



# **FFG 2011**

## **TokenTube** **Benutzerverwaltung für LUKS/dm-crypt**

Jürgen Pabel, CISSP  
Akkaya Consulting GmbH

Creative Commons 3.0  
Attribution, Noncommercial, No Derivative Works (GERMANY)



# Über mich

- Jürgen Pabel
  - Berater für IT-Sicherheit (CISSP)
  - Diverse Open-Source Projekte
  - Rugby
  
- Akkaya Consulting GmbH
  - IT-Beratung <http://www.akkaya.de/>
  - Medizinische Software <http://www.ac-stb.de/>



# Agenda

- Einführung
  - Kontext und Motivation
  - LUKS & dm-crypt
  - Praktische Auswirkungen des LUKS Designs
  
- TokenTube
  - Konzepte, Komponenten & Features
  - Debian/Ubuntu Integration
  - Live Demo
  - TODOs



# Kontext und Motivation

- Festplattenverschlüsselung mit Linux
  - Lösungen für Windows sind oftmals ausgereifter
- Handhabbarkeit für einzelne Systeme akzeptabel
  - Benutzer verwendet ein separat verwaltetes Passwort
  - Es wird keine Recovery-Funktionalität bereitgestellt
- Administrierbarkeit in Firmeneinsatz eher schlecht
  - Keine zentrale Konfiguration
  - Unterstützung für Supportprozesse schwer umsetzbar



# Linux Unified Key Setup (LUKS)

- Platform independent on-disk layout specification
  - Verschlüsselungsalgorithmus und Cipher-Modus
  - Validierungsinformationen für Verschlüsselungsschlüssel
  - Benutzerschlüssel
    - 8 Slots stehen zur Verfügung
- Schutz gegen krypto-analytische & forensische Angriffe
  - Zufällige Anzahl von Hash-Iterationen
  - Anti-forensisches Informations-Splitting



# LUKS/dm-crypt: Status Quo

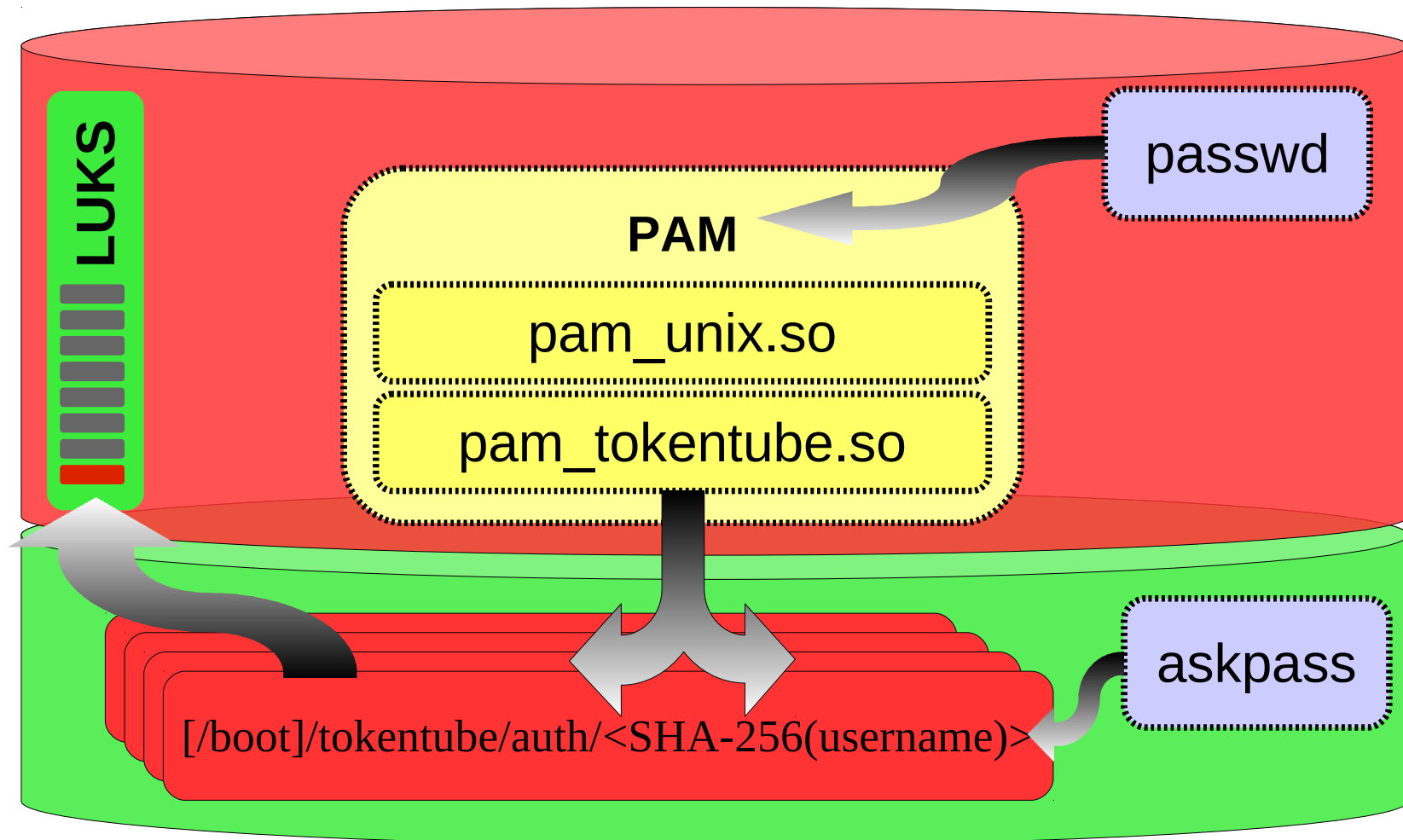
- The good
  - Einsatz dedizierter Verschlüsselungspasswörter
- The bad
  - Wer verwendet wirklich (kryptographisch) starke Passwörter?
    - >100 Bit Entropie “erforderlich” (entspricht >15 stelligem Passwort)
- The ugly
  - Behindert eine breitere Adaptierung von LUKS/dm-crypt
    - Kryptographisch starke Passwörter haben kaum Benutzerakzeptanz
    - Zentrale Administrations-/Supportprozesse sind schwer umsetzbar



# TokenTube

- Benutzerbezogene Pre-Boot-Authentisierung
  - Ablage der Authentisierungsdaten auf der Boot-Partition
  - Entschlüsselung auf Basis des Benutzerpasswortes
    - LUKS Benutzerschlüssel
  
- PAM Modul [sic]
  - Verschlüsselt LUKS Benutzerschlüssel mit Benutzerpasswort
  - Passwortvalidierung erfolgt per Validierungstoken
  
- PBA Benutzerauthentisierung
  - Laden der benutzerspezifischen Authentisierungsdaten
  - Entschlüsselung mittels Benutzerpasswort
  - Ausgabe des LUKS Benutzerschlüssels (pipe'd an cryptsetup)

# TokenTube visualisiert





# Authentisierungsdateien (1/2)

- Ablage im Dateisystem
  - `/boot/tokentube/auth/`
    - Dateiname → `HASH(Benutzername)`
- Datenstruktur
  - 32 Bytes → LUKS Benutzerschlüssel
  - 32 Bytes → Validierungstoken
    - Standard: `HASH(UUID der verschlüsselten Partition)`
- Kryptographische Algorithmen sind konfigurierbar
  - Entsprechend der Unterstützung von *libgcrypt*



# Authentisierungsdateien (2/2)

- Verschlüsselungsalgorithmus
  - Standard Cipher: *AES-256 CFB*
- Schlüsselgenerierung: PBKDF2
  - Standard Hash: *SHA-256*
  - Salz: *Benutzername*
  - Iterationen: *1024 – strlen(Benutzerpasswort)*
- Initialisierungsvektor: PBKDF2
  - Standard Hash: *MD5*
  - Salz: *Benutzerpasswort*
  - Iterationen: *1024 – strlen(Benutzername)*



# Konfiguration

- **Root-Dateisystem**
  - `/etc/tokentube/tokentube.conf`
    - PAM: Benutzer Enrollment
    - Plugins
  
- **Boot-Partition**
  - `[/boot]/tokentube/auth.conf`
  - `[/boot]/tokentube/askpass.conf`
    - Sprachressourcen für PBA
    - Optional: Standard-Benutzername
    - Optional: Aktivierung des Credential-Caching Daemons
  
- **Initramfs**
  - `/etc/tokentube/boot.conf`
    - UUID der Boot-Partition (`/dev/disks/by-uuid/...`)



# Benutzer Enrollment

- **Komfort-Feature: Automatische PBA Freischaltung**
  - Authentisierungsdaten der Benutzer werden bei interaktiver Anmeldung (mittels PAM) übernommen
  
- **Konfiguration**
  - Feature (de)aktivieren
  - Filterkriterien
    - UID Filter (Minimum- & Maximum-Werte)
    - TODO: Gruppenfilter
    - ...?



# Credential-Caching Daemon

- Komfort-Feature: Automatischer Login am Desktop
- Kurzfassung
  - Optional zu startender Daemon
  - Kommuniziert per UNIX Socket
    - Authentisierungsdaten werden von *askpass* bereit gestellt
    - GDM/KDM Plugin liest Authentisierungsdaten aus und führt automatische Anmeldung durch
- Sicherheit
  - Prüfung des UNIX Socket "Clients" mittels `SO_PEERCREC`
  - Unterbindet Debug-Analysen per `PTRACE_TRACEME`
  - Obfuskiert die Authentisierungsdaten in RAM
  - ...



# Plugins

- **Konfiguration**
  - `/etc/tokenube/tokenube.conf`
    - Absoluter Pfad der Plugin-Bibliothek (*dlopen*)
    - Absoluter Pfad der Plugin-Konfigurationsdatei
  
- **Plugin Hooks**
  - LUKS Benutzerschlüssel laden
  - Validierungstoken generieren
  - ...?
  
- **Implementierungen**
  - Default Plugin
  - LUA Skript Plugin



# Challenge-Response Verfahren

## ➤ Ablauf

- Benutzer gibt C/R-String als Benutzername ein (z.B. "SOS")
- Zufällig generierter Challenge-Code wird angezeigt
  - Benutzer teilt den Challenge-Code dem Helpdesk mit
  - Helpdesk generiert Response-Code
  - Helpdesk teilt den Response-Code dem Benutzer mit
- Benutzer gibt Response-Code ein
- Helpdesk setzt neues Login-Passwort (z.B. via LDAP)

## ➤ Perfect Forward Secrecy

- $Key_{file} = Challenge \oplus Response \oplus Secret$
- $Secret = MD5(Key_{luks})^n \rightarrow n \text{ decrements per C/R}$
- $Key_{luks} = AES_{decrypt} („helpdesk.key“, Key_{file})$



# Debian & Ubuntu Paket

- Pre-Boot-Authentication
  - Ersetzen des `askpass` Tools (über *alternatives*)
- System
  - Initramfs neu generieren
    - TokenTube Programme
    - Konfigurationsdatei `/etc/tokentube/boot.conf`
  - PAM Konfiguration (`pam-auth-update`)
- TODO: Debian-Installer
  - `partman-crypto`
  - `user-setup`



# Debian & Ubuntu: askpass

- Konfiguration laden
  - Boot-Partition ermitteln (`/etc/tokenube/boot.conf`)
  - Konfigurationsdatei per `e2fslibs` einlesen
- Eingabe Benutzername und -passwort
  - Leerer Benutzername umgeht TokenTube Authentisierung
- LUKS Benutzerschlüssel ermitteln
  - Benutzerspezifische Authentisierungsdatei einlesen (`e2fslibs`)
  - LUKS Benutzerschlüssel entschlüsseln
  - Ausgabe auf `stdout` (`|cryptsetup`)



# Live Demo

- **Ubuntu**
  - /dev/sda1      Boot-Partition
  - /dev/sda5      Verschlüsseltes LVM
  
- **Funktionen**
  - Installation und Konfiguration (falls zeitlich möglich)
  - Pre-Boot-Authentication
  - Automatischer GNOME Login
  - Wechsel des Benutzerpasswortes



## To-Do Liste (1/2)

- TokenTube binary & library
  - Benutzer Enrollment per Gruppenzugehörigkeit
  - LUKS Bibliothek zur Schlüsselmanagement einbinden
  - Authentisierungsdateien mit Metadaten versehen
- Credential-Caching Daemon
  - Verschlüsselung der Authentisierungsdaten im RAM
  - Validierung der GDM/KDM Greeter-Plugins
- GDM / KDM Greeter Plugins
  - Anpassungen für GNOME  $\geq$  2.21
  - KDE Plugin implementieren

*Anyone?*



## To-Do Liste (2/2)

- Pre-Boot-Authentication
  - Challenge-Response Verfahren
  - Integration für nicht-Debian basierte Distributionen
- Helpdesk
  - Web-basierte Challenge-Response Applikation
- Integration in Debian & Ubuntu (und andere)

*Anyone?*



# URLs

- SourceForge (mailing list, issue tracker, ...)
  - <http://sf.net/projects/tokentube/>
- Ubuntu PPA
  - <https://launchpad.net/~jpabel/+archive/ppa>
- Infos
  - <http://blog.akkaya.de/jpabel/>
  - <http://twitter.com/juergenpabel/>



# TokenTube

Vielen Dank für euer Interesse.

Bitte stellt Fragen!

This presentation is published under the terms of the Creative Commons „Attribution-Noncommercial-NoDerivs 3.0 Germany“ (BY-NC-ND) license.

Any trademarks, registered trademarks and brands mentioned in this document are property of their respective owners.