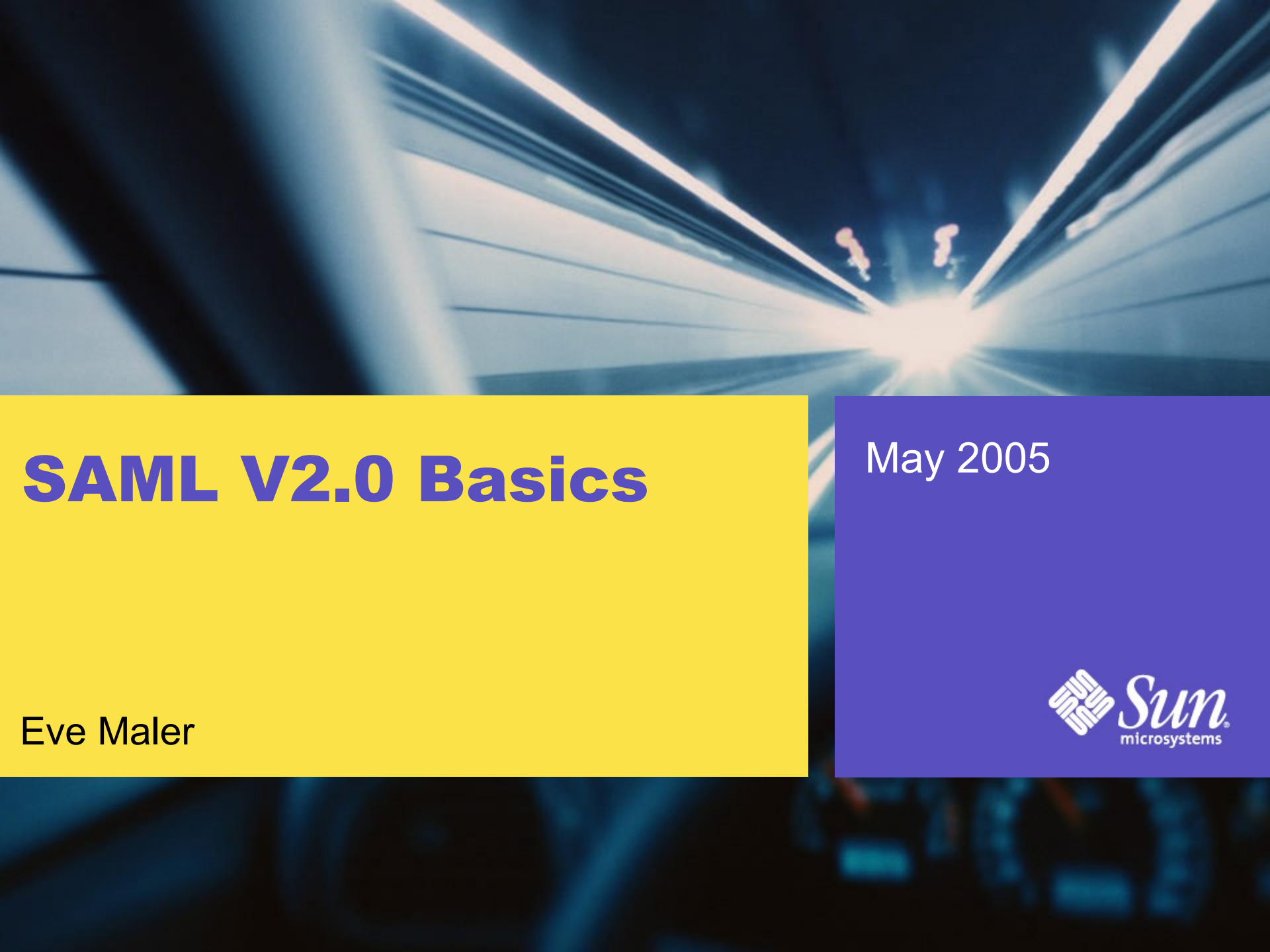


SAML V2.0 Basics

May 2005

Eve Maler

Sun microsystems

SAML in a technical nutshell

- XML-based framework for marshaling security and identity information and exchanging it across domain boundaries
  - Wraps existing security technologies rather than inventing new ones
  - Its **profiles** offer interop for a variety of use cases, but you can extend and profile it further
- At SAML's core: **assertions** about subjects
  - Assertions contain statements: authentication, attribute, entitlement, or roll-your-own

Key use cases covered by SAML out-of-the-box

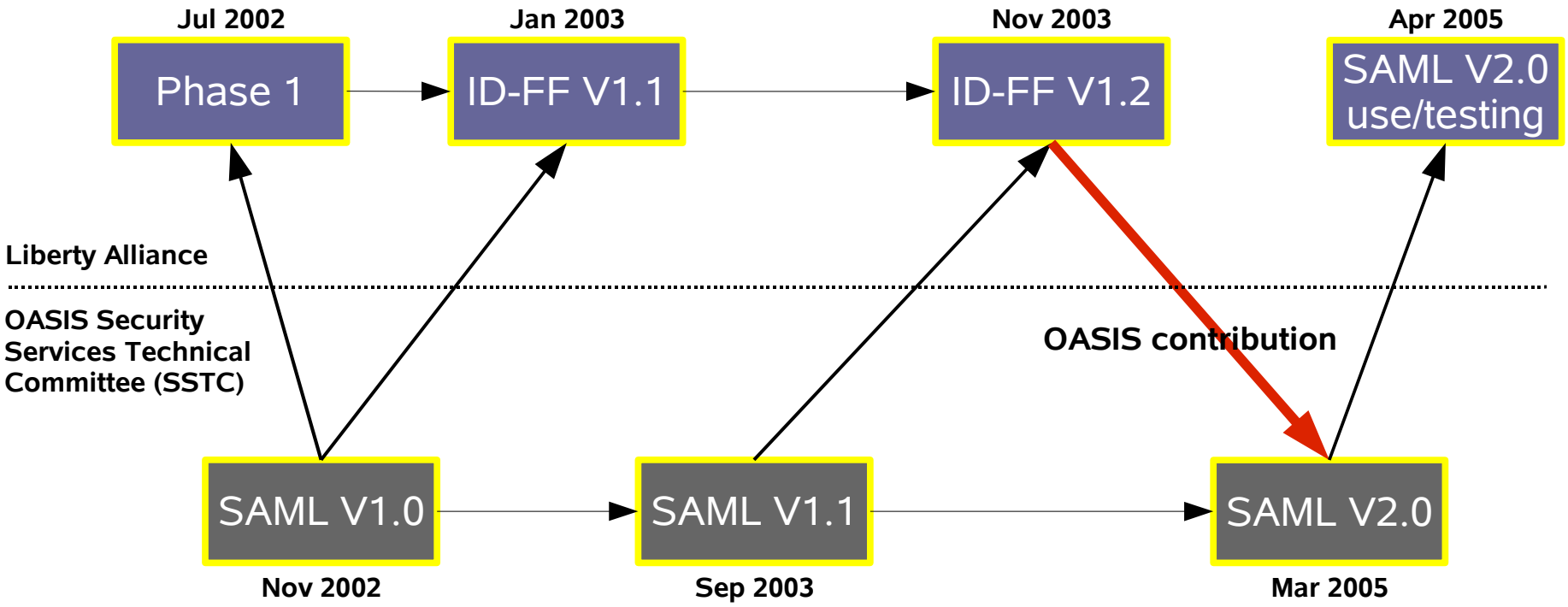
- Single sign-on
  - Using standard browsers
  - Using enhanced HTTP clients (such as handheld devices) that know how to interact with IdPs but are not SOAP-aware
- Identity federation
  - Using a well-known name or attribute
  - For anonymous users by means of attributes
  - Using a privacy-preserving pseudonym
- Attribute services
  - Getting attributes that can be interpreted according to several common attribute/directory technologies
- Single logout

Additional use cases

- Securing web service messages
  - Handled by the OASIS WSS TC as the “SAML Token Profile of WS-Security”
  - Liberty makes use of this token profile
- Profile to support retrieval of attributes for X.509-authenticated principals
- Attribute profile for interpreting XPath URIs as SAML attribute names
  - Both being worked on by the SAML committee now
- Other profiles defined for “private” use by other groups and companies

Identity federation standards: timeline and relationships

- Shown here are dates of final approval
- Internet2 Shibboleth implementation project has influenced SAML throughout and now is moving to a SAML V2.0 basis



Similarities and differences

- In achieving convergence with ID-FF V1.2, SAML V2.0 was not made backwards-compatible
  - Absorbed its functionality, rather than “gluing on Liberty bits”
  - The namespaces, syntax structure, and markup names in SAML V2.0 all differ from ID-FF and SAML V1.x
    - Whereas ID-FF was at least based on SAML V1.1 syntax
  - Its architecture is somewhat more generalized
  - However, it adopted ID-FF's “identity provider” terminology
- We focus exclusively on SAML V2.0 here

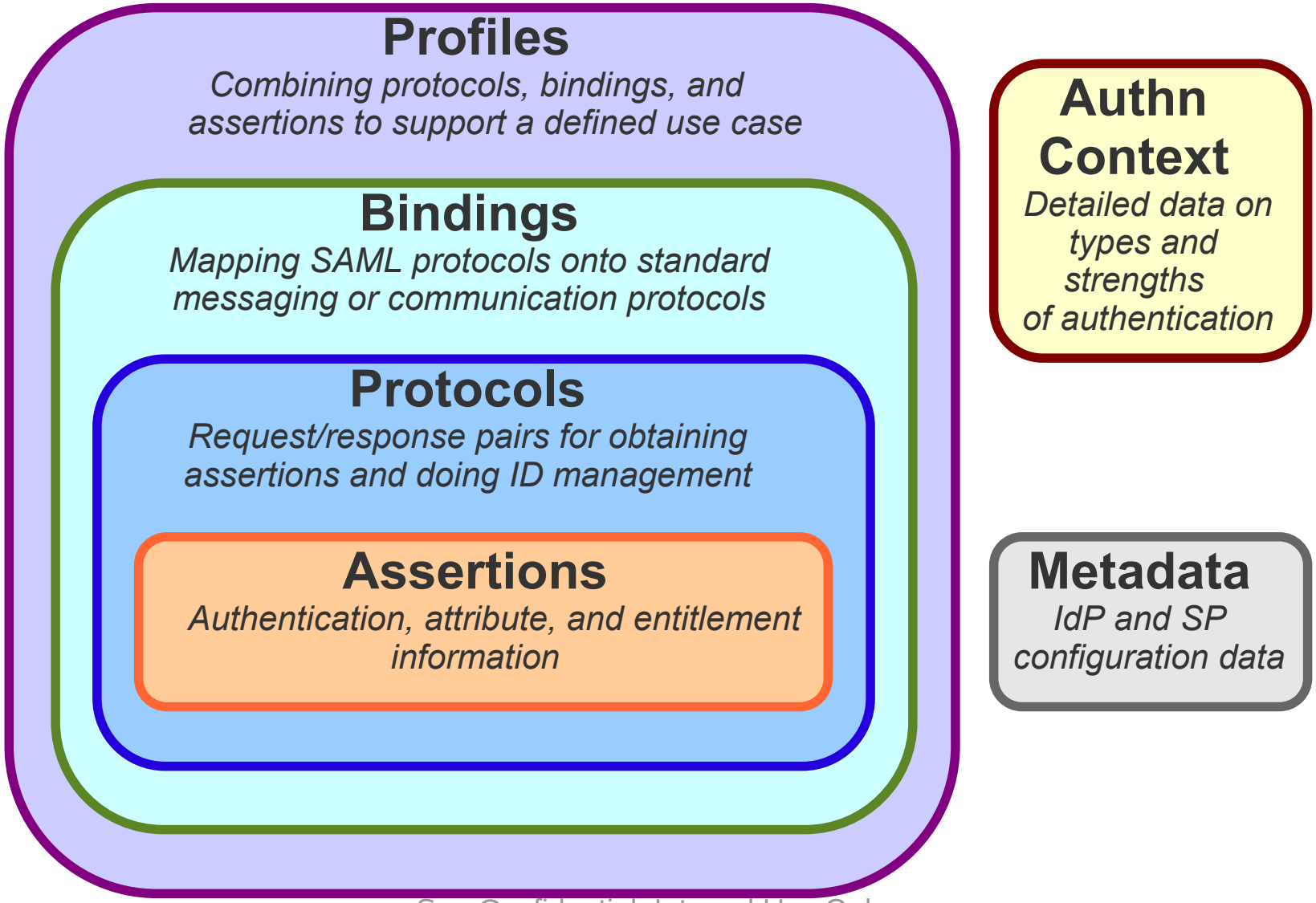
SAML V2.0 spec set

- Conformance Requirements
  - Entry point for the entire set
- Assertions and Protocols
  - SAML assertions and protocols schemas
- Bindings
- Profiles
  - SAML attribute profile schemas
- Metadata
  - SAML metadata schema
- Authentication Context
  - Various authentication context schema files
- Security and Privacy Considerations
- Glossary

Other SAML materials

- All available from the SAML home page: http://www.oasis-open.org/committees/security
  - FAQ
  - Executive Overview (final)
  - Technical Overview (draft)
  - Response to IBM security analysis (draft)
  - New profiles being worked on (drafts)
  - SAML and XACML overview by Yuri Demchenko
  - Tutorial slideware

SAML concepts



Terms and concepts: subjects

- **Entity (or system entity):** An active element of a computer/network system
- **Principal:** An entity whose identity can be authenticated
- **Subject:** A principal in the context of a security domain

Terms and concepts: identities

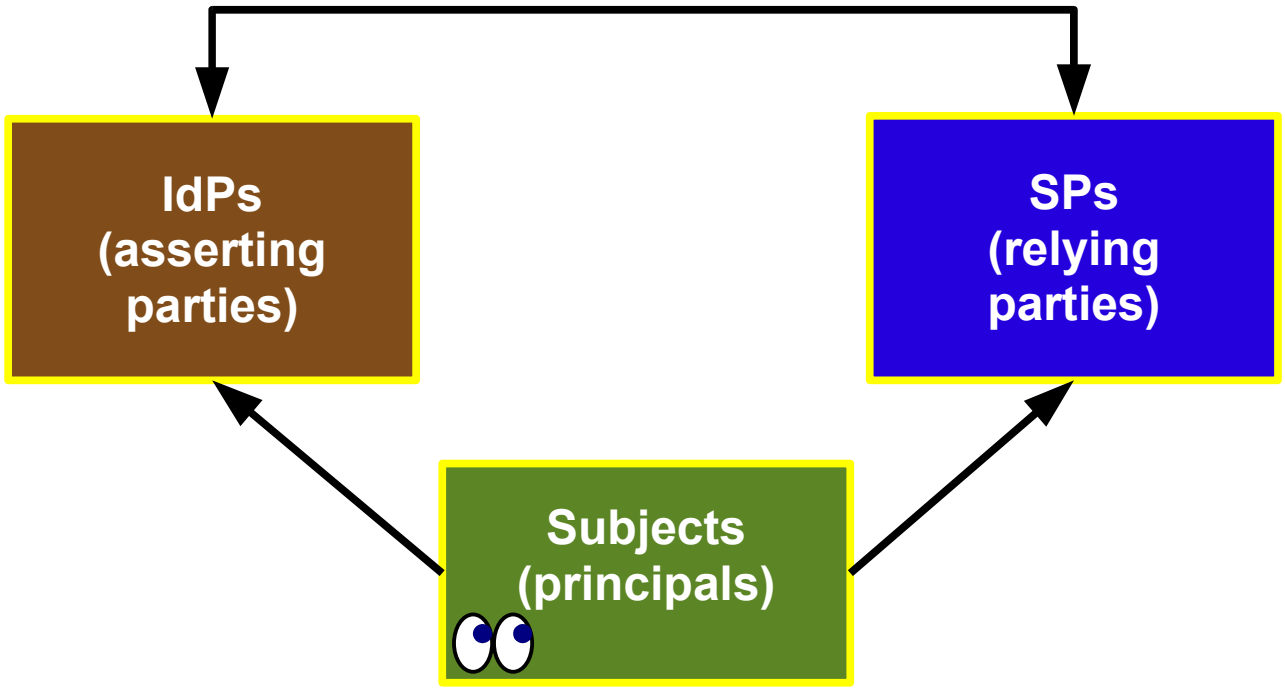
- **Identity:** The essence of an entity, often described by one's characteristics, traits, and preferences
  - **Anonymity:** Having an identity that is unknown or concealed
- **Identifier:** A data object that uniquely refers to a particular entity
  - **Pseudonym:** A privacy-preserving identifier
- **Federated identity:** Existence of an agreement between providers on a set of identifiers and/or attributes to use to refer to a principal
  - **Account linkage:** Relating a principal's accounts at two different providers so that they can communicate about the principal

Terms and concepts: more entities

- **Asserting party (SAML authority):** An entity that produces SAML assertions
  - **Identity provider:** An entity that creates, maintains, and manages identity information for principals and provides principal authentication to other service providers
- **Relying party:** An entity that decides to take an action based on information from another system entity
  - **Service provider:** An entity that provides services to principals or other entities

How these entities interrelate

- Most of the SAML and ID-FF use cases are eyeball-oriented
- But some back-channel (SOAP and other) communication takes place in service of this

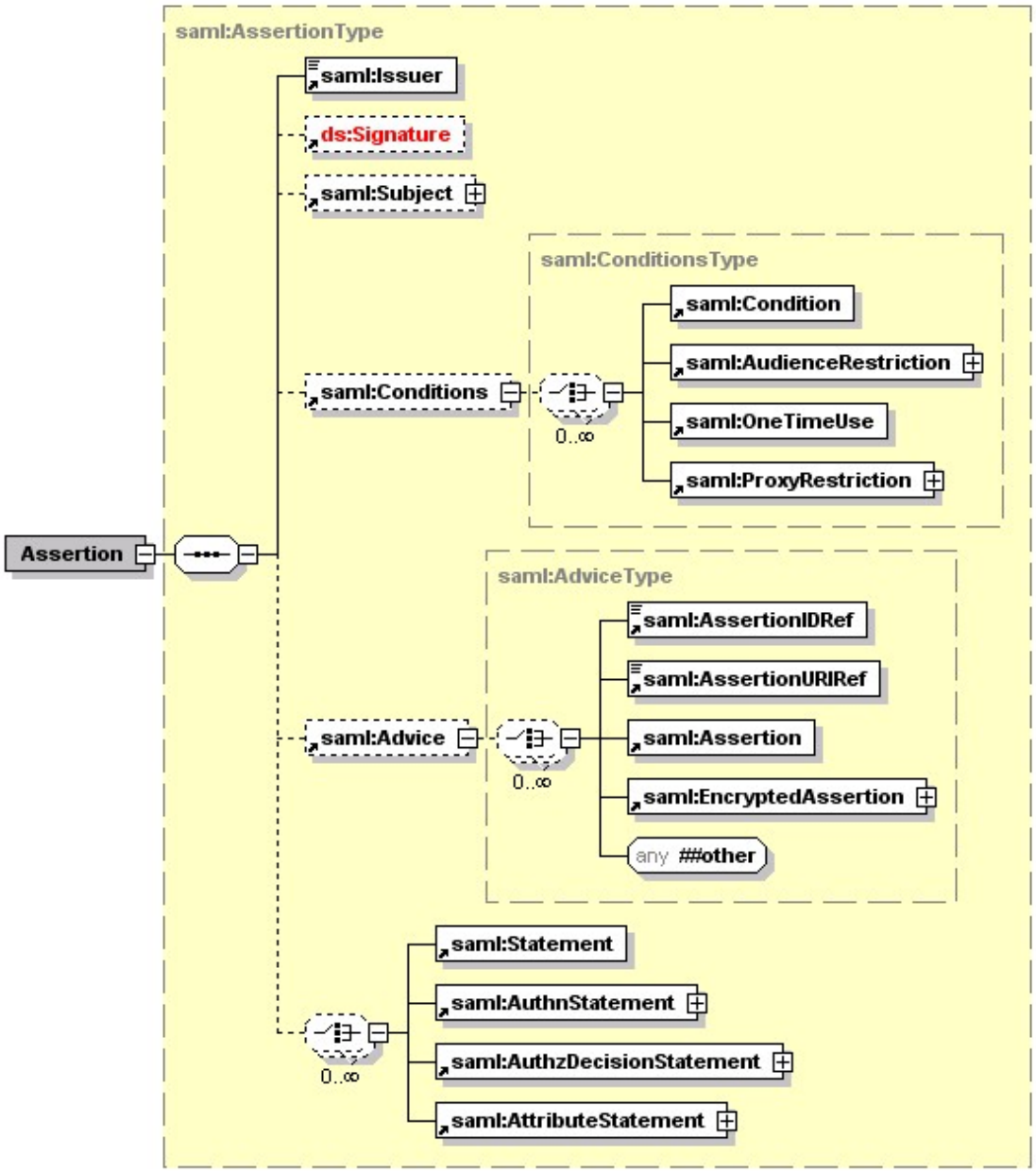


SAML assertions

- An assertion is a declaration of fact, according to someone
- SAML assertions contain one or more statements about a subject:
  - Authentication statement: **“Joe authenticated with a password at 9:00am”**
  - Attribute statement (which itself can contain multiple attributes): **“Joe is a manager with a $500 spending limit”**
  - Authorization decision statement (now deprecated)
  - Roll-your-own

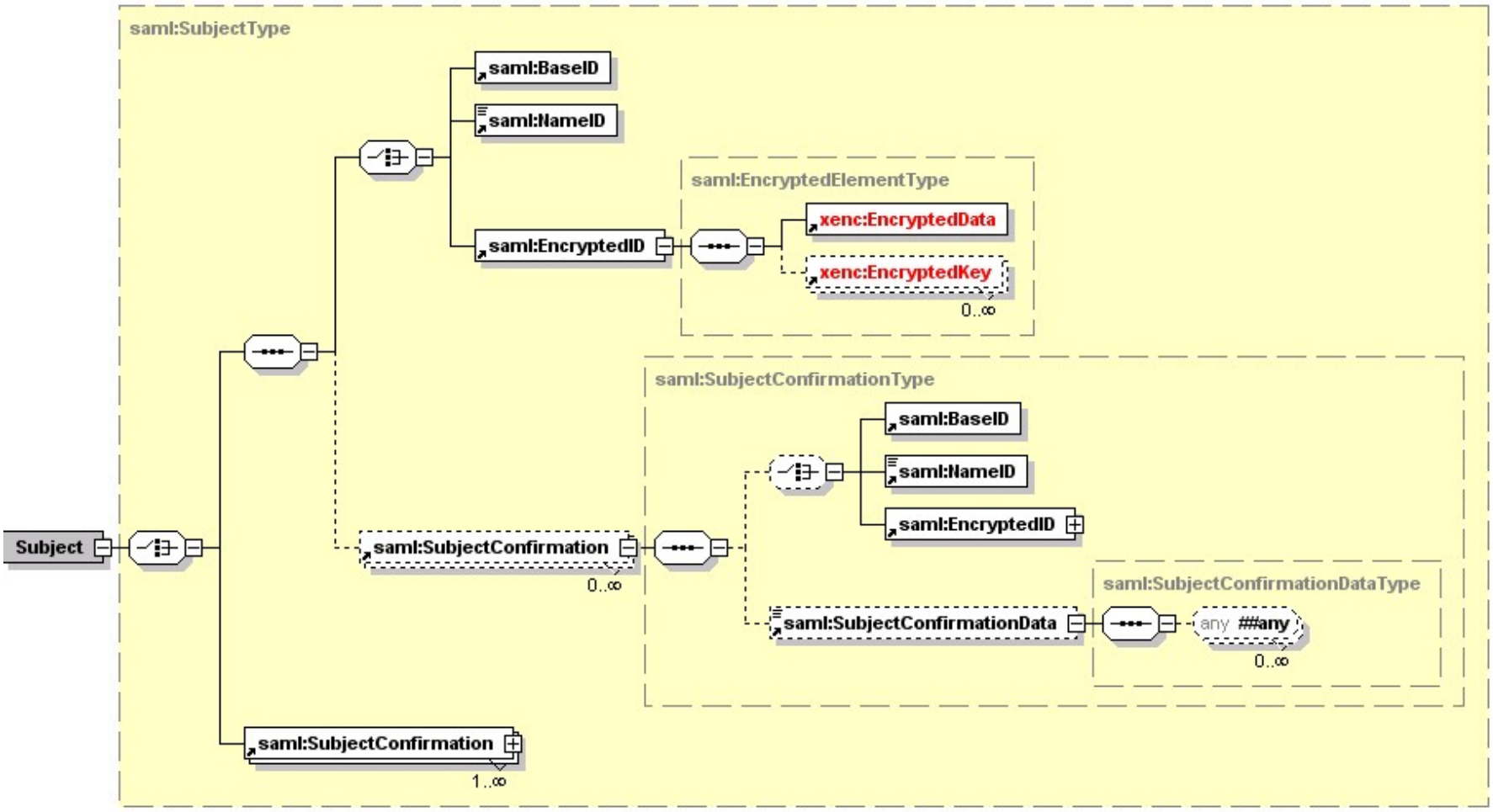
Overall assertion structure

- Attributes on **<Assertion>**: **Version**, **ID**, **IssueInstant**
- Attributes on **<Conditions>**: **NotBefore**, **NotOnOrAfter**



Subject structure

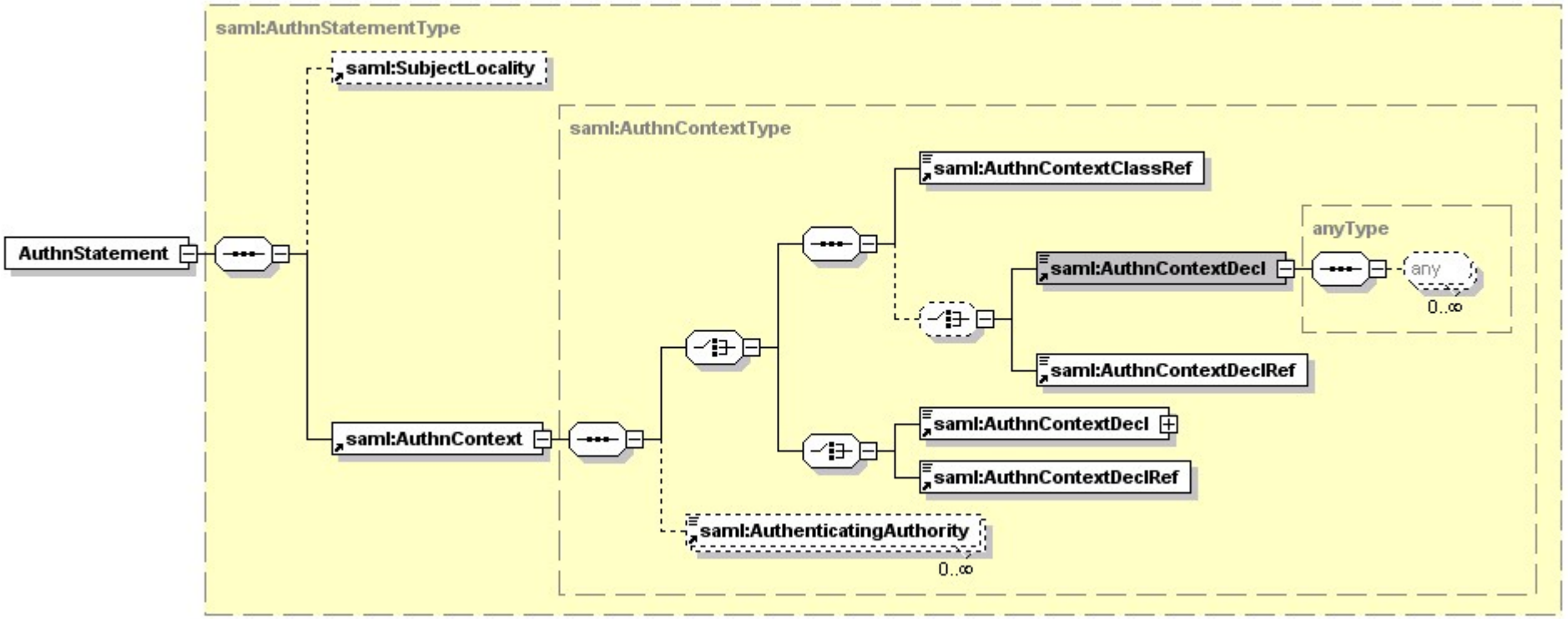
- Attributes on **<NameID>**: **NameQualifier**, **SPNameQualifier**, **Format**, **SPProvidedID**



Example of the common portions of an assertion

```
<saml:Assertion
  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
  Version="2.0"
  IssueInstant="2005-01-31T12:00:00Z">
  <saml:Issuer>
    www.acompany.com
  </saml:Issuer>
  <saml:Subject>
    <saml:NameID
  Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">
      j.doe@company.com
    </saml:NameID>
  </saml:Subject>
  <saml:Conditions
    NotBefore="2005-01-31T12:00:00Z"
    NotOnOrAfter="2005-01-31T12:00:00Z">
  </saml:Conditions>
    ... statements go here ...
</saml:Assertion>
```

Authentication statement structure



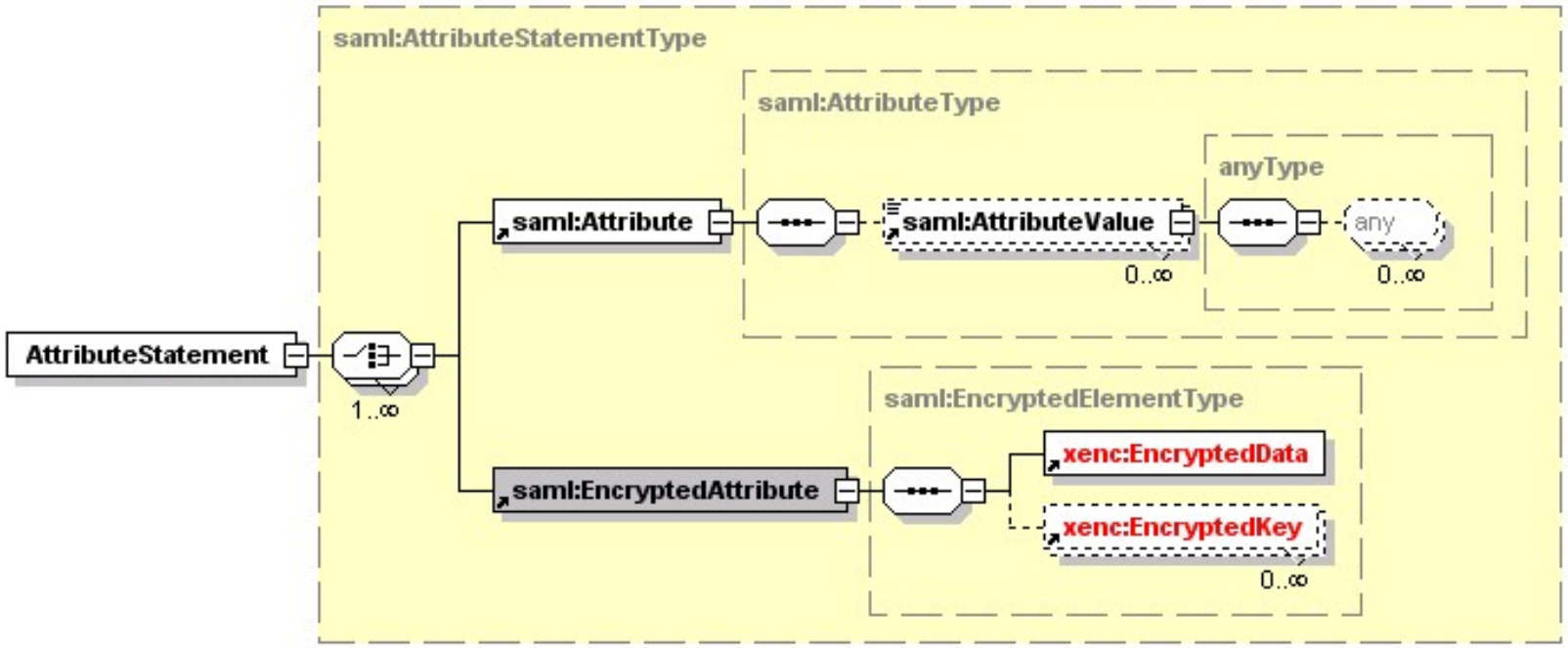
Example of an authentication statement

```
<saml:Assertion ... common info goes here ... >
  ... and here ...
  <saml:AuthnStatement
    AuthnInstant="2005-01-31T12:00:00Z"
    SessionIndex="67775277772">
    <saml:AuthnContext>
      <saml:AuthnContextClassRef>
urn:oasis:names:tc:SAML:2.0:ac:classes:PasswordProtectedTransport
      </saml:AuthnContextClassRef>
    </saml:AuthnContext>
</saml:AuthnStatement>
```

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 Authn
- Time Sync Token
- Unspecified

Attribute statement structure



Example of an attribute statement

```
<saml:Assertion ... common info goes here ... >
  ... and here ...
  <saml:AttributeStatement>
    <saml:Attribute
      NameFormat="http://smithco.com">
      Name="PaidStatus"
      <saml:AttributeValue>
        PaidUp
      </saml:AttributeValue>
    </saml:Attribute>
    <saml:Attribute
      NameFormat="http://smithco.com">
      Name="CreditLimit"
      <saml:AttributeValue xsi:type="smithco:type">
        <smithco:amount currency="USD">
          500.00
        </my:amount>
      </saml:AttributeValue>
    </saml:Attribute>
  </saml:AttributeStatement>
</saml:Assertion>
```

Attribute profiles

- Basic
  - Simple string-based SAML attribute names
- X.500/LDAP
  - Common standardized convention for SAML attribute naming using OIDs, expressed as URNs and accompanied by usage of `xsi:type`
- UUID
  - SAML attribute names as UUIDs, expressed as URNs
- DCE PAC
  - Representation of DCE realm, principal, and primary group, local group, and foreign group membership information in SAML attributes
- XACML
  - How to map SAML attributes cleanly to XACML attribute representation

Example of the DCE attribute profile

```
<saml:AttributeStatement>
  <saml:Attribute
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
    Name="urn:oasis:names:tc:SAML:2.0:profiles:attribute:DCE:realm">
    <saml:AttributeValue
      xsi:type="dce:DCEValueType" dce:FriendlyName="example.com">
      urn:uuid:003c6cc1-9ff8-10f9-990f-004005b13a2b
    </saml:AttributeValue>
  </saml:Attribute>
  <saml:Attribute
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
    Name="urn:oasis:names:tc:SAML:2.0:profiles:attribute:DCE:principal">
    <saml:AttributeValue xsi:type="dce:DCEValueType" dce:FriendlyName="jdoe">
      urn:uuid:00305ed1-a1bd-10f9-a2d0-004005b13a2b
    </saml:AttributeValue>
  </saml:Attribute>
  <saml:Attribute
    NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"
    Name="urn:oasis:names:tc:SAML:2.0:profiles:attribute:DCE:primary-group">
    <saml:AttributeValue
      xsi:type="dce:DCEValueType" dce:FriendlyName="cubicle-dwellers">
      urn:uuid:008c6181-a288-10f9-b6d6-004005b13a2b
    </saml:AttributeValue>
  </saml:Attribute>
  </saml:AttributeStatement>
```

Other assertion-related structures

- `<saml:AssertionIDRef>`
- `<saml:AssertionURIRef>`
- `<saml:EncryptedAssertion>`

Themes in the XML expression of SAML

- URIs as category names for various options
  - No native use of “QNames in content”
  - SAML standardizes a starter set of URIs in each case, but anyone can develop and use other URIs
  - For example, `urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress` for email-based name identifiers
- Controlled extension points
  - Abstract schema types – for example, for name identifiers
  - Wildcards – for example, **`<AttributeValue>`** allows arbitrary XML element content
  - Guidelines for how to develop your own profiles

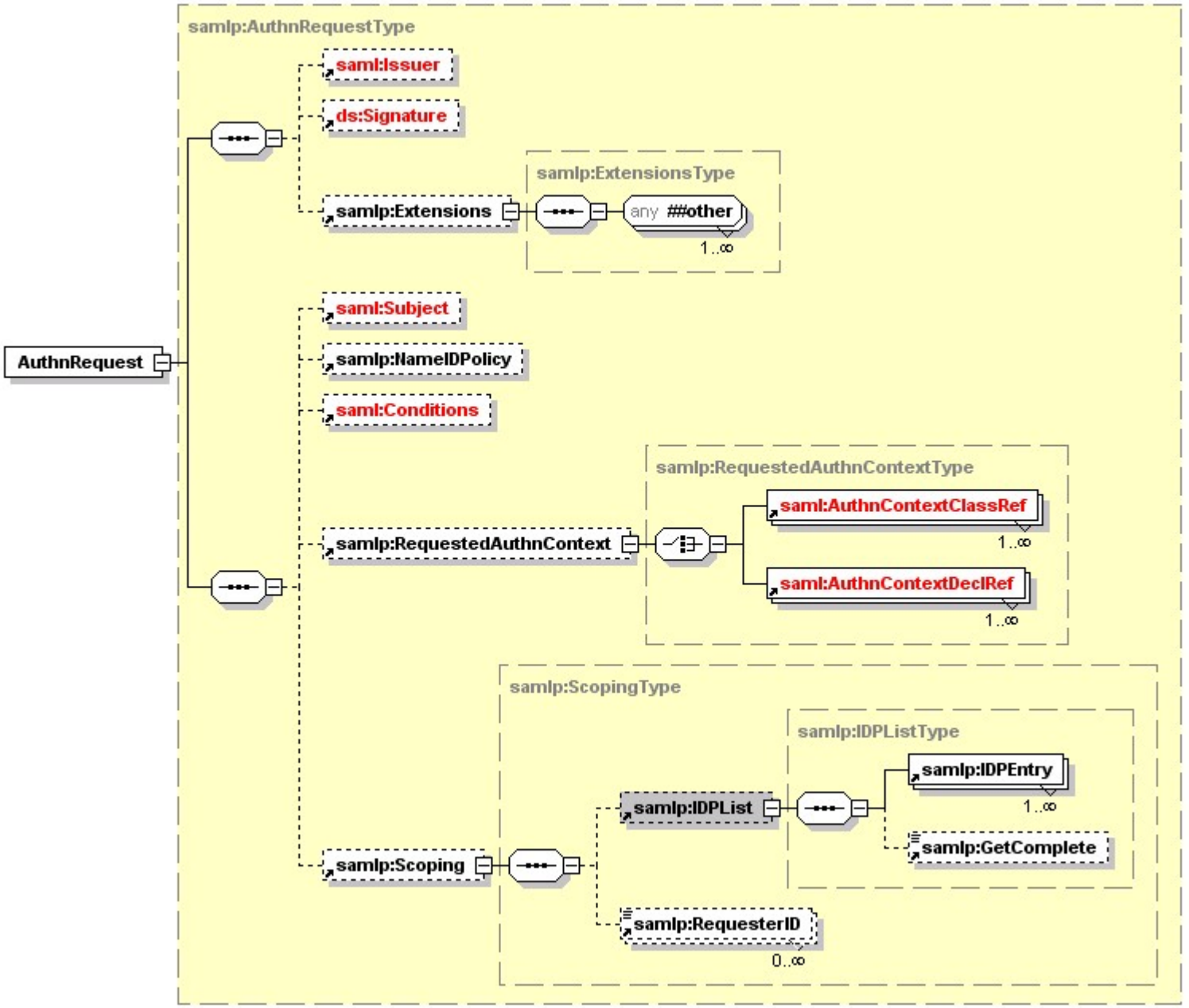
Artifacts

- A small, fixed-size, structured data object pointing to a typically larger, variably sized SAML protocol message
- Designed to be embedded in URLs and conveyed in HTTP messages
- Allows for “pulling” SAML messages rather than having to push them
- SAML defines one artifact format but you can roll your own

Protocols

- Assertion query and request
  - Query for assertion based on simple reference, subject-matching, or statement type
- Authentication request
  - SP requests a fresh authn assertion that adheres to various requirements (specified by means of Authentication Context)
- Artifact resolution (“meta-protocol”)
  - Dereferences an artifact to get a protocol message
- Name identifier management
  - IdPs and SPs inform each other of changes to their mutual understanding of what a principal's name is
- Name identifier mapping
  - Privacy-preserving way for two SPs to refer to the same principal
- Single logout
  - Signals to all SPs using the same session to drop the session

Authn request structure



Example of an authn request protocol message

```
<env:Envelope
  xmlns:env="http://www.w3.org/2003/05/soap/envelope/">
  <env:Body>
    <samlp:AuthnRequest
      xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
        ForceAuthn="true"
        AssertionConsumerServiceURL="http://www.example.com/"
        AttributeConsumingServiceIndex="0"
        ProviderName="string"
        ID="abe567de6"
        Version="2.0"
        IssueInstant="2005-01-31T12:00:00Z"
        Destination="http://www.example.com/"
        Consent="http://www.example.com/">
      <saml:Subject
        xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
        <saml:NameID
        Format="urn:oasis:names:tc:SAML:1.1:nameid-format:emailAddress">
          j.doe@company.com
        </saml:NameID>
      </saml:Subject>
    </samlp:AuthnRequest>
  </env:Body>
</env:Envelope>
```

Bindings

- SOAP
  - Basic way for IdPs and SPs to send SAML protocol messages
- Reverse SOAP (PAOS)
  - Multi-stage SOAP/HTTP exchange that allows an HTTP client to send an HTTP request containing a SOAP response
- HTTP redirect
  - Method to send SAML messages by means of HTTP 302
- HTTP POST
  - Method to send SAML messages in base64-encoded HTML form control
- HTTP artifact
  - Way to transport an artifact using HTTP in two ways: URL query string and HTML form control
- URI
  - How to retrieve a SAML message by resolving a URI

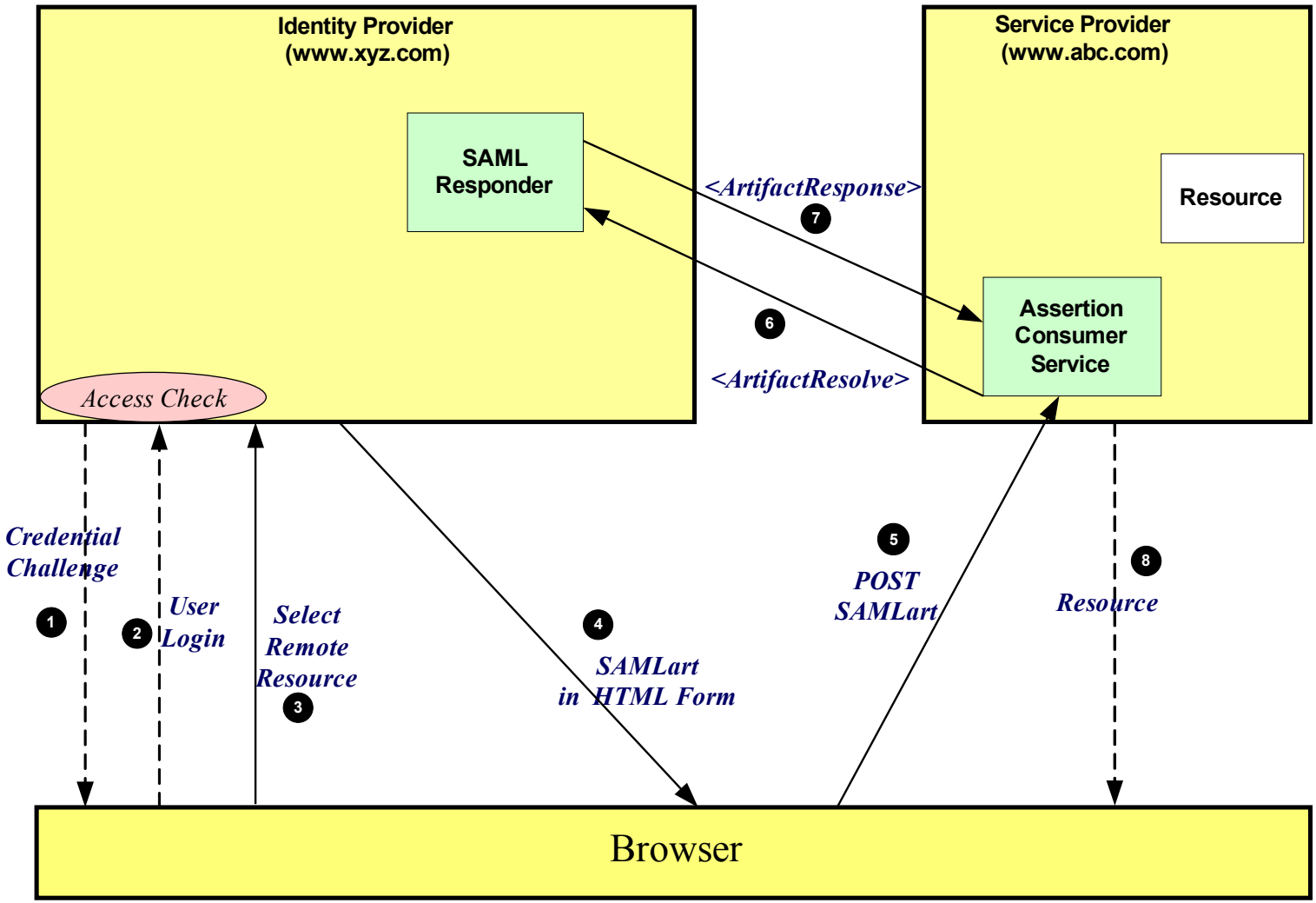
Profiles

- Web browser SSO
  - SSO using standard browsers to multiple SPs: profiles Authn Request protocol and HTTP Redirect, POST, and artifact bindings
- Enhanced client and proxy (ECP)
  - SSO using ECPs: profiles Authn Request protocol and SOAP and PAOS bindings
- IdP discovery
  - One way for SPs to learn the IdPs used by a principal
- Single logout
- Name identifier management
  - Profiles the NIM protocol with SOAP, HTTP redirect, HTTP POST, and HTTP artifact bindings
- Artifact resolution
- Assertion query/request

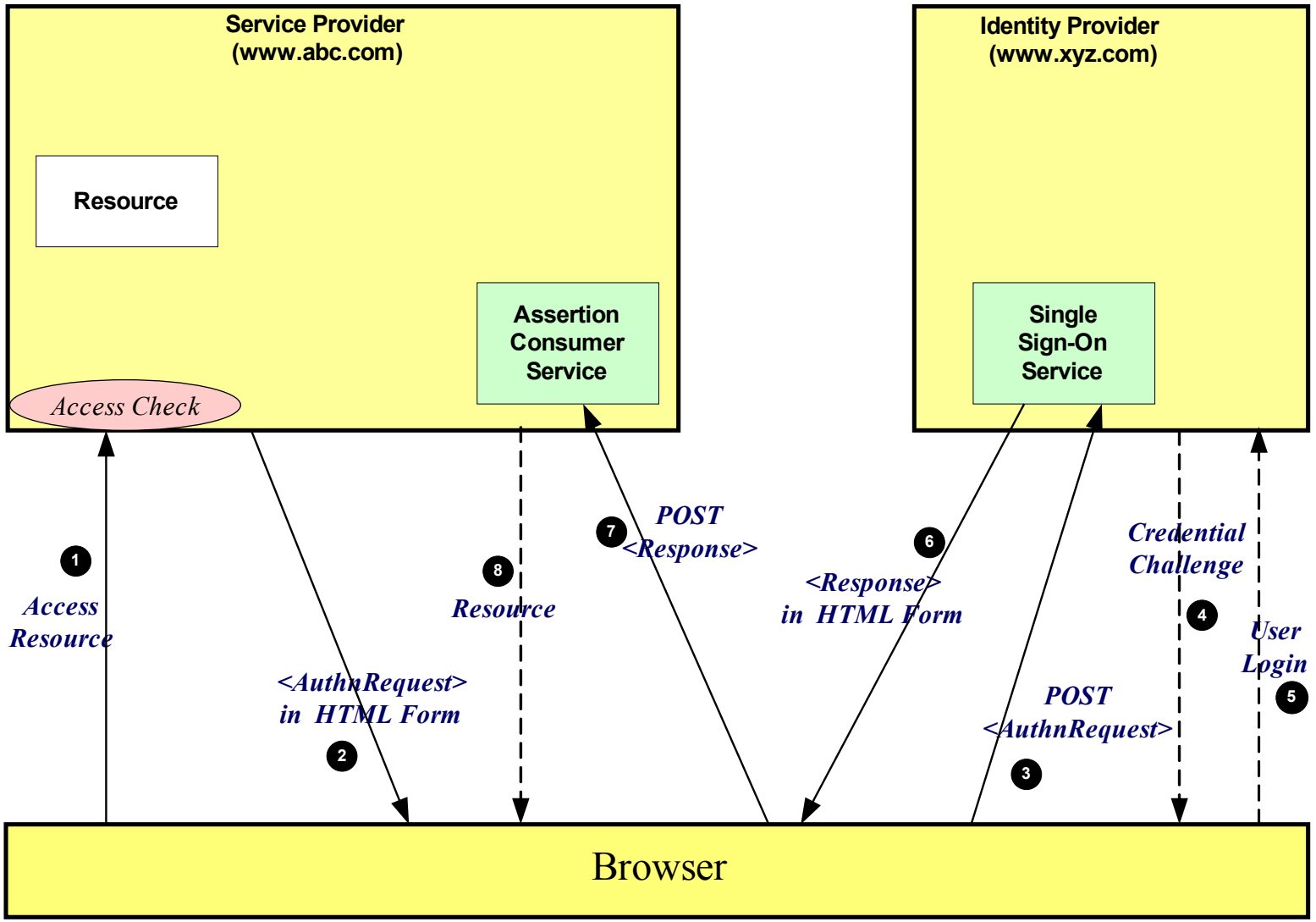
Within profiles, different flows and binding choices are possible

- E.g., in the web browser SSO profile:
  - Authn request from SP to IdP can use any of HTTP redirect or HTTP POST or HTTP artifact
  - IdP response to SP can use either HTTP POST or HTTP artifact
- E.g., in the ECP SSO profile using the PAOS binding, two flows are possible:
  - ECP to SP, SP to ECP to IdP
  - IdP to ECP to SP, SP to ECP

Browser/artifact flow, IdP-initiated



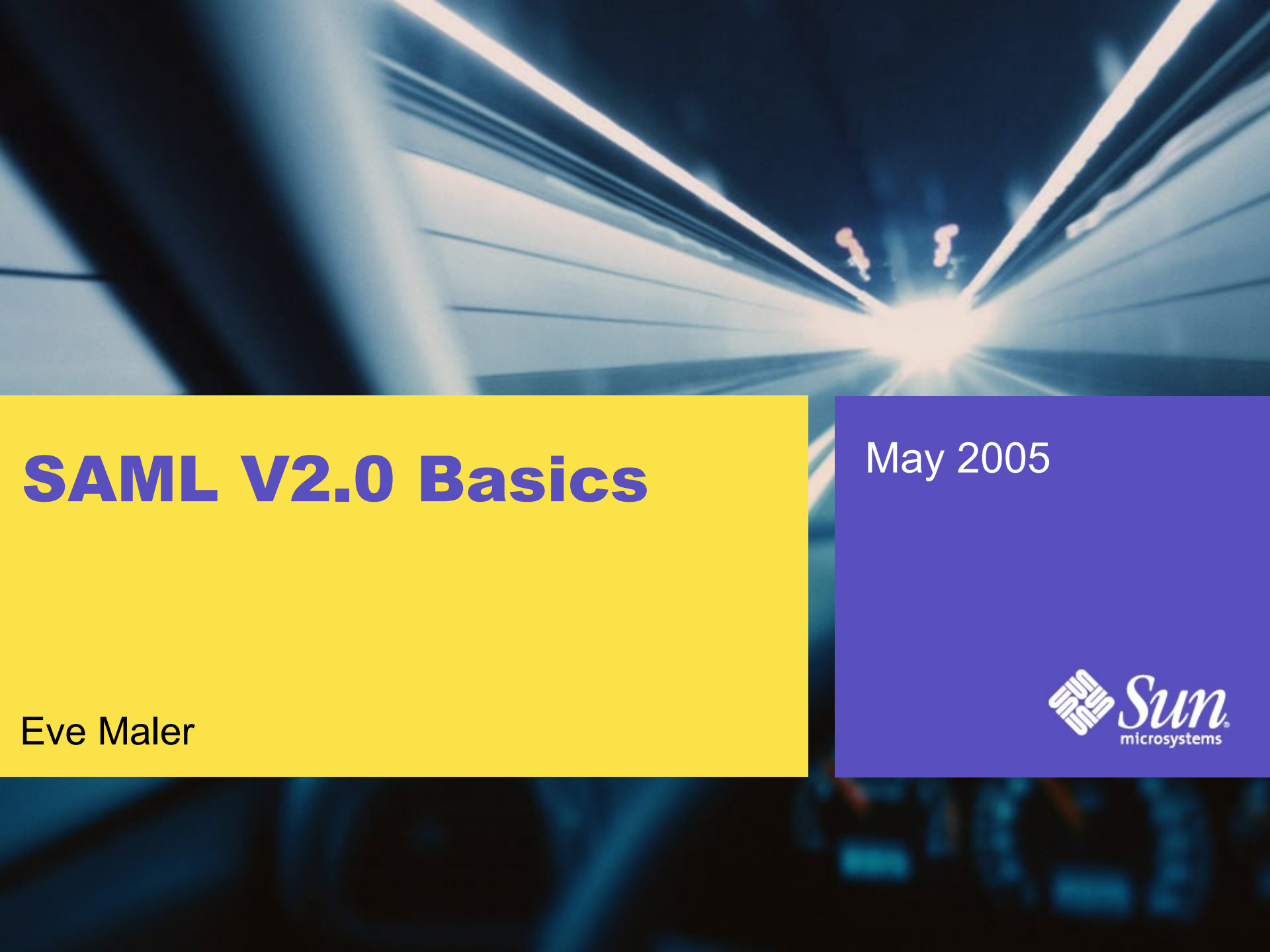
Browser/POST flow, SP-initiated



SAML conformance and operational modes

- Profiles are the "minimum unit of interoperability"
- But ***operational modes*** are the "minimum unit of conformance"
- Each one requires support for a particular set of profiles

- IdP
- IdP Lite
- SP
- SP Lite
- ECP
- SAML Authn Authority
- SAML Attribute Authority
- SAML Authz Decision Authority
- SAML Requester



SAML V2.0 Basics

May 2005

Eve Maler

Sun microsystems