AGETO

PersoApp

Workshop

PersoApp

Transparenz. Anpassbarkeit. Mobilität.

Vorstellung

Victoria Kühn
Marketing Manager

André Gutwirth
Projektleiter

Christian Kahlo
Research Manager

Agenda

- Einführung in die eID-Infrastruktur

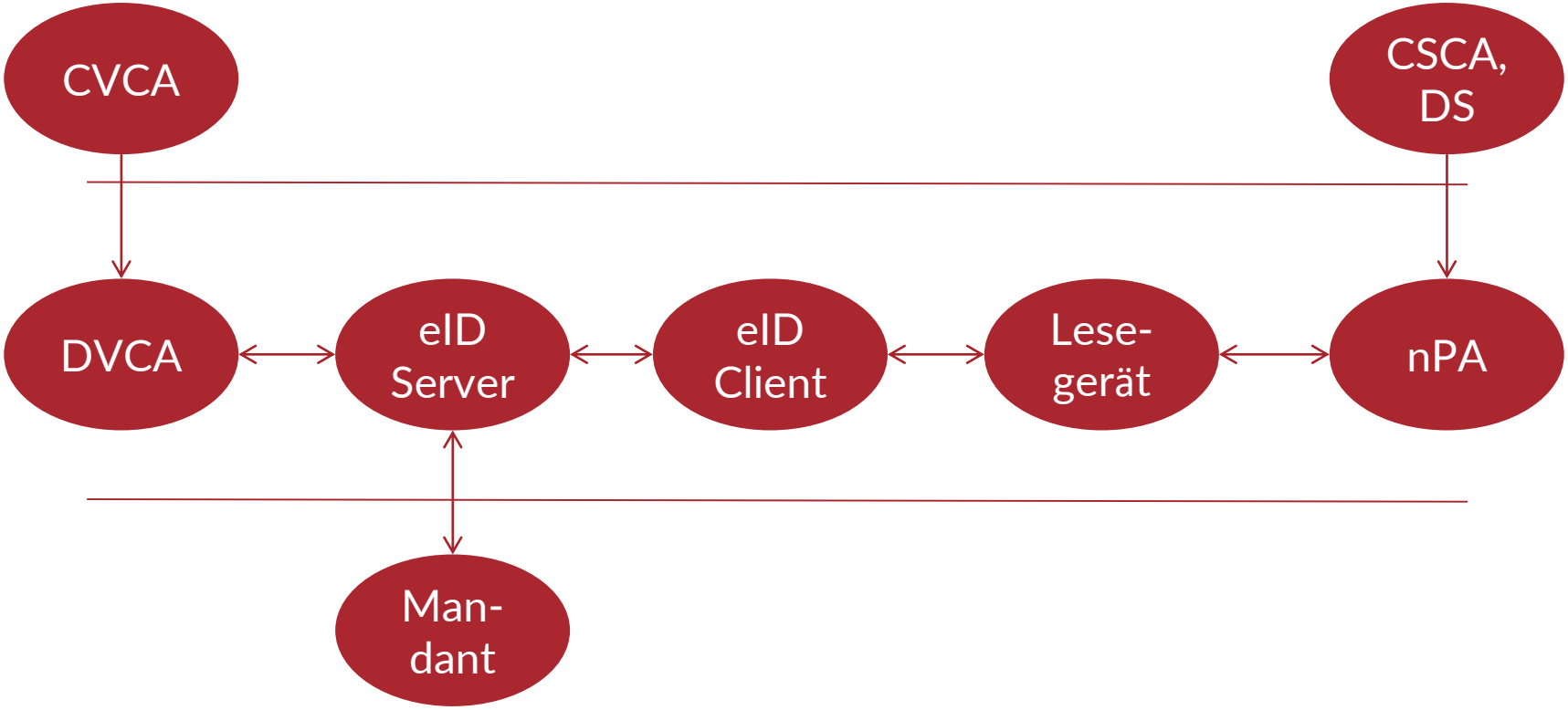
- PersoApp
- PersoApp Struktur und Entwicklungsumgebung

- Übersicht über entscheidende Komponenten

- Kurze Pause

- Erweiterbarkeit und Wiederverwendbarkeit
- Fragen und Antworten, freie Themen

Einführung in die eID Infrastruktur



CVCA, CSCA, DS, DVCA

- CVCA Country Verifier Certification Authority
- CSCA Country Signer Certification Authority
- DS Document Signer
- DVCA Document Verifier Certification Authority

- CVCA und CSCA sind in staatlicher Hand, für DE -> BSI
- CVCA stellt DVCAs (Berechtigungs-CA) Zertifikate aus
- CSCA stellt DS (Ausweishersteller) Zertifikate aus
- CVCA und CSCA legen Schlüsselparameter fest, z.B. brainpool256r1 oder brainpool384r1

eID-Server und Mandant

- eID-Server (oder –Service) hält das Berechtigungs-zertifikat (Terminalzertifikat) und Schlüssel vor
- eID-Server erhält Zertifikate und Sperrlisten von DVCA
  - BSI TR-03129
- Mandant (Diensteanbieter) kommuniziert mit eID-Server
  - BSI TR-03130
- eID-Server kommuniziert mit eID-Client
  - BSI TR-03112-7, BSI TR-03124-1

eID-Client, Lesegerät und nPA

- Im Besitz und unter Kontrolle des Anwenders (Bürgers)
- eID-Client in mehreren Ausführungen verfügbar
  - PersoApp, AutentApp, AusweisApp1, AusweisApp2, BeID-lab, Open eCard
- Lesegeräte mit RFID / ISO-14443 inzwischen breit verfügbar
- nPA abgesichert mit eigener PIN des Anwenders

Was ist PersoApp?

- Open-Source eID-Client Bibliothek

- im April 2013 aufgesetzt
- Projektpartner
  - AGETO Innovation, Fraunhaufer SIT, TU Darmstadt
- Initial commit August/September 2013
- Ergänzung um Desktop-Anwendung ebenda
- Ergänzung um Android-Anwendung ab März 2014
- Projektdokumentation
  - http://www.persoapp.de/fuer-entwickler/dokumente/

PersoApp Struktur

- SVN Repository auf Google Code:
  - https://persoapp.googlecode.com
- Vorbereitungen für Umzug auf z.B. GitHub laufen
  - (https://github.com/PersoApp)
- Modular gegliedert in 3 Hauptprojekte
  - Core enthält TLS-RSA-PSK, PACE, SAL, EAC, CardHandler
  - Desktop enthält GUI, Localization, Configuration
  - Android wie Desktop, NFC, JAXWS/JAXB, Android Fixes

PersoApp Struktur

- Code rein in Java
- Lizenz GNU LGPL
- Abhängigkeiten sind Open-Source (z.B. BouncyCastle)
- Entwicklungsumgebung
  - Core Eclipse, Ant, IDEA + Ant
  - Desktop Eclipse (einfachster Weg), Ant, IDEA + Ant
  - Android IDEA oder AndroidStudio mit gradle, da unter Eclipse Android Plugin nicht mehr unterstützt

PersoApp-Core

- https://persoapp.googlecode.com/svn/trunk/PersoApp-Core/
  - Abhängig von Unterprojekt
    - https://persoapp.googlecode.com/svn/trunk/PersoApp-eCardAPI-1.1.2/
- Enthält Basisklassen für interne Kommunikation
  - z.B. IMainView, EventListener, ICardHandler
- Haupteintrittspunkte
  - ECApiHttpHandler für embedded Webserver auf 127.0.0.1:24727
  - ECardWorker.start(URL url) für HttpHandler, Intent, Tests, custom

PersoApp-Desktop

- https://persoapp.googlecode.com/svn/trunk/PersoApp-Desktop/
  - Abhängig von PersoApp-Core
- Beispielimplementierung zur Verwendung von Core
- „as-is“ im Design angelehnt an proprietären AGETO Client
- Enthält GUI-Elemente (Dialoge, Panels, Buttons, Grafiken)
- Ausgangspunkt für Build-Prozess einer Applikation
  - ./build/build.xml

PersoApp-Android

- https://persoapp.googlecode.com/svn/trunk/PersoApp-Android/
  - Abhängig von PersoApp-Core und PersoApp-Android/PersoApp-Lib-JAXB
- Beispielimplementierung für Android 4.1+
- Eigenständiges Design, nun Dagger, Gradle, IDEA
- Abstrahiert Android-spezifische Anpassungen
  - MainViewFacade, NFCTransportProvider, Intent
- ./PersoApp-Client/app ist der Ausgangspunkt für Build

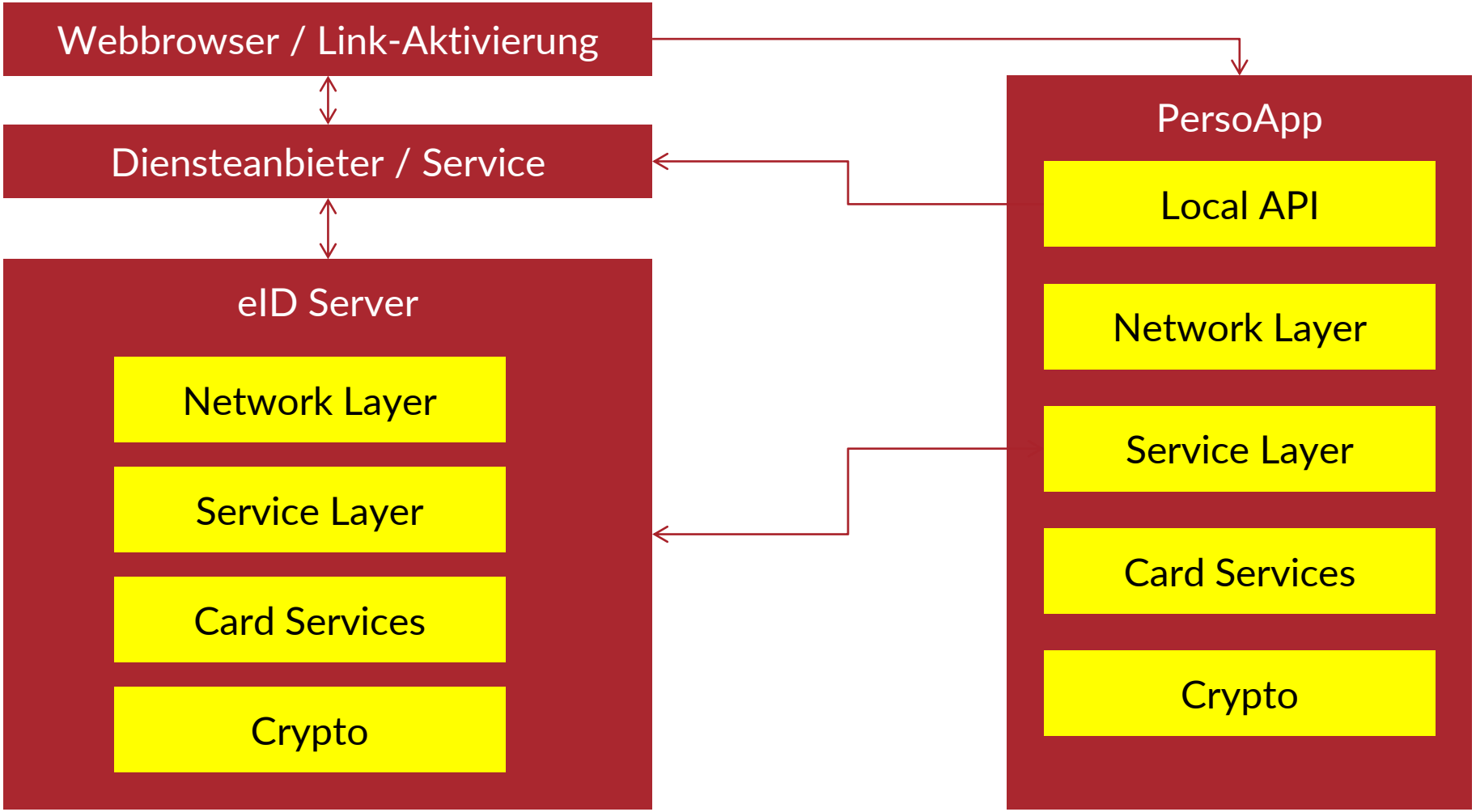
PersoApp-Core <-> GUI #1

- Interface de.persoapp.core.client.IMainView
  - Desktop: de.persoapp.desktop.MainView
  - Android: de.persoapp.android.core.adapter.MainViewFacade
    - de.persoapp.android.activity.AbstractNfcActivity
      - de.persoapp.android.core.adapter.MainViewFragment
- Initialisierung
  - Desktop: de.persoapp.desktop.PersoApp
  - Android: Injection der Facade via Dagger

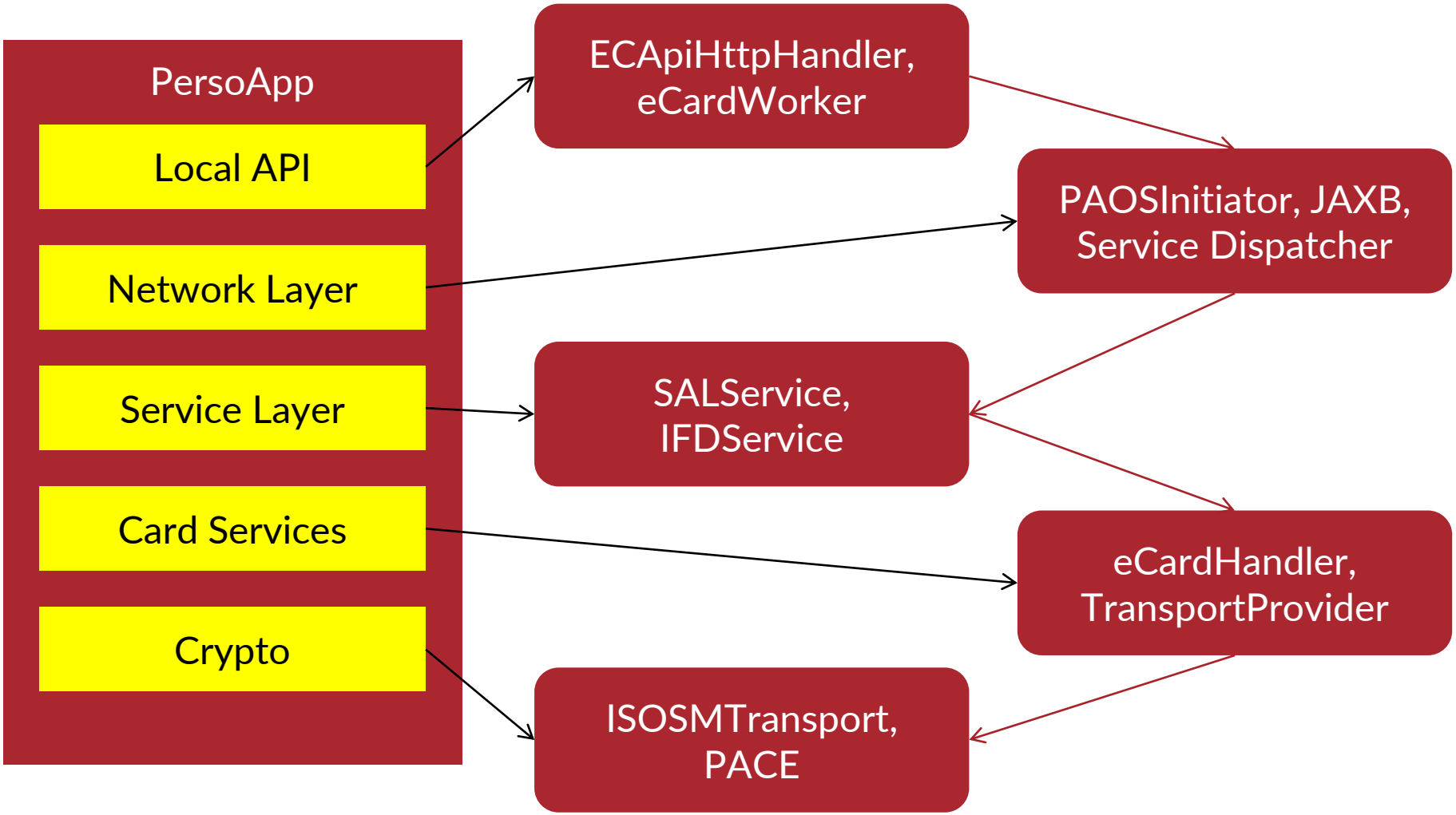
Persoapp-Core <-> GUI #2

- setEventLister(EventListener listener) mit Instanz von de.persoapp.core.client.MainViewEventListener
  - ICardHandler und IMainView als Parameter
- GUI sendet Nachrichten mit triggerEvent(int event, Object... eventData) an Core
  - Card State, PIN State, PIN Change, eSign
- Core ruft Interface Methoden von GUI
  - showMainDialog, showCANDialog, showProgress, showMessage

Übersicht über entscheidende Core-Komponenten #1



Übersicht über entscheidende Core-Komponenten #2



Übersicht über entscheidende Core-Komponenten #3

- TLS mit BouncyCastle
  - de.persoapp.core.tls.* enthält Abstraktionen für JRE SocketFactory, TLSSession, TLSClient, TLSPSKClient
- SALService implementiert EAC-Protokoll
  - de.persoapp.core.ws.SALService basiert auf WSDL und XSD des BSI (iso.std.iso_iec._24727.tech.schema.SAL)
  - eID-Server steuert didAuthenticate(…) an
- CardHandler abstrahiert nPA-Implementation und PC/SC
  - de.persoapp.core.card.CardHandler

Pause

Wiederverwendbarkeit und Erweiterbarkeit

- Szenario 1 GUI Anpassung
- Szenario 2 Einbettung in Rich-/Fat-Client
- Szenario 3 Erweiterung

Szenario 1 – GUI Anpassung

- Einfachster Weg für Form und Farbe durch Ableitung von PersoApp-Desktop
- de.persoapp.desktop.Configuration
  - LookAndFeel, Font, Icon, Logo, Claim Text
- de.persoapp.desktop.MainView
  - Layout, Dialogaufbau, -inhalt und –struktur
- Ohne Kenntnisse der Innenstruktur möglich

Szenario 2 – Einbettung in Rich-/Fat-Client

- einfachster Weg Anpassung von de.persoapp.desktop.PersoApp
- Initialisierung beibehalten, embedded Webserver und Splashscreen ggf. entfernen
- ECardWorker.start(URL tcTokenURL) mit eigener (fester) Dienst-URL rufen
  - kehrt mit Result-URL zum Aufrufer zurück
  - Abholen der ausgelesenen Daten vom eigenen Dienst
  - Weiterverarbeitung in eigener Applikation

Szenario 3 - Erweiterung

- Variante 1
  - eigener, custom eID-Server mit Sonderfunktionen (QES, eIDAS)
  - de.persoapp.core.ws.engine.WSContainer.addService() in PersoApp Hauptklasse
- Variante 2
  - Anbindung von speziellen Lesegeräten, custom TransportProvider (z.B. Lesegeräte mit LAN-Interface – siehe eGK-Umfeld)
  - Ergänzung um neue Funktionen, z.B. eIDAS Token
  - Integration mit Secure Elements für abgeleitete Ausweise

Erweiterung – Variante 2 #1

- Zuarbeit als Prototyp für FU Berlin, Institut für Informatik
- Implementierung von „PersoApp-SE-Base“
- Proof-Of-Concept verwendet PersoApp-Desktop
- Integration über angepassten CardHandler
- Protokollbasisfunktionen in Software realisiert
- Abstraktion der Chip-Authentication für Secure Element

Erweiterung – Variante 2 #2

- Verhält sich gegenüber dem eID-Server wie ein nPA
- Wird in der Test-PKI als Testausweis erkannt
- Kann vorgegebene Datensätze ausliefern
- Implementierungstiefe Anwendungsabhängig
- Erweiterbar
- Sinnvoll zum Erproben und Testen von Konzepten

PersoApp und eIDAS #1
Rechtliche Grundlagen

- eIDAS = eID and eTS
  - Electronic IDentification And trustServices
- EU Verordnung Nr. 910/2014
  - COM(2012) 238
  - Aufhebung von 1999/93/EG (EU Signatur Richtlinie)
- Vereinheitlichung nicht nur von Signatur, sondern auch von Identifizierungs- und Vertrauensdiensten
- eIDAS Token von BSI und ANSSI -> BSI TR-03110 v2.20

PersoApp und eIDAS #2
Auswirkungen und Reichweite

- BSI TR-03110-4 definiert Applikationen und Profile
  - ePassport, eID, eSign
  - European Passport, ID Card + MRTD, ID Card + opt. EU MRTD
- einheitliches Kartenprofil für eID in der gesamten EU
- technische Basis nPA
  - RFID, PACE, EAC (Terminal Authentication, Chip Authentication)
- umsetzbar auf existierender nPA-Infrastruktur

PersoApp und eIDAS #3
Vorteile

- Für alle EU-Mitgliedsstaaten eine einheitliche Infrastruktur
- Datenschutz besser umgesetzt, keine einfachen Signaturkarten mehr (Österreich, Belgien, Estland, …)
- Middleware für eID kann Ausweise aller EU-Staaten lesen
- eID-Server können generisch mit allen Ausweisen arbeiten
- Aufwändige Konstrukte aus STORK, FutureID, Open eCard nicht länger notwendig

PersoApp und eIDAS #4
Neuerungen

- Pseudonyme Signaturen (PSA, PSM, PSC)
  - ChipAuthentication V3 ohne Gruppenschlüssel umsetzbar
  - in Verbindung mit eID ist Signatur über Pseudonym möglich
- mERA, modular Enhancend Role Authentication
  - Authorization Extensions für feinere Rollen von Terminals
- Attribut Provider
  - Aufbringen von zusätzlichen Merkmalen, z.B. Versichertennummer
- Erweiterungen im SALService und neuer CardHandler

PersoApp Zukunftsoptionen

- Erweiterter Testdienst
  - Öffentlicher eID-Service Mandant (TR-03130 Webservice)
- eIDAS Integration
- Minimalistischer Open-Source eID-Server
- USB OTG CCID Unterstützung für Androids ohne NFC
- embedded nPA im Mobilbereich (SE, SIM-Karte)
  - PersoApp Secure Element mit FIDO-Support

Fragen und Antworten
Freie Themen

www.persoapp.de

Kontaktdaten

- AGETO Service GmbH
  Winzerlaer Straße 2
  07745 Jena