Humboldt-Universität zu Berlin

Mathematisch-Naturwissenschaftliche Fakultät II

Institut für Informatik

SAML Identity Federation und die eID-Funktionalität des nPA

Martin Schröder

Studienarbeit

Betreuer: Dr. Wolf Müller

Frank Dietrich

Berlin, den 13. Juni 2011

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

Abkürzungsverzeichnis

AP	Attribute Provider
CAS	Claim Assertion Service
DA	Diensteanbieter (eID-Kontext)
ECP	Enhanced Client or Proxy
IdP	Identity Provider
nPA	neuer Personalausweis
PKI	Public Key Infrastructure
PSK	Pre-shared Key
rID	Restricted Identifier
SAML 2.0	Security Assertion Markup Language Version 2.0
SP	Service Provider (SAML-Kontext)
VfB	Vergabestelle für Berechtigungszertifikate
XML	Extensible Markup Language

1 Einleitung

1.1 Motivation

In der heutigen Informationsgesellschaft ist die eigene digitale Identität ein wichtiges schützenswertes Gut. Leider sehen sich Nutzer immer häufiger dazu gezwungen im Internet persönliche Daten weiterzugeben, um einen bestimmten Dienst verwenden zu können. Die Spanne dieser Daten kann von der Zusicherung der Volljährigkeit bis zur Preisgabe der kompletten Adressdaten reichen. Hinzu kommt, dass man sich für jeden dieser Dienste unterschiedliche Zugangsdaten merken muss, was eine Unbequemlichkeit darstellt, welcher meist durch Merken gleicher oder zumindest ähnlicher Passwörter entgangen wird. Somit entsteht eine Sicherheitslücke über die bei einem erfolgreichen Angriff auf einen Dienst nicht nur die Identitätsinformationen der Kunden dieses Dienstes sondern auch weiterer Dienste eventuell abgegriffen werden können.

Mit Hilfe der eID-Funktionalität des nPA soll die Lücke insofern geschlossen werden, als dass sich die Dienste nicht mehr selber um die Authentifizierung des Benutzers und die Speicherung sensibler Kundendaten kümmern müssen. Möchte der Nutzer solch einen Dienst benutzen, authentisiert er sich mit Hilfe seines nPA bei einem vertrauenswürdigen Identity Provider, welcher ihm ein Identitätstoken ausstellt, das die vom Dienst benötigten Informationen enthält. Mit diesem Token kann sich der Benutzer nun bei dem Dienst anmelden und ihn verwenden.

Dieser Ansatz setzt jedoch voraus, dass alle Informationen, die der Dienst zur Authentifizierung des Nutzers benötigt, von einem Identity Provider gestellt werden. Der Fall, dass ein Dienst im selben Vorgang auch auf Daten von weiterer Attribute Providern angewiesen ist, wurde bisher noch nicht betrachtet. Zwar ließe sich das in einem naiven Ansatz durch einzelne Abfragen des Dienstes an die jeweiligen Attribute Provider realisieren, jedoch muss sich dabei der Benutzer für jede Abfrage neu authentisieren. Somit muss für dieses Anwendungsszenario eine alternative Lösung gefunden werden.

1.2 Zielstellung

Ziel der Studienarbeit ist es, eine auf den SAML 2.0 Standard basierende Lösung zu finden, bei der, wie in Abbildung 1.1 ersichtlich, Identitätsinformationen aus dem neuen Personalausweis mit denen verschiedener Attribute Provider kombiniert und zur Nutzung eines Dienstes verwendet werden können. Hierbei wird vor allem auf die Benutzerfreundlichkeit und den Schutz der Privatsphäre Wert gelegt. So soll der Nutzer seine Ausweis-PIN möglichst nur einmal eingeben müssen und die Attribute Provider nur Zugriff auf diejenigen Attribute der Nutzeridentität haben, die von ihnen selber bereit gestellt werden.

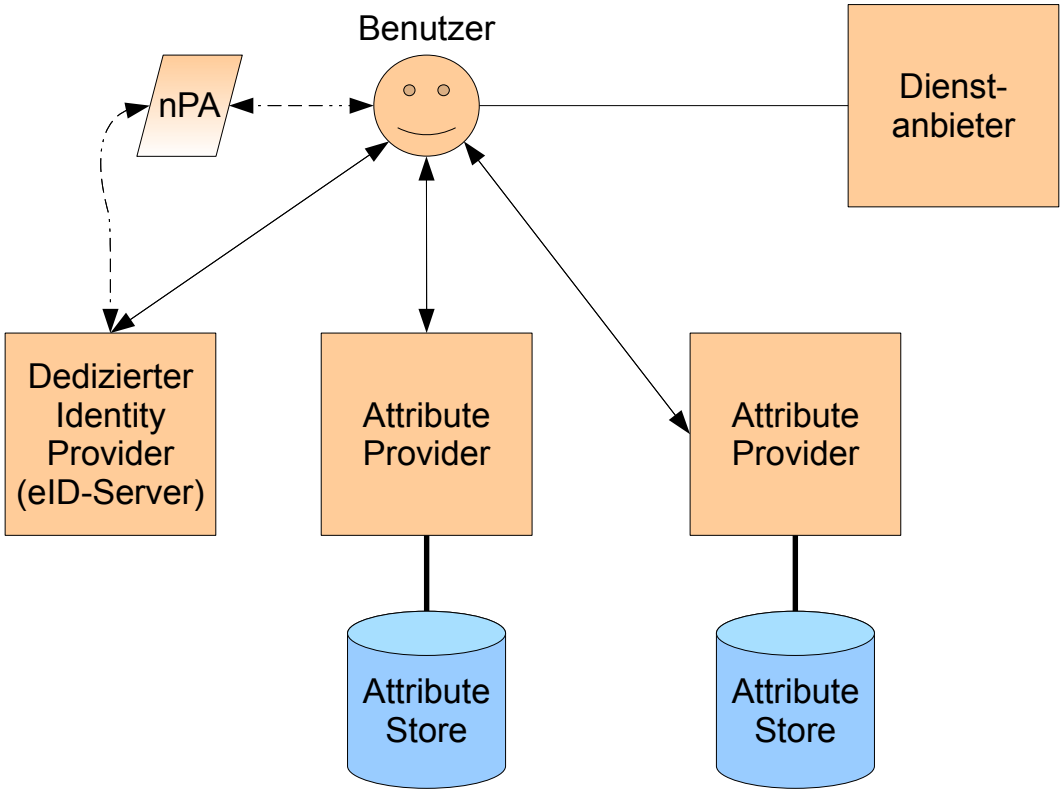

Abbildung 1.1: Übersicht

2 Grundlegende Begriffe

2.1 XML-Enc und XML-Sig

Da im Folgenden über XML-Strukturen und deren Verschlüsselung bzw. Signierung geschrieben wird, sollen in diesem Kapitel die dafür zuständigen Techniken XML-Encryption [9] und XML-Signature [10] kurz erwähnt werden. Für eine Erklärung, um was es sich bei XML handelt, sei auf die entsprechende Spezifikation [14] des World Wide Web Consortium (W3C) verwiesen.
Verschlüsselung und Signaturen sind kryptografische Vorgänge, welche verschiedenen Schutzzielen der IT-Sicherheit dienen. Durch das Verschlüsseln von Nachrichten wird sicher gestellt, dass diese *vertraulich* sind, also nur von dazu berechtigten Person gelesen werden können. Elektronische Signaturen haben mehrere Aufgaben. Zum einen lässt sich durch sie die *Integrität* einer Nachricht, also dessen Unversehrtheit, feststellen. Zum anderen kann der Ersteller der Nachricht *nicht abstreiten*, dass sie von ihm angefertigt wurde.

XML-Encryption

Mit XML-Enc können komplette Dokumente, einzelne Elemente oder auch nur deren Inhalte verschlüsselt werden. Zusätzlich können die Elemente auch für mehrere Empfänger verschlüsselt werden, indem unterschiedliche Schlüssel verwendet werden. Der Verschlüsselungsalgorithmus wird nicht vom Standard vorgegeben und kann im *EncryptionMethod* Element angegeben werden. Meistens werden hybride Verschlüsselungsmethoden benutzt. Dabei wird der eigentliche Text symmetrisch verschlüsselt und zusammen mit dem asymmetrisch verschlüsselten Schlüssel übertragen. Die wichtigsten Elemente von XML-Encryption sind folgende:

EncryptedMethod ist ein optionales Element, das Informationen über den verwendeten Verschlüsselungsalgorithmus enthält. Ist es nicht angegeben, so muss der Algorithmus dem Empfänger auf einem anderen Weg gemacht werden.

CipherData ist ein obligatorisches Element, das den verschlüsselten Text in Base64-kodierter Form oder eine Referenz auf den Text enthält.

EncryptedData ist das Elternelement von *CipherData*. Es ersetzt in dem Dokument das Element, welches mit XML-Enc verschlüsselt wurde.

KeyInfo enthält den benötigten Schlüssel oder eine Referenz auf diesen, welche mit dem *RetrievalMethod* Element angegeben ist. Liegt der Schlüssel selber verschlüsselt vor, so wird er zusätzlich vom *EncryptedKey* Element umschlossen.

ReferenceList stellt eine Zuweisungsliste dar, die angibt, welche Schlüssel zu welchen verschlüsselten Daten passen.

EncryptionProperties kann zusätzliche Informationen über den verschlüsselten Teil enthalten, wie zum Beispiel einen Zeitstempel.

XML-Signature

XML-Signature ermöglicht das Signieren von Daten jeden Typs. Einzige Voraussetzung ist, dass sie im XML-Dokument der Signatur enthalten sind (Enveloping Signature) oder in dem Dokument referenziert werden (Detached Signature). Die einzelnen Bestandteile sind folgende:

Signature ist das Wurzelelement einer XML Signatur und enthält die folgenden Elemente.

SignatureValue enthält die Base64-kodierte Signatur.

SignedInfo enthält wichtige Informationen zum Bilden der Signatur. Dabei handelt es sich um den verwendeten Kanonisierungsalgorithmus[1], den Signaturalgorithmus und den Hash der signierten Daten.

KeyInfo ist ein optionales Element, das Informationen zum verwendeten Schlüsselmaterial enthält und dem Empfänger somit hilft, die Signatur zu validieren.

Object ist ein optionales Element, welches jede Art von Daten enthalten kann. Es wird meistens für *Enveloping Signatures* benötigt, wenn die zu signierenden Daten im Signaturelement enthalten sein sollen.

Bei der Erstellung und Überprüfung der Signatur ist darauf zu achten, dass die Signatur über das korrekte Objekt ausgeführt wurde. Diese Sicherheitsmaßnahme klingt zwar trivial, ist aber nötig um unter anderem Signature-Wrapping Attacken zu verhindern. Hierbei wird der signierte Teil im XML Dokument so verschoben und dafür eigenes XML eingefügt, dass die Signaturprüfung noch das original signierte XML, die Programmlogik jedoch den eingefügten Teil auswertet[17].

[1] Beim Kanonsieren wird ein XML-Dokument in eine Standardform überführt, wobei der Inhalt nicht verändert wird. Das ist nötig, weil bei der Bildung der Signatur viele Dinge eine Rolle spielen (u.a. Leerzeichen zwischen zwei XML-Elementen), die von einem XML-Parser jedoch ignoriert werden.

2.2 SAML

Die Security Assertion Markup Language (kurz SAML) ist ein XML-Framework mit dessen Hilfe Geschäftspartner online sicherheitsbezogene Informationen über eine Entität, in diesem Fall Authentisierungs- und Autorisierungsinformationen, sicher austauschen können. SAML wurde 2001 vom OASIS-Konsortium verabschiedet und liegt mittlerweile in der Version 2.0 vor. Bei der Entwicklung wurde dem Web Single Sign-On (Web SSO), das vereinfacht in Abbildung 2.1 dargestellt ist, besondere Bedeutung geschenkt.

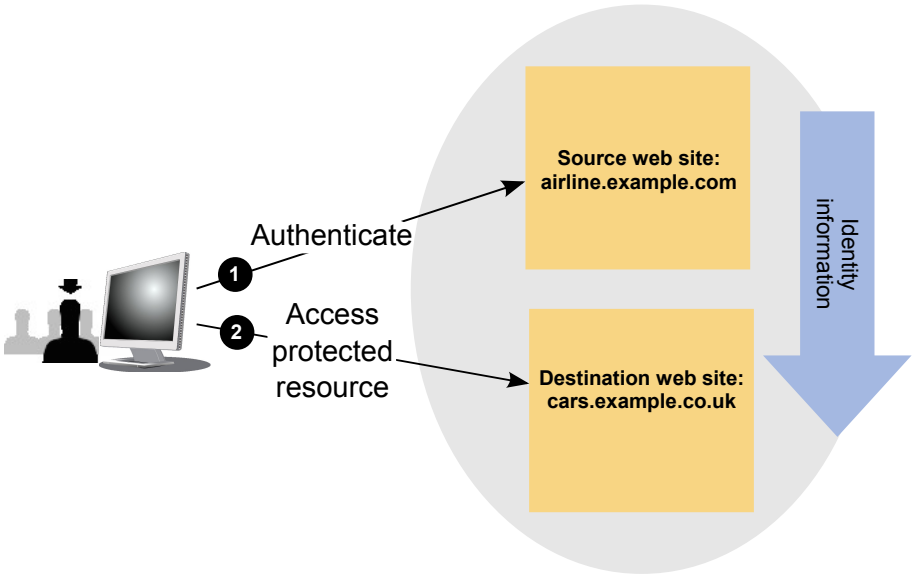

Abbildung 2.1: Web SSO Use Case. Quelle: [12], Abb. 2

Hierbei kann ein Benutzer, der bei einem Onlinedienst angemeldet ist, seine schon vorhandene Session weiter nutzen, wenn er die Seite eines Vertragspartners des Dienstes besucht. Er muss sich also nicht neu authentifizieren, um das Angebot nutzen zu können. Dafür wird bei SAML eine dritte Instanz, eine sogenannte Asserting Party eingeführt, dem die beiden Diensteanbieter, in SAML auch Relying Party genannt, vertrauen. Dieser übernimmt an deren Stelle die Authentifizierung des Benutzers und stellt eine Assertion aus, mit welcher die Relying Partys den Benutzer identifizieren können.

Assertions bestehen aus Aussagen über ein Subjekt, das in diesem Fall der Benutzer ist aber auch eine andere Entität sein kann. Sie können Informationen über den Authentifizierungsvorgang, die Zugriffsrechte und Attribute des Benutzers enthalten, aber auch nur aus einer Fehlermeldung bestehen.

Desweiteren gibt es bei SAML folgende Komponenten:

Protokolle Diese legen fest in welcher Reihenfolge welche Nachrichten übertragen werden. Für die Authentifizierung eines Benutzers wird das Authentication Request Protokoll genutzt. Des Weiteren gibt es unter anderem das Assertion Query and Request Protokoll, mit dem bei einer Asserting Party weitere Informationen über eine schon bestehende Session, zum Beispiel weitere Attribute des Benutzers, abgefragt werden können.

Bindings Diese legen fest wie SAML (request oder response) Nachrichten über ein Transportprotokoll übertragen werden. Gab es in SAML 1.0 nur das SOAP-Binding, so werden im 2.0 Standard zusätzlich unter anderem HTTP-Post, HTTP-Redirect oder PAOS unterstützt.

Profile Profile haben in SAML zwei Bedeutungen. Zum einen sagen sie aus wie Assertions, Protokolle und Bindings für einen bestimmten Anwendungsfall kombiniert werden sollen. Des Weiteren legen sogenannte „Attribute Profile“ fest, wie SAML Attribute in andere Systeme eingebunden werden. Für das Web SSO Szenario existieren unter anderem das Web Browser SSO und das Enhanced Client or Proxy (ECP) Profil, wobei beim Zweiten ein zusätzliches Programm an Stelle des Browsers für die Authentifizierung genutzt wird.

2.2.1 SAML Identity Federation

Um den Begriff Identity Federation erklären zu können, werden zunächst die einzelnen Bestandteile des Begriffes erklärt.
In SAML besteht eine *Identität* aus den Attributen, die benötigt werden, um eine Entität (zum Beispiel einen Benutzer) eindeutig zu identifizieren. Der Begriff Federation hat in SAML zwei Bedeutungen (vgl. [11]):

(a) Der Vorgang um zwei Entitäten miteinander zu verknüpfen

(b) Eine Verbindung zwischen verschiedenen Service Providern und Identity Providern.

Bei einem *Service Provider* (SP) handelt es sich um eine Entität, welche einem Benutzer bestimmte Dienste anbietet.
Eine spezielle Art Service Provider stellt der *Identity Provider* (IdP) dar. Dieser stellt bestimmten Service Providern innerhalb einer Federation Identitätsinformationen eines Benutzers zu Verfügung, wofür er den Benutzer meistens authentifiziert. Wichtig ist hierbei das Vertrauen, das die SP und der IdP zueinander haben müssen. Zum einen müssen die SP auf die Korrektheit der vom IdP erhaltenen Daten vertrauen. SAML kann hierbei mit XML-Signature nur sicherstellen, dass die Daten bei der Übertragung nicht verändert wurden, das Vertrauen in den Authentifizierungsvorgang des IdP muss jedoch vertraglich hergestellt werden. Zum anderen besteht meistens auch ein Vertrauen des IdP in die SP, dass diese die erhaltenen Daten nicht an Dritte weitergeben. Somit kann es geschehen, dass aus vertraglichen Gründen ein

SP nicht als IdP für weitere SP dienen darf, obwohl dies technisch mit SAML möglich wäre.
Alles in allem ist Identity Federation als Vorgang, bei dem eine *Federated Identity*, eine zwischen verschiedenen Providern verknüpfte Identität, entsteht, ein eher schwer zu fassender Begriff, der unter technischen und rechtlichen Aspekten betrachtet werden sollte.

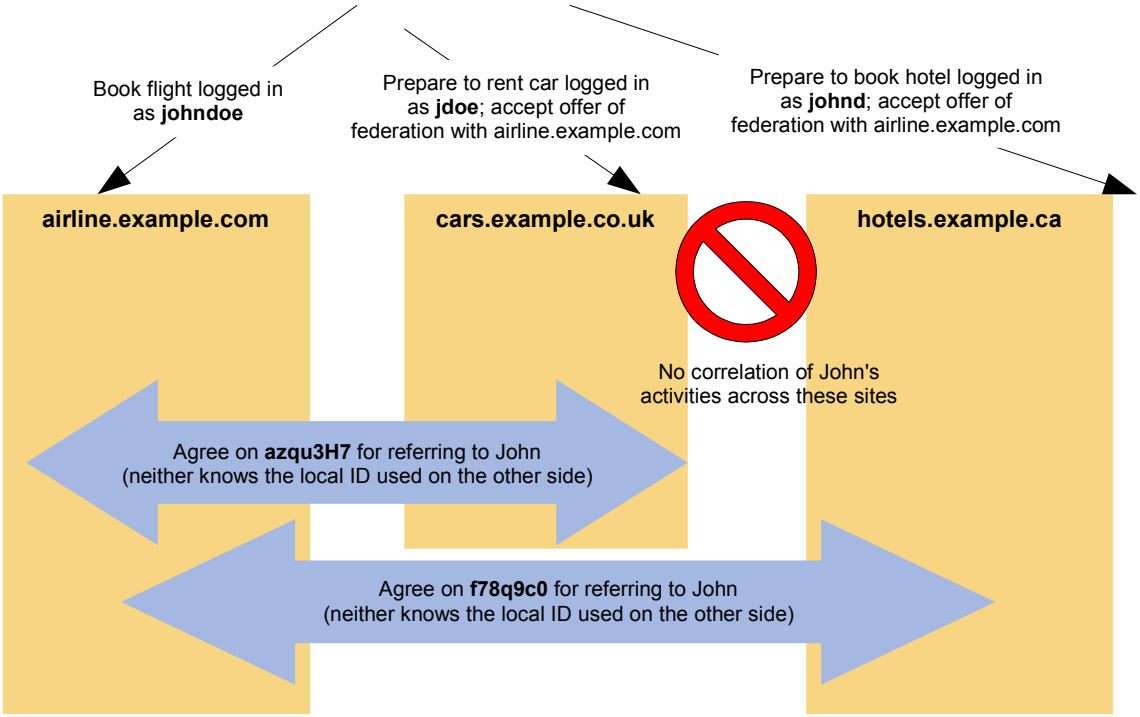

Abbildung 2.2: SAML Identity Federation. Quelle: [12], Abb. 3

Federated Identities

Bei der in Grafik 2.2 vereinfacht dargestellten Bildung einer Federated Identity wird die Nutzung einer Identity Federation deutlich. Nachdem sich ein Benutzer namens John Doe bei airline.example.com einen Flug gebucht hat, möchte er sich noch einen Mietwagen besorgen. Hierfür besucht er über einen Link auf der Seite cars.example.co.uk, wobei zum Beispiel als GET-Variable die Information übertragen wird, dass er ein Kunde von airline.example.com ist. Direkt nach der Anmeldung auf der neuen Seite wird ihm nun die Möglichkeit geboten, seine beiden Accounts zu verlinken. Stimmt er dem zu, einigen sich beide Diensteanbieter auf ein gemeinsames Pseudonym **azqu3H7** für John Doe und ermöglichen ihm somit, fortan seine

bestehende Session bei einem der beiden Webseiten für die jeweils andere weiter nutzen zu können. Das Verfahren ist natürlich ebenfalls für airline.example.com und hotels.example.ca möglich, wobei eine Transitivität allerdings ausdrücklich ausgeschlossen ist. Hotel und Autovermietung können die Aktivitäten des Benutzers trotzdem nicht miteinander verbinden, auch wenn beide eine Federation mit der Fluggesellschaft eingegangen sind.

Für das beschriebene Szenario bietet SAML verschiedene Möglichkeiten, wie die Verlinkung der Accounts hergestellt werden kann. Insgesamt gibt es vier unterschiedliche Arten:

Out-of-Band Account Linking Hierbei wird die Federation über einen außerhalb von SAML liegenden Kanal hergestellt, zum Beispiel über sich synchronisierende Datenbanken oder eigens dafür entwickelte Programme. In SAML V1 war es die einzige Möglichkeit der Verlinkung.

Identity Attributes Sofern sich die Benutzer-Datenbestände der beiden SP bei identifizierenden Attributtypen überschneiden, können diese Attribute für die Verlinkung genutzt werden.

Persistent Pseudonym Identifiers Gibt es keine (hinreichenden) Überschneidungen der Datenbestände oder sollen nicht bei jeder Abfrage Daten des Benutzers mitgeschickt werden, können mit SAML auch Pseudonyme erstellt werden. Dafür leitet ein SP den Benutzer mit einem *AuthnRequest*, in dem zusätzlich ein persistentes Pseudonym angefordert wird, an den entsprechenden SP weiter. Nachdem der Benutzer sich nun dort und auf der Ausgangsseite authentisiert hat, kann diese das Pseudonym dem entsprechenden Datensatz zuordnen. Da persistente Pseudonyme auch nach der Nutzungs-Session des Benutzers bestehen bleiben, müssen sie nur einmal erstellt werden und können ab dann ohne Einschränkungen bei allen folgenden Sessions als Identifier für den Benutzer zwischen den beiden Service Providern genutzt werden.

Transient Pseudonym Identifiers Sofern die Verlinkung nicht dauerhaft sondern nur für die aktuelle Session existieren soll, wird ein flüchtiges Pseudonym benutzt, welches wie ein persistentes Pseudonym erstellt wird. Einsatz findet es zum Beispiel im deutschen eID-System, wobei der eID-Server in der Response als *NameID* für den Benutzer einen zufällig generierten flüchtigen Identifier mitsendet, der allerdings im weiteren Verlauf keine weitere Bedeutung hat.

Generell ist bei der Erstellung einer Federated Identity wichtig, dass sich der Benutzer darüber im Klaren ist, dass seine Accounts verlinkt werden und er den Vorgang gegebenenfalls unterbinden kann. Auch sollte er die Verlinkung hinterher wieder aufheben können, was in SAML mit Hilfe des *Name Identifier Management Protocol* geschieht. Nichtsdestotrotz gilt auch hier wieder, dass SAML dafür zwar Empfehlungen gibt, das allerdings vertraglicher und nicht technischer Natur ist. So wäre es denkbar, dass ohne Wissen des Benutzers ein angeblich flüchtiges Pseudonym dazu verwendet wird, ihn über mehrere Sessions hinweg zuordnen zu können.

2.2.2 Ausgewählte SAML-Techniken

Der SAML-Standard besteht aus zahlreichen einzelnen Dokumenten, welche sich gegenseitig ergänzen. Hinzu kommt eine Vielzahl an proprietären Erweiterungen verschiedener Identitätssysteme, die einzelne Unzulänglichkeiten des Standards bereinigen zu versuchen. Da eine Betrachtung aller Aspekte von SAML zu umfangreich wäre, werden nachfolgend nur die für die Arbeit relevanten berücksichtigt.

POST- und Redirect-Binding

Wie bereits erwähnt, legen SAML-Bindings fest, wie die Nachrichten über ein Transportprotokoll übertragen werden. Da sich HTTP als ein wichtiges Protokoll im Internet etabliert hat, wird es auch in SAML unterstützt. So existieren dafür unter anderem das Redirect- und das POST-Binding.
Im Ersten werden die SAML-Nachrichten an die aufgerufene URL als GET-Variable angehangen und der Benutzer mit einem HTTP-Redirect auf die entsprechenden Seiten geleitet. So leitet der SP ihn zusammen mit einem AuthnRequest auf die Seite des IdP, welcher nach erfolgter Authentifizierung ebenfalls eine Weiterleitung mit der SAMLResponse auf die Seite des SP durchführt. Damit das XML als GET-Variable übertragen werden kann, wird es meistens zuerst gezipt und danach immer Base64- und URL-encodiert. Da die Adresslänge bei Webbrowsern typischerweise beschränkt ist, eignet sich das Binding nicht zum Übertragen komplexer SAML-Nachrichten, die womöglich noch signiert sind und Zertifikate beinhaltet. Außerdem werden URLs meistens in Browsern abgespeichert, sodass die übertragenen Nachrichten auch später noch über die Browserhistorie abgerufen werden können.
Diese Schwächen beseitigt das POST-Binding, bei dem die Nachrichten als POST-Variable im HTTP-Body übertragen werden. Typischerweise wird dabei die SAML-Nachricht ebenfalls Base64-kodiert und in einem speziell benannten Hidden-Feld eines Formulars hinterlegt, welches automatisch per Javascript versendet wird. Allerdings ist Javascript manchmal in Browsern deaktiviert, sodass der Benutzer jede einzelne Umleitung manuell durchführen muss.

Proxying

Der Proxy Mechanismus wird in SAML Core [13] beschrieben. Ist ein Identity Provider nicht in der Lage einen Nutzer zu authentifizieren, kann er unter bestimmten Umständen selber einen eigenen AuthnRequest an einen weiteren IdP senden. Dessen Antwort kann er wiederum dazu verwenden, eine eigene Response zu generieren. Durch setzen bestimmter Optionen kann der Aussteller eines AuthnRequest Einfluss auf den Proxy Mechanismus nehmen. So bestimmt der Wert in *ProxyCount* die maximale Proxytiefe und das *IDPList* Element enthält zusätzlich eine Liste an bevorzugten IdPs. Des Weiteren existieren noch einige Vorschriften, die der Proxy

bei der Erstellung seiner Antwort beachten muss. So soll unter anderem das *Subject* Element den Anforderungen der originalen Anfrage genügen, auch muss dem Anfragenden mitgeteilt werden, welcher Server im Endeffekt den Benutzer authentifiziert hat. Wichtig dabei ist, dass es sich hierbei um Verabeitungsvorschriften handelt, bei denen die beteiligten Instanzen darauf vertrauen müssen, dass sich die betroffenen Identity Provider daran halten. Auch wird die Abarbeitung nicht genau im Standard festgeschrieben, weswegen diese vorher vertraglich festgelegt werden muss.

2.3 Attribute Provider

Der Begriff Attribute Provider wird im SAML-Standard nicht erwähnt. Nach dem Glossar des verwandten Liberty Alliance Projekts stellt ein AP sogenannte „Identity Personal Profile (ID-PP) information“ zur Verfügung [15], also persönliche Identitätsinformationen einer Entität. Auch wenn ein eID-Server ebenfalls in dieses Schema fällt, soll dieser trotzdem in der Arbeit extra behandelt werden. Grund dafür ist die besondere Rolle, die er gegenüber anderen AP einnimmt, da die von ihm zur Verfügung gestellten Daten immer aus dem nPA ausgelesen wurden.

2.4 eID-Funktionalität des nPA

Ab dem 1. November 2010 wurde im Zuge der bereits im März 2005 beschlossenen eCard-Strategie des Bundes [16] der neue (elektronische) Personalausweis eingeführt. Dieser bietet neben den hoheitlichen Funktionen und der qualifizierten elektronischen Signatur (QES) auch eine eID-Funktion, mit der sich der Ausweisinhaber im Internet authentisieren kann. Möchten Diensteanbieter die Funktion nutzen, müssen sie zuvor ein Berechtigungszertifikat bei der Vergabestelle für Berechtigungszertifikate (VfB) beantragen, in dem festgelegt wird, auf welche Datenfelder des Ausweises sie zugreifen dürfen. Dabei ist die Vergabe dieser Zertifikate an bestimmte Bedingungen geknüpft, welche in § 21 Abs. 2 des Personalausweisgesetzes [1] festgelegt sind und in einer Leitlinie zur Vergabe von Berechtigungszertifikaten von der VfB [3] erläutert werden. So müssen unter anderem die Daten für den Geschäftsvorgang notwendig sein (Erforderlichkeitsgrundsatz) und ein kreditorisches Risiko für den Diensteanbieter bestehen. Des Weiteren dürfen sie nicht geschäftsmäßig an Dritte weitergegeben werden, um einen Vertrauensverlust von Ausweisinhabern in den elektronischen Identitätsnachweis zu vermeiden. Interessanterweise ist allerdings nach [3] die nicht geschäftsmäßige Übermittlung der Daten zugelassen, wofür unter anderem sogar bei dem koordinierten Anwendungstests ein Anwendungsbeispiel namens „ID Safe“ betrachtet wurde. Voraussetzung hierfür ist, dass der Ausweisinhaber nach § 13 Telemediengesetz aktiv jeder Weitergabe seiner Daten zustimmen muss. Auch gilt wieder das Gebot der Datensparsamkeit, nach dem selbst mit Zustimmung des

Benutzers und ausreichender Berechtigungen nicht mehr Daten übertragen werden sollen, als unbedingt notwendig sind.

Ein Beispiel für den Erhalt eines Berechtigungszertifikates nach diesen Vorgaben stellt die VZnet Netzwerke Ltd. (bekannt unter anderem durch StudiVZ) dar. Diese hat schon in der ersten Vergaberunde eine Berechtigung zum Auslesen von Vor- und Nachnamen, RID[2] und der Angabe, ob ein bestimmtes Alter über- oder unterschritten wird, bekommen, damit sich ihre Mitglieder gegenüber anderen Nutzern des VZ-Netzwerks authentisieren können [4]. Leider existiert zum aktuellen Zeitpunkt noch keine frei zugängliche Implementation, weswegen offen ist, wie oft und in welcher Weise Personen, die das Angebot nutzen, der Freigabe zustimmen müssen.

2.4.1 eID-Server

Damit sich ein Benutzer mit dem nPA im Internet authentisieren kann, wird ein eID-Server benötigt, der sich entweder beim SP direkt oder bei einem externen Anbieter befinden kann. Wichtig ist jedoch, dass ihm Nutzer und Service Provider gleichermaßen vertrauen. Sein Aufbau wird unter anderem in der TR-03130 [6] beschrieben [3]. Da eine vollständige Betrachtung des eID-Servers den Kontext der Arbeit bei weitem sprengen würde, wird der Fokus auf die SAML-Schnittstelle gelegt. Um diese korrekt einordnen zu können, soll trotzdem der komplexe Authentifizierungsvorgang (vereinfacht dargestellt in Abbildung 2.3) vorher kurz angerissen werden.

(a) Eine Person möchte einen bestimmten Dienst nutzen.

(b) Der Betreiber des Dienstes verlangt eine Authentifizierung und leitet den Benutzer mit Hilfe des SAML-POST Bindings auf eine SSL-gesicherte Seite eines eID-Servers weiter.

(c) Die Seite enthält ein Object-Tag zum Ansteuern eines Browserplugins.

(d) Das Browserplugin startet wiederum eine eCard-API-konforme Anwendung (AusweisApp) und übergibt ihr die Parameter des Object-Tags (unter Anderem ein Preshared Key (PSK), der im nächsten Schritt benötigt wird).

(e) Die Anwendung baut eine weitere TLS-gesicherte Verbindung zum eID-Server auf, die mit Hilfe des PSK kryptografisch mit der ersten verschränkt ist.

(f) Hierüber findet nun die eigentliche Authentifizierung und ein Großteil der Kommunikation statt. Dabei werden nacheinander eine Reihe von Protokollen abgearbeitet, bei denen der Benutzer mit seiner PIN den Auslesevorgang bestätigen und optionale Attribute abwählen kann.

[2] Siehe Kapitel 2.4.2

[3] Wobei noch auf weitere technische Richtlinien zu verweisen sei, wie die TR-03110 [5] und TR-03112 [2], welche sich mit den Interna der Kommunikation zwischen nPA, AusweisApp und eID-Server auseinandersetzen.

(g) An deren Ende liest der eID-Server die entsprechenden Datengruppen aus dem Personalausweis aus und erstellt eine signierte SAML-Response, die an die AusweisApp weitergereicht wird.

(h) Die Response wird an das Browserplugin weitergegeben, welches sie per HTTP-Post an den SP schickt.

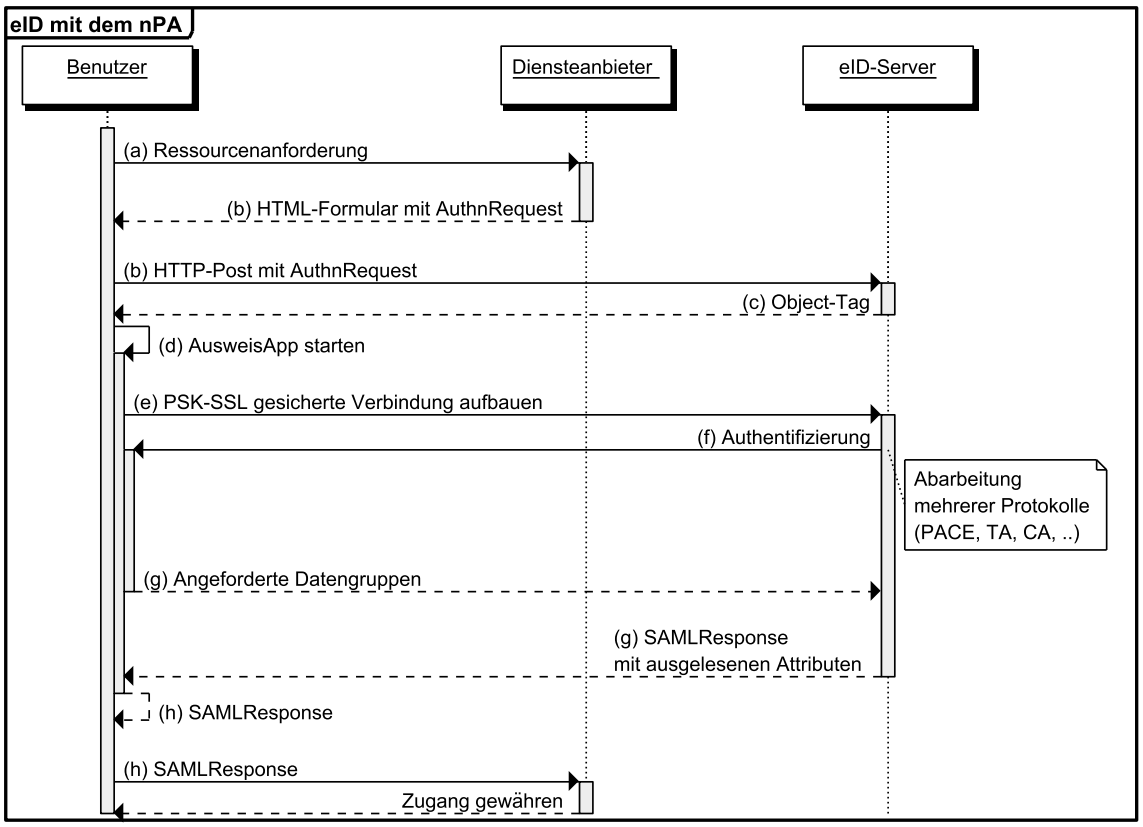

Abbildung 2.3: Authentifizierung mit dem neuen Personalausweis

Auffallend daran ist, dass SAML nur in sehr einfacher Form zum Übertragen der Authentifizierungsanfrage und der Attribute verwendet wird, nicht jedoch während der Authentifizierung selber. Grund dafür ist nach der TR-03130 [6], dass die Authentisierungsprotokolle des nPA grundsätzlich von genau zwei miteinander kommunizierenden Entitäten ausgehen, auf der einen Seite der Benutzer, der sich mit seinem nPA ausweist und auf der anderen Seite der Diensteanbieter, welcher mit seinem Berechtigungszertifikat Attribute auslesen möchte. Des weiteren wird die Sicherheit durch die verwendeten Protokolle (PACE, TA, CA,..), nicht jedoch durch das darunter liegende Transportprotokoll hergestellt. Dem gegenüber steht SAML mit seinem 3-Entitäten-Modell (Nutzer, Relying Party, Asserting Party), wobei zwischen Asserting Party und Relying Party eine Vertrauensbeziehung besteht. Unter anderem um die beiden Systeme doch kompatibel zueinander zu bekommen, wurde ein eID-Server als vertrauenswürdige dritte Instanz eingeführt. Dieser übernimmt in

SAML trivialerweise die Rolle der Asserting Party und authentifiziert anstelle des Diensteanbieters den Benutzer[4]. Anders als in SAML eigentlich vorgesehen, werden die Attribute jedoch mit Hilfe des Berechtigungszertifikates des SP aus dem Personalausweis ausgelesen und liegen nicht schon vorher auf dem Server, auch dürfen sie weder auf diesem gespeichert noch an Dritte weitergegeben werden. Allerdings werden nach der TR-03130 Erweiterungen des eID-Servers nicht ausgeschlossen. So ist denkbar, dass weitere Authentifizierungsmethoden oder andere Serviceleistungen hinzugefügt werden, auch wurden schon „Implementierungen im internationalen Zusammenhang“ bedacht.

SAML-Schnittstelle

Als Bestandteile von SAML werden wie bereits erwähnt nur die Elemente *AuthnRequest* (mit Erweiterung) und *Response* in Verbindung mit dem Web Browser SSO Profile verwendet. Als Binding wird für Anfrage und Antwort das HTTP Post Binding benutzt. Auch wenn der SAML-Teil relativ einfach gehalten ist, gibt es einige Besonderheiten, auf die im folgenden Bezug genommen wird.

AuthnRequest Erweiterung Der SAML 2.0 Standard bietet normalerweise mit dem *AuthnRequest* Element keine Möglichkeit, bei einer Authentifizierungsanfrage eine Liste an benötigten Attributen mit anzugeben. Zwar existiert dafür das Element *AttributeQuery*, jedoch wird dafür eine schon bestehende Authentifizierung vorausgesetzt. Dies bedeutet, dass eine Abfrage in zwei verschiedene (Authentifizierung + Attributabfrage) aufgespalten werden muss, wodurch nicht nur überflüssiger Netzwerkverkehr sondern auch ein komplizierteres Sessionmanagement entsteht. Eine einfachere, wenn auch nicht elegantere, Variante lässt sich mit der in SAMLCore [13] beschriebenen Erweiterungsmöglichkeit realisieren. Dabei enthält der AuthnRequest ein zusätzliches *Extensions* Element, in dem beliebiges eigenes XML untergebracht werden kann. Von dieser Möglichkeit wurde beim eID-Server Gebrauch gemacht. Hier enthält die Extension ein mit Hilfe von XML-Encryption verschlüsseltes *RequestAttributes* Element, welches wiederum folgende Attribute und Elemente beinhaltet:

Version (Attribut) gibt die Schnittstellenversion des Diensteanbieters an und wird hier nicht weiter betrachtet.

RequestAllDefaults (Attribut) Ist das Attribut „true“ so werden die im Berechtigungszertifikat angegebenen Datengruppen ausgelesen. Andernfalls lassen sich die auszulesenden Attribute mit dem nachfolgend beschriebenen *RequestAttributes* Element steuern.

[4]Formal gesehen authentisiert sich der Bürger eigentlich nicht beim eID-Server, sondern immernoch beim SP, da die Berechtigung zum Auslesen der Attribute dem SP gehört.

PreSharedKey (Attribut) Der Pre-Shared Key wird zum Aufbau der bereits erwähnten SSL-Verbindung zwischen AusweisApp und eID-Server benötigt, um diese mit der bereits bestehenden verschlüsselten Verbindung zwischen Browser und eID-Server zu verschränken.

RequestAttributes (Element) Sofern RequestAllDefaults auf „false“ gesetzt ist, kann innerhalb des *RequestAttributes* Element eine Liste von *Attribute* Elementen auftauchen. Solch ein Element beinhaltet wiederum den Namen des angeforderten Attributes und die Angabe, ob dieses optional sind. Des Weiteren bietet es die Möglichkeit, bei bestimmten Attributen einen Vergleichswert mit zu übertragen[5].

Auch wenn Erweiterungen Bestandteil von SAML sind, stellt das Resultat eine Insellösung dar, welche unter anderem ein eigenes XML-Schema[6] mit sich bringt. Nichtsdestotrotz wurde auch bei der SuisseID eine ähnliche Lösung mit Hilfe des *Extensions* Elements entwickelt, die in Abschnitt 2.5 behandelt wird.

Attribute Die aus dem Personalausweis ausgelesenen Datengruppen werden im *Assertion* Element einer SAML-Response als *Attribute* Elemente übertragen. Allerdings gibt SAML weder den Typ noch die semantische Bedeutung dieser Elemente vor, sodass die Abbildungen der Datengruppen auf die *Attribute* in der TR-03130 [6] und in dem beiliegenden XML-Schema beschrieben werden müssen. Bei einigen der Daten wie Name und Titel wird der primitiven Datentypen *xs:string* verwendet. Andere besitzen eigene Datentypen, welche die jeweiligen Besonderheiten der Datengruppen berücksichtigen. So werden bei der *CommunityIdVerification* und *AgeVerification* die Vergleichswerte der Anfrage mitgeschickt. Weitere *Attribute* Elemente mit komplexen Datentypen sind *DateOfBirth*, *PlaceOfBirth*, *PlaceOfResidence* und *DocumentValidity*.

Signatur / Verschlüsselung SAML empfiehlt zur Sicherung und Vertraulichkeit der Kommunikation eine bestehende PKI zu verwenden. Außerdem existiert die Möglichkeit, die ausgestellten Assertions ganz oder teilweise mit Hilfe von XML-Encryption zu verschlüsseln und mit XML-Signature zu signieren. Das deutsche eID-System macht von dem Angebot regen Gebrauch, wobei als PKI die bereits etablierte X.509-PKI des Internets verwendet wird. Grundsätzlich müssen alle SAML-Nachrichten über einen TLS-gesicherten Kanal verschickt werden, wodurch die Kommunikation zwischen Browser und Diensteanbieter bzw. eID-Service verschlüsselt

[5] Genauer gesagt gibt es zwei Attribute bei denen das der Fall ist:
CommunityIdVerification Wohnt der Ausweisinhaber in dem angegebenen Gebiet?
AgeVerification Hat der Ausweisinhaber das angegebene Mindestalter erreicht?

[6] Zu finden unter https://www.bsi.bund.de/ContentBSI/Publikationen/TechnischeRichtlinien/tr03130/tr-03130.html

stattfindet. Trotzdem kann unter anderem ein Browserplugin AuthnRequest und Response mitlesen, sodass zusätzlich im AuthnRequest das *Extensions* Element (mit den angeforderten Attributen) für den eID-Server und in der Authentisierungsantwort das *Assertion* Element für den Diensteanbieter mit XML-Encryption verschlüsselt werden. Zwar kann sich der Nutzer dadurch sicher sein, dass seine Daten auf dem Weg vom eID-Server zum Diensteanbieter von keinem Dritten abgegriffen werden können, jedoch kann er sie selber auch erst frühstens beim Diensteanbieter einsehen, sofern dieser das zulässt. Des Weiteren ist ein Browserplugin zumindest dazu in der Lage, herauszufinden, wann der Benutzer sich für welchen Dienst mit seinem nPA authentisieren will.
Das Signieren der Nachrichten mit XML-Signature ist relativ betrachtet einfacher. Um XML-Signature-Wrapping-Attacken zu verhindern, werden immer die kompletten Nachrichten, also AuthnRequest und Response, vom Diensteanbieter bzw. vom eID-Server signiert.

2.4.2 Restricted Identification

Bei manchen Szenarien wird die eID-Funktionalität nur dazu benötigt, eine Person einer früheren Authentifizierung zuordnen zu können. Dies ist unter anderem dann der Fall, wenn der nPA als alternative Loginmethode für bestimmte Dienste (z.B. ein soziales Netzwerk) benötigt wird. Auch besteht die Möglichkeit, dass ein Diensteanbieter nur ein eindeutiges Pseudonym für eine Person benötigt. So wäre es denkbar, dass bei einer Onlineabstimmung nur sichergestellt werden muss, dass eine Person nicht mehrmals abstimmen kann, jedoch alle weiteren Angaben über diese Person überflüssig sind. Aus diesem Grund wurde eine sektor- und kartenspezifische Kennung als Pseudonym eingeführt. Innerhalb eines Sektors ist diese Kennung, auch restricted Identifier (rID) genannt, für jeden Ausweis einzigartig. Des Weiteren werden für einen Ausweis A und zwei verschiedene Sektoren S_1, S_2 auch zwei verschiedene rIDs rID_1, rID_2 berechnet, welche unlinkbar sind. Das heißt, dass es für einen Diensteanbieter aus dem Sektor S_1 unmöglich ist, rID_2 mit Hilfe von rID_1 zu berechnen. Sogar wenn er an rID_2 gelangen würde, sollte er keinen Zusammenhang zu rID_1 herstellen können. Idealerweise befindet sich jeder DA in einem eigenen Sektor, wodurch die Unlinkbarkeit über alle Diensteanbieter hinweg gegeben ist.

2.4.3 Problem Attributaggregation

Damit Attribute aus unterschiedlichen Quellen zusammen geführt werden können, müssen verschiedene Anforderungen erfüllt sein, die in Kapitel 3 näher erläutert werden. Möchte man die ausgelesenen Daten des nPA mit Attributen aus weiteren Attribute-Providern kombinieren, stellt sich jedoch die Zuordnung des Benutzers zwischen den einzelnen Datenquellen als am kompliziertesten heraus. Sie kann auf verschiedene Arten geschehen, zum Beispiel über einen eindeutigen Identifier, wie

bei der SuisseID (Kapitel 2.5). Der einzige Identifier beim deutschen eID-System, die rID, ist hierfür jedoch nicht geeignet, da sie für jeden Sektor unterschiedlich ist, damit Benutzerkonten gerade nicht verlinkt werden können. Einzig in dem Fall, dass sich zwei Diensteanbieter im selben Sektor befinden, ist der Einsatz der rID denkbar. Da das jedoch, wenn überhaupt, eher die Ausnahme als die Regel sein dürfte und einer der Diensteanbieter zum Auslesen der rID auch nicht berechtigt sein kann, ist diese Möglichkeit nicht von praktischer Relevanz. Des Weiteren gäbe es auch die Möglichkeit, eine Zuordnung über eindeutig identifizierende Attribute zu treffen. Dafür müssen sich die Datensätze beider Attribute Provider allerdings soweit überschneiden, dass eine eindeutige Zuordnung des Nutzers möglich ist. Außerdem widerspricht der Ansatz dem Gebot der Datensparsamkeit, da unter Umständen mehr Nutzerinformationen übertragen werden, als für den Anwendungsfall erforderlich sind. Ein Ausweg bietet SAML mit Federated Identities (siehe Kapitel 2.2.1). So wäre es möglich, *Persistent Pseudonym Identifiers* bzw. *Transient Pseudonym Identifiers* zu verwenden, um die Verlinkung herzustellen.

Erschaffung einer neuen Identität

Allerdings wird sogar im SAML-Standard folgendem Problem keine Beachtung geschenkt: Um eine Federated Identity bilden zu können, muss sich der Benutzer, wie in Abschnitt 2.2.1 beschrieben, auf den Seiten beider Provider anmelden. Dabei wird leider nicht überprüft, ob es sich bei beiden Anmeldevorgängen um die selbe Person handelt. Tatsächlich ist solch eine Überprüfung ohne eine dritte Instanz nur sehr schwer realisierbar, da die Verlinkung der beiden Accounts ja gerade noch nicht besteht und somit auch kein eindeutiger Identifier existieren muss, über den man beide Accounts abgleichen könnte.
Dass das Problem durchaus ernst zu nehmen ist, soll an einem Beispiel, das in Abbildung 2.4 dargestellt wird, demonstriert werden. Angenommen ein Onlineshop hat ein spezielles Angebot für Studenten, über das sie kostengünstiger an bestimmte Artikel gelangen. Voraussetzung dafür ist allerdings, dass sich der Student sowohl online mit dem neuen Personalausweis ausweist, als auch eine (fiktive) digitale Immatrikulationsbescheinigung seiner Universität vorlegt. Um an die Bescheinigung zu gelangen, bieten die Universitäten einen Service an, der nach erfolgreicher Authentifizierung einen AuthnRequest ausstellt, dessen Assertion nur ein einziges Attribut mit der Angabe des Studentenstatus besitzt. Ohne weitere Vorkehrungen könnte nun jemand, der selber zwar kein Student ist, jedoch einen kennt, sich zuerst mit seinem eigenen nPA online ausweisen und dann den Bekannten bitten, sich mit seinen Daten bei der Universität zu authentisieren, um an den Studentenstatus zu gelangen. Für den Onlineshop ist diese Person nun ebenfalls Student, sodass sie unberechtigter Weise an die Vergünstigungen gelangen kann. Das eigentlich Prekäre daran ist, dass hierbei nicht nur die Identität einer anderen Person genommen wurde, um sich Vorteile zu verschaffen, sondern Attribute mehrerer Identität kombiniert werden, um quasi eine neue real nicht existierende Identität zu schaffen. Deswegen muss dieses

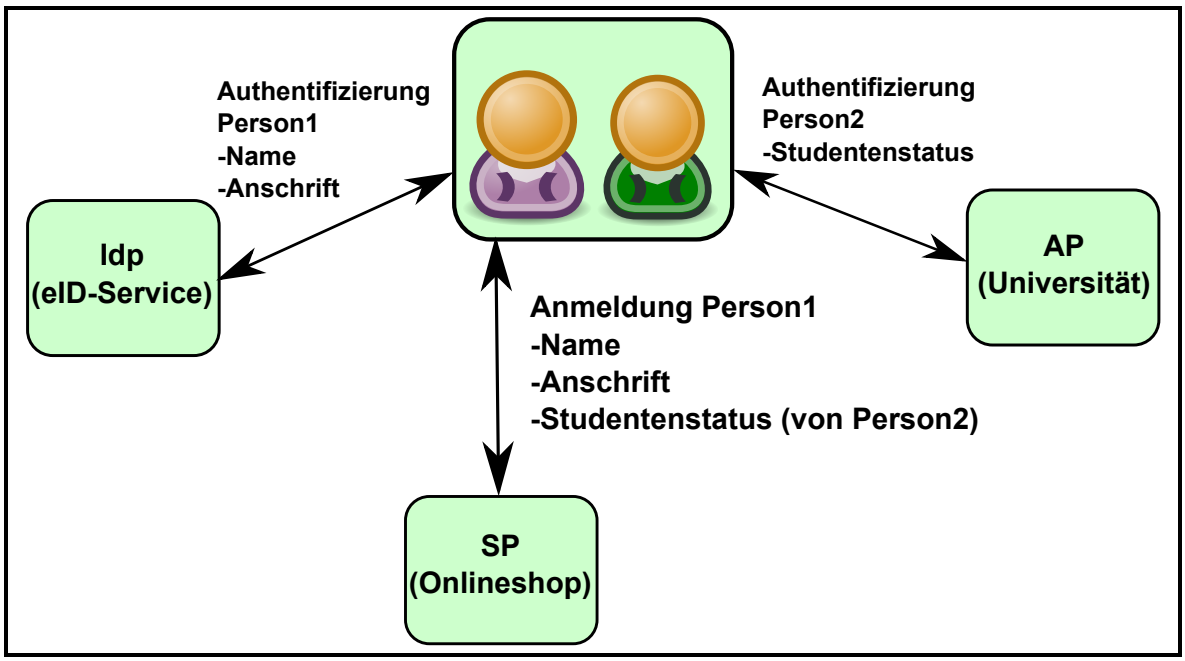

Abbildung 2.4: Missbrauch der Attributaggregation mit zwei zusammen arbeitenden Personen

Angriffszenario unbedingt mitberücksichtigt werden. Im vorliegenden Beispiel wäre eine Lösung, dass die Universität im AuthnRequest nicht nur den Status, sondern auch weitere identifizierende Angaben über die Person mitsendet, wodurch die Anonymität bei der Nutzung des Services wiederum verloren gehen würde. Ebenfalls denkbar wäre es, dass der Onlineshop die Attribute des Benutzers an die Universität schicken würde, da diese sowieso schon über Angaben wie Name und Anschrift der Studenten verfügt. Das Versenden der Attribute könnte dabei in einer für die Universität verschlüsselten Extension eines AuthnRequests geschehen. Sofern sich nun der Benutzer bei der Uni authentifiziert, zu dem die mitgeschickten Attribute passen, ist sicher gestellt, dass es sich um die selbe Person handelt.
Auch wenn in dem konkreten Fall ein Ansatz für das Problem existiert, bleibt die Frage nach einer allgemeinen Lösung. Leider kann diese zumindest in der vorliegenden Arbeit nicht zufriedenstellend beantwortet werden.

2.5 SuisseID

Mit der SuisseID[18] wurde bereits im Mai 2010 in der Schweiz ein standardisierter elektronischer Identitätsnachweis eingeführt, welcher das Abfragen von Attributen aus unterschiedlichen Quellen erlaubt. Die Attribute Provider, bei SuisseID CAS (Claim Assertion Service) genannt, müssen dabei eine vorgegebene SAML und WS-Trust Schnittstelle implementieren. Ausgangspunkt ist, dass sich ein Benutzer zur

Nutzung eines bestimmten Dienstes authentifizieren muss. Dafür muss er zuerst auswählen, bei welchem SuisseID Anbieter er Kunde ist, woraufhin die RP einen AuthnRequest, welcher als Erweiterung die benötigten Attribute enthält, für den IdP dieses Anbieters generiert. Durch die Seite des IdP wird nun eine Anwendung beim Benutzer gestartet, über die er sich, nachdem er die korrekte PIN eingegeben hat, mit Hilfe eines privaten Schlüssels auf einer SmartCard oder einem Security-Token authentifiziert. Daraufhin bekommt er eine Auflistung der ausgelesenen Attributwerte angezeigt. Stimmt der Benutzer dem Vorgang zu, werden diese an die RP weiter übertragen. Sofern sie nun weitere Attribute benötigt, kann die RP einen AttributeQuery, welcher als NameIdentifier des Subjekts die SuisseID des Benutzers enthält, an einen CAS stellen [7]. Auch hier bekommt der Nutzer wieder angezeigt, welche Attributwerte übermittelt werden sollen und kann den Vorgang abbrechen, bevor die Daten an die RP weitergeleitet werden. Einen Überblick über die Infrastruktur und die Vertrauensbeziehungen bietet die Grafik 2.5.

Gegenüber dem deutschen System fallen einige Gemeinsamkeiten aber auch viele Unterschiede auf. Gemein ist trivialerweise die Nutzung von SAML. Auch wird gleich mit der Unzulänglichkeit von SAML umgegangen, in einem AuthnRequest keine Liste an benötigten Attributen angeben zu können. Die dafür entwickelten Erweiterungen (SAML Extensions) sind jedoch, wie für Insellösungen üblich, inkompatibel zueinander. So enthalten zwar beide Arten eine Auflistung der abgefragten Attribute und der Angabe, ob diese optional sind, der Aufbau des XML ist jedoch von Grund auf verschieden. Auch unterscheiden sie sich in der Verschlüsselung, die bei der SuisseID grundsätzlich nur auf der Transportebene mittels SSL/TLS geschieht, im deutschen System jedoch noch zusätzlich mittels XML-Encryption stattfindet.

Auch an anderen Stellen scheint man bei der SuisseID Probleme pragmatischer gelöst zu haben, als im deutschen System. Anstatt zusätzlich CV-Zertifikate einzuführen, um die Berechtigungen von Diensteanbietern festzustellen, verlässt man sich auf die weit verbreitete X.509-PKI. Da Diensteanbieter sowieso dazu verpflichtet sind, ein gültiges X.509-Zertifikat zu besitzen, brauchen sie sich nicht auch noch zusätzlich auszuweisen. Dem Benutzer wird dadurch jedoch zusätzlich die Entscheidung überlassen, ob solch ein Diensteanbieter auch vertrauenswürdig mit den Daten umgeht und diese für den Geschäftszweck unbedingt notwendig sind. Er übernimmt somit selber die Rolle, die im deutschen System den CV-Zertifikaten zufällt, und muss für sich selber bestimmen, ob ein Diensteanbieter berechtigt ist, bestimmte Benutzerdaten zu erhalten. Ein vom IdP/CAS gesteuertes Blacklisting wird in der SuisseID Spezifikation ausdrücklich nicht behandelt (vgl. 3.10.1.3 [18]).

Als weiterer Unterschied lässt sich nennen, dass IdP/CAS dazu verpflichtet sind, dem Benutzer alle abgefragten Attribute inklusive ihrer Werte auf einer Statusseite anzuzeigen bevor sie an den Diensteanbieter übermittelt werden (vgl. 3.10.1.4

[7] Aus der SuisseID Spec übernommene Beispiele für den AuthnRequest und den AttributeQuery befinden sich unter Listing 1 und 2 im Anhang

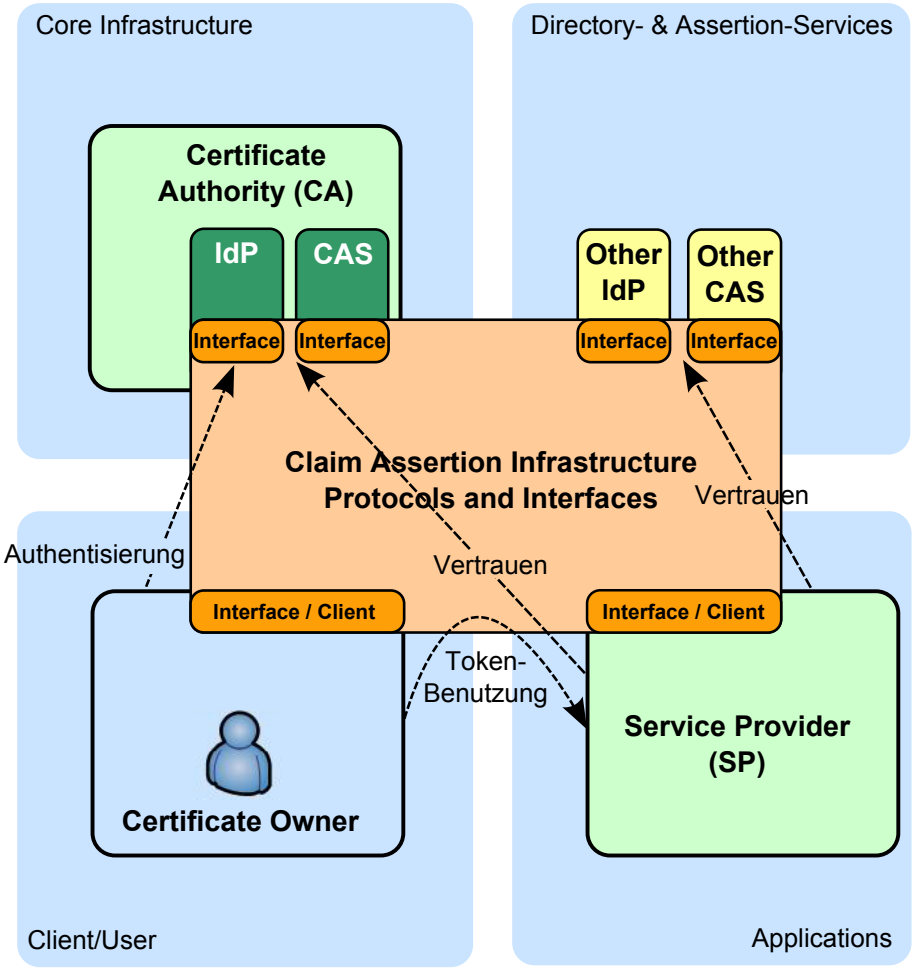

Abbildung 2.5: Suisse ID Infrastruktur (nach [18], 3.2.1 und 3.3.2)

[18]). Dies steht im krassen Unterschied zum deutschen System, in dem nirgends Attributwerte während des Vorgangs zu sehen sind [8].

Allerdings ist es gängige Praxis, dass ein SP meistens kurz vor Abschluss einer Transaktion dem Nutzer alle getätigten Angaben übersichtlich anzeigt. Hierbei dürfte auch ein Großteil der vom nPA ausgelesenen Attribute, wie Name und Anschrift, zu sehen sein, was das Anzeigen der Werte schon beim IdP nicht mehr allzu unattraktiv erscheinen lässt. Immerhin könnte sich hierbei der Nutzer von der Korrektheit der Daten schon vor der Übermittlung zum Diensteanbieter überzeugen, was nicht nur angesichts des holprigen Starts bei der Einführung des nPA [8] sinnvoll sein könnte.

[8] Grund hierfür ist, dass dann eine dritte Person mit Sicht auf den Bildschirm die Werte unberechtigter Weise sehen könnte

2.5.1 Attributaggregation

Im Gegensatz zum deutschen eID-System ist in der SuisseID bereits eine Attributsaggregration integriert. Ein Grund dafür ist, dass es in der Schweiz mit der SuisseID eine Art Personenkennzahl gibt. Da diese jedoch nur für jedes Zertifikat eindeutig ist, man also als Person mehrere Zertifikate beantragen kann, ist die Assoziation Person-SuisseID global gesehen nicht unbedingt eine 1 zu 1 sondern eher eine 1 zu n Beziehung. Trotzdem lässt sich die SuisseID innerhalb eines Vorgangs als eindeutigen Identifier benutzen über den die Attribute zusammengeführt werden können. Genau genommen gibt es bei der SuisseID sogar zwei unterschiedliche Arten von Aggregationen.

Bei der ersten handelt es sich um sogenannte Combined Assertions, welche von einem IdP/CAS Verbund ausgestellt werden. Erhält ein IdP einen Combined Request, in dem über den reinen AuthnRequest hinaus noch weitere Attribute mittels einer Erweiterung abgefragt werden, ruft er sie mit Hilfe der SuisseID von seinem CAS ab und stellt die Werte dann signiert oder unsigniert zur Verfügung. Bei den abgefragten Attributen handelt es sich um sogenannte Kern Attribute, wie Name und Anschrift, die bei der Registrierung des Benutzers einmalig erfasst wurden und unveränderlich beim CAS abgespeichert sind.

Die zweite Art von Aggregation findet statt, wenn zusätzlich noch Attribute eines weiteren CAS abgefragt werden, nachdem sich der Benutzer bereits mit seiner SuisseID authentisiert hat. Dafür verschickt der SP einen AttributeQuery, der als NameID des Subjects die SuisseID des Benutzers enthält, an den CAS. Dieser kann mit einer Attribute Assertion antworten, nachdem der Benutzer auch hier wieder der Übermittlung zugestimmt hat.

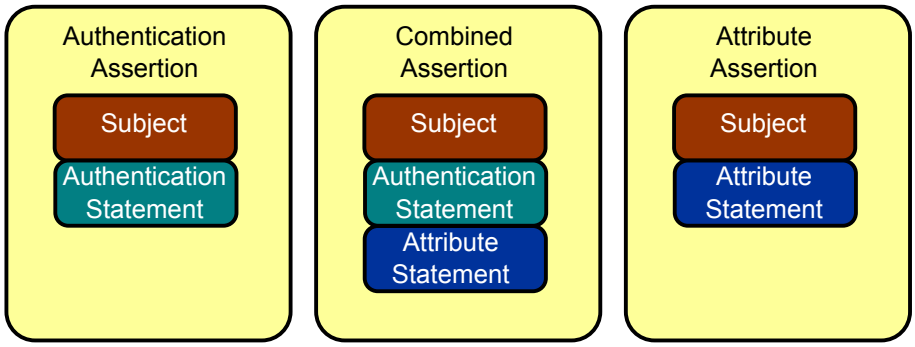

Abbildung 2.6: Typen an Assertions in SuisseID (nach [18], 3.6.1.2)

Beiden Arten ist gemein, dass die Zusammenführung über die SuisseID geschieht. Als Unterschied lässt sich der Ort der Zusammenführung nennen, welcher im ersten Fall beim IdP und im zweiten beim SP liegt. Um dem Benutzer trotzdem die Kontrolle darüber zu geben, wann welche Daten über ihn versendet werden, muss er jeder einzelnen Übertragung der Attribute zustimmen. Die durch die vielen Dialoge

entstehende „Klickmüdigkeit“, in der der Benutzer genervt nur noch alles bestätigt, dürfte jedoch eher kontraproduktiv sein. Besser wäre es, ihm kurz vor Ende der Transaktion eine Zusammenfassung aller Daten anzuzeigen, wodurch er auch zusätzlich einen Überblick über eventuell für ihn ungünstige Datenkonstellationen bekommen würde. Außerdem würde der SP im Falle eines Abbruches durch den Benutzer so nicht bereits einen Teil der Attribute bekommen.

3 Anforderungen

Wie aus den vorhergehenden Kapiteln ersichtlich ist, müssen bestimmte Anforderungen erfüllt werden, um Daten aus zwei Attributquellen verlinken zu können, welche ganz unterschiedlicher Natur sind. So sollte trivialerweise der Diensteanbieter die erhaltenen Attribute verarbeiten können, aber es muss ebenfalls sichergestellt werden, dass diese Attribute authentisch sind und vertraulich behandelt werden, was eine nicht mehr ganz so einfache Aufgabe darstellt. Diese und weitere Anforderungen werden im folgenden Kapitel besprochen. Im nachfolgenden Kapitel kann dann betrachtet werden, welche SAML basierten Lösungen sich am besten eignen und ob diese eher policybasiert sind oder einen technischen Hintergrund haben.

3.1 Kommunikation zwischen den Providern

Bei der Kommunikation zwischen den Providern muss nicht zwangsweise SAML zum Einsatz kommen. Denkbar sind auch andere Standards, wie Protokolle aus der WS-* Familie (insbesondere WS-Trust, WS-Federation und WS-Security), die hier aufgrund der Komplexität allerdings nicht weiter behandelt werden. Ebenfalls wäre es möglich U-Prove Token zu versenden, welche aber wiederum ein Transportprotokoll, zum Beispiel SAML, benötigen. Da die eID-Server schon jetzt *AuthnRequest* und *Response* Nachrichten verstehen, bieten sich SAML-basierte Lösungen an, bei denen nötigenfalls die SAML-Schnittstelle der eID-Server um weitere Konstrukte des Standards, wie das Proxying, ergänzt werden. Noch offen ist dabei die Frage nach Verschlüsselung und Signierung der Nachrichten. Diese soll im nachfolgenden Abschnitt *3.2 Schutzziele* behandelt werden.

3.2 Schutzziele

In der IT-Sicherheit wird zwischen verschiedenen Schutzzielen unterschieden, die auch hier beachtet werden müssen. Dies ist um so wichtiger, da zu den drei Akteuren Bürger, Diensteanbieter und eID-Service des deutschen eID-System eine unbestimmte Anzahl an Attribute Providern hinzukommen, wodurch sich ebenfalls die Komplexität des Systems erhöht. Die Schutzziele für den eID-Server und der Kommunikation zwischen diesem und weiterer Parteien des eID-Systems werden in der

TR-03130 [6] in Anhang C „Anforderungen an den Betrieb von eID-Servern“ Abschnitt 4 „Schutzbedarf“ beschrieben. Des Weiteren werden dort mögliche Gefährdungen und entsprechende Maßnahmen behandelt. Noch betrachtet werden muss nun die Kommunikation der Akteure mit den neu hinzu gekommenen Attribute Providern.

3.2.1 Gefährdungen

Tabelle 3.1: Zusätzliche Gefährdungen durch Attribute Provider

Nr.	***Verbindung/Komponente***	
	Spezifische Gefährdung	***Grundwert(e)***
1	Verbindung 1: *AP - Browser*	
1.1	Einsehen und Verändern von übertragenen Daten (durch Schadprogramme) auf dem Client.	Vertraulichkeit Integrität
1.2	Eingriff in den Verbindungsaufbau für einen Man-in-the-middle Angriff oder um den Benutzer auf einen falschen AP umzuleiten.	Vertraulichkeit Authentizität
1.3	Angabe falscher Benutzerinformationen bei der Authentisierung am AP	Authentizität
2	Verbindung 2: *AP - SP/AP*	
2.1	Einsehen von Nutzerdaten oder abgefragten Attributen durch einen Man-in-the-middle.	Vertraulichkeit
2.2	Einsehen von Nutzerdaten oder abgefragten Attributen durch andere AP.	Vertraulichkeit
2.3	Fälschung eines SP, um an Benutzerdaten zu gelangen.	Authentizität
2.4	Unberechtigte Anfragen von einem eigentlich legitimen SP, um an Benutzerdaten zu gelangen.	Vertraulichkeit Authentizität
3	Verbindung 3: *AP - eID-Server*	
3.1	Einsehen oder Verändern von übertragenen Daten durch einen Angreifer.	Vertraulichkeit Integrität

3.2	Fälschen von Anfragen / Antworten.	Authentizität
4	Attribute Provider	
4.1	Unbefugtes Einsehen und Weitergeben von Nutzerdaten (auch aus dem nPA).	Vertraulichkeit
4.2	Veränderung von Benutzerdaten / Herausgabe falscher Daten.	Integrität
4.3	Verfügbarkeitsstörung der Komponenten.	Verfügbarkeit
4.4	Komprimitierung des SSL-Zertifikates	Authentizität

3.2.2 Maßnahmen

Tabelle 3.2: Einsehen und Verändern von übertragenen Daten auf dem Client

1.1 Einsehen und Verändern von übertragenen Daten auf dem Client	
Anmerkung	Hierbei ist nicht unbedingt ein Angriff durch den Benutzer, sondern durch ein Schadprogramm auf dem Rechner des Benutzers, das übertragene Daten mitliest, gemeint. Dieser Angriff ist umso gefährlicher, da im SAML Redirect- und POST-Binding die gesamte Kommunikation über den Webbrowser geleitet wird. Eine Transportverschlüsselung wie SSL reicht hier außerdem nicht aus, da zum Beispiel Browserplugins Zugriff auf die unverschlüsselten Daten haben.
Maßnahmen	Grundsätzlich sollten SAML-Nachrichten auf XML-Ebene verschlüsselt und signiert werden.
Grenzen	Trotz allem kann ein Schadprogramm ermitteln, an welche Server AuthnRequest und SamlResponse Nachrichten geschickt wurden. Außerdem muss damit gerechnet werden, dass es alle Informationen, die einem Benutzer auf dem Monitor angezeigt werden, ebenfalls sehen kann.

Tabelle 3.3: Eingriff in den Verbindungsaufbau

1.2 Eingriff in den Verbindungsaufbau	
Anmerkung	Denkbar wären Sicherheitslücken durch Schadsoftware auf dem Client oder ein fehlerhaftes DNS. Dies ermöglicht einen Man-in-the-middle Angriff oder das Umleiten des Benutzers auf einen falschen AP.
Maßnahmen	Grundsätzlich sollten Verbindungen SSL gesichert sein und AP gültige SSL-Zertifikate besitzen, sodass der Benutzer diese prüfen kann. Auch gilt wieder, dass SAML-Nachrichten auf XML-Ebene verschlüsselt und signiert werden sollten.
Grenzen	Das Abgreifen von Passwörtern durch gefälschte APs lässt sich bei unvorsichtigen Nutzern trotzdem nicht verhindern.

Tabelle 3.4: Angabe falscher Benutzerinformationen bei der Authentisierung am AP

1.3 Angabe falscher Benutzerinformationen bei der Authentisierung am AP	
Anmerkung	Hierbei ist mit „falsche Benutzerinformationen“ nicht gemeint, dass der Nutzer sich willkürlich Anmeldedaten ausdenkt, sondern dass, wie in Kapitel 2.4.3 *Erschaffung einer neuen Identität* erklärt, Anmeldedaten einer anderen Person angegeben werden. Werden bei anderen APs ebenfalls falsche Angaben gemacht, so lassen sich dadurch neue virtuelle Identitäten generieren.
Maßnahmen	Findet die Authentisierung an den APs über identifizierende Attribute statt, könnte der Benutzer die Obermenge dieser Attribute bei einer vertrauenswürdigen Instanz mit einem mal abrufen. Somit ist sichergestellt, dass alle Logindaten zu der selben Person gehören.
Grenzen	Das Vorgehen funktioniert nur, wenn diese Art der Authentifizierung von allen beteiligten APs unterstützt wird. Auch benötigt man eine bzw. mehrere zusammenhängende Attributquellen, die die benötigten Attribute bereit stellen. Hinzu kommt das Problem, dass AP Zugriff auf mehr Attribute bekommen könnten, als sie für die Authentifzierung eigentlich benötigen.

Tabelle 3.5: Einsehen von Nutzerdaten oder abgefragten Attributen durch einen Man-in-the-middle

2.1 Einsehen von übertragenen Daten durch einen Man-in-the-middle	
Anmerkung	Wird zum Beispiel durch den in Tabelle 3.3 beschrieben Angriff verursacht.
Maßnahmen	Verschlüsselung der SAML-Nachrichten auf XML-Ebene.

Tabelle 3.6: Einsehen von Nutzerdaten oder abgefragten Attributen durch andere AP

2.2 Einsehen von Nutzerdaten oder abgefragten Attributen durch andere AP	
Anmerkung	AP müssen mit Teil der Anfrage etwas anfangen können, möglicherweise sogar mit dem Teil einer Response (Login mit nPA / Zuordnung von Benutzern). Außerdem können sie bei konkatenierten Anfragen Kenntnis von weiteren APs gelangen. Dabei besteht die Gefahr, dass sie an zu viel Informationen kommen, die sie für den Vorgang eigentlich nicht benötigen würde.
Maßnahmen	Durch geeignete Verschlüsselung ist sicher zu stellen, dass alle beteiligten Instanzen nur so viele Informationen wie nötig bekommen.
Grenzen	Bei konkatenierten Anfragen lässt sich nicht verhindern, dass AP gegenseitig voneinander erfahren.

Tabelle 3.7: Fälschung eines SP, um an Benutzerdaten zu gelangen

2.3. Fälschung eines SP, um an Benutzerdaten zu gelangen	
Anmerkung	Bei der Nutzung eines eID-Servers zur Authentisierung werden die Berechtigungen eines SP standardmäßig mit Hilfe der CV-Zertifikate überprüft. Werden allerdings nur AP genutzt, kann ein Angreifer so in den Besitz von Benutzerdaten gelangen.
Maßnahmen	Die Anfragen sollten signiert sein und deren Aussteller mit einer Whitelist abgeglichen werden.

Tabelle 3.8: Unberechtigte Anfragen von einem eigentlich legitimen SP

2.4. Unberechtigte Anfragen von einem eigentlich legitimen SP	
Anmerkung	SP könnten selbstständig AuthnRequests an AP stellen, für Personen, die sich gar nicht auf deren Webseite angemeldet haben. Auch könnte ein SP absichtlich zu viele Attribute abfragen oder womöglich sogar an zu viele AP einen Request senden.
Maßnahmen	Benutzer sollten darüber in Kenntnis gesetzt werden, von welchen AP welche Daten von ihnen abgerufen werden. Dies könnte zum einen eine vertrauenswürdige Instanz, wie der eID-Server oder ein spezieller Client erledigen. Außerdem ist es denkbar, dass der Benutzer sich bei jedem AP neu authentifizieren muss und auf dessen Seite die abgefragten Attribute angezeigt bekommt.
Grenzen	Ständiges Authentisieren ist für den Benutzer sehr umständlich und belastet somit die Akzeptanz des Systems. Besser ist es, ihm kurz vor dem Ende der Transaktion alle abgefragten Daten aufzulisten, was allerdings bedeutet, dass sie zusätzlich für ihn verschlüsselt werden müssen.

Tabelle 3.9: Einsehen oder Verändern von übertragenen Daten durch einen Angreifer

3.1 Einsehen oder Verändern von übertragenen Daten durch einen Angreifer	
Anmerkung	Ein Angreifer kann z.B. als Man-in-the-middle versuchen, übertragene Anfragen oder Attribute auszuspähen oder zu verändern. Auch denkbar ist, dass er nur den Kommunikationsweg in Erfahrung bringen will.
Maßnahmen	Standardmäßig sind Anfragen an und Antworten vom eID-Server auf XML-Ebene verschlüsselt und signiert und auf Transportebene verschlüsselt.
Grenzen	Auch wenn der Angreifer keine Kenntnis darüber erlangen kann, was übertragen wird, kann er erfahren, welche AP abgefragt wurden.

Tabelle 3.10: Fälschen von Anfragen / Antworten

3.2 Fälschen von Anfragen / Antworten	
Anmerkung	Hierbei ist nicht nur gemeint, dass ein unbekannter Dritter eigene Anfragen erstellt, sondern auch, dass ein AP selber unberechtigter Weise versucht, an Benutzerdaten zu gelangen.
Maßnahmen	Durch die CV-Zertifikate und Verschlüsselung für den DA wird sicher gestellt, dass nur berechtigte Provider bestimmte Daten des Ausweises bekommen. Das PACE-Protokoll wiederum sorgt dafür, dass das Auslesen nur durch Benutzerinteraktion geschieht.
Grenzen	Hat ein Angreifer Kenntnis von einem AP, der die ausgelesen Attribute eines nPA anzeigt, Zugriff auf den nPA und kennt die Ausweis-Pin, kann er sich als der Besitzer ausgeben und die Ausweisdaten abgreifen.

Tabelle 3.11: Unbefugtes Einsehen und Weitergeben von Nutzerdaten

4.1 Unbefugtes Einsehen, Verändern oder Weitergeben von Nutzerdaten	
Anmerkung	Hierbei handelt es sich um das altbekannte Problem des Datenmissbrauchs. Schon immer haben Datensammlungen kriminelle Energien dazu verführt, diese weiter zu verkaufen oder anderes damit zu tun. Trotzdem muss ein Benutzer einem AP vertrauen, wenn er dessen Dienste in Anspruch nehmen will. Nicht nur darauf, dass er dessen Daten nicht weiter gibt, sondern auch, dass der Server ausreichend gegen Angriffe gesichert ist.
Maßnahmen	Neben sonstigen Sicherheitsmaßnahmen des Servers könnten zusätzlich die Benutzerdaten verschlüsselt abgelegt sein, wobei der Schlüssel aufgrund der Logindaten des Benutzers generiert wird.
Grenzen	Auch wenn die Daten verschlüsselt werden, müssen sie spätestens beim Abrufen entschlüsselt werden.

Tabelle 3.12: Veränderung von Benutzerdaten / Herausgabe falscher Daten

4.2 Veränderung von Benutzerdaten / Herausgabe falscher Daten	
Anmerkung	Durch einen Fehler beim AP können Benutzerdaten verändert werden oder bei der Authentifizierung falsche Attribute herausgegeben werden.
Maßnahmen	Dem Benutzer sollte die Möglichkeit gegeben werden, die übertragenen Daten vor dem Abschluss der Transaktion zu kontrollieren und diese auch abbrechen zu können.
Grenzen	Durch das Anzeigen der Attribute, wird wiederum der in Tabelle 3.2 besprochene Angriff ermöglicht.

Tabelle 3.13: Verfügbarkeitsstörung der Komponenten

4.3 Verfügbarkeitsstörung der Komponenten	
Anmerkung	Mit der Zunahme der beteiligten Akteure und der Kommunikationswege steigt auch das Risiko des Ausfalls von einem Akteur. Ist ein AP dabei betroffen, fehlen für den Vorgang benötigte Attribute.
Maßnahmen	Wie in Tabelle 25 der TR-03130 beschrieben, sollten die Netze und Komponenten lastabhängig ausgestaltet sein. Hinzu kommt die Verwendung von hochverfügbaren Architekturen für Server und die redundante Auslegung der Netzkomponenten. Eventuell kann sogar ein Alternativserver beim Ausfall eines AP gefunden werden.
Grenzen	Auch hochverfügbare Komponenten können (mit einer sehr geringen Wahrscheinlichkeit) ausfallen. Ist ein dringend benötigter AP nicht erreichbar, muss der Vorgang entweder abgebrochen oder verzögert werden.

Tabelle 3.14: Kompromittierung des SSL-Zertifikates

4.4 Kompromittierung des SSL-Zertifikates	
Anmerkung	Diese Bedrohung zieht weitere mögliche Angriffe nach sich. So können Benutzer auf eine imitierte Seite des AP gelockt werden, ohne dass es ihnen auffällt. Wobei es dafür auch ausreichen würde, dessen Zertifikat zu fälschen[1]. Auch lässt sich so die Kommunikation zwischen Benutzer und dem AP entschlüsseln.
Maßnahmen	Grundsätzlich gilt, dass es für Angreifer so schwer wie möglich sein sollte, an den privaten Schlüssel eines AP zu gelangen. Sollte ein Zertifikat doch mal kompromittiert sein, ist eine funktionierende Sperrlistenverwaltung unumgänglich. Zusätzlich könnte ein Hinweis des Browsers, wenn sich der Fingerabdruck eines Providers geändert hat, für den Benutzer hilfreich sein, solche Art von Angriffen zu erkennen.
Grenzen	Und zusätzliche Hilfsmittel ist es für eine normale Person sehr schwer, diese Art von Angriff zu erkennen.

Datensparsamkeit

Wie bereits verschiedentlich erwähnt, ist die Datensparsamkeit ebenfalls ein wichtiges Schutzziel, welches sogar im § 3a des Bundesdatenschutzgesetzes (BDSG) genannt wird. Danach dürfen nur die personenbezogenen Daten verarbeitet werden, die für einen Anwendungsfall unbedingt erforderlich sind. Grund dafür ist, dass man versucht große Datensammlungen zu vermeiden, mit deren Hilfe man detaillierte Benutzerprofile erstellen kann. Im konkreten Fall hat dies zum einen Auswirkungen darauf, welche Benutzerdaten die AP heraus geben. So dürfen trivialerweise Diensteanbieter nicht mehr Informationen erhalten, als sie benötigen und es ist darauf zu achten, dass ein eigentlich anonymer Benutzer durch für ihn ungünstige Datenkonstellationen unterschiedlicher Attribute Provider nicht mehr anonym ist. Zum anderen dürfen AP wiederum nicht an Benutzerdaten gelangen, die eigentlich nicht für sie bestimmt sind. Das kann zum Beispiel passieren, wenn ein Satz identifizierender Attribute dazu genutzt wird, den Benutzer bei allen in einem Anwendungsfall beteiligten AP zu authentifizieren.

[1] Erfolgreiche Angriffe auf SSL-Registrare, bei denen die Kriminellen an Zertifikate und deren zugehörigen Schlüssel größerer Webseiten gelangten, sind bereits Realität, wie in [7] nachzulesen ist.

Anonymität / Pseudonym

Neben dem Gebot der Datensparsamkeit wird in § 3a des BDSG auch festgelegt, dass personenbezogene Daten wenn möglich zu anonymisieren und pseudonymisieren sind. Bei der Anonymisierung werden personenbezogene Daten derart verändert, dass sie nicht oder nur mit großem Aufwand einer Person zugeordnet werden können. Bei der Pseudonymisierung werden die identifizierenden Attribute durch ein Pseudonym ersetzt, sodass die Daten ebenfalls keiner natürlichen Person mehr zugeordnet werden können. Bei der mehrfachen Verwendung eines Pseudonyms kann es jedoch passieren, dass durch die Verlinkung der Daten die Person nicht mehr anonym agiert bzw. das Anonymity Set, also die Menge an Personen auf die diese Daten ebenfalls zutreffen, stark verkleinert wird. Das deutsche eID-System ist darauf ausgelegt, dass Benutzer anonym (z.B. reine Altersverifikation) oder über ein Pseudonym (rID) agieren können. Bei der Hinzunahme von Attribute Providern ändert sich das jedoch. Wie bereits erwähnt kann durch die Aggregation mehrerer Attribute die Anonymität nicht mehr gewährleistet sein, da mit jedem hinzugefügten Attribut das Anonymity Set kleiner wird oder gleich bleibt. Deswegen ist es um so wichtiger für den Nutzer, zu wissen, welche Provider über welches Wissen von ihm verfügen. Gleichzeitig entsteht durch Anonymität auch das in Kapitel 2.4.3 beschriebene Problem, dass mehrere Attribute Provider sich nicht untereinander darüber austauschen können, wessen Attribute sie gerade ausliefern. Aus technischen Gründen kommt die rID hierfür nicht in Frage, da für jeden DA eine eigene rID generiert wird. Die Alternative per identifizierender Attribute widerspricht allerdings der Anonymität und evtl. dem Gebot der Datensparsamkeit. Hierfür ist somit ein Trade-off zwischen dem Schutz des Benutzers und des DA abzuwägen.

3.3 Zuordnung des Benutzers zwischen den AP

Da zu dem eID-Server mit den Attribute Providern noch weitere Datenquellen hinzukommen, muss ein Weg gefunden werden, wie diese Daten dem selben Benutzer zugeordnet werden können. Des Weiteren eröffnet solch eine Zuordnung auch die Möglichkeit, dass man sich nur bei einer Stelle authentisieren muss, um von allen weiteren AP die benötigten Attribute zu bekommen. Dabei gibt es mehrere Möglichkeiten, welche unterschiedliche Vor- und Nachteile aufweisen:

Globale ID Ähnlich einer PKZ besitzt jeder Benutzer eine bei allen Providern bekannte ID. Dies ist zum Beispiel bei der SuisseID der Fall, bei der sich der Benutzer nur bei seinem IdP authentisieren muss, um von anderen CAS über seine SuisseID Attribute zu erhalten. Dabei wird auch deutlich, dass ein Benutzer auch mehrere IDs haben kann. Wichtig ist nur, dass jede ID einem Benutzer eindeutig zugeordnet werden kann. Ebenfalls denkbar sind Benutzerzertifikate, dessen Einsatz jedoch aufgrund der geringen Verbreitung und der fehlenden

Unterstützung im nPA unrealistisch ist. Der größte Nachteil globaler IDs ist deren Globalität, sodass Anonymität nicht mehr gegeben ist.

Pseudonym Pseudonyme ermöglichen Anonymität, sofern die mit ihnen verbundenen Attribute keine Identifizierung erlauben. In SAML existieren spezielle Pseudonym Identifiers (siehe Kapitel 2.2.1), die diese Aufgabe übernehmen. Schwierig ist dabei die einmalige Einigung der Attribute Provider auf eine ID für den Benutzer, da dabei auch die Konten verschiedener Personen verlinkt werden könnten (siehe Kapitel 2.4.3). Die rID des nPA eignet sich übrigens nicht als Pseudonym für den Anwendungsfall, da Nutzer Prinzip bedingt bei verschiedenen DA auch verschiedene rIDs besitzen.

Identifizierende Attribute Bei genügend großer Überschneidung der Attributmengen von APs können diese zur Zuordnung genutzt werden. Da es sich um persönliche Daten handelt, ist eine geeignete Verschlüsselung hier besonders wichtig. Diese muss für alle beteiligten Attribute Provider einzeln geschehen, sodass mit einem großen XML-Overhead zu rechnen ist.

3.4 Attribute

Die Einigung zwischen den Providern auf die verwendeten Attributen und deren Typen ist eher ein vertragliches als ein technisches Problem. So ist es relativ einfach, für die verwendeten SAML-Attribute ein XML-Schema zu erstellen, wenn der verwendete Satz an Attributen fest steht. Die verwendeten Attribute des eID-Servers leiten sich aus den Datengruppen des nPA ab und werden in der TR-03130 beschrieben. Zusätzlich existiert ein XML-Schema in der die komplexen Typen mancher Attribute, wie zum Beispiel DateOfBirth oder PlaceOfResidence, erklärt sind. Stellt nun ein AP Benutzerdaten für andere Provider zur Verfügung, so sollte er einen ähnlichen Weg gehen und Syntax sowie Semantik der bereit gestellten Attribute angeben. Dabei ist darauf zu achten, dass Attribute verschiedener AP mit eigentlich gleicher Semantik eine andere Benennung bzw. Syntax haben können. Gleichzeitig kann es aber auch zu dem entgegen gesetzten Fall kommen, dass gleich benannte Attribute unterschiedliche Bedeutungen haben. Somit sollten AP grundsätzlich einen eigenen Namespace für ihre Elemente festlegen, um Namenskollisionen in aggregierten SAML-Response Nachrichten zu vermeiden.

3.5 Aggregation der Attribute

Die Aggregation der Benutzerattribute ist prinzipiell in allen beteiligten Instanzen möglich, wenn auch nicht immer sinnvoll. Ausdrücklich nicht gewünscht ist zum

Beispiel eine Zusammenfassung der unverschlüsselten Attribute bei einem AP oder dem eID-Server. Mögliche Aggregationsorte sind:

Diensteanbieter Bei diesem „klassischen“ Weg sendet der DA an jeden AP eine eigene Anfrage, und sammelt sich somit die Attribute selber zusammen. Ein Beispiel sind Onlineshops, welche selbständig Schufa-Anfragen tätigen.

Client Sofern der Benutzer einen speziellen Client oder Proxy besitzt, können die Attribute auch dort aggregiert werden. Dadurch hat er mehr Kontrolle über seine Daten und über die Kommunikation zwischen den Providern. Dies ist allerdings nur der Fall, wenn die Attribute Provider die Daten für ihn verschlüsseln können, er also ein eigenes Zertifikat besitzt. Andernfalls würden sie bei ihm nur für den SP verschlüsselt vorliegen, sodass er keine Kontrolle darüber hätte.

Attribute Provider Hierbei ist nicht gemeint, dass ein AP alle Attribute des Benutzers aggregiert, sondern dass die Aggregation nacheinander stattfindet. Denkbar ist eine Verkettung der AP, sodass jeder AP seine Daten an eine SAML Response anfügt.

eID-Server Der eID-Server nimmt unter den Providern eine besonders vertrauenswürdige Rolle ein. Somit könnte er stellvertretend für DA nach der Authentifizierung des Benutzers SAML Requests an die AP senden, welche entsprechende Informationen zur Identifizierung des Benutzers enthalten.

3.6 Auswahl der AP

Grundsätzlich muss der DA die Attribute Provider kennen, von denen er Benutzerdaten bekommt. Dabei ist nicht nur wichtig, dass er die Adresse kennt, zu der der Benutzer geleitet wird, sondern vielmehr, dass der DA den erhaltenen Daten trauen kann. Die Reihenfolge der AP bei der Abfrage wird durch verschiedene Faktoren beeinflusst. So spielen der Aggregationsort, die Zuordnung des Benutzers zwischen den AP und das eventuelle Vorhandensein einer Zentralen Instanz, die den Benutzer identifiziert, eine Rolle, wie in den vorherigen Abschnitten deutlich wird. So kann zum Beispiel wichtig sein, dass ein AP zuerst den Benutzer authentifiziert und die nachfolgenden ihre Daten nur an die Response anhängen.

3.7 Bestimmung der abgefragten Attribute

Um einem Attribute Provider die erforderlichen Attribute mitzuteilen gibt es mehrere Möglichkeiten. Der SAML-Standard sieht ursprünglich das *AttributeQuery* Element vor, welches in Abschnitt 3.3.2.3 SAML Core [13] beschrieben wird. Dabei

wird vorausgesetzt, dass sich der Benutzer zuvor authentifiziert hat und einen Name Identifier besitzt. Bei der SuisseID werden die angeforderten Attribute in der Extension eines AuthnRequests mitgeschickt. Das deutsche eID-System verwendet ebenfalls Extensions, aber bietet durch die Berechtigungszertifikate eine zusätzliche Einschränkung. Um kompatibel zu dem System zu bleiben, sollten die Attribute von den AP ebenfalls in AuthnRequests spezifiziert werden.

4 Mögliche Ansätze

Aufbauend auf die vorhergehenden Kapitel und unter Berücksichtigung der Anforderungen können nun mögliche Ansätze diskutiert und dabei deren Vor- und Nachteile aufgezeigt werden. Hauptsächlich unterscheiden sich diese Ansätze in dem Ort der Attributaggregation und somit der Frage, welche Instanz für die Koordination der Anfragen und Antworten zuständig ist.

4.1 DA orientiert

Bei diesem naiven Ansatz authentisiert sich der Benutzer zuerst beim eID-Server. Die dabei erhaltenen Attribute benutzt nun der Diensteanbieter, um selbstständig Anfragen an die einzelnen AP zu stellen. Die Aggregation der Attribute findet also erst beim DA statt. Voraussetzung dafür ist, dass der DA alle erforderlichen Attribute für die Anfragen erhält, damit die AP die Person zuordnen können. Auch müssen die AP Benutzerdaten ohne zusätzliche Nutzerinteraktion weitergeben, was eventuell durch eine einmalige Zustimmung erreicht werden kann.

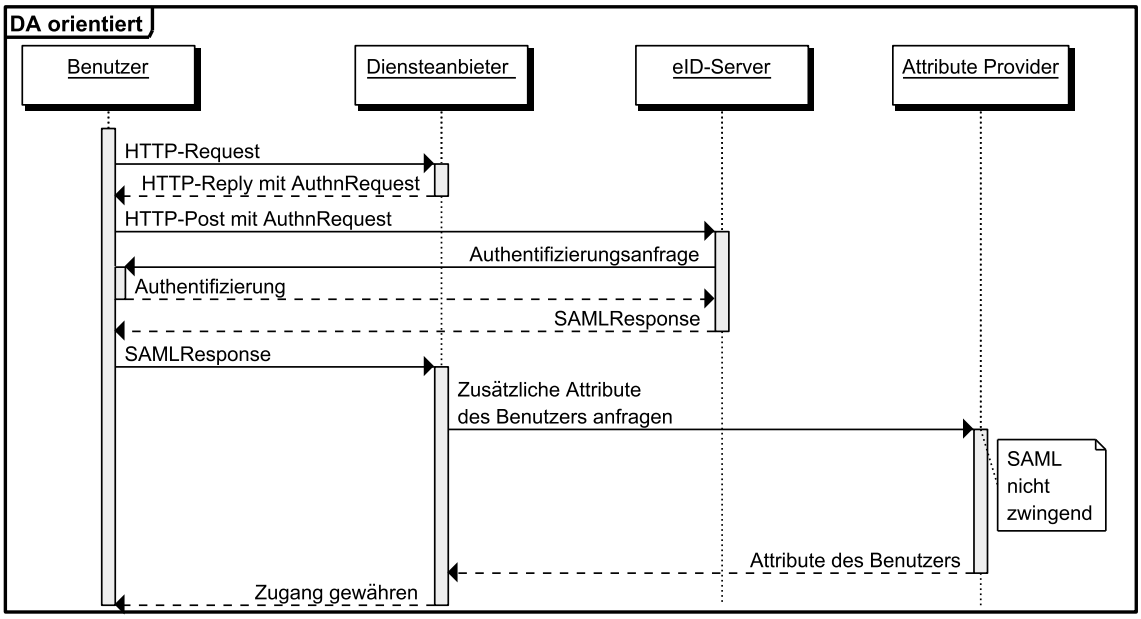

Abbildung 4.1: Sequenzdiagramm des SP orientierter Ansatzes

Als Vorteil ist die relative Einfachheit der Lösung zu nennen. Beim deutschen eID-System müssen hierfür keine Änderungen durchgeführt werden. Auch bewahrt es den DA vor der Gefahr, dass er Attribute eines anderen Benutzers erhält. Deswegen existieren bereits Lösungen, die älter als der nPA sind, um zum Beispiel die Kreditwürdigkeit von Kunden zu erfahren.

Die Nachteile liegen klar auf der Seite des Benutzers. Dieser kann weder anonym handeln, noch hat er Kontrolle darüber, welche Daten der DA bekommt. Er muss somit darauf vertrauen, dass sich der DA an das Gebot der Datensparsamkeit hält.

4.2 Client orientiert

Im Gegensatz zum ersten Ansatz erfordert dieser um einiges mehr Benutzerinteraktion. Voraussetzung ist, dass der Benutzer über einen sogenannten Enhanced Client verfügt, ein Programm, welches mehr Möglichkeiten für die Authentifizierung bietet, als ein Webbrowser. Dieses generiert selber AuthnRequests und kann die entsprechenden Antworten auswerten und zusammen führen. Vorzugsweise geschieht das so, dass sich der Benutzer nur beim eID-Server oder bei einem anderen speziellen AP authentisiert und dabei die entsprechenden Attribute erhält, über die er Zugriff auf die Daten anderer AP erlangt, ohne sich neu anmelden zu müssen. Auch ist denkbar, dass sich die AP untereinander bei der ersten Authentifizierung des Benutzers auf eine entsprechende ID für ihn einigen.

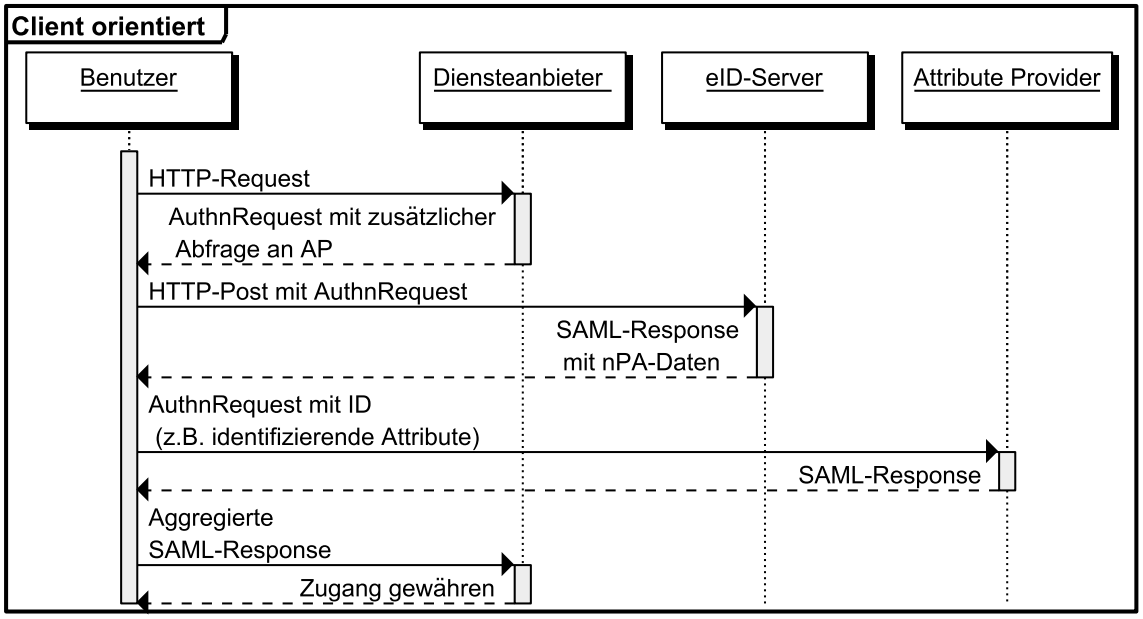

Abbildung 4.2: Sequenzdiagramm des Client orientierter Ansatzes

Dies bietet dem Nutzer einige Vorteile bezüglich des Datenschutzes. Da die AuthnRequests kein Zertifikat des DA enthalten müssen, können die AP keinen Rückschluss darauf ziehen, für welchen DA sie die Response ausstellen. Außerdem kann der Benutzer einsehen, welche Attribute bei welchem AP von ihm abgefragt werden und den Vorgang gegebenenfalls abbrechen, bevor der DA überhaupt Daten von ihm erhalten hat.

Ein Nachteil sind die speziellen Voraussetzungen, die erfüllt sein müssen. So benötigt der Benutzer einen speziellen Client und womöglich sogar ein Clientzertifikat. Des Weiteren müssen die AP auf clientgenerierte Anfragen auch antworten und haben Schwierigkeiten bei der Abrechnung dieser, sofern sie nicht den Benutzer zur Kasse bitten wollen. Er muss außerdem selber darauf achten, nicht Phishingseiten zum Opfer zu fallen und für sie Daten zur Verfügung zu stellen.

4.3 eID-Server orientiert

Diese Lösung macht sich zu Nutze, dass der eID-Server bereits eine vertrauenswürdige Instanz darstellt, welcher sogar kurzzeitigen Zugriff aus Ausweisdaten eingeräumt wird. Wie in Abbildung 4.3 zu sehen sendet der DA zuerst einen AuthnRequest mit den entsprechend geforderten Attributen an den eID-Server. Dieser überprüft, welche er aus dem nPA zur Verfügung stellen kann und für welche zusätzliche AP benötigt werden. Werden weitere AP benötigt, so nutzt er für die ein eigenes Berechtigungszertifikat, um an die von den AP zur Authentifizierung benötigten Attribute zu gelangen. So kann er eigene AuthnRequests bzw. AttributeQuerys an die AP senden, deren Antworten aggregieren, signieren und verschlüsseln und die Informationen in einer Assertion an den DA schicken.

Der Vorteil für den DA liegt darin, dass er die Antwort aus genau einer Quelle bekommt, der er als einziges nur vertrauen muss. Um die Beschaffung der Attribute und deren Beglaubigung kümmert sich der eID-Server. Auch für den Benutzer stellt der Vorgang eine Vereinfachung dar, da er sich nur einmal authentisieren muss.

Der Nachteil liegt auf Seiten des eID-Servers, der nun nicht mehr nur zum Auslesen des nPA zuständig ist, sondern zusätzlich die Verwaltung von Attribute Providern und das eigenständige Versenden von Anfragen bzw. Aggregieren und Signieren von Antworten kümmern muss.

4.4 Proxying

Hierbei handelt es sich um einen rein hypothetischen Ansatz, welcher zum Ziel die eindeutige Zuordnung des Benutzers zwischen den AP und einem möglichst guten Datenschutz hat. Voraussetzung ist, dass die Authentifizierung bei allen AP mit

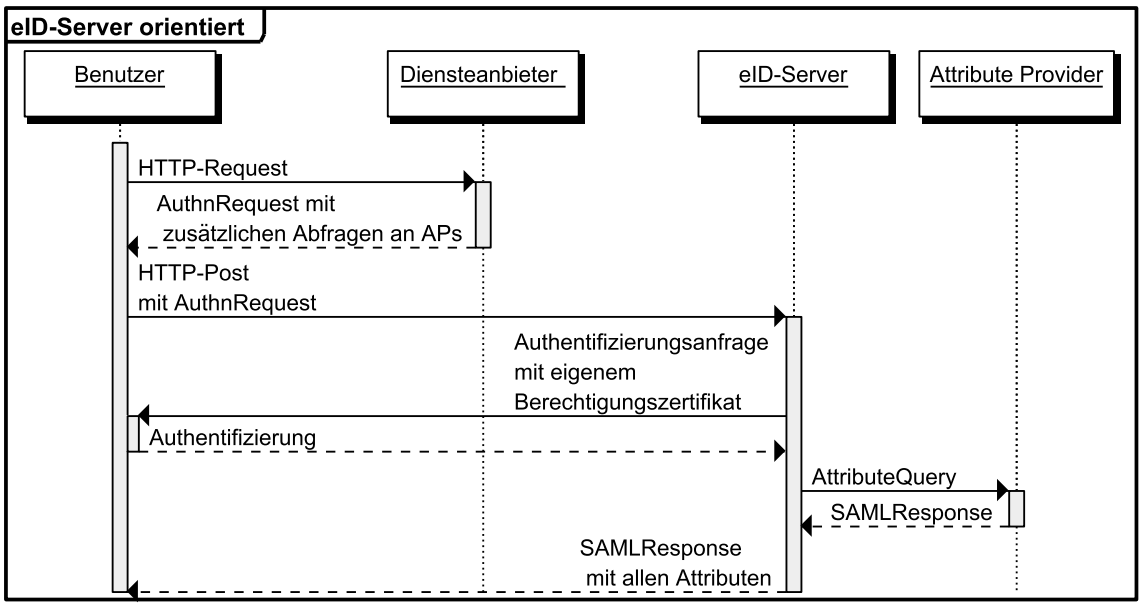

Abbildung 4.3: Sequenzdiagramm des eID-Server orientierter Ansatzes

Hilfe des nPA erfolgt. Des Weiteren muss es eine Föderation des DA und der AP geben, welche über ein eigenes „Federation-Berechtigungszertifikat“ verfügt, das eine Obermenge der einzelnen Berichtigungen darstellt. Trotzdem sollten die DA der Föderation nur Zugriff auf die Daten erhalten, die sie auch benötigen. Ebenfalls ist zu beachten, dass die rID des Benutzers für alle Mitglieder der Föderation gleich ist. Ein ähnlicher Effekt kann auch durch Verlegen der Berechtigungszertifikate in einen gleichen Sektor erreicht werden, wird hier allerdings nicht betrachtet.
Der Ablauf des in Abbildung 4.4 dargestellten Ansatzes ist wie folgt:

(a) Der DA generiert einen AuthnRequest mit mehreren verschlüsselten Extensions für den eID-Server und jeden AP, welche die angeforderten Attribute enthalten, und einer Proxy-List, in denen alle AP und zum Schluss der eID-Server stehen.

(b) Die Anfrage wird nacheinander an die AP der Proxy-List geschickt, welche in dem AuthnRequest die an sie gerichteten Anfragen entfernen und durch Anfragen mit den für die Authentifizierung erforderlichen Attributen an den eID-Server ersetzen. Nachdem der eigene Eintrag aus der Proxy-List entfernt wurde, wird der so veränderte AuthnRequest an den nächsten Eintrag weitergeleitet.

(c) Der eID-Server überprüft zum Schluss, ob alle enthaltenen Anfragen von Mitgliedern der Föderation stammen, authentifiziert den Benutzer und erstellt für jeden AP und den DA eine extra Assertion in der nur die für den jeweiligen Anbieter erforderlichen Daten stehen. Die so erstellte Response schickt er nun an den letzten AP zurück.

(d) Der AP wertet die entsprechende Assertion aus, ersetzt sie durch die signierten und für den DA verschlüsselten Benutzerdaten und sendet sie an den vorherigen AP bzw. an den DA.

(e) Der DA entschlüsselt alle Assertions, überprüft deren Signaturen, wobei besondere Rücksicht auf Signature Wrapping Attacken zu nehmen ist und kann nun mit der Transaktion fortfahren.

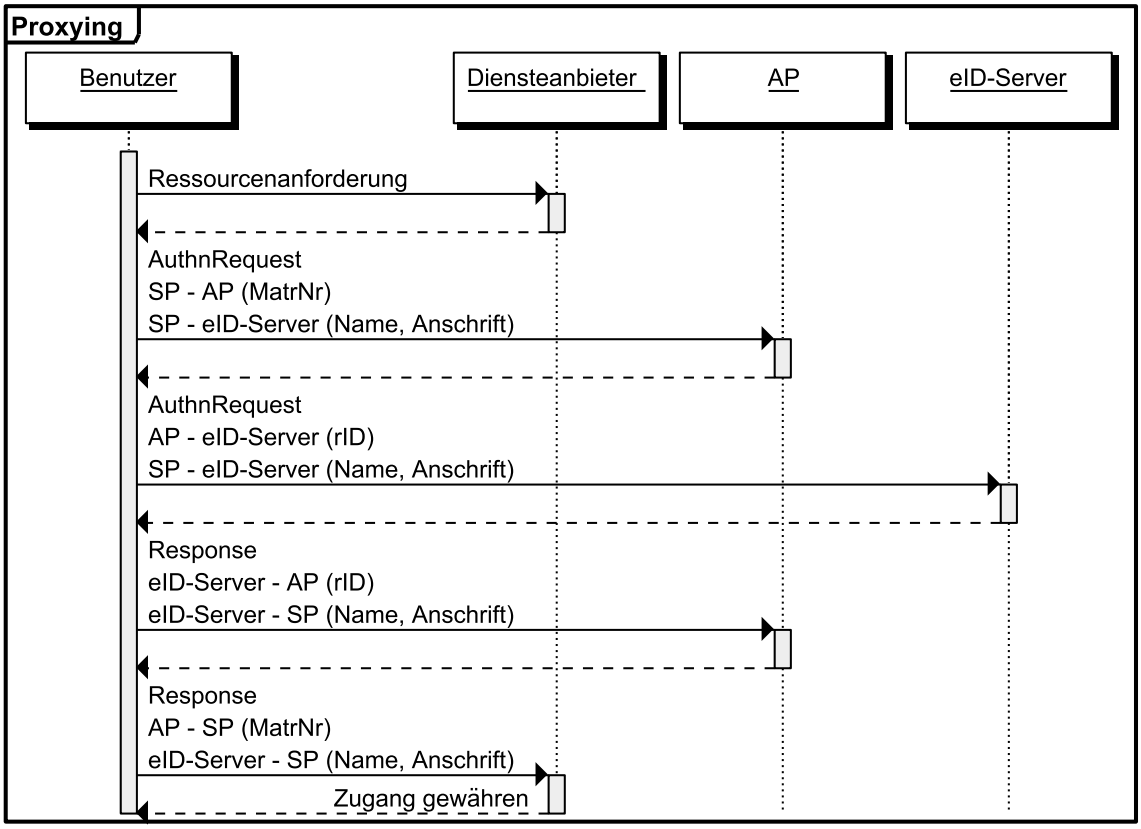

Abbildung 4.4: Sequenzdiagramm des Proxying Ansatzes

Der Vorteil liegt zum einen auf Seiten des DA, da dieser sich sicher sein kann, nicht aggregierte Attribute unterschiedlicher Personen bekommen zu haben. Zum anderen hat der Benutzer hier eine bessere Übersicht über die Verwendung seiner Daten als im DA orientierten Ansatz. Bei geeigneten zusätzlichen Maßnahmen kann sogar sichergestellt werden, dass auch bei ausreichenden Berechtigungen des „Federation-Berechtigungszertifikates“ die einzelnen Teilnehmer der Föderation nur auf eine benötigte Teilmenge an Daten zugreifen dürfen.

Der größte Nachteil ist der komplexe Verlauf der Anfragen und Antworten, wobei bei, bis auf den Client, jedem Teilnehmer Anpassungen vorgenommen werden müssen. Besonders beim eID-Server ist der Eingriff kritisch, da hier am System der Berechtigungszertifikate eingegriffen wurde, sodass diese Lösung gegen derzeit geltende Anforderungen verstoßen würde. Der Vorteil der gleichen rID und somit

die Verlinkbarkeit stellt sich als Nachteil für den Benutzer heraus, da dieser seine Anonymität verliert.

5 Fazit

In der Studienarbeit wurde das Problem behandelt, wie sich Daten aus dem neuen Personalausweis mit denen weiterer Attribute Provider verknüpfen lassen. Dafür wurden zuerst die entsprechenden Teile des deutschen eID-Systems und seine Grenzen bezüglich der Attributaggregation betrachtet. Folgend wurde mit der SuisseID eine bereits existierende Lösung für das Problem vorgestellt, welche jedoch aufgrund der globalen ID, nämlich der SuisseID, nicht den hiesigen Datenschutzansprüchen genügt. Zum Schluss wurden Anforderungen festgelegt, welche ein mögliches eID-System leisten muss, und auf deren Basis potentielle Lösungen vorgestellt. Bei diesen muss jedoch immer zwischen Komfort und Schutz des Benutzers vor Datenmissbrauch abgewägt werden. Auch zeigt sich an allen Lösungen, dass Änderungen bei den beteiligten Instanzen oder womöglich sogar beim bestehenden eID-System vorgenommen werden müssen. Ein besonders schwieriges Problem stellt die Verknüpfung der Benutzerkonten zwischen den Attribute Providern dar, die aufgrund fehlender Identifier bei einigen Anwendungsfällen, fälschlicher Weise auch für Konten verschiedener Personen hergestellt werden kann.

Anhang

Listing 1: Extended AuthnRequest for Combined Assertions

```
<?xml version="1.0" encoding="utf-8"?>
<samlp:AuthnRequest
  Destination="https://a-suisseid-idp.ch/samlprovider/
     authenticate"
  ProviderName="SuisseID Service Provider AG"
  ForceAuthn="true"
  ID="identifier_1"
  IsPassive="false"
  IssueInstant="2009-11-21T11:43:08.3344Z"
  Version="2.0">
  <saml:Issuer>https://a-suisseid-sp.ch/saml/acs</saml:Issuer>
  <samlp:Extensions>
    <saml:Attribute
      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-
         format:uri"
      Name="http://www.ech.ch/xmlns/eCH-0113/1/
         identificationNumber"
      eCH-0113:required="true" />
    <ic:PrivacyNotice Version="1">
      http://a-suisseid-sp.ch/privacy_policy.html
    </ic:PrivacyNotice>
  </samlp:Extensions>
</samlp:AuthnRequest>
```

Listing 2: SAML Attribute Request to the CAS

```
<samlp:AttributeQuery
  ID="acf23494-1743-2155-4334a-fc453464ab56"
  Version="2.0"
  IssueInstant="2009-11-26T20:31:40Z"
  Destination="https://a-suisseid-idp.ch/samlprovider/query">
  <saml:Issuer>https://a-suisseid-sp.ch/saml/acs</saml:Issuer>
  <samlp:Extensions>
    <ic:PrivacyNotice Version="1">http://a-suisseid-sp.ch/
       privacy_policy.html
    </ic:PrivacyNotice>
    <eCH-0113:assertionConsumerServiceUrl>
```

```
11        https://www.a-suisseid-sp.ch/saml/acs
12      </eCH-0113:assertionConsumerServiceUrl>
13    </samlp:Extensions>
14    <saml:Subject>
15      <saml:NameID
16        Format="urn:oasis:names:tc:SAML:1.1:nameid-
              format:unspecified">
17        1234-5678-9012-3456
18      </saml:NameID>
19    </saml:Subject>
20    <saml:Attribute
21      NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
22      Name="http://www.ech.ch/xmlns/eCH-0113/1/
            identificationValidUntil"
23      FriendlyName="Identification Valid Until"
24      eCH-0113:required="true" />
25  </samlp:AttributeQuery>
```

Literaturverzeichnis

[1] Gesetz über Personalausweise und den elektronischen Identitätsnachweis (Personalausweisgesetz). http://www.buzer.de/gesetz/8806/index.htm (Abruf am 29. September 2010), Juni 2009.

[2] Technische Richtlinie TR-03112, eCard-API-Framework Version 1.1. https://www.bsi.bund.de/cln_183/ContentBSI/Publikationen/TechnischeRichtlinien/tr03112/index_htm.html (Abruf am 10. Dezember 2010), Juli 2009. Bundesamt für Sicherheit in der Informationstechnik.

[3] Leitlinie für die Vergabe von Berechtigungen für Diensteanbieter nach § 21 Abs. 2 Personalausweisgesetz. http://www.personalausweisportal.de/SharedDocs/Downloads/DE/Leitlinie_VfB_Vergabe_Berechtigungszertifikate.pdf?__blob=publicationFile (Abruf am 30. November 2010), November 2010. Vergabestelle für Berechtigungszertifikate.

[4] Liste der Diensteanbieter mit Berechtigungszertifikat. http://www.personalausweisportal.de/SharedDocs/Downloads/DE/Liste_Diensteanbieter.html (Abruf am 8. Dezember 2010), Dezember 2010. Vergabestelle für Berechtigungszertifikate.

[5] Technische Richtlinie TR-03110, Advanced Security Mechanisms for Machine Readable Travel Documents - Extended Access Control (EAC), Password Authenticated Connection Establishment (PACE) and Restricted Identification (RI), Version 2.05. https://www.bsi.bund.de/cln_183/ContentBSI/Publikationen/TechnischeRichtlinien/tr03110/index_htm.html (Abruf am 10. Dezember 2010), Oktober 2010. Bundesamt für Sicherheit in der Informationstechnik.

[6] Technische Richtlinie TR-03130, eID-Server Version 1.4.1. https://www.bsi.bund.de/cln_174/ContentBSI/Publikationen/TechnischeRichtlinien/tr03130/tr-03130.html (Abruf am 10. Dezember 2010), Oktober 2010. Bundesamt für Sicherheit in der Informationstechnik.

[7] Daniel Bachfeld. SSL-GAU zwingt Browser-Hersteller zu Updates. http://heise.de/-1212986 (Abruf am 4. Juni 2011), März 2011. Heise Online.

[8] Carolin Eckenfels (dpa). Fehlerhafte Personalausweise in Hessen. http://heise.de/-1141762 (Abruf am 30. November 2010), November 2010. Heise Online.

[9] D. Eastlake et al. XML Encryption Syntax and Processing. http://www.w3.org/TR/xmlenc-core/ (Abruf am 16. Mai 2011), Dezember 2002. W3C.

[10] D. Eastlake et al. XML Signature Syntax and Processing (Second Edition). http://www.w3.org/TR/xmldsig-core/ (Abruf am 16. Mai 2011), Juni 2008. W3C.

[11] J. Hodges et al. Glossary for the OASIS Security Assertion Markup Language (SAML) V2.0. http://docs.oasis-open.org/security/saml/v2.0/saml-glossary-2.0-os.pdf (Abruf am 27. August 2010), März 2005. OASIS SSTC.

[12] N. Ragouzis et al. Security Assertion Markup Language (SAML) V2.0 Technical Overview V.2.0 CD02. http://docs.oasis-open.org/security/saml/Post2.0/sstc-saml-tech-overview-2.0-cd-02.pdf (Abruf am 26. August 2010), März 2008. OASIS SSTC.

[13] S. Cantor et al. Assertions and Protocols for the OASIS Security Assertion Markup Language (SAML) V2.0. http://docs.oasis-open.org/security/saml/v2.0/saml-core-2.0-os.pdf (Abruf am 19. November 2010), März 2005. OASIS SSTC.

[14] T. Bray et al. Extensible Markup Language (XML) 1.0 (Fifth Edition). http://www.w3.org/TR/xml/ (Abruf am 16. Mai 2011), November 2008. W3C.

[15] Jeff Hodges. Liberty Glossary v2.0. http://www.projectliberty.org/resource_center/specifications/liberty_alliance_specifications_support_documents_and_utility_schema_files/liberty_glossary_v2_0/ (Abruf am 11. Juni 2011), 2006. Liberty Alliance Project.

[16] Bernd Kowalski. Die eCard-Strategie der Bundesregierung im Überblick. http://subs.emis.de/LNI/Proceedings/Proceedings108/gi-proc-108-008.pdf (Abruf am 10. Dezember 2010). Bundesamt für Sicherheit in der Informationstechnik.

[17] L. Liao J. Schwenk S. Gajek, M. Jensen. Analysis of Signature Wrapping Attacks and Countermeasures. http://www.nds.rub.de/media/nds/downloads/mjensen/ICWS09.pdf (Abruf am 23. Dezember 2010), Juli 2009. Ruhr Universität Bochum.

[18] M. Zweiacker U. Bürge. SuisseID Specification - Digital Certificates and Core Infrastructure Services. http://www.suisseid.ch/unternehmen/technik/index.html?lang=de&download=NHzLpZeg7t,lnp6I0NTU042l2Z6ln1acy4Zn4Z2qZpnO2Yuq2Z6gpJCDdIJ9e2ym162epYbg2c_

`JjKbNoKSn6A--` (Abruf am 27. September 2010), Juni 2010. Staatssekretariat für Wirtschaft (SECO).