Bernd Zwattendorfer, Ivo Sumelong: "Interoperable Middleware-Architektur für sichere, länderübergreifende Identifizierung und Authentifizierung", Sicher in die digitale Welt von morgen: Tagungsband zum 12. Deutschen IT-Sicherheitskongress, 2011, pp. 175-189

Interoperable Middleware-Architektur für sichere, länderübergreifende Identifizierung und Authentifizierung

Bernd Zwattendorfer[1], Ivo Sumelong[2]

Kurzfassung:
Informations- und Kommunikationstechnologien spielen immer mehr eine wichtige Rolle in unserem Leben. Viele Verfahren oder Prozesse können heutzutage bereits online abgewickelt werden. Speziell in den Bereichen des E-Business oder des E-Governments gewinnen diese Technologien vermehrt an Bedeutung. Nachdem in diesen Bereichen oft sensitive Daten ausgetauscht werden, entwickelt sich auch die sichere Identifizierung und Authentifizierung im Internet zu einem immer stärker werdenden Thema. Einige EU-Mitgliedsstaaten haben diese Thematik bereits aufgegriffen und sichere, nationale eID Lösungen ausgerollt. Üblicherweise sind diese eID-Lösungen speziell auf die Anforderungen des jeweiligen Landes zugeschnitten und im länderübergreifenden Einsatz nur bedingt tauglich. Das von der EU geförderte Projekt STORK versucht diesem Missstand entgegenzuwirken und hat eine interoperable Plattform für elektronische Identitäten innerhalb der EU geschaffen. Die Plattform baut dabei im Wesentlichen auf zwei Ansätzen auf, dem sogenannten PEPS-Modell (ein Proxy-Modell mit Identitäts-Intermediären) und dem MW-Ansatz (Middleware-Modell), die miteinander interoperabel verbunden werden. Dieser Beitrag beschreibt vor allem die von Deutschland und Österreich gemeinsam entwickelte Middleware-Architektur für eine sichere, länderübergreifende Identifizierung und Authentifizierung. Neben der Architektur werden auch deren Implementierung sowie der angewandte Entwicklungs- und Auslieferungsprozess erläutert.

Stichworte: eID, elektronische Identität, Middleware, STORK, Interoperabilität, Österreichische Bürgerkarte, elektronischer Personalausweis, nPA

1. Einleitung

In den Welten des Internets spielen Identifizierung und Authentifizierung eine wesentliche Rolle. Zugriff auf schützenswerte Daten oder Services werden durch unterschiedlichste Authentifizierungsmechanismen geregelt. Der zurzeit am häufigsten verwendete Ansatz mit Benutzername und Passwort bietet aber nicht immer ausreichende Sicherheit. Eine Vielzahl an Schwachstellen dieser Authentifizierungsmethode sind bekannt **[1]**. Sind solche schwachen Authentifizierungsmechanismen für einzelne, hauptsächlich informelle Services noch ausreichend, stößt man bei komplexeren Anwendungen wie z.B. im E-Government oder im E-Business, wo oft sensiblere Daten von Nöten sind, schnell an seine Grenzen. Aus diesem Grund wird vermehrt auf sicherere Authentifizierungsmechanismen, wie z.B. Zwei-Faktor-Authentifizierung mittels Smart-Cards gesetzt. Außerdem bedarf es bei Anwendungen mit großen möglichen Nutzerzahlen, wie es z.B. die BürgerInnen eines Staates darstellen, im Rahmen der Identifizierung einer unverwechselbaren Zuordnung dieser. Diese Eindeutigkeit ist

[1] EGIZ - E-Government Innovationszentrum, Graz

[2] OpenLimit, Berlin

notwendig, um in der Anwendung dem Gebot der Datenqualität und des Datenschutzes Rechnung tragen zu können.

Einzelne Mitgliedsstaaten der EU haben bereits die Notwendigkeit einer eindeutigen Identifizierung und sicheren Authentifizierung im Rahmen von Online Anwendungen erkannt und nationale eID-Lösungen - meist basierend auf Smart-Cards - ausgerollt. Diese Lösungen sind aber sehr länderspezifisch und im grenzüberschreitenden Einsatz nicht anwendbar. Diesen Missstand hat die EU aufgegriffen und deshalb in den Ministererklärungen von Manchester 2005 **[2]**, Malmö 2009 **[3]** bzw. der Dienstleistungsrichtlinie **[4]** auch einen Fokus auf sichere elektronische Identifizierung gelegt. Die Wichtigkeit dieser Thematik wird weiters durch das von der EU-Kommission geförderte Projekt STORK **[5]** aufgezeigt, welches das Ziel einer interoperablen Plattform für elektronische Identitäten innerhalb der EU besitzt.

Dieser Artikel stellt einen wesentlichen Teil der STORK-Plattform basierend auf dem sogenannten Middleware-Ansatz vor. Die Architektur dieses Ansatzes sowie deren Implementierung werden in den Abschnitten 4 und 5 beschrieben. Eine Einführung in unterschiedliche Identitätsmodelle sowie Interoperabilitäts-Konzepte zwischen den einzelnen Modellen der STORK-Plattform werden in den Abschnitten 2 und 3 gegeben. Letztendlich wird die Arbeit zusammengefasst.

2. Identitätsmodelle

Die Identifizierung und Authentifizierung von elektronischen Identitäten stellen keine neuen Probleme dar. Unterschiedlichste Systeme für das Management von Identitäten existieren bereits und sind im Einsatz. Nicht alle diese Systeme verfolgen aber den gleichen methodologischen Ansatz. So basieren einzelne System eher auf dem Ansatz der zentralen Speicherung von Identitätsdaten, währenddessen andere Systeme eher auf einen föderierten Ansatz setzen. In Anlehnung an **[6]** stellt dieses Kapitel unterschiedliche Identitätsmodelle (Benutzer-zentrierter, zentraler sowie föderierter Ansatz) vor. Unterscheidungskriterium ist einerseits, wo die Identitätsdaten gespeichert werden (z.B. Smart-Card, zentrale Datenbank), und andererseits die Ende-zu-Ende-Sicherheit.

Benutzer-zentrierter Ansatz:

Üblicherweise erfolgt die Identifizierung bzw. Authentifizierung eines Benutzers an einem Service Provider über einen sogenannten Identity Provider (Identitäts-Provider). Dieser Identity Provider ist dafür zuständig, dass die Identitäts- und Authentifizierungsdaten des Benutzers an den Service Provider entsprechend weitergeleitet werden. In diesem Benutzer-zentriertem Modell bleibt immer der Benutzer selbst der Eigner seiner Identitätsdaten. Die Identitätsdaten werden alle beim Benutzer verwaltet (z.B. mittels Smart-Card) und werden nur an den Service Provider weitergegeben, wenn der Benutzer explizit dazu zustimmt. Im Endeffekt entsteht eine direkte Kommunikation zwischen dem Service Provider und dem Benutzer, sodass eine Ende-zu-Ende-Sicherheit gewährleistet werden kann.

Zentraler Ansatz:

Dieser Ansatz ist das derzeit dominierende Modell im Internet. Bevor ein Benutzer ein bestimmtes Service nutzen kann, muss er sich zuerst beim Service Provider bzw. Identity Provider registrieren. Diese Daten werden nun vom Identity Provider zentral verwaltet und gespeichert. Für eine Anmeldung am Service Provider muss sich der Benutzer zuerst ordnungsgemäß beim Identity Provider authentifizieren, welcher anschließend die Daten an den Service Provider weiterleitet. Im Gegensatz zum Benutzerzentriertem Modell besitzt der Benutzer in diesem Modell keine Kontrolle mehr darüber, welche Daten genau gespeichert und an den Service Provider weitergeleitet werden.

Föderierter Ansatz:

In diesem Modell sind die Identitäts- bzw. Benutzerdaten über mehrere Identity Provider verteilt, die über ein gemeinsames Vertrauensverhältnis verfügen (meist auf organisatorischer Ebene). Üblicherweise sind die Datenspeicher der einzelnen Identity Provider miteinander verbunden und ein Datenaustausch kann einfach vollzogen werden. Meist erfolgt die Übereinkunft für einen Datenaustausch über einen gemeinsamen Identifier für einen bestimmten Benutzer.

3. Interoperabilitäts-Konzepte und Modelle

Die STORK-Plattform basiert im Wesentlichen auf zwei Ansätzen, dem sogenannten PEPS-Ansatz (ein Proxy-Modell mit Identitäts-Intermediären) und dem MW-Ansatz (Middleware-Modell), welcher vertiefend in den weiteren Kapiteln dieses Artikels beschrieben wird **[7]**.

Beim PEPS-Ansatz (Pan-European Proxy Service) existiert pro Mitgliedsstaat ein nationales Gateway (PEPS), welches alle Services der nationalen eID Lösung (z.B. Kommunikation zum Service Provider und den Identitäts- bzw. Attribut-Providern) sowie alle Funktionen für eine länderübergreifende Authentifizierung kapselt. In diesem Fall stellt der PEPS einen Intermediär zwischen dem Service Provider und dem eigentlichen Identitäts-Provider dar und garantiert dem Service Provider, dass sich ein Benutzer ordnungsgemäß authentifiziert hat. Abbildung 1 skizziert ein länderübergreifendes PEPS-Modell. Ein PEPS kann entweder als S-PEPS (PEPS im Mitgliedsstaat des Service Providers) oder als C-PEPS (PEPS im Mitgliedsstaat des Benutzers) agieren. Ein S-PEPS kommuniziert mit dem Service Provider und dem C-PEPS und stellt somit einen Intermediär zwischen den beiden dar. Ein C-PEPS empfängt Authentifizierungsanfragen vom S-PEPS und triggert den Identifizierungs- und Authentifizierungsprozess an einem Identitäts- und/oder Attribut-Provider für einen Benutzer.

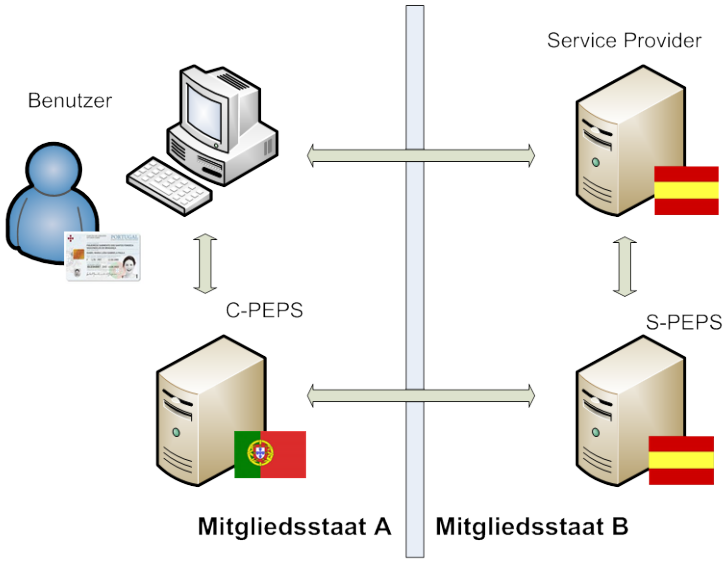

Abbildung 1 - PEPS Modell

Im Gegensatz zum PEPS-Modell authentifiziert sich im MW-Modell (siehe Abbildung 2) ein Benutzer direkt am Service Provider, d.h. der Service Provider unterstützt direkt alle gewünschten Identifizierungs- und Authentifizierungsmethoden, meist basierend auf Smart-Cards. Zur Unterstützung wird meist eine server-seitige Middleware (in Abbildung 2 der VIDP – Virtual Identity Provider) eingesetzt, welche direkt in der Domäne bzw. Infrastruktur des Service Providers läuft. Dieses MW-Modell kann dem Benutzer-zentrierten Ansatz aus Kapitel 2 zugeordnet werden. Da die Identitätsdaten direkt beim Benutzer liegen und er explizit einer Weitergabe an den Service Provider zustimmen muss, wird der Datenschutz entsprechend gewahrt. Ein weiterer Vorteil dieses Ansatzes liegt in der Möglichkeit der Ende-zu-Ende Sicherung, da eine direkte Kommunikation zwischen dem Service Provider und dem Benutzer bzw. dessen eID Karte stattfindet, welche im PEPS-Modell nicht gegeben ist.

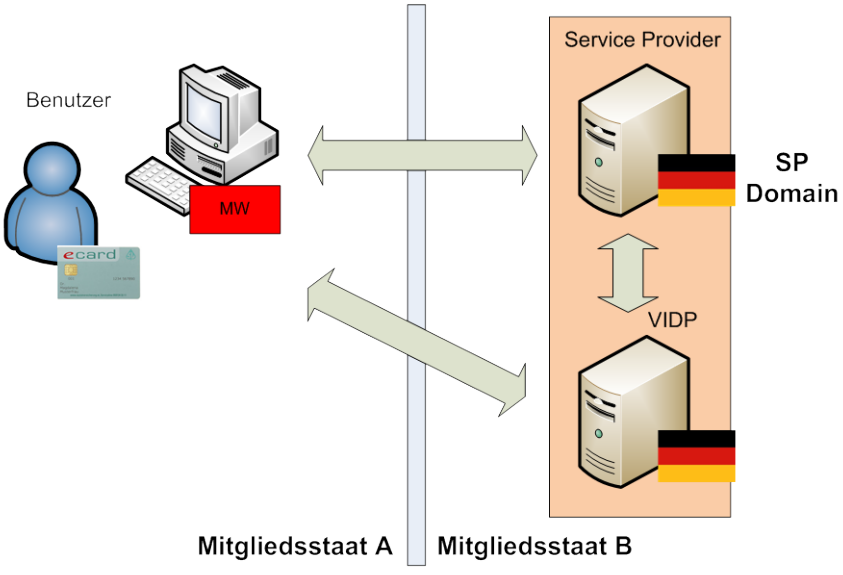

Abbildung 2 - Middleware Modell

Im Rahmen des STORK-Projekts wird versucht, diese beiden Modelle miteinander zu verbinden. D.h. beispielsweise, dass sich Bürger, deren Heimatland eine nationale MW-Lösung im Einsatz hat, auch in fremden Ländern mit ihren nationalen eIDs sicher identifizieren und authentifizieren können, obwohl der jeweilige andere Mitgliedsstaat national den PEPS-Ansatz verfolgt. STORK schafft also ein übernationales Interoperabilitäts-Framework für eID Authentifizierung, ohne dass eine bereits existierende nationale Infrastruktur massiv verändert werden muss.

Verbindet man beide Modelle, so ergeben sich vier unterschiedliche Szenarien:

1. Ein Bürger eines PEPS-Landes (PEPS-Infrakstuktur national deployed) möchte sich in einem anderen PEPS-Land sicher authentifizieren (siehe Abbildung 1)
2. Ein Bürger eines MW-Landes (MW-Infrakstuktur national ausgerollt) möchte sich in einem anderen MW-Land sicher authentifizieren (siehe Abbildung 2)
3. Ein Bürger eines PEPS-Landes möchte sich in einem MW-Land sicher authentifizieren (siehe Abbildung 3)
4. Ein Bürger eines MW-Landes möchte sich in einem PEPS-Land sicher authentifizieren (siehe Abbildung 4)

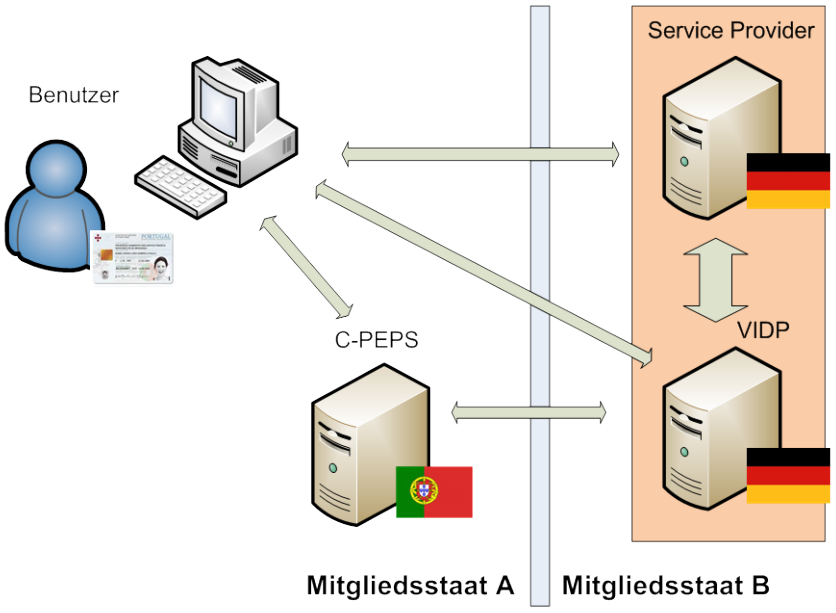

Abbildung 3 - MW - PEPS Interoperabilitätsmodell

Abbildung 3 illustriert den Authentifizierungsfall, bei dem sich ein Benutzer eines PEPS-Landes an einem Service Provider eines Middleware-Landes anmelden möchte. Abbildung 4 stellt den umgekehrten Fall dar, wobei sich hier ein Benutzer aus einem MW-Land bei einem Service Provider eins PEPS-Landes authentifizieren möchte. Das im Rahmen von STORK entwickelte Framework versucht also, beide Modelle miteinander zu verbinden. Der Austausch der Identifizierungs- und Authentifizierunsdaten erfolgt auf Basis der Security Assertion Markup Language (SAML) **[8]**. Details zum

länderübergreifenden Protokoll können in der entsprechenden STORK Interface Spezifikation **[9]** nachgelesen werden.

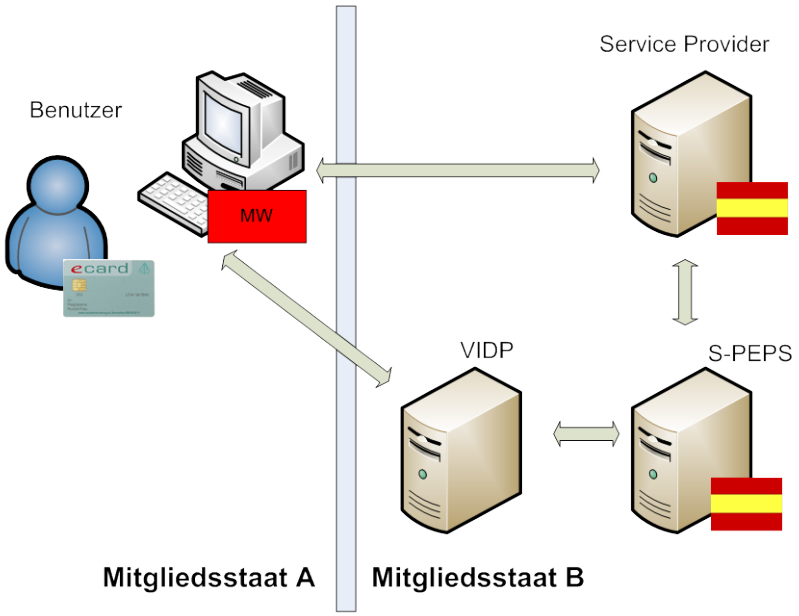

Abbildung 4 - PEPS - MW Interoperabilitätsmodell

4. Middleware-Architektur

Aufgrund der sich aus dem MW-Modell ergebenen Ende-zu-Ende-Sicherheit haben sich Länder wie z.B. Deutschland und Österreich für den MW-Ansatz entschieden. Österreich hat bereits seit einigen Jahren eine nationale eID Lösung basierend auf Middleware flächendeckend im Einsatz (*Österreichische Bürgerkarte* **[10]**), Deutschland ist gerade im Aufbau einer MW-Infrastruktur basierend auf dem *neuen Personalausweis* (nPA) **[11]**.

Die im Rahmen des Projekts STORK entwickelte gemeinsame MW-Architektur (VIDP - Virtual Identity Provider) versucht beide nationalen Lösungen miteinander zu kombinieren. D.h. auf Basis dieser Architektur wird es österreichischen bzw. deutschen Service Providern ermöglicht, länderübergreifende Authentifizierung mittels eID über Middleware anbieten zu können. Abbildung 5 zeigt die von Deutschland und Österreich entwickelte Middleware-Architektur, welche den hohen Modularitäts- und Skalierbarkeitsansprüchen genügt.

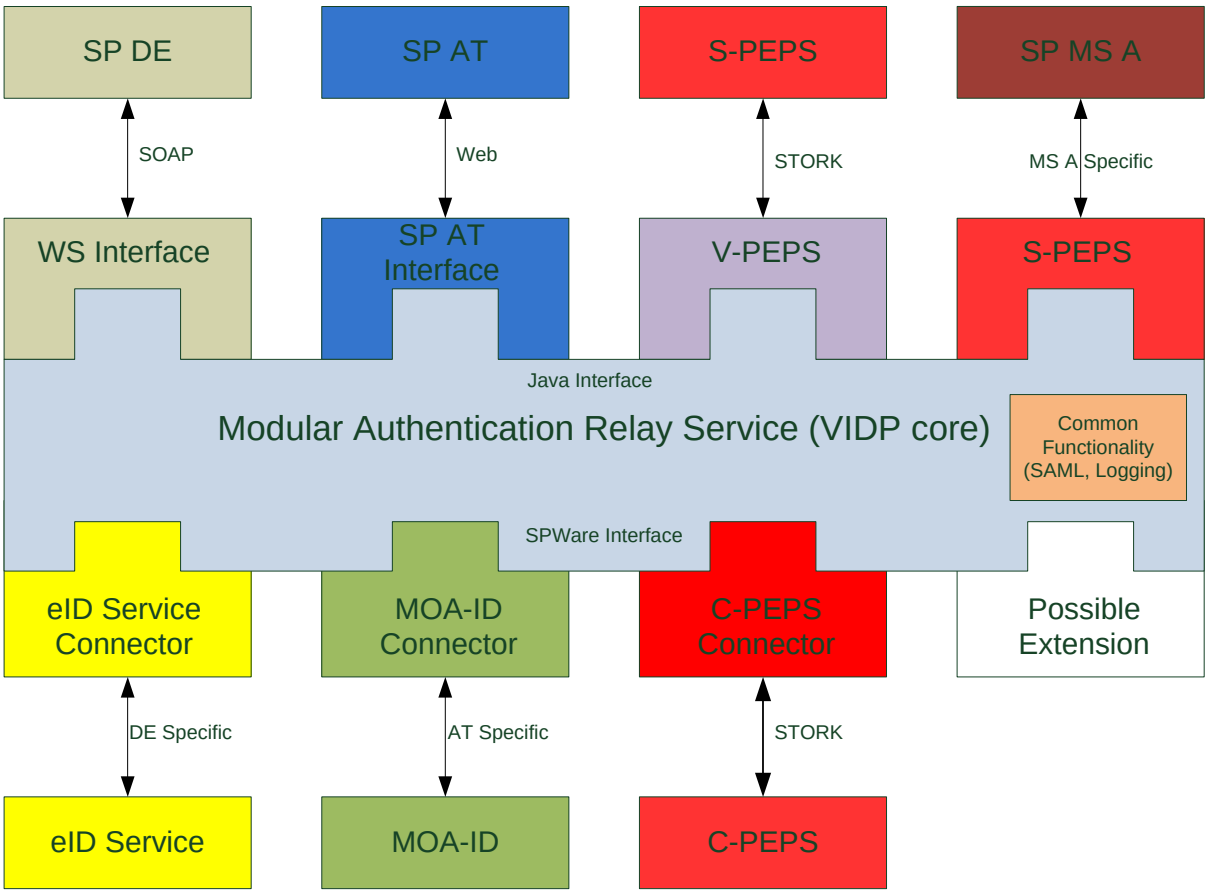

Abbildung 5 - Middleware-Architektur

Aufgrund des modularen Aufbaus können aber sowohl weitere MW-Lösungen anderer Länder als auch Länder, die den Proxy-basierten Ansatz verwenden, integriert werden. Für die Integration eines anderen MW-Landes müssen nur die beiden Interfaces (Java Interface und SPWare Interface) des Modular Authentication Relay Services (MARS) implementiert werden. Länder die den PEPS-Ansatz verfolgen – in weiterer Folge als PEPS-Länder bezeichnet – werden bereits unterstützt. In diesem Fall (Szenario 3 aus Abschnitt 3 – siehe auch Abbildung 3) leitet der sogenannte C-PEPS Connector einen Authentifizierungs-Request an den jeweiligen Länder-PEPS (C-PEPS) weiter. Der Benutzer authentifiziert sich anschließend ordnungsgemäß an seinem nationalen PEPS und dieser schickt die Identifikations- und Authentifizierungsdaten verpackt in einem SAML-Token **[8]** an den VIDP zurück. Der VIDP überprüft die Gültigkeit des Tokens und übermittelt bei erfolgreicher Validierung die Daten über die entsprechende nationale Schnittstelle an den Service Provider. Das für den länderübergreifenden Datenaustausch verwendete Protokoll basiert auf der entsprechenden STORK Interface Spezifikation **[9]**, die zum Großteil auf SAML aufbaut.

Der modulare Ansatz bietet aber nicht nur die Möglichkeit eID-Lösungen anderer Länder zu integrieren, sondern gestattet auch die Umgestaltung eines VIDP zu einem kompletten PEPS. Durch den gemeinsamen Einsatz der Module S-PEPS und C-PEPS Connector kann mittels der gewählten Architektur einfach ein PEPS realisiert werden.

Die aktuelle und für die Pilotphase ausgerollte Implementierung dieser MW-Architektur (VIDP) beinhaltet folgende Komponenten **[12] [13]**:

- WS Interface: Diese Schnittstelle basiert auf SOAP und wird von deutschen Service Providern verwendet.
- SP AT Interface: Eine Web-Schnittstelle zur Unterstützung von bereites existierenden österreichischen Service Providern.
- V-PEPS: Über diese Schnittstelle empfängt der VIDP einen Authentifizierungsrequest vom PEPS (SAML AuthnRequest).
- eIDService Connector: Dieses Plug-In ist für die Kommunikation mit dem deutschen eID-Service zuständig.
- MOA-ID Connector: Dieses Modul leitet eine Authentifizierungsanfrage an die österreichische Middleware MOA-ID (server-seitig) weiter.
- C-PEPS Connector: Mit Hilfe dieses Moduls können Benutzer aus PEPS-Ländern sich an einem Service Provider, welcher auf das MW-Modell setzt, authentifizieren.

5. Implementierung, Auslieferung und Deployment

Dieses Kapitel beschäftigt sich mit Überlegungen hinsichtlich der Implementierung, der Auslieferung und dem Deployment der zuvor beschriebenen Middleware-Architektur. Es werden eine von Deutschland und Österreich gemeinsam entwickelte Software-Architektur auf Basis von J2EE[3] Referenzkomponenten sowie unterschiedliche Deployment-Strategien vorgestellt. Letztendlich wird auch auf die Sicherheits-Architektur eingegangen.

Die wichtigsten Überlegungen für diese Strategie waren **[14]**:

- Entkopplung der einzelnen Module für dynamisches Deployment
- Dynamisch konfigurierbare Modell-Architektur
- Sicherheit auf Nachrichten- sowie Kommunikationsebene
- Unterstützung der gängigsten Datenbank- und Applikationssever

2.1 Entwicklungs- und Auslieferungsprozess

Die gemeinsam vom STORK-Konsortium entwickelte und abgestimmte Interface-Spezifikation **[9]** beschreibt ein „lebendes“ Dokument und wurde deshalb im Rahmen des Projektes immer wieder aufgrund unterschiedlicher Anforderungen der Mitgliedsstaaten bzw. notwendiger Verbesserungen oder Fehlerbehebungen adaptiert und somit leicht verändert. Diese Dynamik im Lebenszyklus der Spezifikation bedarf daher eines

[3] http://www.oracle.com/technetwork/java/javaee/tech/index.html

geeigneten und angepassten Entwicklungs- und Auslieferungsprozesses. Die folgenden Ziele und Prozesse wurden deshalb für den gemeinsamen Entwicklungs- und Auslieferungsprozess des VIDP gesetzt:

- Agiler Entwicklungs- und Auslieferungsprozess für das Abfedern von Erweiterungen oder störenden Ereignissen
- Sicherer Entwicklungs- und Auslieferungsprozess
- Automatisierung und Überwachung der Prozesse
- Sicherer und konsistenter Konfigurationsprozess
- Ein gemeinsamer Entwicklungs- und Auslieferungsprozess für unterschiedliche Infrastrukturen (z.B. Server)

Abbildung 6 veranschaulicht diesen definierten Entwicklungs- und Auslieferungsprozess, der auf dem Prinzip der kontinuierlichen Integration **[15]** basiert. Prinzipiell wird zwischen drei Ebenen für die Entwicklung und Auslieferung der VIDP-Komponenten unterschieden:

1. CI (Continous Integration) für Entwickler
2. QA (Quality Assurance) zum Testen
3. LIVE für die finale Auslieferung

Auch die Konfiguration der einzelnen Module bedarf eines automatisierten und verfolgbaren Konfigurationsprozesses. Zuerst werden alle Konfigurationen in das Staging System übernommen. Nach erfolgreicher Überprüfung werden die Konfigurationseinstellungen an das Master-System übergeben, von dem ein entsprechendes Backup angefertigt wird. Die Aufgabe des sogenannten Snapshoters ist es nun, die Konfigurationen zu replizieren und den einzelnen Ebenen (CI, QA, LIVE) zugänglich zu machen. Globale Konfigurationen werden an allen Systemen deployed während systemspezifische oder kontextbezogene Konfigurationen nur auf bestimmten Systemen deployed werden.

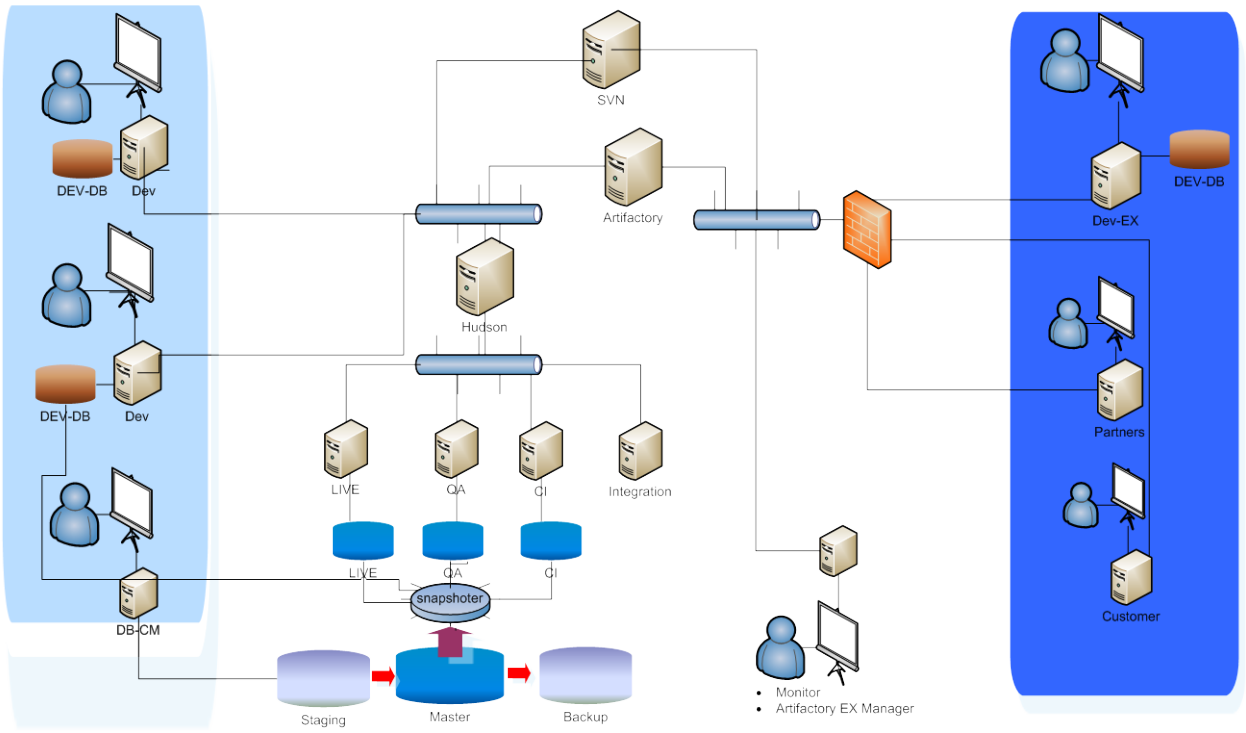

Abbildung 6 - Sicherer und automatisierter Entwicklungs- und Auslieferungsprozess für den VIDP

Die folgenden Tools bzw. Softwaremodule wurden installiert und verwendet, um einen nachhaltigen Entwicklungs- und Auslieferungsprozess gewährleisten zu können:

- Subversion[4]

  Ein Source Code Management System welches eine Versionierung und einfache Verwaltung des Quellcodes ermöglicht.

- Hudson[5]

  Vereinfacht die Softwareentwicklung auf Basis der kontinuierlichen Integration.

- Artifactory[6]

  Unterstützt einen einzigen Zugangs- und Kontrollpunkt für externe Libraries.

- Maven[7]

  Dieses Build-Tool ermöglicht eine modulare Realisierung des Build-Prozesses.

- Glassfish[8] Applikationsserver

[4] http://subversion.tigris.org/

[5] http://hudson.java.net/

[6] http://www.jfrog.org/products.php

[7] http://maven.apache.org/

Dieser Applikationsserver wird einerseits als Server für die Module der kontinuierlichen Integration (Hudson, Artifactory) und andererseits als Referenz-Instanz für automatisierte Integrationstests genutzt.

- JBoss[9], Weblogic[10] Applikationsserver

  Andere Applikationsserver zur Unterstützung von unterschiedlichen Infrastrukturen.

- MySQL[11], Oracle[12] Datenbanken

  Unterstützung der verwendeten Persistenz für unterschiedliche Datenbank-Anbieter.

2.2 Implementierung der Middleware-Architektur

Dieser Abschnitt beschreibt kurz die eigentliche Implementierung der Middelware-Architektur und die Überlegungen, die für die Implementierung angestellt wurden. Um möglichst viel Flexibilität und Dynamik gewährleisten zu können, wurde eine Implementierung auf Basis von EJBs[13] (Enterprise Java Beans) bzw. Web Services gewählt. Die Verwendung von definierten Schnittstellen gibt dazu genug Raum für die Entkopplung einzelner Module und für dynamische Deployment Möglichkeiten. Das Hinzufügen oder Entfernen von Modulen während der Laufzeit bedeutet keine Beeinträchtigung für das System. Abbildung 7 zeigt das Komponenten-Diagramm der implementierten Middleware.

[8] https://glassfish.dev.java.net/

[9] http://jboss.org/

[10] http://www.oracle.com/us/products/middleware/application-server/index.html

[11] http://www.mysql.com/

[12] http://www.oracle.com/de/products/database/index.html

[13] http://www.oracle.com/technetwork/java/index-jsp-140203.html

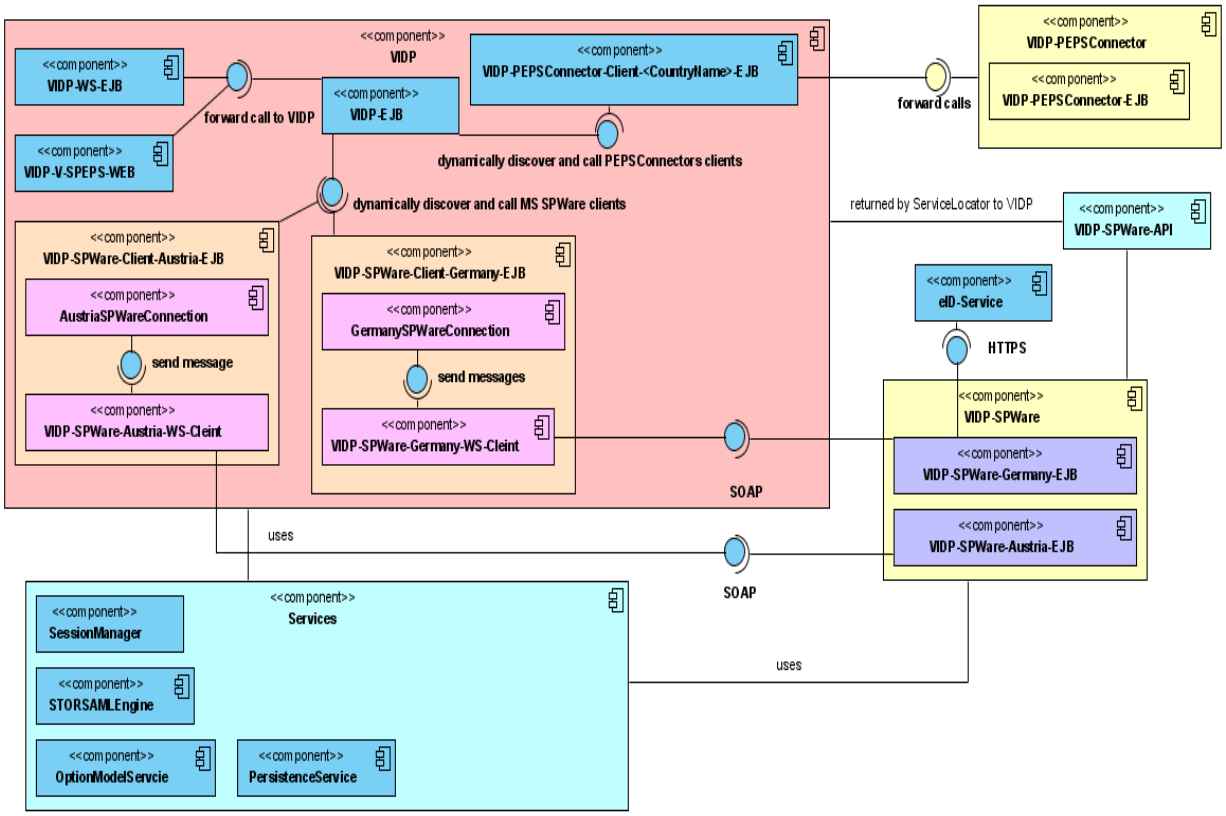

Abbildung 7 - Komponentendiagramm der STORK Middleware

2.3 Deployment Möglichkeiten

Eine auf Abbildung 7 basierende Middleware-Implementierung ermöglicht eine flexible Gestaltung des Deployments. Je nach Verfügbarkeit von Ressourcen oder gewünschten anderen Eigenschaften wie Flexibilität oder Wartungsaufwand kann eine unterschiedliche Deployment-Strategie gewählt werden. Darüber hinaus wird sowohl eine statische als auch dynamische Erweiterbarkeit des VIDP unterstützt. Dynamisch heißt in diesem Zusammenhang, dass zur Laufzeit Module (z.B. C-PEPS Connector, SPWare) zur Erweiterung einfach hinzugefügt bzw. entfernt werden können, ohne den Betrieb der Middleware zu beeinträchtigen.

Die folgenden Deployment-Möglichkeiten werden unterstützt:

- Loses Deployment

  Module wie z.B. PersistenceService oder SPWare können als einzelne, verteilt laufende Instanzen deployed werden.

- Gekoppeltes Deployment

  Alle Module werden auf einer gemeinsamen Server-Instanz deployed.

- Globales Security Gateway

  Vor dem VIDP wird ein globales Security Gateway deployed welches den VIDP global absichert.

- Modulare Security Gateways

  Vor jedem einzelnen Modul wird ein Security Gateway deployed, welches die bidirektionale Kommunikation der Module intern absichert.

- Unterschiedliches Architektur-Modell zwischen SP und Middleware

  Eine Feinabstimmung der Architektur zwischen SP und Middleware (z.B. synchrones oder asynchrones Architektur-Modell) kann beispielsweise durch dynamisches Überschreiben einer vordefinierten statischen Konfiguration zur Laufzeit erfolgen.

Um diese unterschiedlichen, flexiblen und skalierenden Deployment-Ansätze anbieten zu können, wurden jeweils geeignete APIs für die einzelnen Module basierend auf den J2EE-Interfaces Local, Remote bzw. auf Web Services (SOAP) definiert. Ein Wechsel von einem Interface zu einem anderen kann einfach und dynamisch während der Laufzeit erfolgen, ohne dass das beteiligte Modul davon eine Beeinträchtigung erfährt.

2.4 Sicherheit

Die Implementierung der Middleware versucht ein hohes Maß an Sicherheit zu gewährleisten. Dafür wird auf unterschiedlichen Ebenen auf bereits existierende und erprobte Sicherheitsmechanismen gesetzt:

- Authentifizierung und Autorisierung

  Einerseits ist die Implementierung für den Support von Role Based Access Control (RBAC) Modellen und Attribute Based Access Control (ABAC) Modellen zwischen Modulen vorbereitet, anderseits wird z.B. bei Web Services die Authentifizierung und Autorisierung durch beidseitige SSL/TLS Authentifizierung erreicht.

- Nachrichtensicherheit

  Das Maß an Sicherheit auf Nachrichtenebene kann dynamisch konfiguriert werden. Das Signieren und/oder Verschlüsseln von einzelnen Nachrichten wird dabei unterstützt.

- Sicherheit auf Transportebene

  Für die Sicherheit auf Transportebene wird auf den weit verbreiteten Standard der SSL/TLS-Verschlüsselung gesetzt.

- Bidirektionale Sicherheit

  Modular verfügbare Security Gateways können die bidirektionale Kommunikation absichern. Dies reicht von Authentifizierungsservices, über Signatur- und Verschlüsselungsservices bis hin zu Services zur Abwehr von Denial-of-Service-Attacken.

6. Zusammenfassung

Die in diesem Artikel vorgestellte Architektur basiert auf dem sogenannten Middleware-Ansatz und unterstützt eIDs von unterschiedlichen EU Mitgliedsstaaten. Der MW-Ansatz ist neben dem PEPS-Ansatz das zweite im Rahmen von STORK verwendete Modell für nationale eID-Infrastrukturen. Die Entscheidung, welches Modell national eingesetzt wird, hängt von unterschiedlichen Gesichtspunkten ab. Für einen Einsatz von MW gegenüber PEPS sprechen vor allem die Ende-zu-Ende-Sicherheit, die Skalierbarkeit und die Haftung, da in diesem Fall kein Intermediär zwischen dem Service Provider und dem Benutzer auftritt.

Deutschland und Österreich haben gemeinsam die beschriebene MW-Architektur auf Basis von J2EE-Komponenten implementiert. Dabei wurde besonders auf eine dynamische Konfiguration, die Möglichkeit eines dynamischen Deployments, die Sicherheit zwischen den Komponenten und auf die Unterstützung gängiger Datenbank- und Applikationsserver geachtet. Der Entwicklungs- und Auslieferungsprozess wurde auch entsprechend dahingehend gestaltet und gängige Softwaremodule zur Unterstützung verwendet. Die beschriebene Implementierung wird aktuell in sechs konkreten Pilotenprojekten in Produktionsumgebungen deployed und erprobt.

Literatur:

[1] Gary C. Kessler: Passwords – Strengths and Weaknesses, Internet and Networking Security, J.P. Cavanagh (ed.), Auerbach, 1997

[2] Ministerial Declaration approved unanimously, Manchester, United Kingdom, on 24 November 2005

[3] Ministerial Declaration on eGovernment approved unanimously, Malmö, Sweden, on 18 November 2009

[4] The European Parliament and the Council of the European Union: Directive 2006/123/EC of the European Parliament and of the Council of 12 December 2006 on services in the internal market, 2006

[5] Secure Identity Across Borders Linked (STORK), https://www.eid-stork.eu/

[6] John Palfrey, Urs Gasser: Digital Identity Interoperability and eInnovation, Case Study, November 2007, Berkman Publication Series

[7] Jan Eichholz, Adrian Johnston, Herbert Leitold, Marc Stern, John Heppe: D5.1 Evaluation and assessment of existing reference models and common specs, STORK Deliverable, 2010

[8] Security Assertion Markup Language (SAML), OASIS Security Services (SAML) TC, http://www.oasis-open.org/committees/tc_home.php?wg_abbrev= security

[9] Joaquín Alcalde-Morano, Jorge López Hernández-Ardieta, Adrian Johnston, Daniel Martinez, Bernd Zwattendorfer, Marc Stern: D5.8.1b Interface Specification, STORK Deliverable, 2009

[10] Herbert Leitold, Arno Hollosi, Reinhard Posch: Security Architecture of the Austrian Citizen Card Concept, Proceedings of the 18th Annual Computer Security Applications Conference, 2002

[11] Bundesamt für Sicherheit in der Informationstechnik (BSI): Das eCard-API-Framework (BSI TR-03112), 2009

[12] Diana Berbecaru, Eva Jorquera, Joaquín Alcalde-Morano, Renato Portela, Wolfgang Bauer, Bernd Zwattendorfer, Jan Eichholz, Tim Schneider: D5.8.1a Software Architecture Design. STORK Deliverable, 2009

[13] Herbert Leitold, Bernd Zwattendorfer: STORK: Architecture, Implementation and Pilots

[14] Ivo Sumelong, Armin Lunkeit, Bernd Zwattendorfer: Technical Concept of the STORK Interoperability Framework Architecture

[15] Jez Humble, David Farley: Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation, Addison-Wesley