A fully distributed OpenID Connect deployment based on domain names: ID4me

Challenges, lessons learned and take-aways

OAuth Security Workshop 2018, Trento

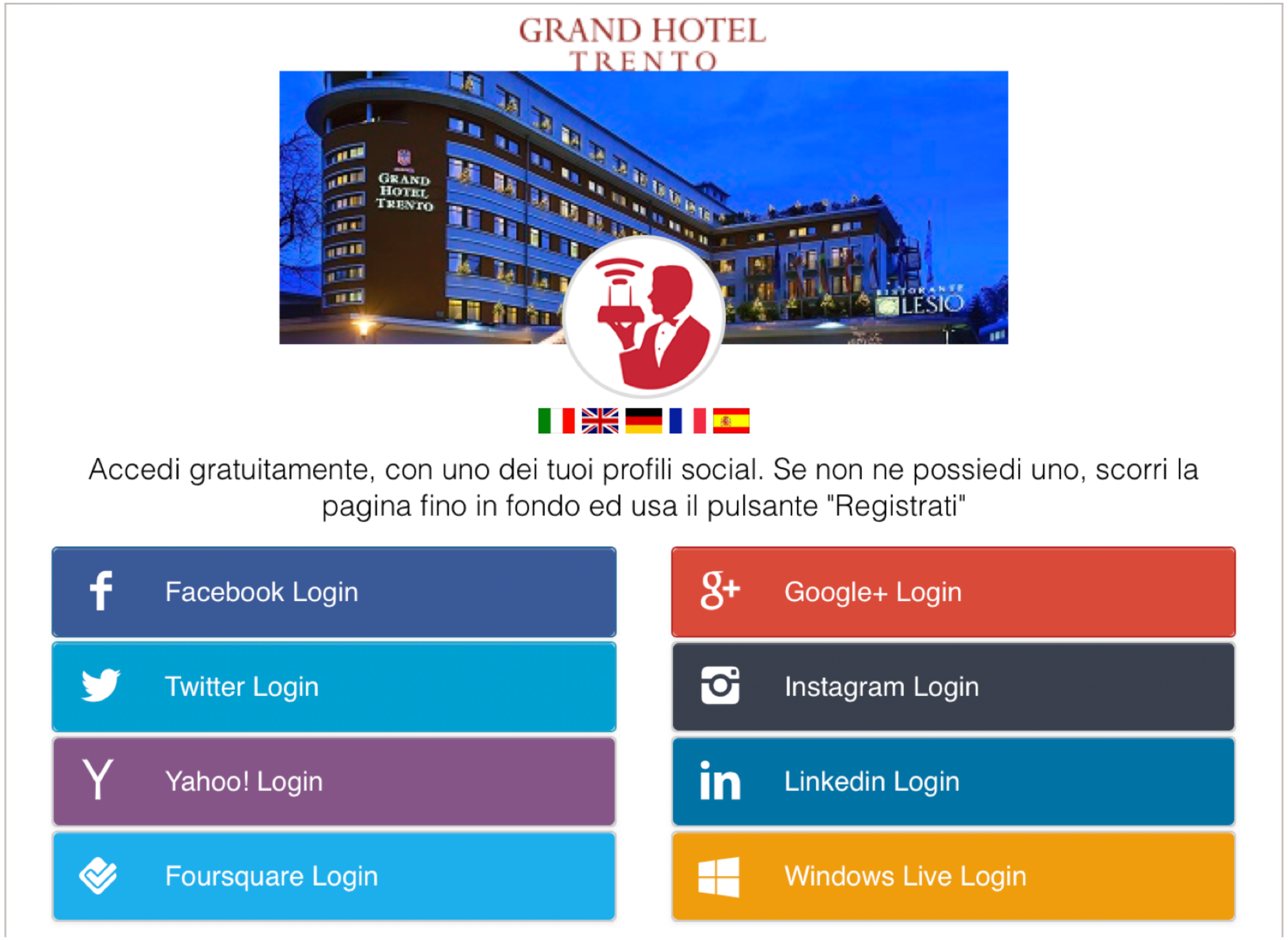
Vittorio Bertola <vittorio.bertola@open-xchange.com>
Marcos Sanz <sanz@denic.de>

Online credentials for the average user

- Most people just reuse usernames and passwords across hundreds of websites and services
  - Usability issues
  - Security issues

- Single-sign-on systems in private namespaces gaining ground
  - Users like them, but:
  - Fragmentation, lack of interoperability
  - Clients have to implement each of them separately
  - Users cannot choose their provider

Grand Hotel Trento's wi-fi login form



GRAND HOTEL TRENTO

Accedi gratuitamente, con uno dei tuoi profili social. Se non ne possiedi uno, scorri la pagina fino in fondo ed usa il pulsante "Registrati"

Advantages of public, federated SSO

- Why can’t your online identity work like your email address?
- You only need one account to interoperate with everyone
- You get to choose and even to change your provider
  - You can keep your address if it is in your own domain name
- You only need to remember and secure one set of credentials
- Any additional security mechanisms can be implemented just once by a specialized party (not by any website operator)
- You have an easy way to control the sharing of your information and to keep it updated (a legal requirement in many countries)
- You don’t need to register for new websites, just identify yourself

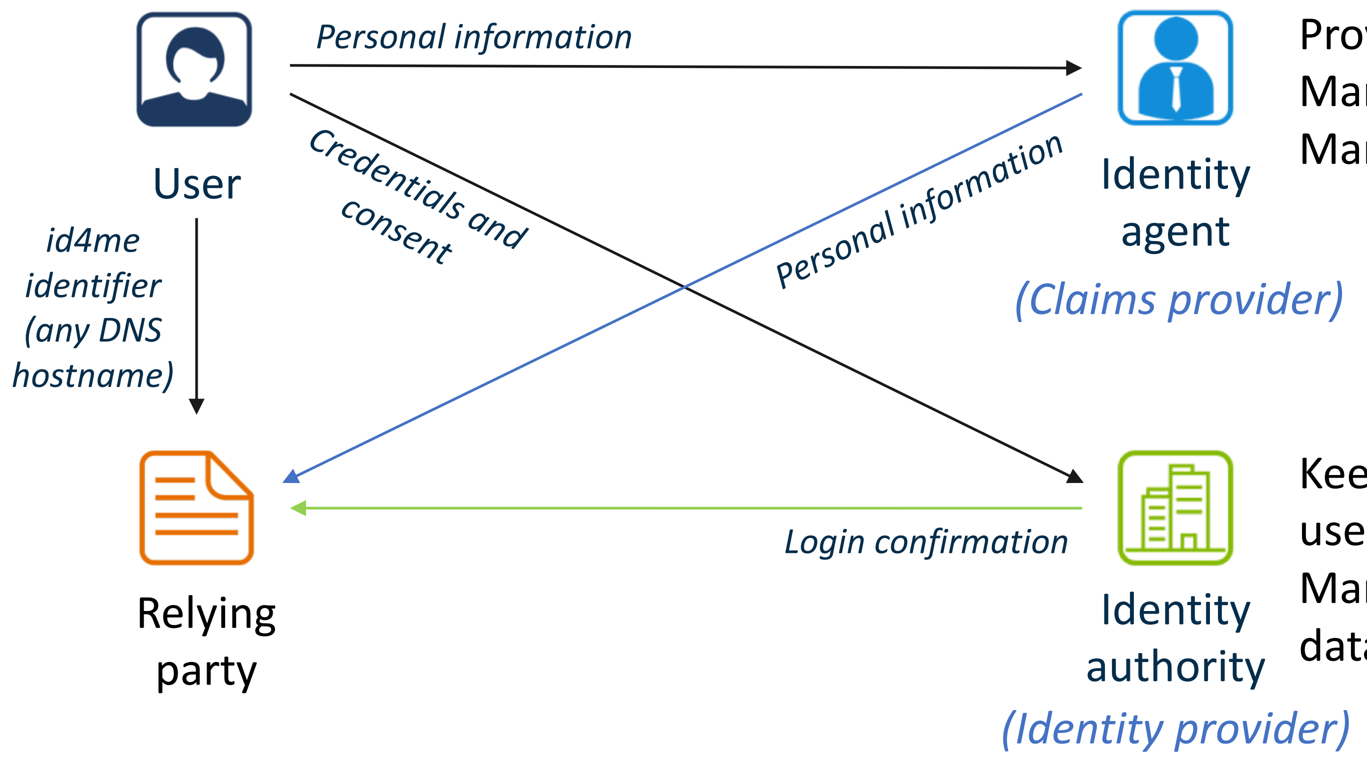
Design principles for the solution

- Be public and federated
  - Prevent a chat-like mess of incompatible competing services
- Reduce the implementation effort
  - Build on widely used technologies: OpenID Connect/OAuth, DNS
  - Allow easy integration of existing OAuth-based sets of identities
- Flatten the user’s learning curve
  - Users are already familiar with DNS-based identifiers (hostnames/emails)
- Not deal with real world identification
  - Users can have multiple identities, pseudonymous identities etc
  - Though you could build third-party certification as an option in the scheme

How it works

- We add a DNS-based discovery mechanism to OpenID Connect
  - Any hostname or email address can be mapped to an identity provider
  - A string with name-value couples in a TXT record specifying pointers
  - You only have to add the DNS piece, the rest is standard OpenID Connect
  - draft-ietf-oauth-discovery leaves issuer discovery out of scope
  - DNSSEC and DANE provide security
- We use the OpenID distributed claims mechanism to separate roles
  - Distinction between an identity authority doing authorization and authentication, and an identity agent managing users and their data
  - Separating functions and data sets increases privacy and security
- We (plan to) add an ontology for any useful claim

The roles in ID4me



Provides service to user
Manages customer
Manages user data

Keeps and verifies user credentials
Manages consent to data sharing

Performing a login

1. The relying party only asks for the domain-based identifier
2. The relying party performs a DNS query and discovers the identity authority and the identity agent for that identifier
3. A standard OpenID Connect Authorization Code Flow is performed
4. During the flow, the authority also verifies consent to data sharing
5. The authority uses the distributed claims mechanism in the claims object returned by its userinfo endpoint
6. The relying party retrieves the actual claim values from the agent's userinfo endpoint

Project status

- A joint project by three companies (public name “ID4ME”)
- A prototype up and running, with new features being added
- An international association in formation
- Presented to several relevant companies in Europe
  - Interest by TLD registries willing to become identity authorities
  - Interest by domain name registrars willing to become identity agents
  - Interest by telcos / ISPs willing to supply identities to their users
- Two Internet drafts independently submitted in October
  - Recently updated, but still missing lots of stuff
- Looking for feedback and participation

Challenges and lessons learned

Using DNS-based identifiers

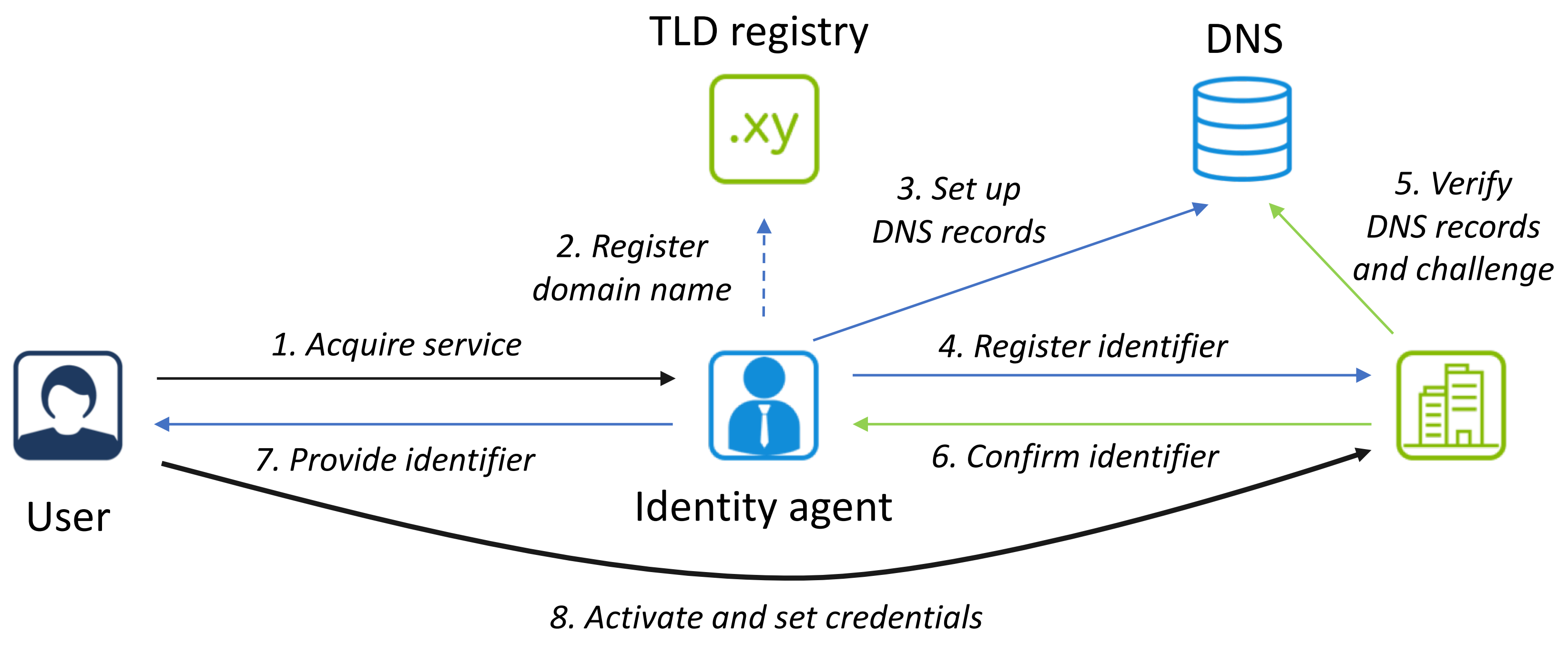
- Immediately offers federation
- Any identifier format used by an existing identity provider can be mapped into a sub-namespace inside one of the provider’s domains
- Email addresses can also be used as identifiers, by specifying a mapping mechanism
  - See draft-sanz-openid-dns-discovery
- Problem: Even if DNS-based OpenID Connect identifiers are fully compliant with the standard, not all the implementations accept non-scoped identifiers (without an @ sign)
  - Including the OpenID Foundation’s own certification software

Proving control of the identifier

- B.Y.O.I. (Bring your own identifier) => The identity authority needs to be sure that the user actually controls the domain
  - »Control» may just mean permission to use (e.g. in the identity agent’s own zone)
- Using ACME with a DNS challenge
- At the end of the process, the identity authority will provide a one-time link for the user to set up the credentials
- The credentials will be agreed directly between the user and the identity authority
  - The identity agent never gets to see them

The identifier creation procedure

(Simplified)



DNS-based discovery mechanism

- Webfinger is impractical if you distribute the identifiers across (potentially) millions of DNS zones
  - Each domain would be required to set up a web server and get a certificate for HTTPS
  - Also unnecessary: you need the DNS anyway to perform Webfinger
  - You can use DNSSEC to secure the reply
- Several DNS resource record types exist for service discovery
  - But not in wide use
- So we followed the trend of TXT records with name-value couples

```
_openid.yourname.example.de IN TXT
"v=OID1;iss=auth.freedom-id.de;clp=identityagent.de"
```

The distributed claims mechanism

- We use OpenID Connect’s «distributed claims» feature
  - The identity authority’s userinfo endpoint returns a claims JSON object pointing at the identity agent’s userinfo endpoint for all consented claims
  - The identity agent’s userinfo endpoint returns the actual values
- But the mechanism is underspecified and not well supported
  - The OpenID Foundation’s certification software tries to validate the claims provider’s JWT with the identity provider’s public key
  - Also there is no standard way for an identity provider to discover dynamically which claims provider is managing the claims for a given identifier
  - So the authority uses the TXT record as well
- The access token includes the information on which claims have been consented or rejected by the user

Using DANE instead of CAs

- All the infrastructure is focused on domain names
  - We only need to secure domain names and server-to-server communications
  - The system does not want to provide «real world» authentication of people and claims, so offline identity checks are not necessary
- Certification authorities would introduce additional attack surfaces without any benefit
- So we go for DANE instead
  - We mandate the use of DANE to validate the certificate for all HTTPS-based server-to-server communications

Discovery of supported claims

- In a centrally run deployment with a single identity provider, claim names and availability can be defined *a priori*
- In a distributed and federated environment, you need naming standards and claims discovery mechanisms
- We will define an ontology of all claims supported by the platform
    - Relying parties will be able to ask for specific pieces of information, no matter which authority/agent is managing the identifier
    - The authority/agent will be able to communicate to the relying party what is available and what is not, also considering user consent
- Identity authorities will have a way to tell agents the claims they support
- Work in progress

Subject identifiers

- Domain names may change hands, bringing with them their DNS-based identity identifiers
- Relying parties need a unique identifier that dies when the identifier changes hands and is reconfigured by someone else
- We’d use «pairwise Subject Identifiers» as defined in OpenID Connect
  - The authority gives a different one to each relying party
  - Though the relying party still gets to see the DNS-based identifier (so no additional privacy/security)
  - Also, again, this feature’s implementation is often buggy
- For the moment, we use «public Subject Identifiers»

Further technical working areas

- Formal security analysis (help welcome)
- Proof-of-possession access tokens (e.g. OAuth Token Binding)
- Technical compatibility with other DNS-based identity systems
- Identifier transfer and deletion
- Claims discovery and ontology

draft-bertola-dns-openid-pidi-architecture

draft-sanz-openid-dns-discovery

id4me.org