



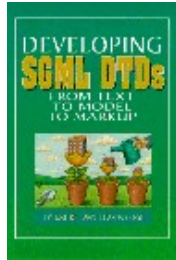
Web Services and Identity

2.2 Federated Identity Technologies

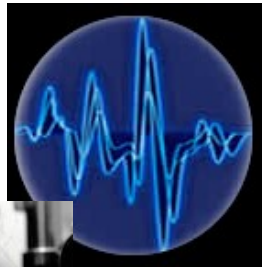
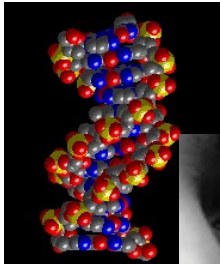
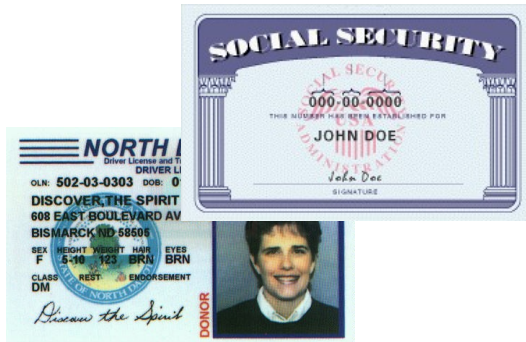
Eve Maler, Sun Microsystems

Introducing myself

- Technology Director at Sun
 - Currently specialising in Sun-Microsoft interoperability and “identity R&D”
- One of the original XML geeks
 - And an old SGMLer
- Leadership roles in various standards efforts and technology projects:
 - Security Assertion Markup Language (SAML)
 - Liberty Alliance
 - Universal Business Language (UBL)
 - DocBook



Identity & web services: two great tastes...



- Most web services are ultimately performed on behalf of people and things with unique identities
- Digital identities are often managed with the help of web services
- Web services can do tasks (or help other humans do tasks) relating to you when you're offline

Topics we'll cover in this session

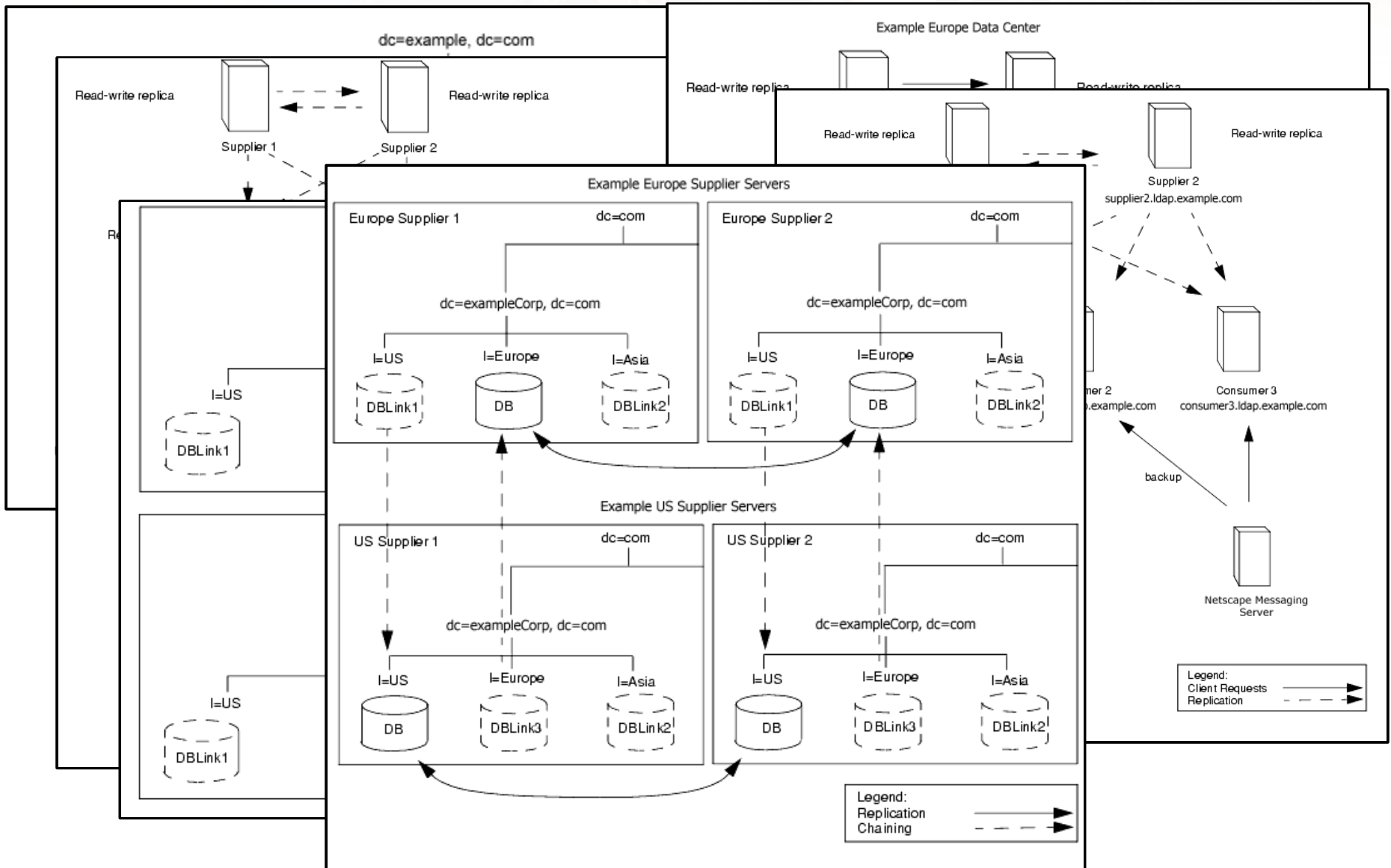
- Opportunities, risks, use cases, and challenges for federated identity
 - Including the famous Single Sign-On
- Examining three protocol technologies you need to know about, in light of the above (in roughly “chronological” order):
 - SAML
 - OpenID
 - Windows CardSpace (more in 2.3)
- Resources for taking technology suitability assessments to the next level



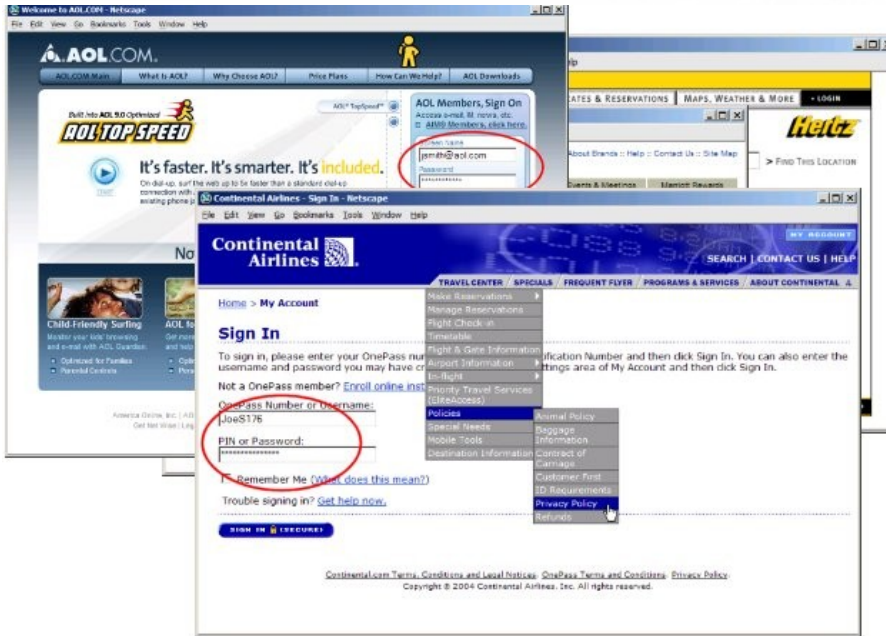
Opportunities, risks, use cases, and challenges for federated identity



Enterprise identity has grown out of directories



At the same time, the web identity experience has become personal



Inbox

There are messages requiring your attention. [View messages.](#)

Network updates since May 21

Questions from your network:

[ask your network a question](#)

Asked by friends of friends:

- Do you have low cost manufacturing connections for a hair...
- X-ray degrades IC performance?
- What the current Travel 2.0 applications and what kind of...

[View more questions](#)



Browse
Books, Music & Movies

Get It By Wednesday
Why wait for [London: The Novel](#)

Eve's Plog™ • 25 new posts

Federated means highly distributed: the good...

Excellent! I can add partner sites quickly and still retain control over user info!...

Identity provider
(login site)

Great! I can delegate user authn and mgmt tasks, like password resets!...

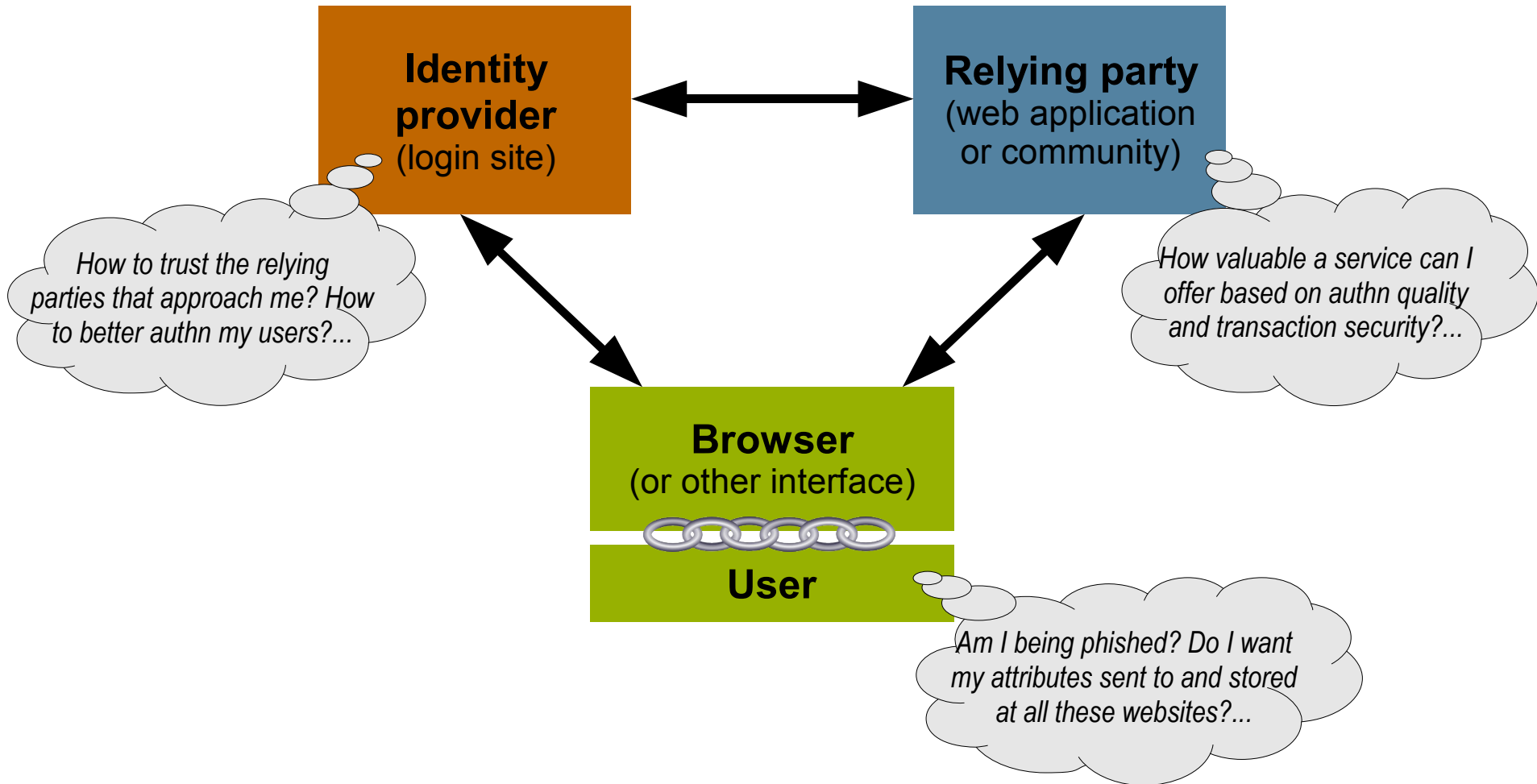
Relying party
(web application or community)

Browser
(or other interface)

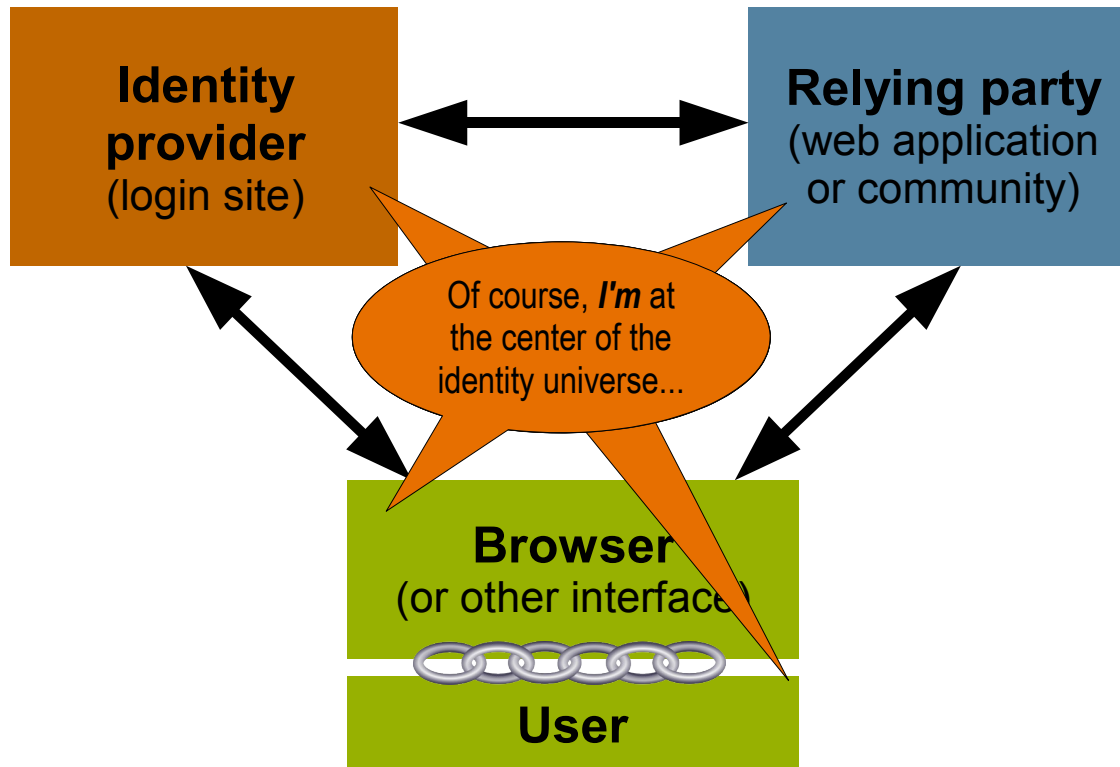
User

Cool! I can log in only once and still get to use multiple websites!...

...the bad...

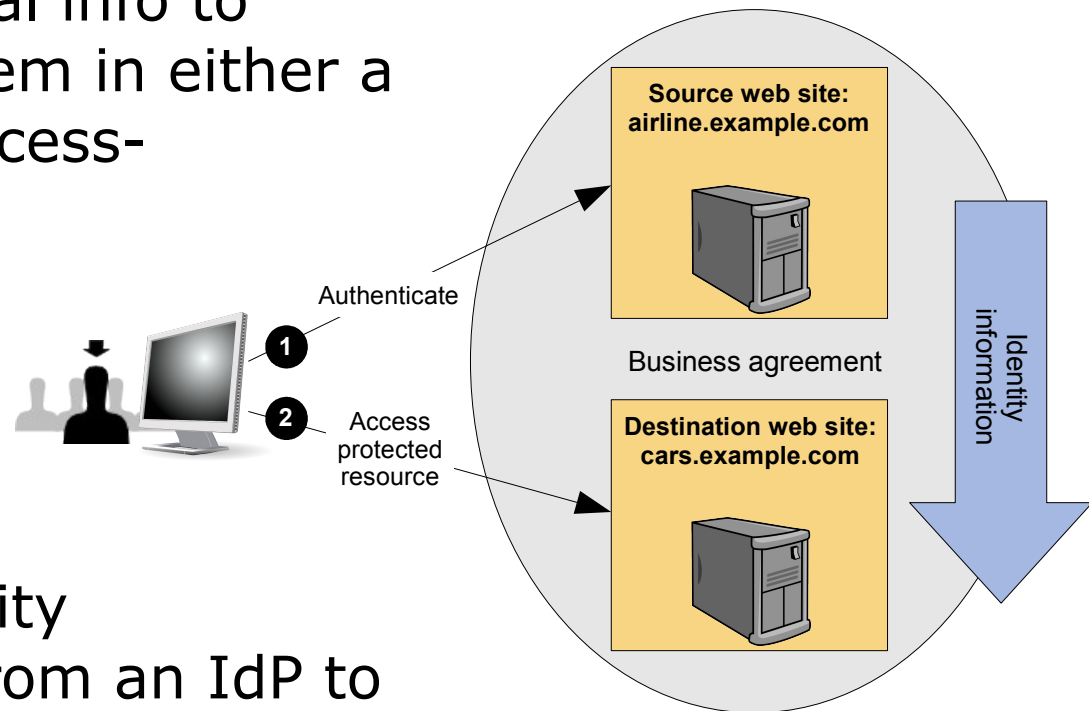


...and the ugly



Cross-domain single sign-on: the “classic” use case

- Strictly defined, it allows a user to authenticate at a *single* site and provide *zero* additional info to *other* sites, to use them in either a personalised or an access-controlled way
 - “Reusing” the same identifier, act of authentication, and login session across multiple sites
- It involves your identity information *flowing* from an IdP to an RP, somehow
 - Many different flow patterns, to satisfy different circumstances



Some examples found in the wild

- **Typepad** requires blog-commenting logins
- **Windows Live ID** logs you in to accepting apps
- **Sun** outsources human resource web apps
 - Logged-in employees gain access to the external HR site
- **T-Online** Net ID-card customers log in once for T-Pay, Meine T-Mobile, T-Com Rechnung Online...
- **Mon.Service-Public.fr** lets French citizens manage government services from a dashboard
 - Health care expenses, job agencies, tax administration...
- Yours?

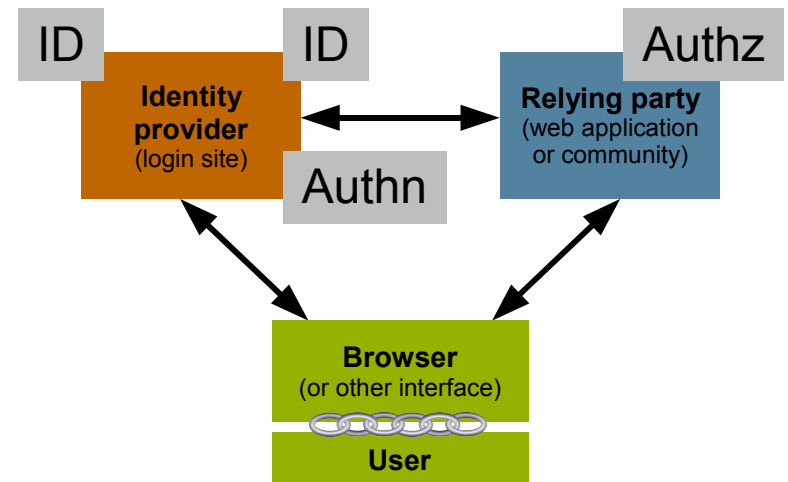


Representing identity data: a familiar challenge

- Using XML (or not)
 - What schema(s)?
 - How to represent user attributes?
 - Remember to keep the meanings of “attribute” straight!
- Messaging models
 - How to transport the data and in what pattern?
- In federated identity, exchanging data *securely* has added importance
 - Is the channel secure?
 - Can we be sure who the message sender was?
 - Can we ensure only the recipient can read it? (one form of ensuring privacy)
 - Can we apply other application-level protections?

Drawing some close distinctions

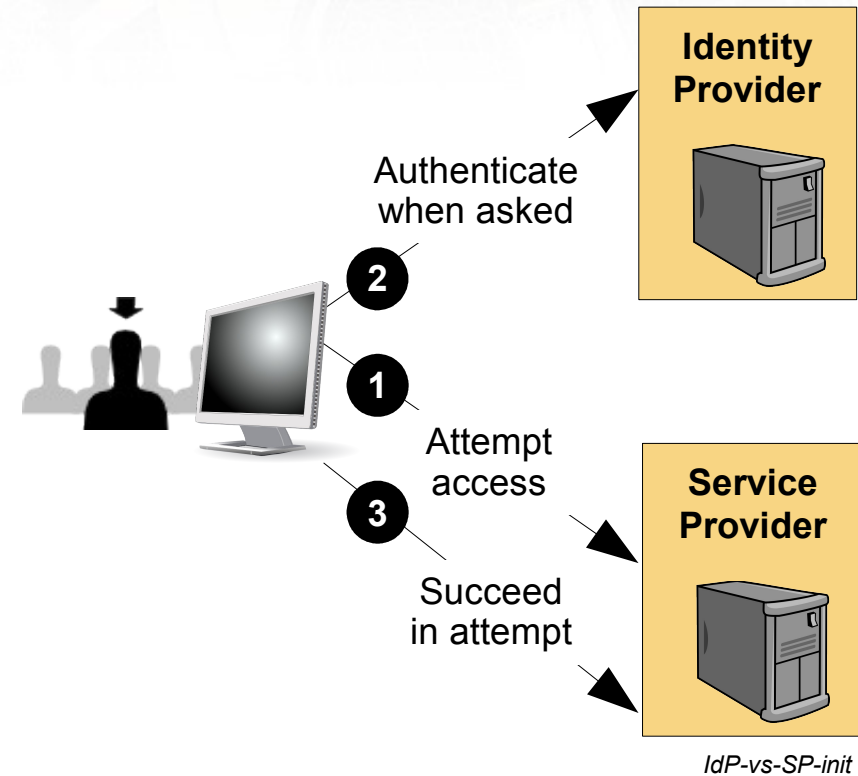
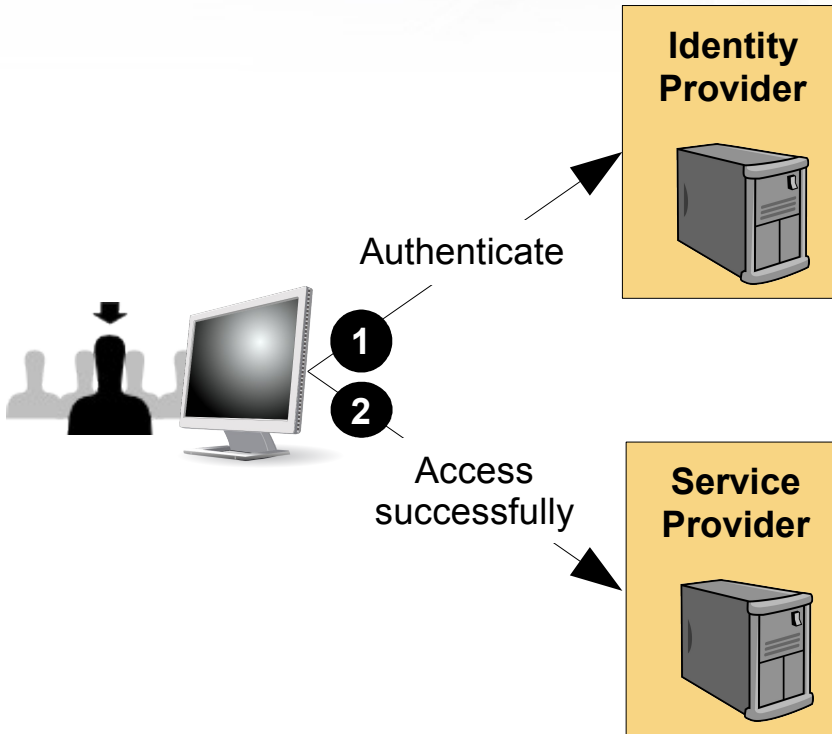
- **Identification:** when they “know” your identity
 - Its mapping to “you in the real world” can vary
 - Sometimes a strong mapping is undesirable or illegal
 - People, groups of people, and non-people can have identity
- **Authentication:** verifying your credentials
 - The method can be weak or strong
 - An RP could be satisfied knowing you've been authenticated without requiring a strong identification mapping
- **Authorization:** a decision an RP makes about which info resources to let you use and actions to let you take
 - Its judgment can be based on anything at all



A bit about identification schemes

- To talk about you, apps often use an identifier – a unique value in some namespace
 - Account IDs: employee numbers, government numbers, customer numbers...
 - Data network IDs: local, IP, email, IM, DNS, URIs, IRIs, and now XRIs (more on them in 2.3)
 - Other network IDs: phone numbers, postal addresses
- Substitutes for these can be used to weaken your identification
 - More about this anon
- Sometimes apps use one or more non-unique attributes to describe a class you're in or a role you have
 - Ephemeral and best for temporary (session-length) use

Two common SSO patterns



- Lois logs in to mobile telco operator IdP
- Lois gets directly into account bill-viewing application SP (RP) on a site at a separate domain
- Lois tries to get into account bill-viewing application
- It asks mobile telco operator IdP to check her out before letting her see sensitive account information

Some other SSO variations

- **SSO plus attributes:** The IdP sends along additional info about you to the RP
- **Step-up authentication:** The RP sends you back to the IdP for stronger authentication so it can authorize you for a more sensitive operation
 - Example: looking at your stock portfolio vs. trading shares
- **Simplified sign-on:** The process can be made smoother but not perfectly smooth
 - You might have to hint where your IdP is
- **Single logout:** Ending all local sessions linked to a shared login session

Select an Identity Provider

In order to fulfill the request for a web resource you have just attempted to access, information must be obtained from your identity provider. Please select the provider with which you are affiliated.

Choose from a list:

or

Search by keyword:

Identity provider discovery: an architectural challenge

- When SSO is RP-initiated, how does it find the IdP?
- Obvious choices:
 - The user tells the RP
 - By choosing from GUI selections
 - Assumes an RP-IdP federation (**circle of trust** or **CoT**)
 - Or by directly typing in a location
 - The RP is configured to know the IdP already
 - Assumes a CoT again
 - The client device can tell it
 - Through a cookie
 - Or by having extra “smarts” (**identity agent**)
 - The client device *is* the IdP!

Identity
provider
(login site)



Relying party
(web application
or community)

Web authentication: a leaky challenge

- Username/password authentication is weak and phishable
 - Used widely for both local and federated logins
 - COTS browsers are a huge vulnerability
- What helps?
 - Mutually authenticating user↔website
 - Stronger authentication
 - Increased user sophistication around using web UIs
- Smart clients (identity agents) can help but have their own challenges
 - Much stronger authentication is possible
 - But client-dependent solutions can limit portability, become an attractive single point of failure, and be hacked themselves

Phishing

From Wikipedia, the free encyclopedia

In [computing](#), **phishing** is a [criminal](#) activity using information, such as usernames, [passwords](#) and communication. [eBay](#) and [PayPal](#) are two of the most frequently targeted sites by email phishing campaigns.

Account linking: the “life is messy” use case



The screenshot shows the homepage of the French public service portal. At the top, there is a search bar with the text "Tapez les mots de votre recherche" and a search button. Below the search bar, there are navigation tabs for "Particuliers", "PME", "Professionnels", and "Citoyens". The main content area is divided into several sections:

- Bienvenue dans l'espace particuliers**: A central banner with the text "Nos droits et démarches" and icons for "Papiers", "Formation", "Déménagement", "Couples", "Enfants", and "Succession".
- Actualités**: A section on the right with news items such as "CONSUMMATEURS - Soldes d'été 2007", "DOSSIER - Elections législatives 2007", "IMPÔT REVENU - Déclaration en ligne", "INTEMPÉRIES - Etat de catastrophe naturelle", "PRÉVENTION - Notices des médicaments", "HALDE - Ecoles", "ECHÉANCE - ISF", and "DECOUVRIR - EVÉNEMENT - fête de la musique 2007".
- Formulaires**, **Démarches en ligne**, and **Adresses utiles**: Three tabs at the bottom of the main content area.
- Se documenter**: A sidebar on the left with links to "Sites internet publics", "Annuaire de l'administration", and "Travailler dans l'administration".

At the bottom of the page, there are links for "Mentions légales", "Presse", "Avis et suggestions", "Statistiques", "Plan du site", "Informations sur service-public.fr", "Faire des liens vers service-public.fr", and "Co-marquage". The footer also includes "© La Documentation française".

- You're a French citizen with a Mon.Service-Public.fr login
- But you already have accounts with the health agency, tax agency, motor vehicle bureau...
- The portal and each agency need to learn which account pair is “yours” and agree on a way to refer to that link for joint operations
 - Like the dashboard or SLO
- The “hub identifier” the portal uses for you might be good enough for this purpose...or it might not

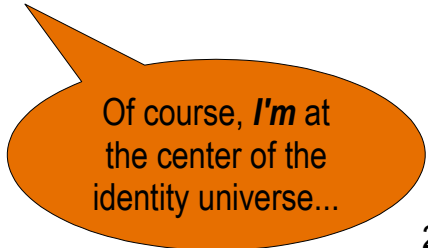
Sharing personal data: a privacy challenge



- Identifiers may constitute **personally identifiable information (PII)**
 - Account IDs and network IDs can strongly identify you!
- Instead of a “veronym”, each RP might only ever see a pseudonym
 - So that multiple RPs can't get together to correlate your activities
- Sharing some attributes, or enough together, can identify you too
 - Email addresses, postal addresses...
 - And may be essential to the functioning of the RP, assuming you want its service

User centricity type 1: the “me generation” use case

- You're tired of self-registering three new logins every week
 - For such trivial uses! – commenting on blogs, editing wikis, logging in to conference websites, playing with new Web 2.0 apps
 - These aren't exactly exclusive clubs
- You want to be the ultimate arbiter of your own digital identity(ies)
 - Hosted by your own website and portable to wherever you say
 - Serving up to RPs only what you let them share
 - Letting others know which activities on the web are truly yours

An orange speech bubble with a tail pointing towards the top left, containing the text: "Of course, *I'm* at the center of the identity universe..."

Of course, *I'm* at the center of the identity universe...

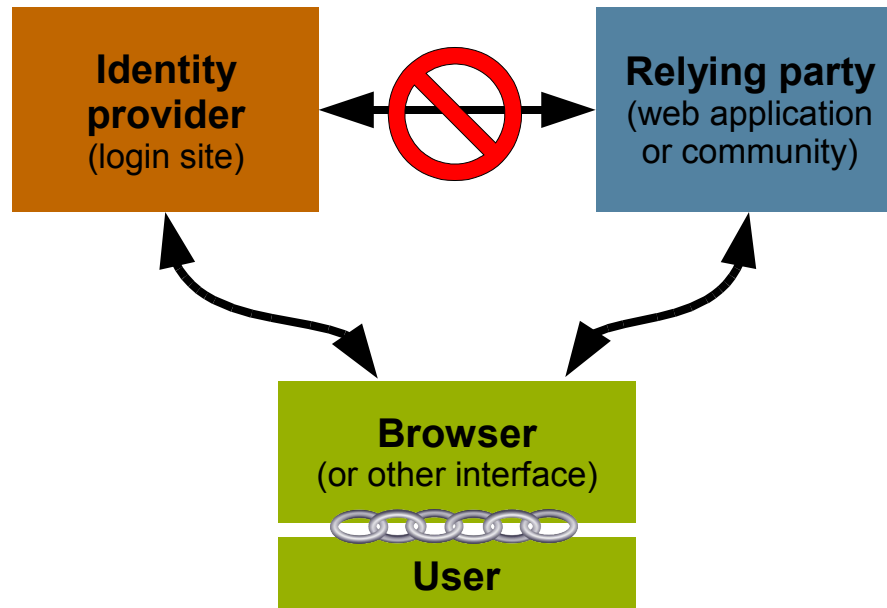
Self-assertion: a credibility challenge

"Oh yeah?! Says who?"

- No one is obliged to believe information you provide about yourself
- Ever lied when filling out a "required" field in yet another account registration form?
- It's more powerful when another is willing to vouch for you

User centricity type 2: the “trust no one” use case

- Even your trusted IdPs should never know what RPs you visit
- All information, even if vouched for by others, must be routed through an identity agent under the user's direct control



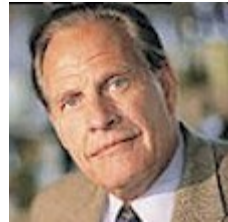
User as intermediary: a lonely challenge



- You (with a suitably clever agent) “must be present to win” service from the RP
- UI ennui is a security vulnerability all by itself
 - Having to press Enter, even (or especially) when the defaults are reasonable, trains us not to pay attention
- Accumulating state on one client device may confine you (in practice) to that device
 - Erecting synchronisation barriers to using other devices

User centricity type 3: the “do what I mean” use case

- People are fickle! Sometimes...
 - ...we want others to see/use data about us, or not
 - ...we want to grant permission explicitly, or not
 - ...we want to “set it and forget it”, or not
- User *consent* is a hallmark of privacy-sensitive identity systems
 - But users hate interruptions
 - Why would anyone want SSO otherwise?



- User *control and empowerment* are a more sophisticated recognition of user desires
 - Both the real-time *and* the pre-set kind
 - Useful even for life-or-death scenarios: the “break glass” scenario

Governance: an elephantine challenge

- All the parties have rights, responsibilities, capabilities, and requirements
- How do these get expressed and enforced?
- For valuable data and transactions, the nontechnical portion is 70-80% of the work
 - Contracts, contextual policies, and SLAs
 - Legal liability when something goes wrong
 - Regulations and laws
 - Certificate management between online partners




Tomassi

"How long has **THAT** been there?"

Summary so far

- Distributing identity gives new service-oriented options – but online identity can be risky
- Use case categories:
 - **Federated identity:** SSO (and variations), single logout, attribute exchange, account linking, ...
 - **User centrality:** types 1, 2, 3
- Challenges:
 - The formats, conveyance, and security of identity data
 - Web authentication methods
 - Smart clients and their demands
 - Identity provider discovery
 - PII sharing and other privacy concerns
 - Self assertion vs. vouched-for information
 - The governance aspects of federated identity



Examining three technologies you need to know about

The Venn of identity

IdP- and enterprise-friendly, heavier, codifies existing practices but flexible for many purposes

SAML

- Comprehensive use case coverage
- Comprehensive challenge solutions, except IdP discovery
- Can be deployed to do any user centricity type

RP-friendly, lighter, less concerned with security, proposes discontinuous change à la the Web itself

OpenID

- Simple use case coverage
- Strong on IdP discovery but weak on other challenges
- The very definition of “me generation”

Client-centred, makes security the uppermost concern, can front SSO of many types through APIs

CardSpace

- “Smart client” component
- Addresses web authentication challenges
- The very definition of “trust no one”

“Me generation”

“Do what I mean” philosophy

“Trust no one”, XML message formats

Consistent user experience, “me generation” in part

The basics of SAML

What is SAML?

- According to its designers, it is:

“an XML-based framework for marshaling security and identity information and exchanging it across domain boundaries”

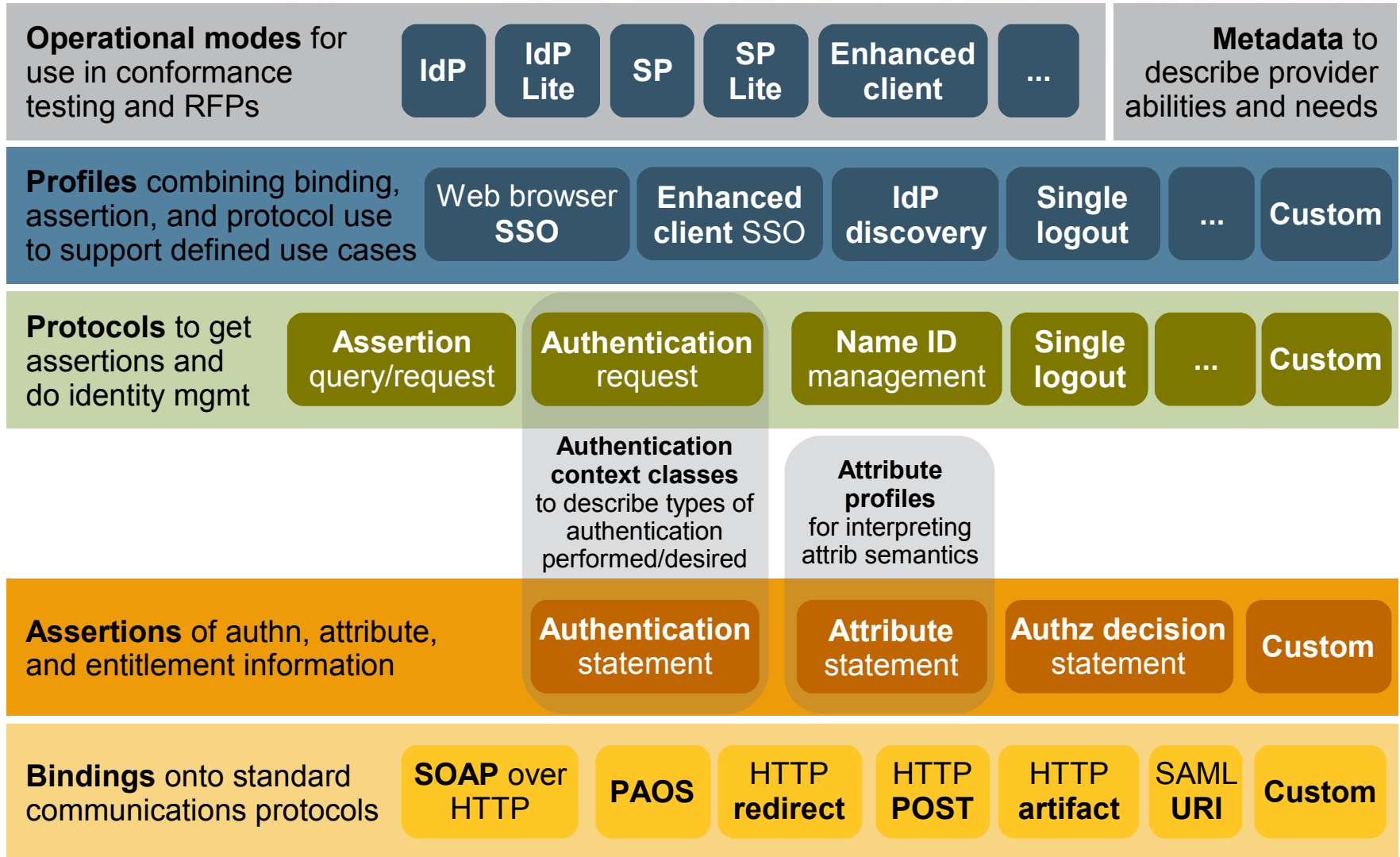


- Strives to be the “universal solvent” of identity
 - Especially SAML V2.0 – based on Liberty ID-FF
 - Has out-of-the-box profiles for interoperability, but can be extended and profiled further
- Driven primarily by serious scenarios where trust, liability, value, and privacy are at stake
 - B2B, B2C, G2C...
- What sorts of adopters does it have?
 - Governments, telcos, financials, aerospace, Google Search Appliance...

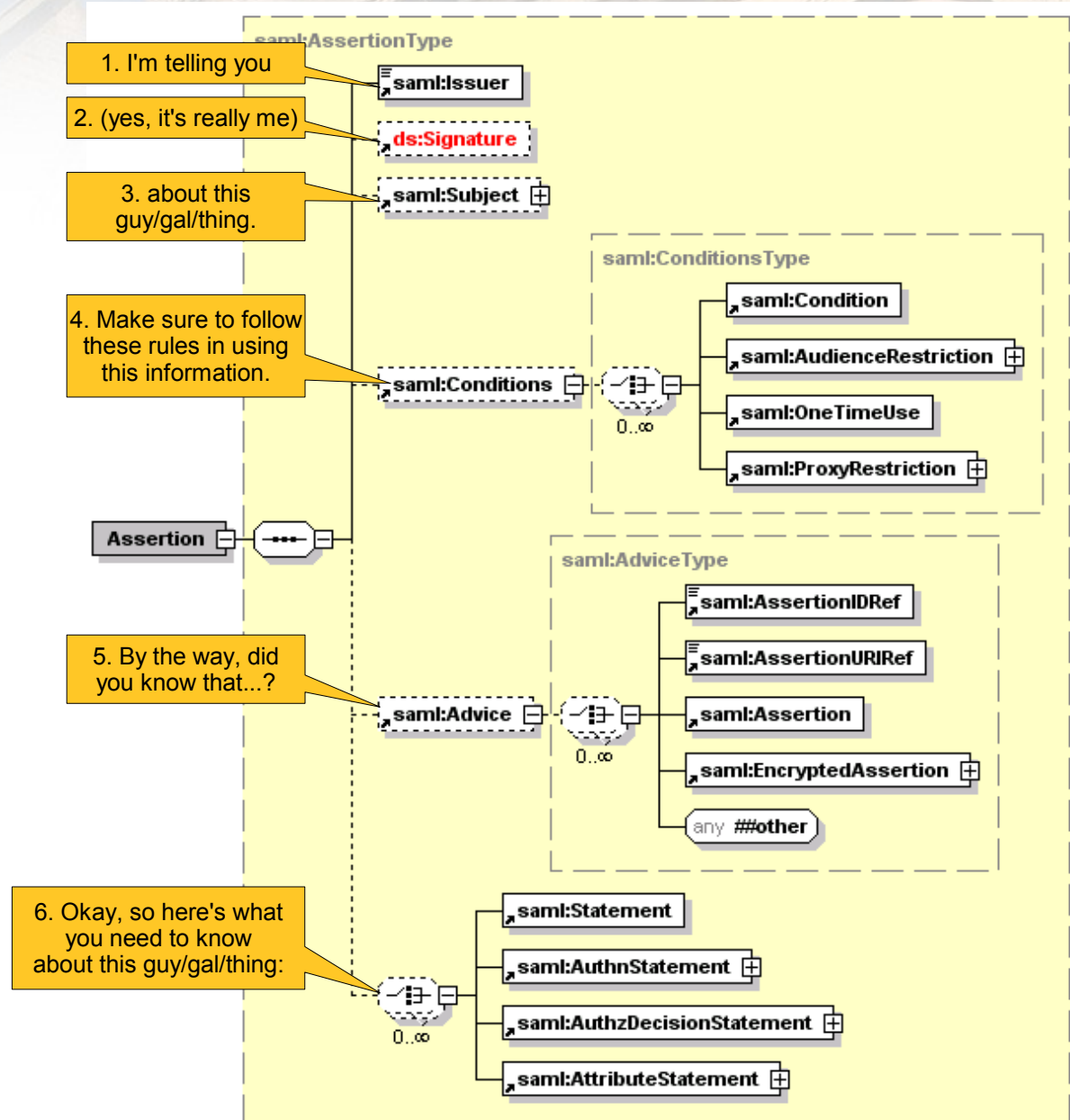
At SAML's core: assertions

- An assertion is a declaration of fact...
 - ...according to someone
 - You have to determine if you trust them
- SAML assertions contain one or more statements about a subject:
 - **Authentication** statement: “Joe authenticated with a smartcard PKI certificate at 9:07am today”
 - **Attribute** statement (which can contain multiple attributes): “Joe is a manager and has a £5000 spending limit”
 - **Authorization decision** statement (use XACML instead for more than simple needs here)
 - Your own customised statements...

Assertions in the bigger SAML framework picture



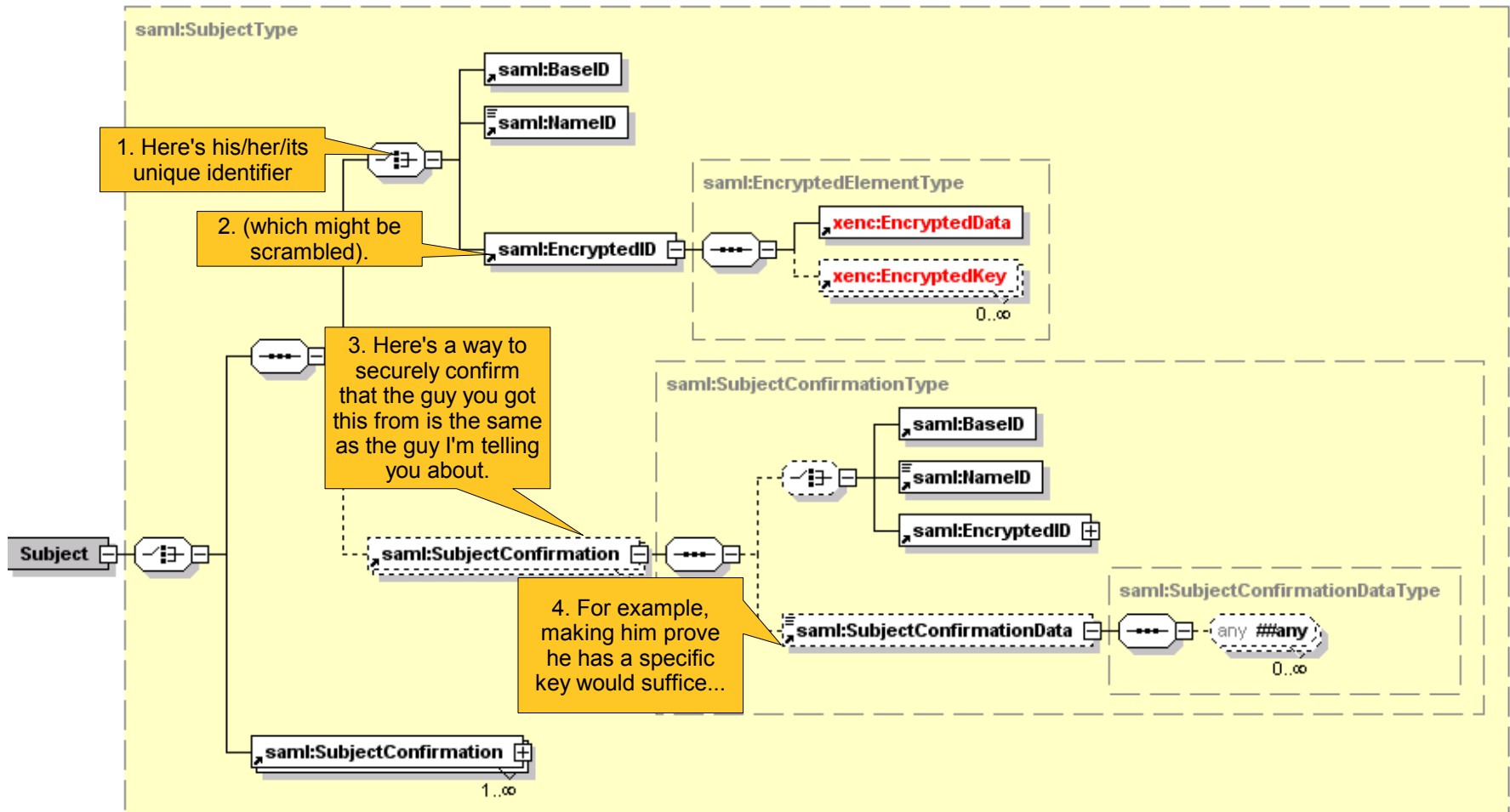
Overall assertion element structure



Example of an assertion's common portions

```
<saml:Assertion
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  Version="2.0"
  IssueInstant="2007-07-27T14:31:05Z">
  <saml:Issuer>
    www.example.com
  </saml:Issuer>
  <saml:Subject>
    <saml:NameID
      Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">
      J.Handy@example.com
    </saml:NameID>
  </saml:Subject>
  <saml:Conditions
    NotBefore="2007-07-27T14:31:05Z"
    NotOnOrAfter="2007-07-27T14:36:05Z">
    </saml:Conditions>
    ... statements go here ...
  </saml:Assertion>
```

Subject element structure



Example of an authentication statement

```
<saml:Assertion ... common info goes here ... >
  ... and here ...
  <saml:AuthnStatement
    AuthnInstant="2007-07-27T14:30:55Z"
    SessionIndex="0">
    <saml:AuthnContext>
      <saml:AuthnContextClassRef>
urn:oasis:names:tc:SAML:2.0:ac:classes:SmartcardPKI
      </saml:AuthnContextClassRef>
    </saml:AuthnContext>
  </saml:AuthnStatement>
```



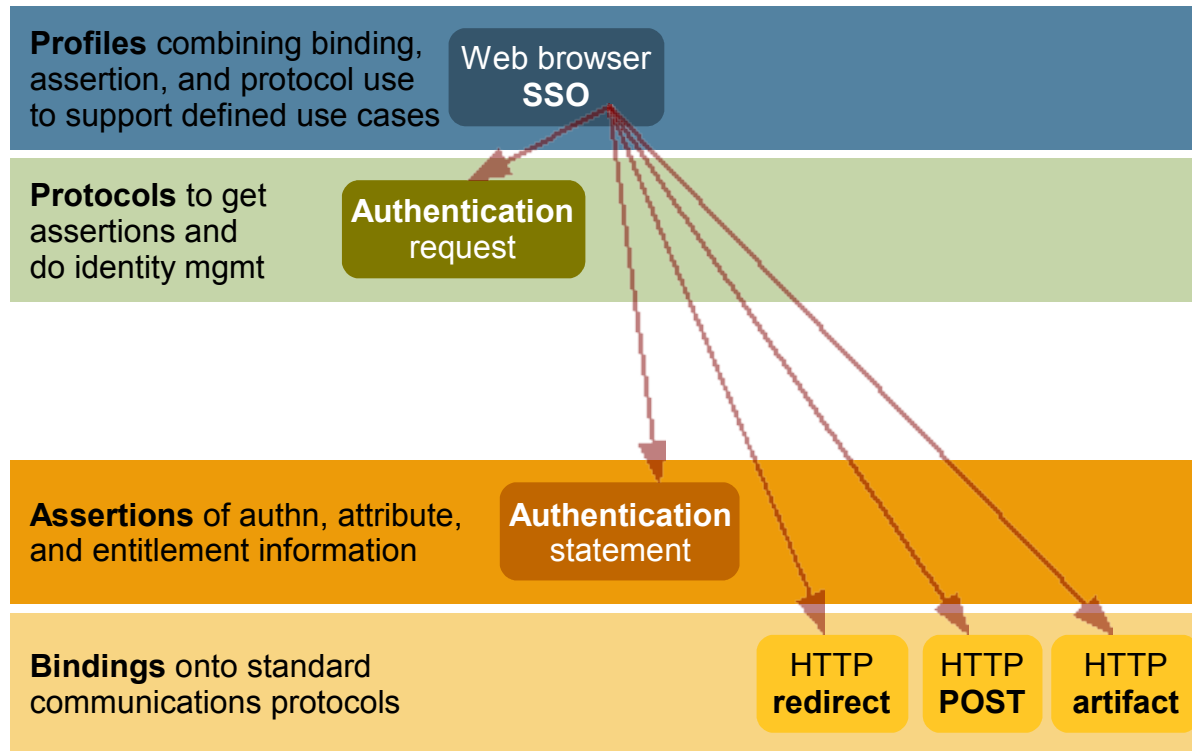
summer school

Authentication context classes

- Internet Protocol
- Internet Protocol Password
- Kerberos
- Mobile One Factor Unregistered
- Mobile Two Factor Unregistered
- Mobile One Factor Contract
- Mobile Two Factor Contract
- Password
- Password Protected Transport
- Previous Session
- Public Key – X.509
- Public Key – PGP
- Public Key – SPKI
- Public Key – XML Signature
- Smartcard
- Smartcard PKI
- Software PKI
- Telephony
- Nomadic Telephony
- Personalized Telephony
- Authenticated Telephony
- Secure Remote Password
- SSL/TLS Cert-Based Client Authentication
- Time Sync Token
- Unspecified
- Your own customised classes...

The most popular profile: web browser SSO

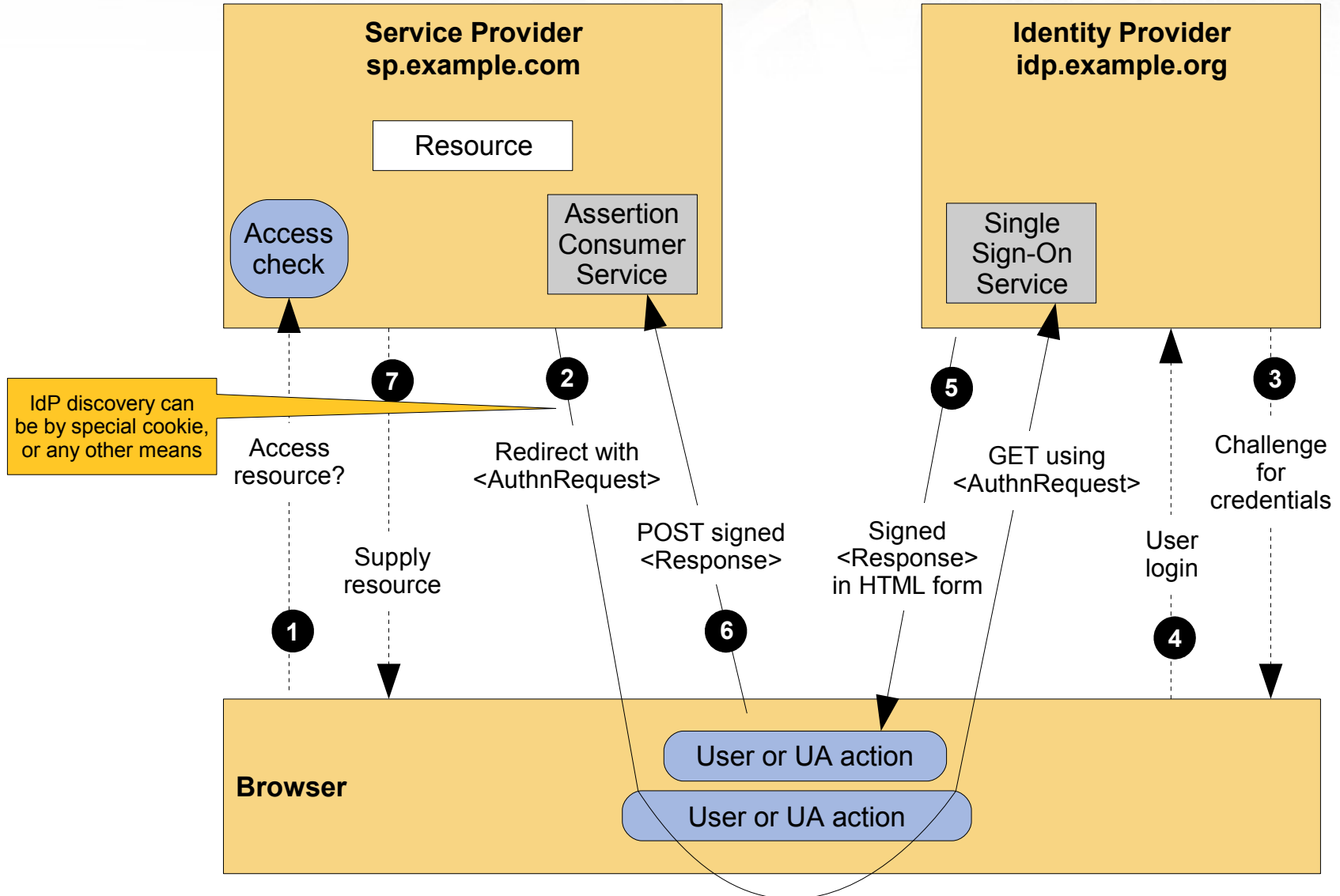
- It picks specific protocols, usage and interpretation of assertion statements, and bindings – but still offers flexibility
- This profile includes basic federation (account linking) but not ongoing name identifier management



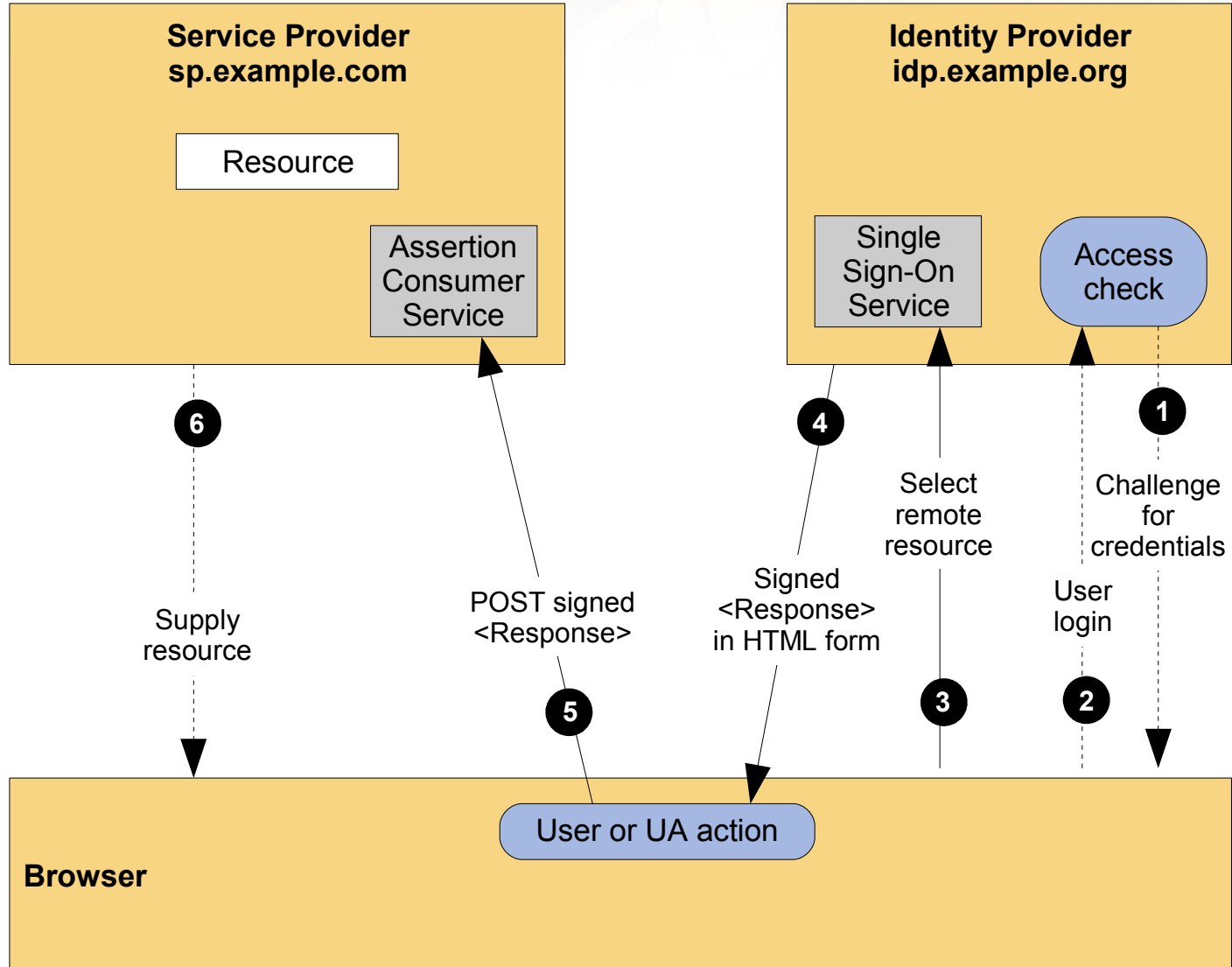
Flow options depend on bindings and initiation style

- IdP-initiated or SP-initiated SSO:
 - IdP-initiated SSO? (involves an unsolicited response)
 - SP-initiated SSO? (requires a request message first)
- Request message:
 - Pushed with HTTP POST?
 - Pulled with an “artifact”?
 - HTTP redirected?
- Response message:
 - Pushed with HTTP POST?
 - Pulled with an “artifact”?
- Let's see...carry the two...that's eight options
 - But some are more common than others

One very common option: SP-initiated/redirect/POST

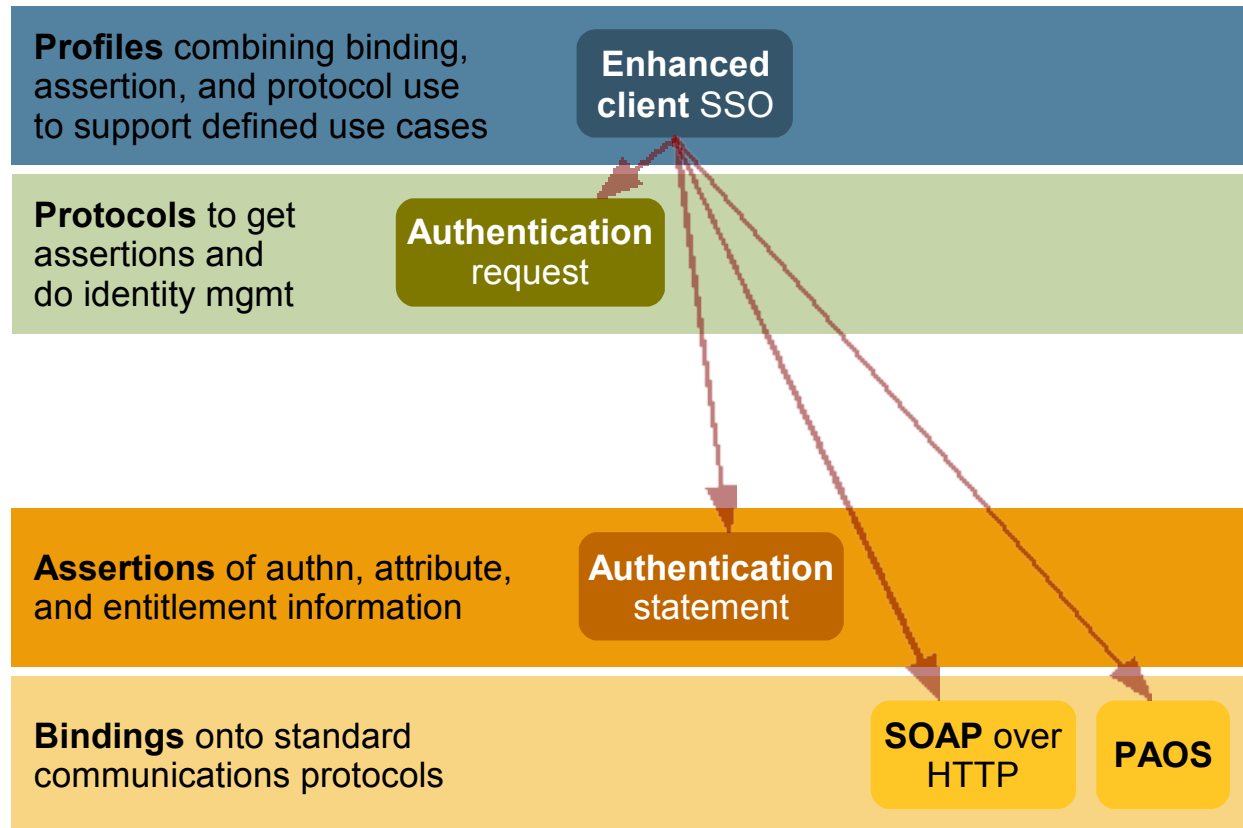


Another common option: IdP-initiated/POST

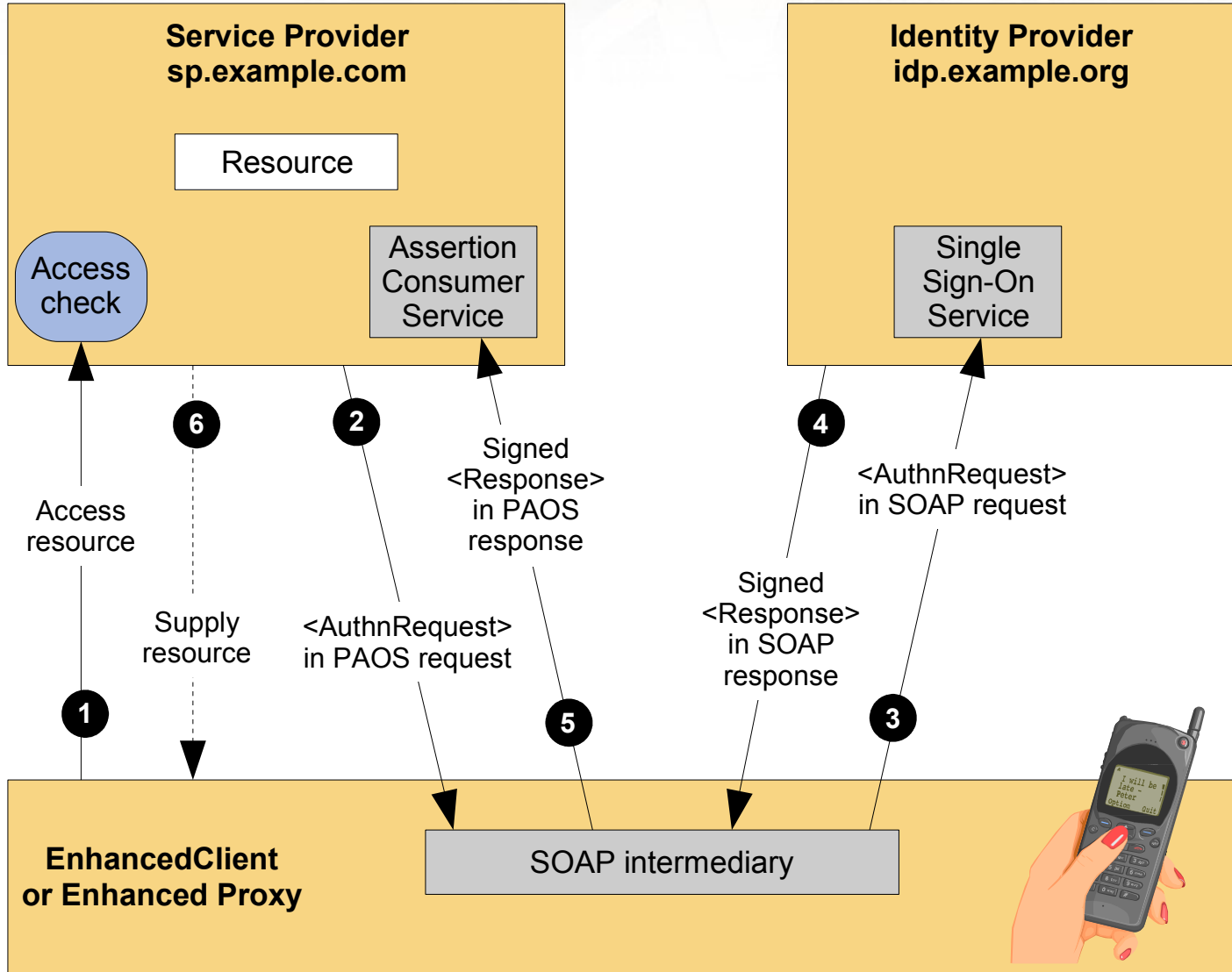


A related profile: SSO using an enhanced client or proxy

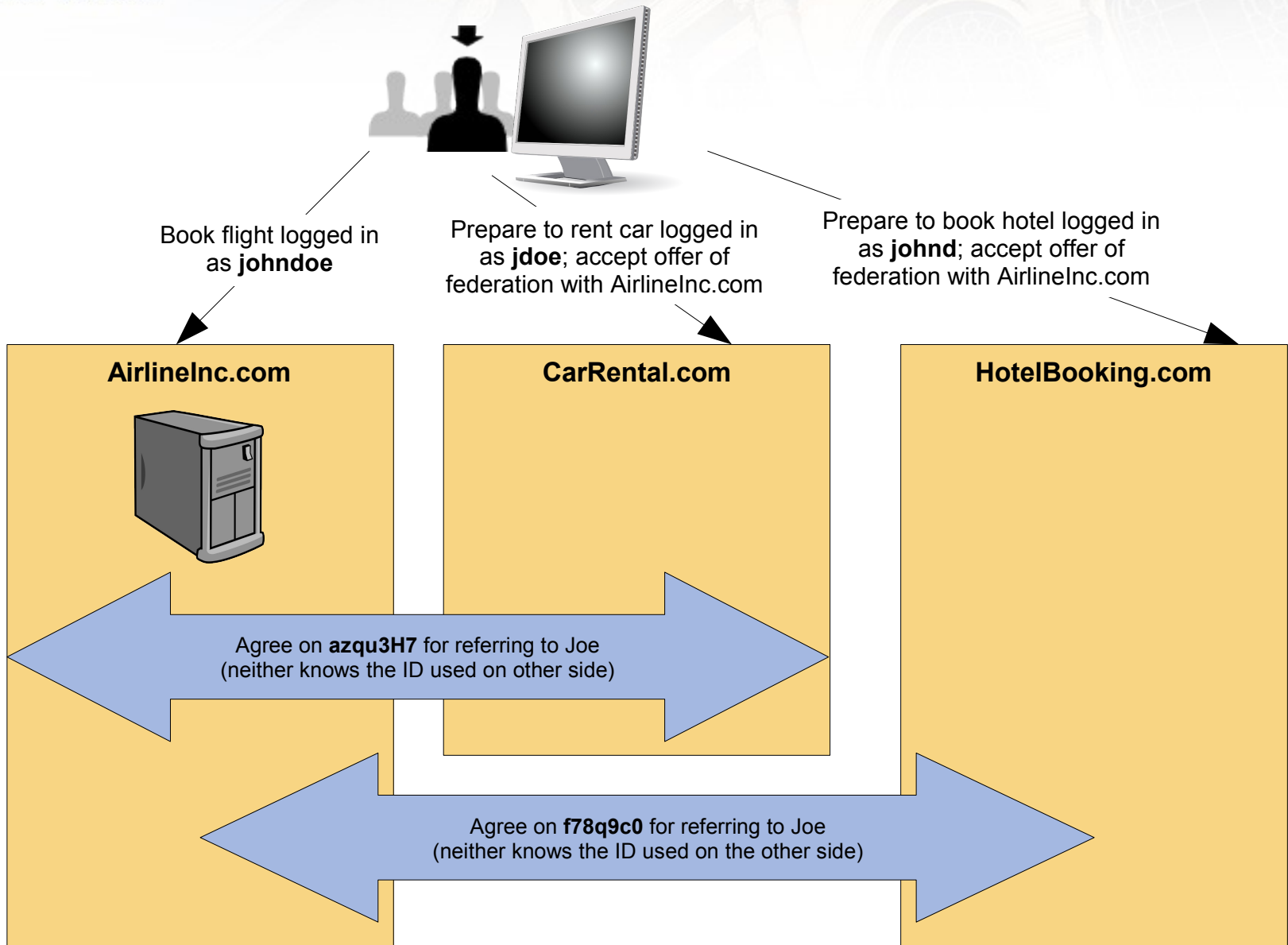
- An enhanced client has to be “smart” enough to know the IdP's location and use the reverse SOAP (PAOS) binding
 - Also discussed in 1.3



SSO using ECP

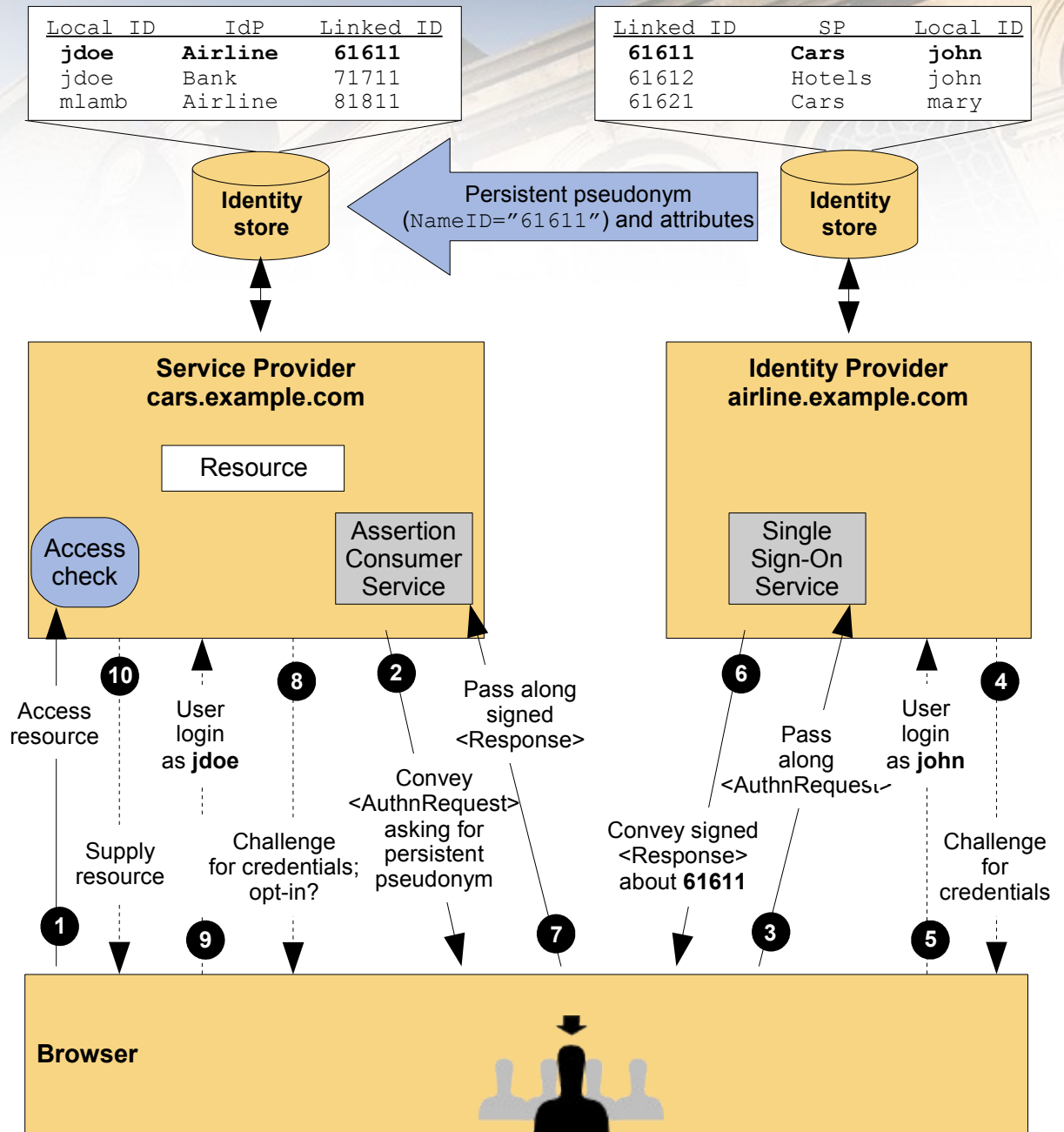


SSO plus account linking with pseudonyms, for privacy



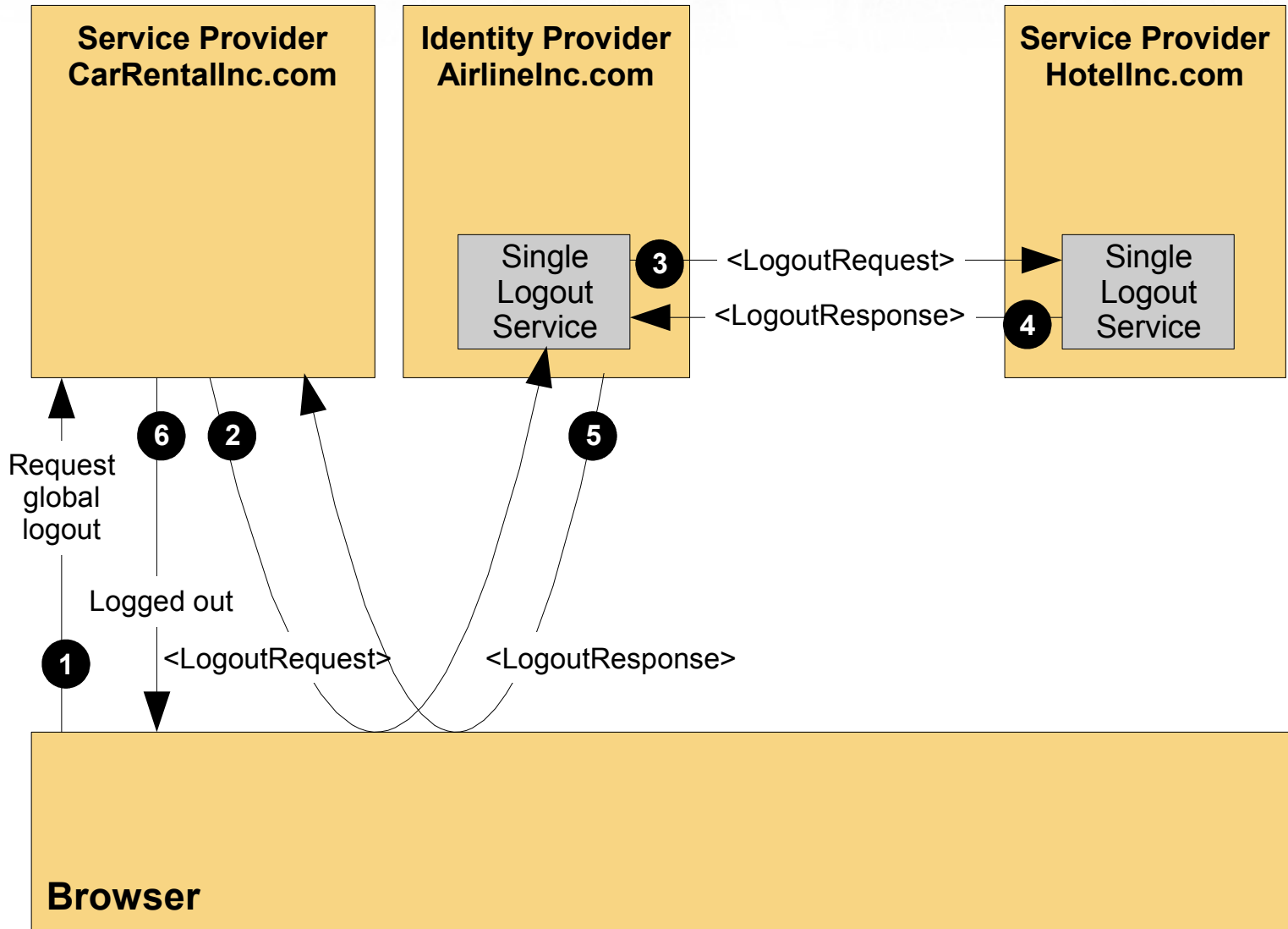
Account linking with a persistent pseudonym

Good for long-term relationships (vs. a pseudonym of the transient type)

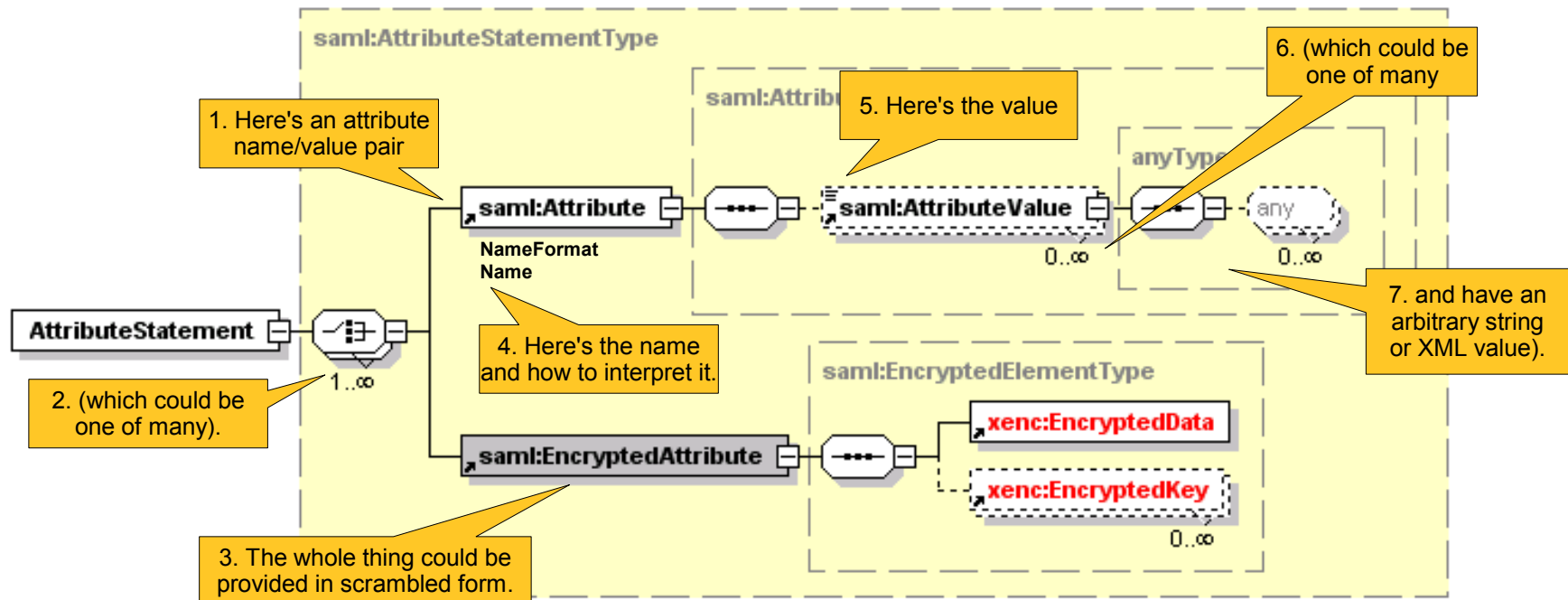


User with local ID **john** at airline.example.com
and local ID **jdoe** at cars.example.com

SP-initiated single logout with multiple SPs



Attribute statement element structure



Example attribute statement with a custom profile

```
<saml:Assertion ... common info goes here ... >
  ... and here ...
  <saml:AttributeStatement>
    <saml:Attribute
      NameFormat="http://example.com" Name="Role" >
      <saml:AttributeValue>repair_tech</saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
      NameFormat="http://example.com">
      Name="Certification"
      <saml:AttributeValue xsi:type="ex:type">
        <ex:CertRecord language="EN">
          <Course>
            <Name>Structural Repair</Name>
            <Credits>3</Credits>
          </Course> ...
        </ex:CertRecord>
      </saml:AttributeValue>
    </saml:Attribute>
  </saml:AttributeStatement>
</saml:Assertion>
```



summer school

Analysis: SAML

- **Federated identity use cases:** Covers a huge range; interoperability and flexibility but also complexity in messaging and data models for IdPs, RPs, and clients
- **User centrality use cases:** Has the *potential* to cover all three, depending on profiling and deployment choices
- **Identity data security:** Offers protection at channel, message, assertion, and application semantics levels
- **Web authentication:** Can convey strong-auth requests and assertions but also allows username/password (always weak)
- **Smart clients:** ECP option allows but does not require them
- **IdP discovery:** The only solution offered (cookie) is weak
- **Privacy concerns:** Strong on privacy of IdP-held data, depending on deployment choices; use ECP for the “trust no one” use case
- **Vouched-for info:** Strong, with strong auth of issuers
- **Governance:** Anticipates nontechnical aspects (Liberty Alliance offers lots of guidance); metadata covers some technical aspects

The basics of OpenID

What is OpenID?

- According to its designers, it is:



“an open, decentralized, free framework for user-centric digital identity”

- Deeply rooted in World Wide Web philosophy:
 - You identify yourself with a URL (or XRI) – a single universal namespace
 - Authentication consists of proving you “own” the corresponding web resource
- Deeply committed to Internet-scale adoption
 - Lots of scripty open source
- Driven by “Web 2.0” scenarios:
 - Blog commenting, contributing to wikis, social networking
- Accepted at, e.g. ...



LIVEJOURNAL™



How does OpenID work?

- An OpenID is simultaneously:
 - A **unique publicly known identifier** string by which your online activities can be correlated
 - A URL or XRI for some **machine-readable information** that redirects an “OpenID Consumer” site (RP) to your “OpenID Provider” site (IdP) – you can host your own or delegate to a chosen provider
 - Often, a URL or XRI for a **human-readable web page** about you
- The provider does authentication and may also send a small set of attributes set by you
 - Through the Simple Registration extension
 - Nickname, email, full name, date of birth, gender, postcode, country, language, timezone

Getting and using an OpenID

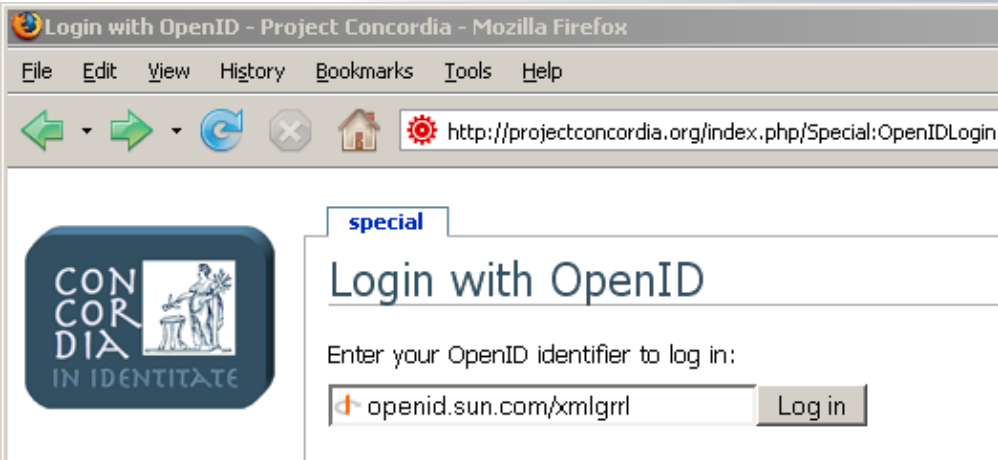
- You can register for a free one at various sites that offer authentication services, e.g.:
 - ProtectNetwork.com (also gives “SAML IDs”), MyOpenID.com, ProoveMe.com...
 - AOL users already have one, of the form `http://openid.aol.com/screenname`
 - Sun employees can get one from `openid.sun.com`
- You can host an authentication service on your own web server
 - E.g., on `xmlgrl.com` (theoretically!)
- You can use delegation to “chain” OpenIDs

Could be hosted, or delegated to (e.g.) ProoveMe

Sign in with OpenID [What is OpenID?](#)

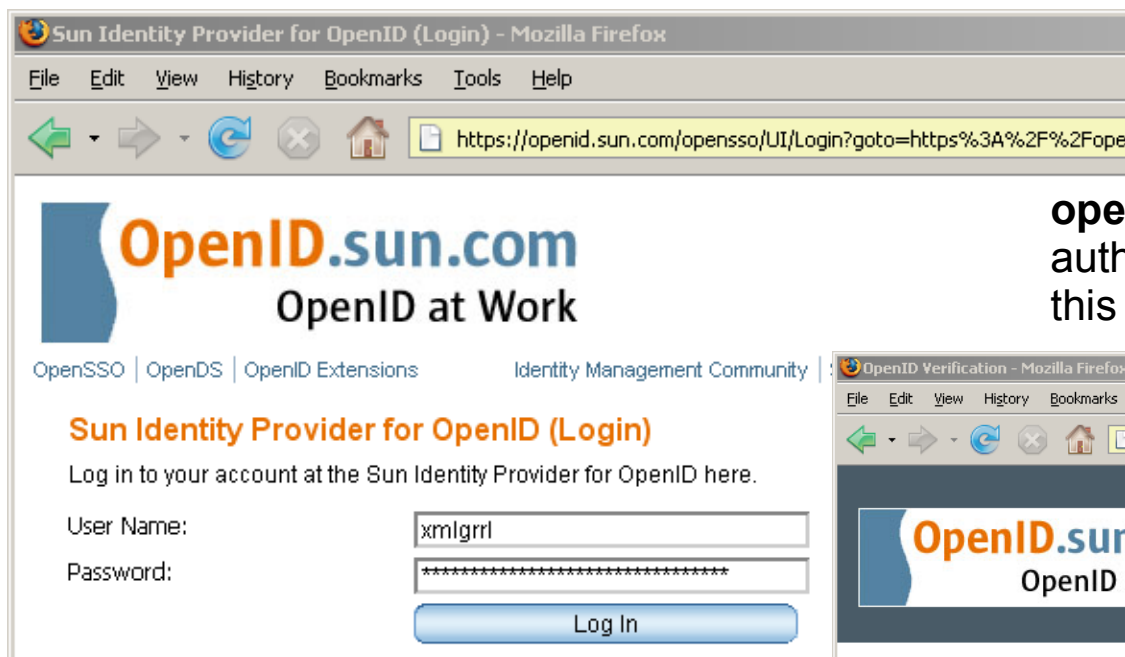
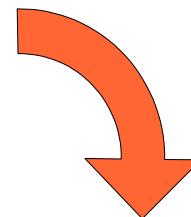
Goes directly to OpenID provider

Sign in with OpenID [What is OpenID?](#)



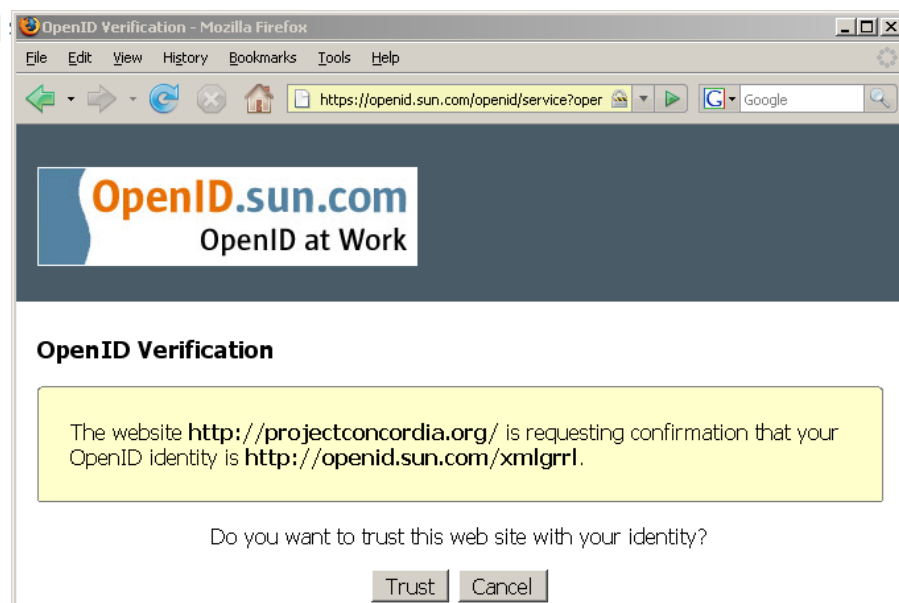
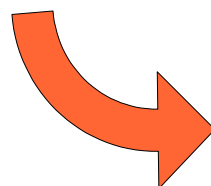
What the user sees: a demo

projectconcordia.org delegates authentication to my OpenID provider, **openid.sun.com**



openid.sun.com has me authenticate, proving I “own” this URL

openid.sun.com checks if I want to share my info with **projectconcordia.org** before getting automatically logged in there

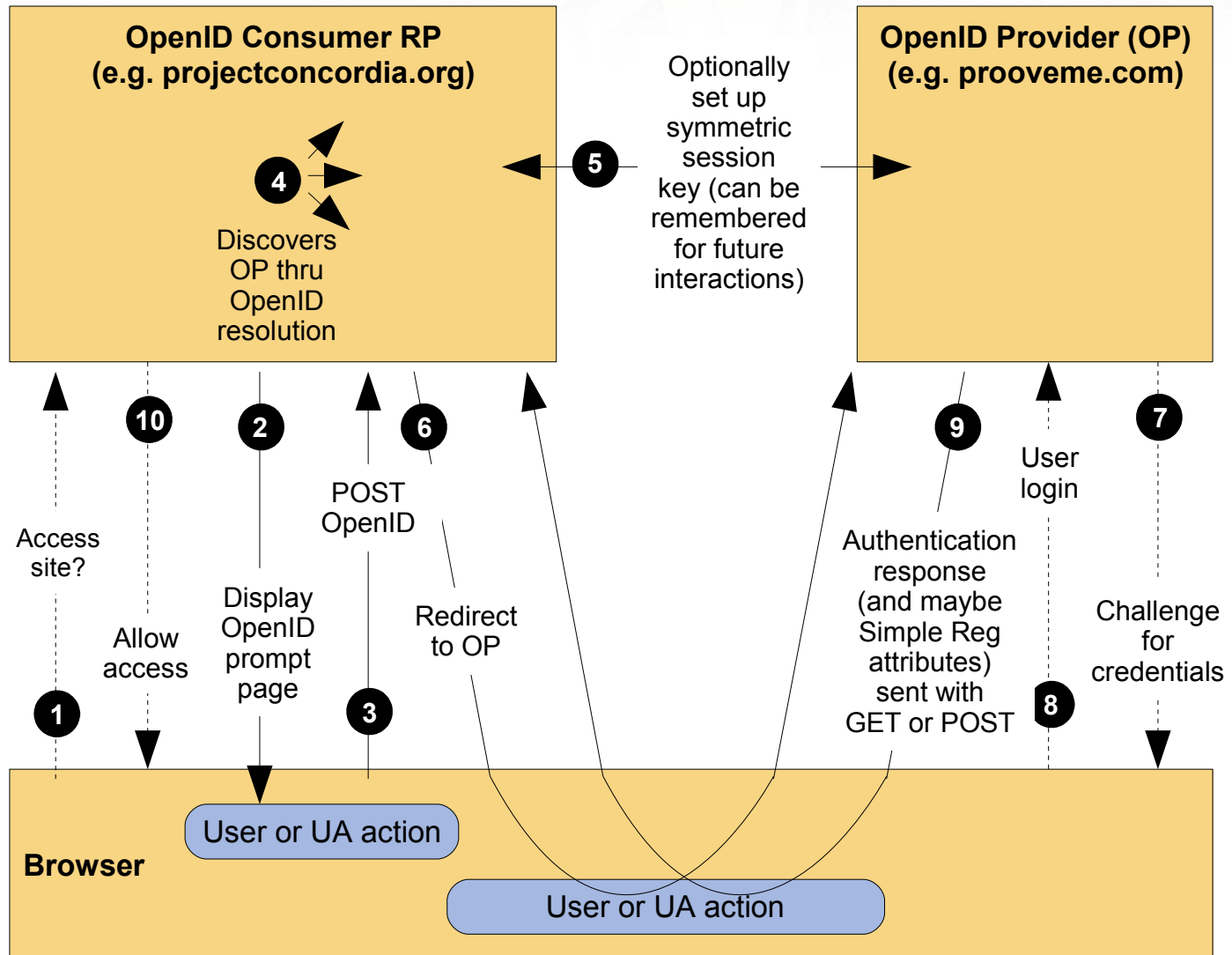


IdP discovery and other metadata through XRDS

- OpenID 2.0 uses the eXtensible Resource Descriptor Sequence metadata format from XRIs
 - More of that “machine-readable code” – your OpenID can resolve to an XML document of the **xrds+xml** media type
 - Helps the RP find the IdP authentication service and any other service endpoint metadata
 - Logical equivalent of a DNS resource record for a URI/XRI
- A locus for interesting web-friendly extensions

```
<xrd:Service>
  <xrd:Type xrd:select='true'>http://openid.net/signon/1.0</xrd:Type>
  <xrd:URI xrd:priority='1' xrd:append='qxri'>
    https://2idi.com/openid/
  </xrd:URI>
  <xrd:URI xrd:priority='2' xrd:append='qxri'>
    http://2idi.com/openid/
  </xrd:URI>
</xrd:Service>
```

SP-initiated simplified sign-on with OpenID



A protocol in transition

- New features in OpenID 2.0 (some also in 1.1 extensions; they may not be widely deployed):
 - XRI and XRDS support
 - IdP-initiated flow (requires it to know about specific RPs!)
 - Directed identity: user input of the provider's location rather than an OpenID
 - One-time generated OpenIDs as transient pseudonyms
- Features being proposed in additional extensions:
 - Arbitrary attribute exchange (name/value pairs)
 - Vouched-for attributes along with arbitrarily structured attributes
 - Ways to describe authentication methods desired/used
 - More to come, no doubt; XRDS can be used to advertise offered support for new features



summer school

Analysis: OpenID

- **Federated identity use cases:** Offers simplified sign-on and some attribute exchange; single unified ecosystem and emphasis on correlation (vs. access control) are most notable characteristics
- **User centrality use cases:** Covers “me generation” by design; is philosophically aligned with “do what I mean” (not “trust no one”)
- **Identity data security:** Offers optional channel protection, Diffie-Hellman message protection
- **Web authentication:** Agnostic as to method, but the username/password mechanism (weak) is common
- **Smart clients:** No native solution
- **IdP discovery:** Very strong – built into the identifier itself!
- **Privacy concerns:** Architectural and philosophical bias towards revelation (one-time OpenID feature could help a little)
- **Vouched-for info:** Weak; all info is self-asserted so far
- **Governance:** Makes a simplifying assumption to trust/serve all IdPs, RPs, and users – no nontechnical governance needed!; metadata extensible for future technical aspects

The basics of Windows CardSpace

What is Windows CardSpace?



- According to its designers, it is:

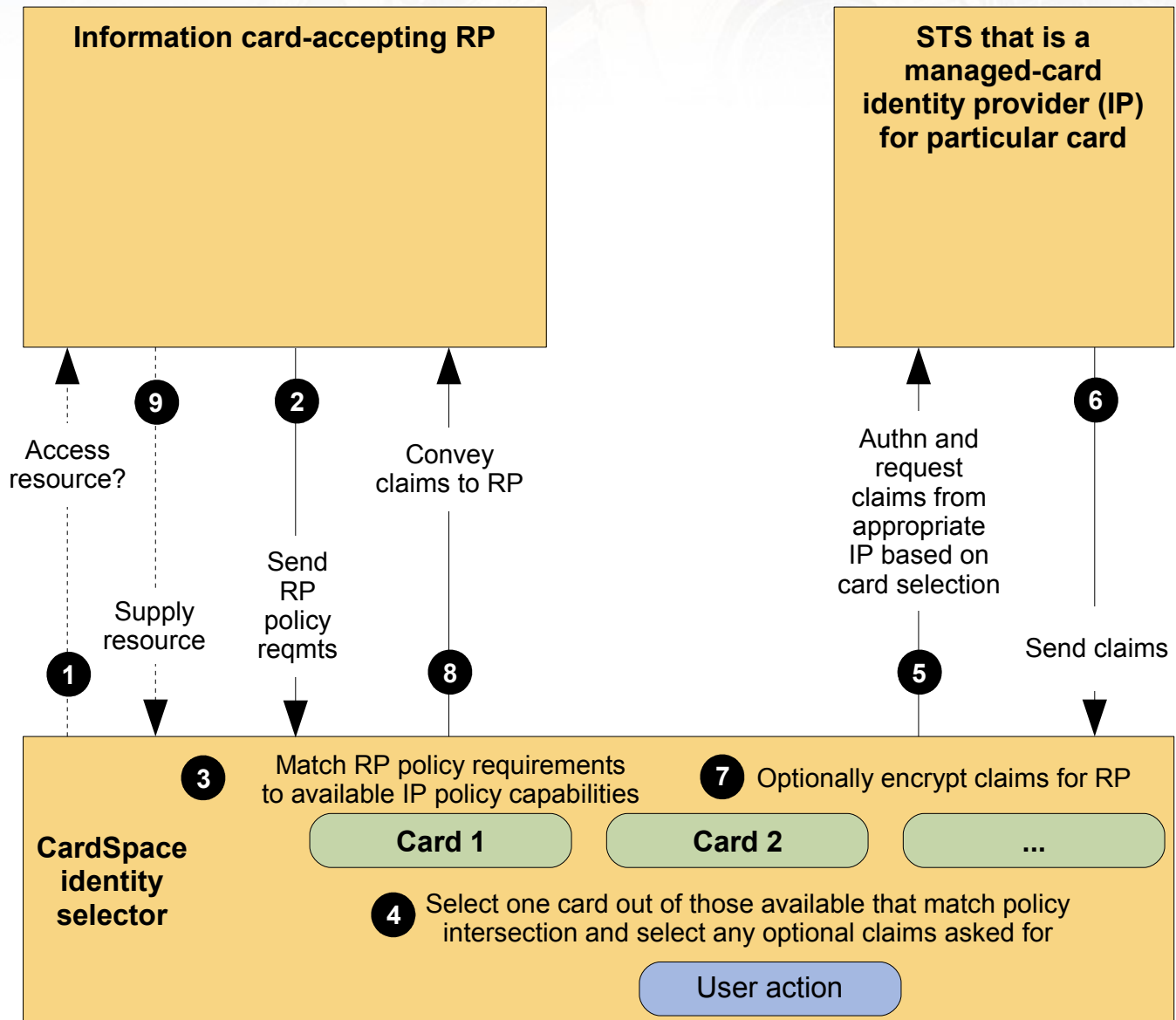
“a Microsoft .NET Framework version 3.0 component that provides the consistent user experience required by the identity metasystem”

- Uses software “cards” to let users manage identities
 - Card selector mediates a “trust no one” IdP/RP relationship
 - Serves up **claims** – authentication and attribute data – associated with a card
- Driven by web authentication security concerns
 - Hardened against tampering and phishing attempts
 - Prepared to tie closely into OS and hardware platform
 - Functions as an identity agent
- Accepted at, e.g. ... [Kim Cameron's Identity Weblog](#) [MIKE JONES: SELF-ISSUED Musings on Digital Identity](#) [SignOn.com BETA](#)

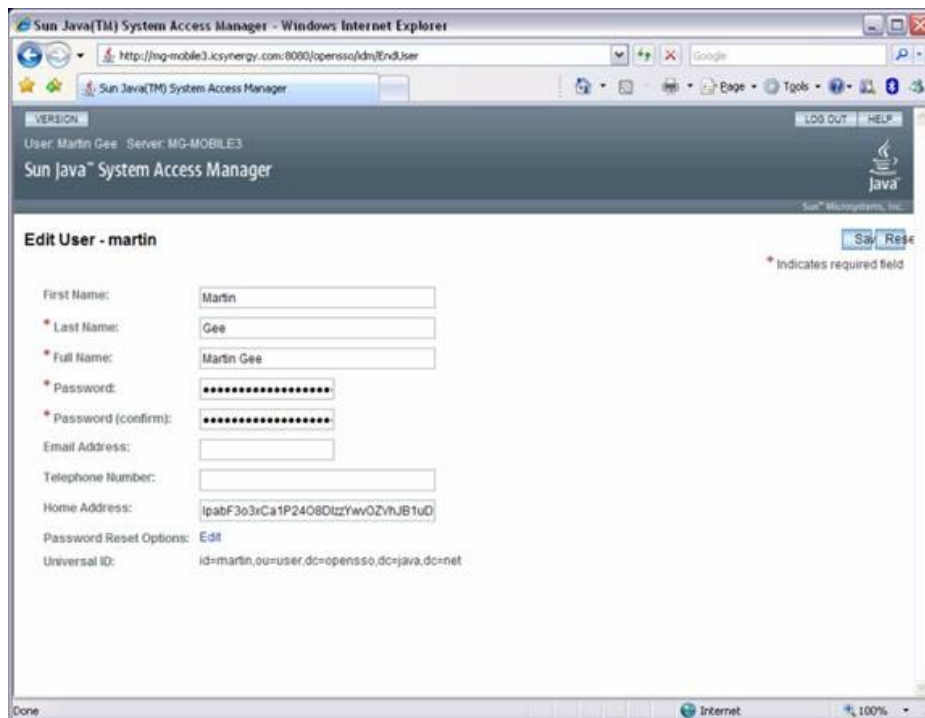
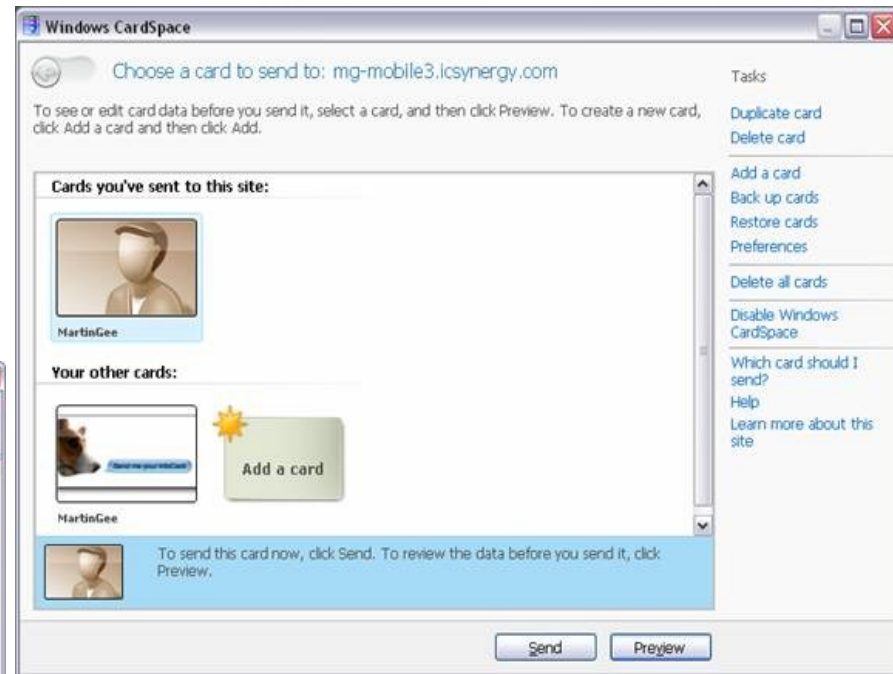
How does CardSpace work?

- You initially use the **identity selector** client component to:
 - Accept **managed cards** from IdPs (**security token services** or **STSs**) after having authenticated to them
 - Your card only *points to* claims made by the IdP; identifiers come from CoT-specific namespaces
 - Create **self-asserted cards** that store your own claims about yourself
 - The identity selector functions as an on-board IdP, with “profile management” features
- Later, when you access a card-accepting RP:
 - You **choose** from among your cards that satisfy the RP's and IdP's policy requirements/abilities

RP-initiated simplified sign-on with a CardSpace managed card



What the user sees



The mechanisms of “trusting no one”

- **Access housekeeping:** When you happen to be online, you delegate access to human and app requesters
 - For their use when you go back offline
- **RP pseudonyms:** The IdP can be made never to see the identifier of the RP in the clear
- **Client-centred:** You use a client device (ideally specially hardened) and special identity selector software
 - The Higgins project offers hosted identity selector web apps (H1, H2, H3 – not supported by CardSpace)
 - A hosted agent avoids smart-client challenges but compromises the goal

Analysis: CardSpace

- **Federated identity use cases:** Offers client-side support for authentication/simplified sign-on and attribute exchange
- **User centricity use cases:** Covers “trust no one” by design; is philosophically aligned with “do what I mean” (self-asserted cards are “me generation”-like)
- **Identity data security:** After initial card provisioning, uses PKI-based strong auth between client and IdP
- **Web authentication:** Client-based selector provides a stronger alternative to weak web authentication, once card is provisioned
- **Smart clients:** Has all the pros and cons of a smart client
- **IdP discovery:** Achieved through the cards themselves
- **Privacy concerns:** Philosophical bias towards privacy, though interaction ceremony and hosted models may weaken this
- **Vouched-for info:** Allows for third-party managed information (as well as self-asserted information)
- **Governance:** Allows for IdP/RP policy matching; no nontechnical guidance

Venn again

IdP- and enterprise-friendly, heavier, codifies existing practices but flexible for many purposes

SAML

- Comprehensive use case coverage
- Comprehensive challenge solutions, except IdP discovery
- Can be deployed to do any user centricity type

RP-friendly, lighter, less concerned with security, proposes discontinuous change à la the Web itself

OpenID

- Simple use case coverage
- Strong on IdP discovery but weak on other challenges
- The very definition of “me generation”

Client-centred, makes security the uppermost concern, can front SSO of many types through APIs

CardSpace

- “Smart client” component
- Addresses web authentication challenges
- The very definition of “trust no one”

“Me generation”

“Do what I mean” philosophy

“Trust no one”, XML message formats

Consistent user experience, “me generation” in part

Resources

Things to think about in technology selection

- Your own use cases, requirements, challenges, and constraints
 - Do your other online partners dictate a choice? Does your existing IT environment mandate a product or vendor or protocol?
 - Note that different *versions of a protocol* are different *protocols*, which may have varying product support
- Is availability of free open-source software a factor?
- What specialty client hardware/software is an option in your environment?
- What aspects of governance pertain to you?
- Is your environment “multi-protocol”?

Some additional reading

- SAML Technical Overview:
<http://www.oasis-open.org/committees/download.php/23920/sstc-saml-tech-overview-2.0-cd-01.pdf>
- Liberty governance guidance:
http://projectliberty.org/index.php/liberty/resource_center/papers
- SAML and Liberty adoption and case studies:
<http://projectliberty.org/index.php/liberty/adoption>
- InCommon federation agreements:
<http://www.incommonfederation.org/join.cfm>
- OpenID specs and resources: <http://openid.net>
- CardSpace documentation and resources:
<http://msdn2.microsoft.com/en-us/netframework/aa663320.aspx>
- XRI and XRDS:
<http://www.oasis-open.org/committees/download.php/17293>
- Planet Identity blog aggregator: <http://www.planetidentity.org>

A selection of free open-source software

- **OpenID** FOSS list – <http://openid.net/wiki/index.php/Libraries>
 - C#, C++, Java, perl, python, Ruby, PHP, ColdFusion
- **OpenSAML** – www.OpenSAML.org
 - Java/C++ libraries for low-level SAML 1.x access (soon 2.0 also)
- **Shibboleth** – <https://spaces.internet2.edu/display/SHIB/WebHome>
 - Support for SAML 1.x-based Shib IdP and SP profile
- **OpenSSO** – <http://opensso.dev.java.net>
 - Java, PHP, Ruby support variously for SAML/ID-FF/OpenID
- **Lasso** – <http://lasso.entrouvert.org/>
 - C libraries for ID-FF low-level support
 - SWIGified bindings for Python, Perl, Java and PHP
- **ZXID** – <http://www.zxid.org>
 - C, SWIGified perl libraries with ID-FF/SAML low-level support
 - C executable CGI and perl and PHP scripts for SP support
- **Higgins** – <http://www.eclipse.org/higgins>
 - Protocol-neutral APIs, focusing first on “I-Card” functionality
- **XMLDAP** – <http://www.xmldap.org/>
 - Java RP and Firefox plug-in for CardSpace identity selector

Thanks!
Questions?

eve.maler@sun.com

<http://www.xmlgrrl.com/blog>

Many thanks to those who reviewed and (knowingly or not) assisted with this presentation, particularly Gerald Beuchelt, Paul Bryan, Jeff Hodges, John Kemp, Hubert Le Van Gong, Paul Madsen, Pat Patterson, David Recordon, Drummond Reed, and Phil Windley – all errors are the author's alone