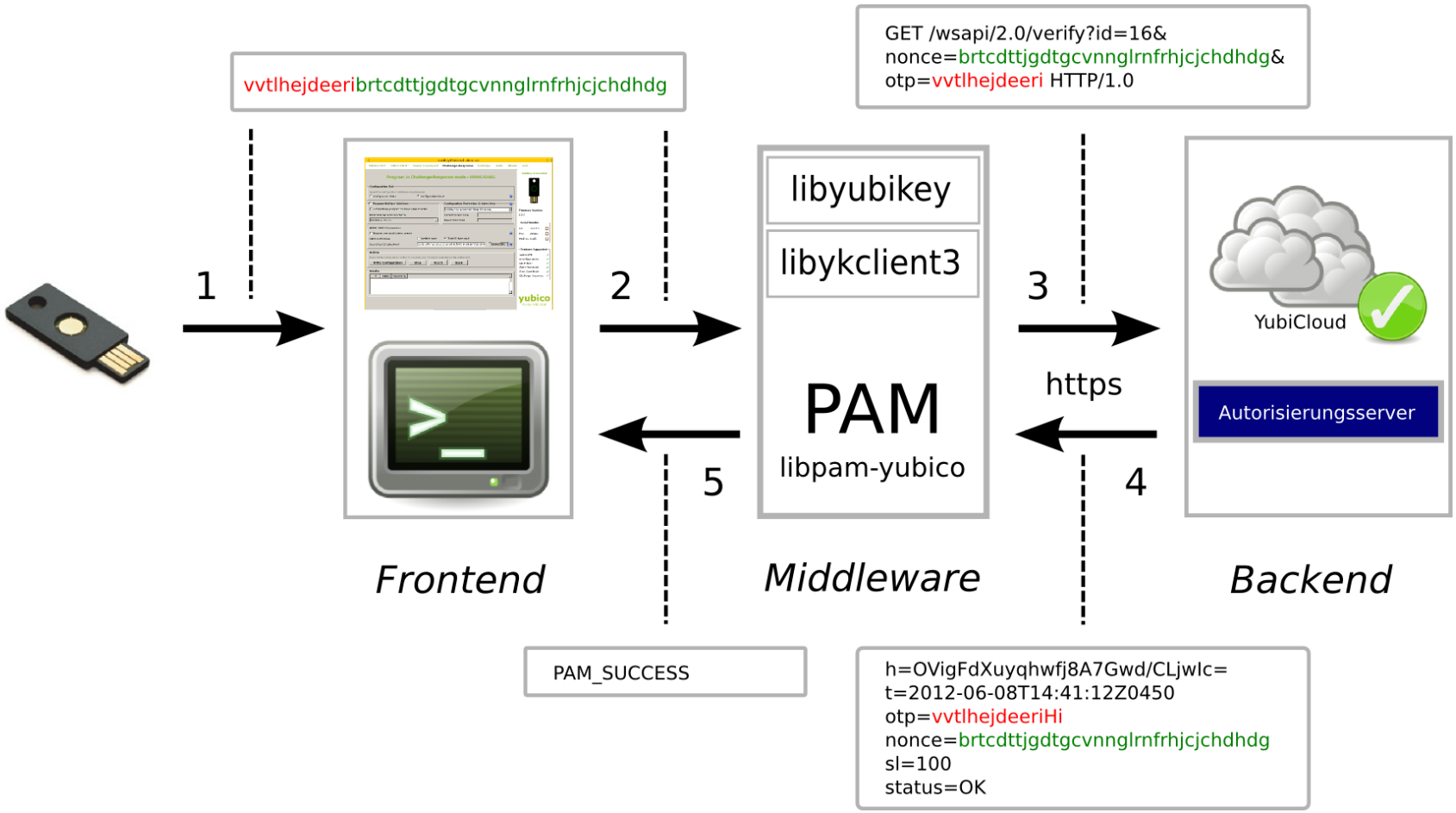
Authentifizierungstoken YubiKey (Teil 2)



YubiKey-Varianten

- YubiKey 2, Nano, Fido U2F
  - 2 Slots für dynamische und statische Passworte
  - OATH, OTP, TOTP
- YubiKey Neo, Neo-N
  - Neo: mit Near Field Communication (NFC) und SmartCard
  - Neo-N: mit U2F Security Key und SmartCard
- Verifizierung durch YubiCloud, YubiHSM (USB), PAM-Module, YubiX (virtuelle Instanz)
- Token mittlerweile akzeptiert von Google, Dropbox und LastPass etc.

Zusammenspiel von YubiKey und dem Authentifizierungsdienst

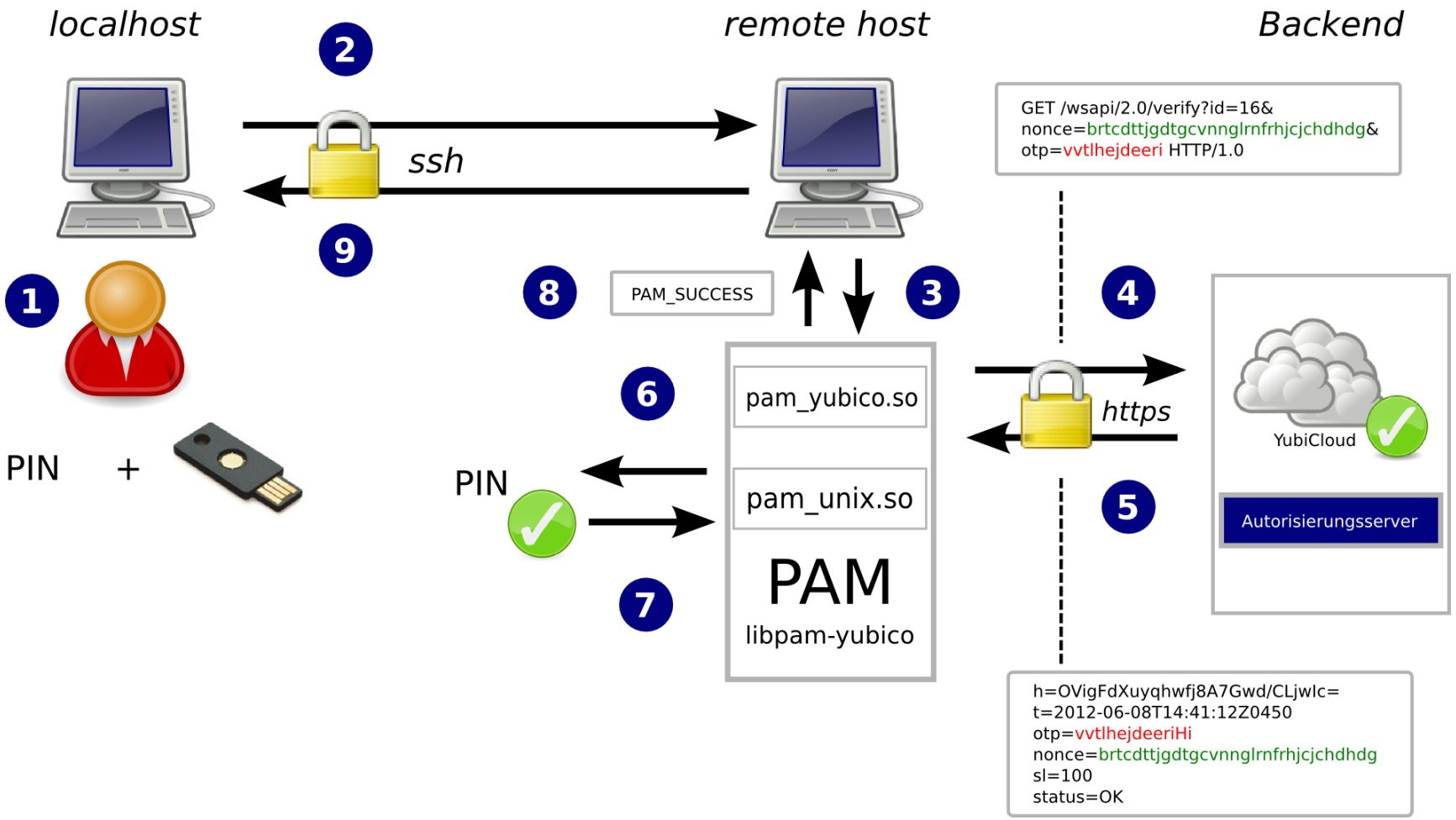


Rechnerzugang mit statischen Passworten

- YubiKey ist mit statischem Passwort konfiguriert (Slot 1 oder 2)
  YubiKey fungiert nur als (verdeckter) Passwortspeicher
- Benutzer bekommt YubiKey und Passwort zur Authentifizierung
- Benutzer kombiniert beide Bestandteile
  er muß wissen, in welcher Reihenfolge er beides eingibt

- Einsatzbereich in der Praxis:
  - Außendienstmitarbeiter mit Laptop
  - Vorteil 1: einfacher Umgang mit Absicherung
  - Vorteil 2: keine Notwendigkeit für eine bestehende Internetverbindung

SSH und dynamische Passworte via PAM (Teil 1)



SSH und dynamische Passworte via PAM (Teil 2)

- Installation der Debian-Pakete `libpam-yubico`, `libpam-runtime` und `libyubikey0`
- SSH-Konfiguration in `/etc/ssh/sshd_config`:
  `UsePAM yes`
  `ChallengeResponseAuthentication no`
- PAM-Konfiguration in `/etc/pam.d/sshd`:

```
10 # Standard Un*x authentication.
11 # @include common-auth
12
13 auth required pam_yubico.so id=16
14 auth required pam_unix.so nullok_secure try_first_pass
15
16 # Disallow non-root logins when /etc/nologin exists.
17 account    required     pam_nologin.so
18
```

Zerlegung in PIN (Passwort) und Yubico OTP und Weiterleitung an PAM-Modul `pam_unix.so` und die YubiCloud zur Authentifizierung

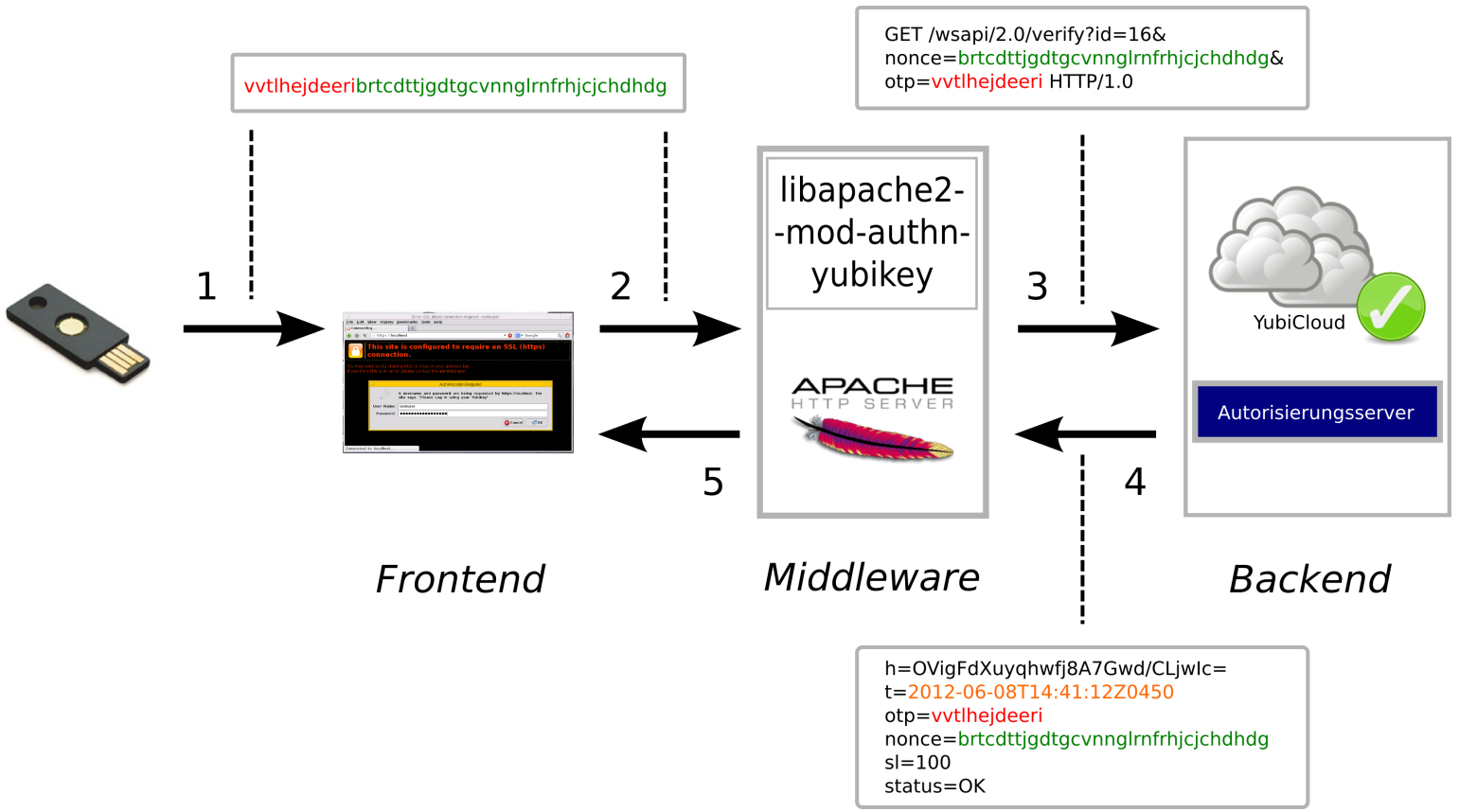
SSH und dynamische Passworte via PAM (Teil 3)

- Hinterlegung der Private-ID ihres YubiKey im Home-Verzeichnis auf dem Zielrechner unter `.yubico`
- Textdatei `authorized_yubikeys`:

```
$ cat /home/fho/.yubico/authorized_yubikeys
fho:ccccccbavaev:vvkiknackeil
```

- Hinweise:
  - Trennung der YubiKey-Private-IDs mittels Doppelpunkt
  - alle Benutzer müssen sich ab jetzt mit YubiKey authentifizieren
  - Authentifizierung über YubiCloud, d.h. Internetzugang erforderlich

Webserver mit Apache-Modul



ausführliche Beschreibung in Uptimes 1/2014

UNIX-Login via PAM und `libpam-yubico`

- pflegt der Hersteller Yubico als Debian-Paket
- Validierung über die YubiCloud
- Zuordnung der Tokens über `/etc/yubikey_mappings` mittels Login und PIN

  `frank:ccccccbevgvr`
- Registrierung des YubiKey und Bezug eines API Key
- PAM-Modul rekonfigurieren

```
# As of pam 1.0.1-6, this file is managed by pam-auth-update by default.
# To take advantage of this, it is recommended that you configure any
# local modules either before or after the default block, and use
# pam-auth-update to manage selection of other modules.  See
# pam-auth-update(8) for details.

# here are the per-package modules (the "Primary" block)
auth    required        pam_yubico.so mode=client try_first_pass id=16997 key=K+dD4VwKyeZR2y0ZI9wLdEdUjsE=
auth    [success=1 default=ignore]      pam_unix.so nullok_secure try_first_pass
# here's the fallback if no module succeeds
auth    requisite                       pam_deny.so
```

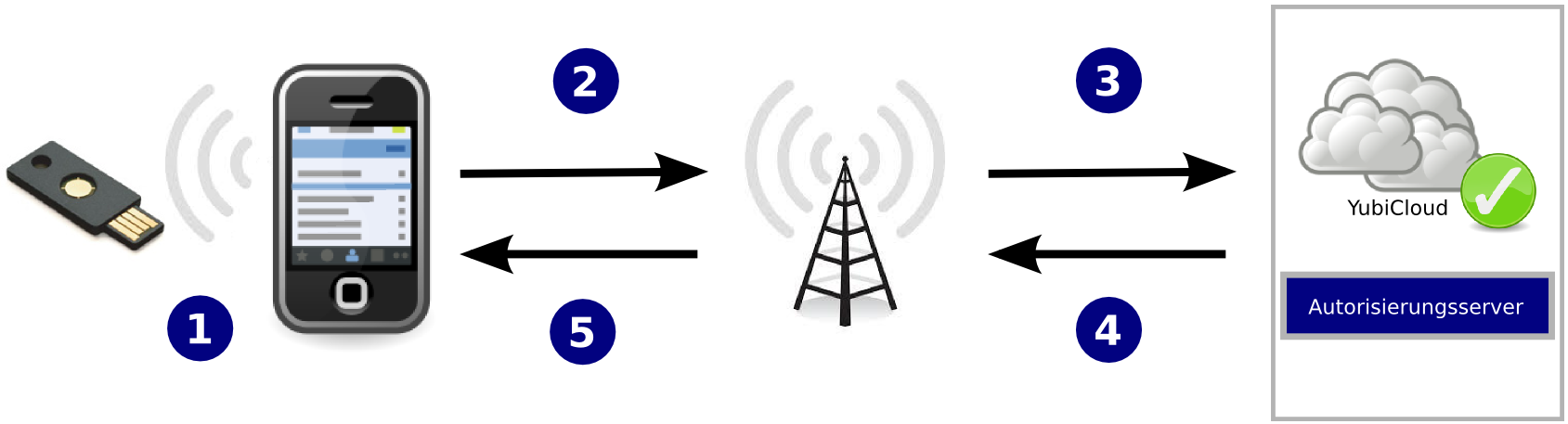
UNIX-Login via PAM und `libpam-yubico`

Einrichtung eines Fallback – Möglichkeiten im PAM

- `requisite,required`
  die Authentifizierung über das Modul muß erfolgreich sein. Im Fehlerfall werden keine weiteren Module abgearbeitet (notwendige Vorbedingung)
- `sufficient`
  wenn die Authentifizierung über das Modul erfolgreich war, reicht das zur Authentifizierung aus und es werden keine weiteren Module abgearbeitet (hinreichende Bedingung).
- `optional`
  das Ergebnis der Authentifizierung über das Modul findet keine Beachtung, es sei denn, es ist das einzige für einen Typ.

`sufficient`: Fallback auf `pam_unix`, falls kein Netz, aber auch genügsam, wenn der YubiKey passt

Authentifizierung mit dem Smartphone

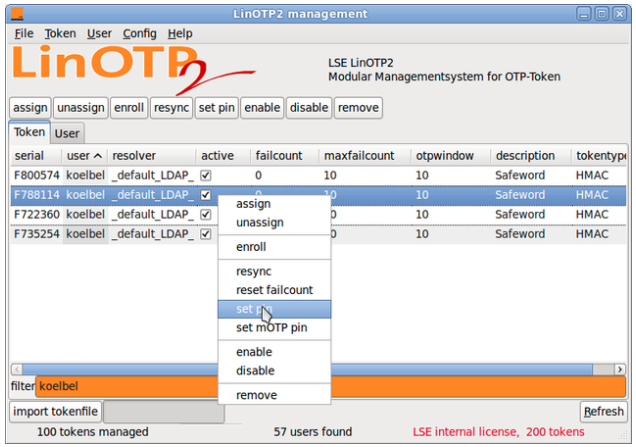


- NFC-fähiges Smartphone in Kombination mit einem YubiKey Neo
- Verifikation mittels App und YubiKey
- Nutzung der YubiCloud zur Überprüfung

Beobachtungen im Alltag

- kompetenter Support des Herstellers Yubico
- Linux-Treiber als Debian-Pakete
- vollständige, lesbare Dokumentation
- Hardware ist ...
  - bruchsicher
  - waschmaschinenfest
  - witterungsresistent
  - transportabel
- Handhabung
  - wie ein zweiter Schlüssel
  - praktikabel

Abhängigkeit vom Authentifizierungsdienst



- Yubico-Authentifizierungsdienst: Internet bzw. Netzzugang
- lokal
  - USB mit YubiHSM
  - eigener Dienst, bspw. auf der Basis von LinOTP
- Fallback
  - Fallback auf UNIX-Login via PAM konfigurieren
  - Rückstufung auf HMAC-SHA-1
  - Einrichtung eines Zweitschlüssels (Admin-YubiKey)

Links

1. Thomas Osterried und Frank Hofmann: YubiKey Basiswissen
   LinuxUser 09/2012
2. Thomas Osterried und Frank Hofmann: YubiKey und SSH
   LinuxUser 10/2012
3. Werner Heuser und Frank Hofmann: YubiKey und Apache-Webserver
   Uptimes 1/2014
4. Werner Heuser und Frank Hofmann: NFC-Authentifizierung mit Smartphones
   LinuxUser 07/2014
5. Frank Hofmann: YubiKey und lokales Login
   LinuxUser 08/2014

Vielen Dank!

Lassen Sie es setzen.

LLS
Linux, Layout & Satz

Kontakt:

Dipl.-Inf. Frank Hofmann
Hofmann EDV – Linux, Layout und Satz
c/o büro 2.0
Weigandufer 45 – 12059 Berlin

Tel. 030 2000 586 80
Email `frank.hofmann@efho.de`
web `http://www.efho.de`
twitter
`http://twitter.com/hofmannedv`