Authentication-as-a service: Theory versus reality

Chris Mitchell
me@chrismitchell.net

Agenda

1. Authentication-as a service
2. Privacy (theory) versus reality – a history
3. Where are we now?
4. Where would we like to be?
5. What can we do?
Acknowledgement

• Much of the work on OAuth 2.0 and OpenID Connect is due to my former PhD student Wanpeng Li, currently of Manchester Metropolitan University.

• He has not only discovered a wide range of real-world vulnerabilities, but has also produced a browser plug-in OAuthGuard to help enhance end user security.

Agenda

1. Authentication-as a service
2. Privacy (theory) versus reality – a history
3. Where are we now?
4. Where would we like to be?
5. What can we do?
Identity and authentication

- When user wishes to access a service via the Internet, the service may want to know who user is (e.g. for charging purposes).
- User must provide identity, and also allow the service provider to authenticate the claimed identity (using credentials).
- In other cases, service provider may simply wish to know certain user characteristics or attributes (e.g. whether the user is over 18).

Single sign on (SSO)

- An Internet single sign on (SSO) system allows a user to log in to multiple web sites with just one authentication.
- Increasingly widely used, e.g. in form of
  - Facebook Connect – using OAuth 2.0;
  - Google SSO service – formerly built using OpenID and now employing OpenID Connect, which is OAuth 2.0 based.
Identity management

• An SSO system is just a special case of an identity (ID) management system.

• In general, in an ID management system, one or more third parties manage aspects of a user's identity on behalf of a user, e.g. they
  – store user attributes;
  – authenticate users on behalf of other parties.

Identity management terminology

• **Identity Provider (IdP)** authenticates user and vouches for **User** identity to ...

• **Relying Parties (RPs)**, typically Service Providers of some kind, which rely on IdP and provide online services to ...

• **Users**, who employ ...

• **User Agents (UAs)** (typically web browsers), to interact with RPs.
SSO operation

- User host and RP host have some kind of session (e.g. an SSL/TLS connection) – i.e. more than stateless http web connectivity.
- User authenticates to the IdP (in context of User/RP session).
- The IdP provides evidence to the RP regarding the identity of the user who shares the session with the RP.
Federation

- **Federation** is an important notion in many real-world identity management systems.
- Enables two entities to link *(federate)* their respective identities for a single user.
- Enables identity management functionality, since allows parties to exchange information about a user.
- Federation process needs to be secure!

Identities

- User may have many identities (with identifiers) used with different relying parties:
  - employee may have an employee number for use with his/her employer;
  - citizen has one or more numbers for interactions with government;
  - user of Internet services (e.g. messaging) may have multiple names, one for each service provider.
Attributes

• More generally, users have many attributes, i.e. properties of them as individuals, e.g.
  – age;
  – sex;
  – nationality;
  – name;
  – credit card number.
• Can define identity to be set of all user attributes.
• Depending on service being provided, a relying party may need to know some but not all attributes.

Credentials

• Service (RP) may ask user to use credentials to prove ownership of identity, e.g.:
  – a password;
  – a biometric sample;
  – a public key certificate;
  – a MAC computed using a shared secret key;
  – a digital signature on a challenge provided by the service provider;
  – an anonymous credential.
Authorisation

• Once entity has been authenticated, the relying party needs to decide whether or not to grant the requested service.
• This is authorisation, i.e. is holder of this identity authorised to access service?
• Could, for example, be supported using server-held Access Control Lists (ACLs).

Privacy goals

• Requester of the service may wish to have a degree of privacy.
• For example, requester may not wish identity to become known to other entities.
• In principle can achieve this by only proving ownership of certain attributes.
• We next consider three different aspects of privacy.
Anonymity

- User may want to access service **anonymously**.
- **Anonymity** means no party will learn any identifiers of the user.
- Providing anonymity for free services is, in principle, ‘easy’.
- If payment needed, then an anonymous payment system is needed.
- True (‘absolute’) anonymity difficult, since revealing IP address (or any attribute) compromises anonymity.

Pseudonymity

- **Pseudonymity** is a lesser form of anonymity
- User reveals special identifier to the service provider – a **pseudonym**.
- Typically, new pseudonyms will be generated regularly, i.e. pseudonyms are often short-lived.
Unlinkability

- **Unlinkability** is a privacy property required to support the use of pseudonyms.
- Two pseudonyms are unlinkable if a third party cannot tell whether or not they belong to same user.
- Absolute unlinkability often difficult to achieve, since authorisation process may reveal information about user.

Agenda

1. Authentication-as a service
2. Privacy (theory) versus reality – a history
3. Where are we now?
4. Where would we like to be?
5. What can we do?
Passport

• In 2000, Microsoft introduced Passport.
• It provided an SSO service for Passport-registered users to Passport-registered SPs.

Passport operation 1

• SSL/TLS used to protect the User host/Passport server and User host/RP channels.
• RP host redirects User browser to the Passport server (i.e. the IdP).
• IdP checks for Ticket Granting Cookie (TGC) in User host – if one found which checks correctly then OK.
Passport operation II

- If not, then User authenticated and TGC created and stored on User host.
- The IdP now uses the TGC to create a set of cookies encrypted using the RP’s secret key.
- User browser redirected back to RP, which reads the cookies.

Vulnerabilities and (lack of) privacy

- Passport is subject to redirection attacks where a malicious RP redirects the User host to a fake IdP.
- Fake IdP can then capture user authentication information.
- Attack made pointless if a ‘one time’ user authentication method used.
- Clearly Passport not anonymous or unlinkable, since Microsoft learns everything.
Negative reactions

- There was a huge negative reaction to Passport.
- This mainly centred around the fact that Microsoft would know who was logging in to which sites.
- Microsoft promised to protect this data, but Passport was soon effectively dead.
- Passport was withdrawn as an SSO service – it lives on as Windows Live ID.

Liberty Alliance and Kantara

- The **Liberty Alliance** was a consortium of companies interested in SSO and identity management.
- It published a series of specifications for an ‘open’ XML-based SSO system as an alternative to Passport.
- The **Kantara Initiative** succeeded Liberty Alliance (and inherited its specifications).
Other systems

- Two other public domain initiatives also merit mention:
  - SAML, an XML-based standard which supports federation, SSO, and attribute management;
  - Shibboleth, a system with similarities to SAML, also designed to enable federation and SSO.

- These systems:
  - offer some limited privacy features;
  - have had some limited use, but none has succeeded in a big way.

Passport fallout

- Microsoft’s experience with Passport was rather painful.
- They tried to become a global identity provider without any privacy protection.
- Idea failed – main lesson Microsoft took is that there will never be such a global identity provider (at least without privacy protection).
- How wrong they were!
CardSpace

- Microsoft’s next big idea was CardSpace.
- CardSpace idea is to provide a unified way for (Windows) users to use many different underlying identity management systems.
- Key ideas here are:
  - provide a simple user model for identity;
  - enable users to control which identity is used for what purpose.

Simple user interface

- Users of CardSpace presented with simple user interface for managing identities.
- Employs a ‘card’ metaphor.
- Simple and appealing to use, and enables a degree of informed consent about privacy-related decisions.
- Also enabled a multi-provider identity landscape.
More failure

• Despite attractive interface and universal level of approval by experts, CardSpace failed to gain widespread use.
• It was quietly dropped by Microsoft in early 2011.

Anonymous credential systems

• These systems enable IdPs to issue credentials to users that can be used to prove selected attributes to RPs.
• System provides anonymity and unlinkability, even to issuing IdP.
• That is, even an IdP witnessing the attribute-proof process, cannot match this to an instance of credential issue.
Existing systems

- There are two widely discussed and implemented anonymous credential systems:
  - U-Prove (Brands);
  - Idemix (Camenisch et al.) – multi-use property.
- Both are subtle cryptographic constructs that build on pioneering work by Chaum.
- Both systems have been extensively analysed and developed, including by the ABC4Trust European project.

Practical impact

- Although these systems have ‘ideal’ privacy properties and implementations exist (and trials have been conducted), in the real world their impact is minimal.
- This may be because they are difficult to implement without installing special software at the client.
- Perhaps the magic bullet is a solution that can be used with a regular, unenhanced, browser...
Information Security Group

Agenda

1. Authentication-as a service
2. Privacy (theory) versus reality – a history
3. Where are we now?
4. Where would we like to be?
5. What can we do?

Internet SSO

• If Passport and other schemes failed because of a lack of user privacy, then we might have expected an anonymous credential system to be in widespread use by now ...
• ... or at least a system like CardSpace which offered users flexibility in choice of the trusted party and some privacy features.
• But no ...
Internet SSO is a reality

- Many (most?) sites requiring login offer AaaS, e.g. provided by Facebook and Google (e.g. via a *login with Facebook* button)
- Such services are almost all based on the OAuth 2.0 protocol.
- OpenID Connect is increasingly used (which is OAuth 2.0 based).

What is OAuth?

- OAuth (Open Authorisation) is an identity management scheme.
- Work began in 2006, to support Twitter’s OpenID implementation.
- OAuth 1.0 protocol published in 2010 as RFC 5849.
OAuth 2.0

• Specifications published in 2012 in three parts:
  – Framework = RFC 6749,
  – Bearer Token Usage = RFC 6750, and
  – Threat Model = RFC 6819.

• Bearer tokens are used by client browsers in HTTP requests to access OAuth 2.0 conformant RPs.

Facebook implementation

• OAuth 2.0, published in 2012 (RFC 6819), is being widely used as the basis of SSO services, e.g. for Facebook Connect.

• Enables Internet SPs to access personal information held by Facebook (with user consent), without user handing over Facebook password.
OAuth design goals

• Original goal of OAuth (1.0 & 2.0) not SSO.
• OAuth allows a Client application to access information (belonging to a Resource Owner) held by a Resource Server, without knowing the Resource Owner's credentials.
• Also requires an Authorization Server, which, after authenticating the Resource Owner, issues an access token to the Client, which sends it to the Resource Server to get access.

Use for SSO

• When used to support SSO:
  – IdP = Resource Server (stores user attributes) + Authorization Server (authenticates user);
  – RP = Client;
  – User = Resource Owner (owns user attributes);
  – UA = web browser.
• Access token used to provide SSO service (not really what it was intended for).
• OAuth supports four ways for a Client to get an access token.
• Of these, we focus on Authorization Code Grant.
OAuth 2.0/SSO – data flows

1. User clicks button on RP website, and UA sends HTTP request to RP.
2. RP sends OAuth 2.0 authorization request to UA, optionally including state variable (used to maintain state between request and response).
3. UA redirects request to IdP.
4. If necessary, IdP authenticates User.
5. IdP generates authorization response containing code (an authorization code), and the state value, and sends it to UA.
6. UA redirects response to RP.
7. RP sends access token request to IdP (directly) containing code and client_secret (shared by IdP and RP).
8. IdP checks request values and responds to RP with access token.
9. RP uses access token to retrieve user attributes (specifically the IdP user identifier) from IdP.

OAuth 2.0 – identity federation 1

- OAuth 2.0 specifications do not provide a standardised approach to identity federation.
- Not surprising given OAuth 2.0 not really designed for SSO.
- Commonly used (ad hoc) means of federation involves the RP binding the user-RP account to the user-IdP account, using the unique user ID generated by the IdP.
- The IdP account ID is fetched from the IdP in step 9 of previous slide.
OAuth 2.0 – identity federation II

- After receiving the access token (step 8), RP retrieves the user’s IdP account ID.
- RP then binds user’s RP account ID to user’s IdP account ID.
- One method of achieving binding is:
  - user initiates binding after logging in to RP;
  - user required to log in to IdP;
  - user grants permission for binding;
  - RP completes binding process.

OAuth – issues I

- OAuth uses http redirects.
- So open to phishing attacks.
- This technology is used to avoid need to install special software on client.
- Enables simple deployment of service.
- Systems using special client software (like CardSpace) have almost no practical use, despite offering greater security.
OAuth – issues II

• OAuth 2.0 has been critically examined by a number of authors.
  – Frostig & Slack (2011) found a Cross-Site Request Forgery (XSRF) attack in the Implicit Grant flow of OAuth 2.0.
  – Wang, Chen & Wang (2012) found a logic flaw in a range of SSO implementations.
  – Sun & Beznosov (2012) found flaws in OAuth 2.0 implementations.
  – Li & Mitchell (2014) found range of flaws in federation process for widely used Chinese language implementations.
  – ... more since then ...

Attack countermeasures

• OAuth 2.0 specifications recommend use of state parameter in authorization request & response to protect against CSRF attacks.
• For it to work state must be non-guessable.
• Otherwise attacker could include guessed value in a CSRF-generated fraudulent authorization response.
• Unfortunately, many real-world RPs either omit the state or use it incorrectly.
Building on OAuth 2.0

- OpenID Connect 1.0 is built as an *identity layer* on top of OAuth 2.0. Used by Google.
- Adds extra functionality aimed specifically at SSO.
- Adds a new type of token to OAuth 2.0, namely the *id token* [a JSON web token].
- The *id token* contains claims about authentication of end user – generated by entity known as OpenID Provider (OP) [=IdP].
- It is digitally signed by the OP.

Vulnerabilities

- Unfortunately, just like with OAuth 2.0, RP implementations are often vulnerable.
- A recent large-scale study found that many websites do not properly implement use of the *state* variable, critical to avoiding CSRF attacks.
- Other sites do not use the *id token* properly.
How about privacy?

- OAuth 2.0 and OpenID Connect are about as non-private as they could be:
  - no anonymity;
  - limited pseudonymity;
  - no unlinkability of pseudonyms;
  - IdP knows everything.
- We do, in principle, have a choice ... but really just Google or Facebook!

Summary

- So we are widely using systems which are both:
  - easy to implement poorly, resulting in significant vulnerabilities to end users;
  - about as non-privacy-respecting as they could be.
- So Microsoft got it wrong?
  - major difference (Passport vs. OAuth 2.0) is not technical but in the business model;
  - Microsoft sought to get RPs to pay for use of Passport, whereas Facebook/Google monetise the data they gather and hence offer a service free to all.
Agenda

1. Authentication-as a service
2. Privacy (theory) versus reality – a history
3. Where are we now?
4. Where would we like to be?
5. What can we do?

Why are we where we are?

- It seems that users care much more about convenience than privacy.
- This is despite very widely discussed concerns about privacy-related behaviour of the major IdPs (independently of SSO service).
- OAuth 2.0-based solution also very easy to adopt for RPs.
- Users can adopt SSO with no software installation.
It’s a tough world out there …

• Even though privacy is being increasingly regulated, it’s still the wild west out there.
• User data is highly valuable, and offering ID management services is a useful source of such data.
• So there is plenty of potential revenue to develop and support free-to-use ID management solutions which are not privacy-respecting.

What we would ideally have

• Of course, in an ideal world and all else being equal, we would all enjoy the benefits of SSO in a privacy-respecting way.
• Technically this is a solved problem – anonymous credential systems work!
• But this is not wholly (or even mainly) a technical problem ...
But we don’t want to pay for it ...

- Cost includes:
  - actual financial cost (charge) to user;
  - cost in terms of work for user, e.g. installing special software, setting up special systems ...
  - work cost to RP.
- However, business model to enable deployment of a free-to-use service in a privacy-respecting way is not obvious.
- Can we find a middle way?

Agenda

1. Authentication-as a service
2. Privacy (theory) versus reality – a history
3. Where are we now?
4. Where would we like to be?
5. What can we do?
Low-cost privacy-respecting AaaS

- The heading says what we want!
- How do we get there?
- Regulators could make it happen, but:
  - this doesn’t seem likely to happen any time soon.
- If current solutions work, then why should RPs change?
- If IdPs don’t get user data, then why should they provide a free service?

Low-cost for users

- Cost-free means:
  - no financial cost for user;
  - no need for users to install any special software or conduct any complex registration processes.
- Maybe this is too demanding – perhaps users might be prepared to install software if it is made simple enough?
Low-cost for Relying Parties

- Current solutions are free for RPs, and development task is simple (although error-prone).
- RPs will not want to adopt a solution if there is a significant charge or implementation is complex.

Privacy-respecting

- Is there a viable middle path between:
  - current state – no privacy at all;
  - ideal solution – e.g. as provided by anonymous credentials?
- That is, can we work towards solutions which enable useful data to be gathered by IdPs without handing over a complete behavioural history for users?
Choice

• Choice of IdP is currently very limited.
• There may be several IdPs, but majority of RPs only support one or two prominent IdPs.
• Can we engineer a solution which enables RPs to easily support multiple IdPs?

How?

• Probably need to evolve from where we are.
• Can we design OAuth 2.0-like, but privacy-respecting, systems which allow easy deployment for users (install-free) and RPs.
• Are there things users can do with current deployments to reduce their privacy exposure to IdPs?
• Are there simple systems that can be implemented (e.g. browser plugins) that enhance user privacy when using existing SSO systems?
Questions?

• Thank you for your attention.