Active-client based identity management

Chris Mitchell
Royal Holloway, University of London
www.chrismitchell.net

Acknowledgements

- This is joint work with Haitham Al-Sinani, also of Royal Holloway.

Agenda

- Introduction
- A problem – not another new protocol
- Active-client identity management
- A solution – not another new protocol!
- Mappings to specific systems
- Other functionality
- Concluding remarks

User authentication

- The need for authentication of human users is a fundamental security requirement (perhaps ***the*** fundamental requirement).
- Despite its importance, it is almost universally acknowledged that providing user authentication remains a huge practical problem.

Context

- Identity management **within an organisation** (or, more generally, in a **managed environment**) is, to a large extent, a solved problem.
- In Windows, for example, Active Directory provides identity management functionality.
- The focus here is on identity management for the unmanaged Internet end user (e.g. you or me at home).
- With the growth in mobile Internet access via a range of devices, this is a problem of ever-increasing importance.

Passwords

- In practice, as many observers have noted, we are still using passwords for almost everything.
- Again, as widely acknowledged, the use of passwords has many shortcomings, not least because users today have so many Internet relationships, all needing authentication.
- In such a context, password re-use and use of weak passwords are almost inevitable.

Solutions

- Usual approach to this problem is to propose yet another new way of doing user authentication, e.g. using a cryptographic protocol.
- However, perhaps there are already enough good technological solutions?
- Maybe the problem is adoption of the solutions we already have? How do we fix this?
- Of course, this is partly a business case and sociological issue, but maybe it is also a problem which requires new technical thinking?

New thinking required

- It is easy for those of us doing technical research to claim that this is not our problem.
- We provide the technology and the commercial world should just get on with it.
- However, life is not so simple.
- We as academics should be thinking about how to devise technological solutions which are easier to adopt.
- Key issues for easy adoption are transparency, ease of use, and backwards compatibility.

Identity management

- Identity management systems have been designed to simplify user authentication.
- Such a system enables an **Identity Provider (IdP)** to support authentication of a User (and assertion of user attributes) to a **Service Provider (SP)**.
- Recent years have seen the emergence of a wide range of such systems, e.g. OpenID, Liberty, Shibboleth, CardSpace and OAuth.
- Each has its own set of protocols governing communications between the main parties.

Infrastructure support

- As well as its own protocols, each system may also have a unique supporting infrastructure, including public key certificates, shared keys, passwords, etc.
- Some systems have gained traction recently, e.g. Facebook's adoption of OAuth (Facebook Connect), and significant use of OpenID.
- However, the systems that have been most widely used are also those which have the most significant problems (e.g. phishing vulnerabilities).

Agenda

- Introduction
- A problem – not another new protocol
- Active-client identity management
- A solution – not another new protocol!
- Mappings to specific systems
- Other functionality
- Concluding remarks

Well known problems

- We start by reviewing some of the well known problems with existing authentication solutions.
- These problems apply very broadly.

The phishing threat

- Many identity management systems are susceptible to phishing attacks, in which a malicious (or fake) SP redirects a user browser to a fake IdP.
- The user then reveals to the fake IdP secrets that are shared with a genuine IdP.
- This arises because, in the absence of a system-aware client agent, schemes rely on browser redirects.

Lack of consistency

- One huge problem faced by any user is that the user experience of every identity management system is different.
- We all know that users fail to make good security decisions, even when confronted with relatively simple decisions.
- The lack of consistency is likely to make the situation much worse, with users simply not understanding the complex privacy- and security-relevant decisions they are being asked to make.

Privacy

- When using third party IdPs which provide assertions about user attributes, there is a danger that a user will damage their privacy by revealing attributes, i.e. **Personally Identifiable Information (PII)**, unintentionally to an SP.
- This is a threat when using systems like OAuth (e.g. as instantiated by Facebook Connect).
- In general, getting privacy settings right is highly non-trivial.

Another new infrastructure?

- It is tempting to try to devise another new scheme which has the practical advantages of OAuth and OpenID, but yet provides robust protection against phishing and privacy loss.
- That is, devise a client-based scheme with the user convenience of other systems, but which somehow avoids the fate of CardSpace.

Problems

- However, it seems that a new solution is:
  - unlikely to succeed when others (some with a great deal of inertia and incorporating very nice features, e.g. CardSpace) have failed;
  - likely to create yet another different user experience, increasing the likelihood of serious mistakes.
- Thus maybe this is not the right approach.

A new approach?

- The goal of this talk is to consider a new approach to the problem.
- It does not involve proposing any new protocols or infrastructures.
- The goal is to try to make it easier to use existing systems, and also to make their use more secure (less prone to phishing) and privacy-enhancing (consistent interface and explicit user consent).

Agenda

- Introduction
- A problem – not another new protocol
- Active-client identity management
- A solution – not another new protocol!
- Mappings to specific systems
- Other functionality
- Concluding remarks

Active and passive clients

- Identity management systems can be divided into two broad classes:
  - **passive-client systems** (e.g. OpenID), which assume only that the client system has a browser;
  - **active-client systems** (e.g. CardSpace), where special software must be installed on the client to support the identity management system.

Active-client systems

- Here a browser incorporates an ‘active client’, which acts as an intermediary between SPs and IdPs, and is aware of the identity system.
- All SP-IdP communications involve this active client.
- The active client might prompt the user to select a digital identity, choose an IdP, review an identity token created by the IdP, and/or approve a transaction.
- Phishing attacks are mitigated.
- The active client can also give a consistent user experience and a greater degree of user control.
- Examples include CardSpace and Liberty (when using a Liberty-enabled client (LEC)).

Passive-client systems

- In such a scheme, the browser is HTTP-redirected by an SP to an IdP (and vice versa).
- No direct client control over site with which it is communicating.
- A major disadvantage is that a malicious SP (e.g. a phishing site) can redirect the browser to a fake IdP (e.g. to fraudulently obtain user credentials).
- Examples include OpenID, Liberty (browser-post profile), Shibboleth, and Facebook Connect (OAuth).

Agenda

- Introduction
- A problem – not another new protocol
- Active-client identity management
- A solution – not another new protocol!
- Mappings to specific systems
- Other functionality
- Concluding remarks

A universal active-client solution

- The scheme we propose involves an active-client user agent.
- This is a single tool which supports a wide range of ID management systems yet provides a single interface to the user.
- The consistent user interface should maximise user understanding of what is happening (and reduce risk of errors).
- It also avoids the need for passive browser redirects, hence mitigating phishing attacks.

Motivation for scheme

- One motivation for the scheme comes from considering CardSpace (and its open source ‘twin’, Higgins).
- Before proceeding we thus need to spend a bit of time describing CardSpace.

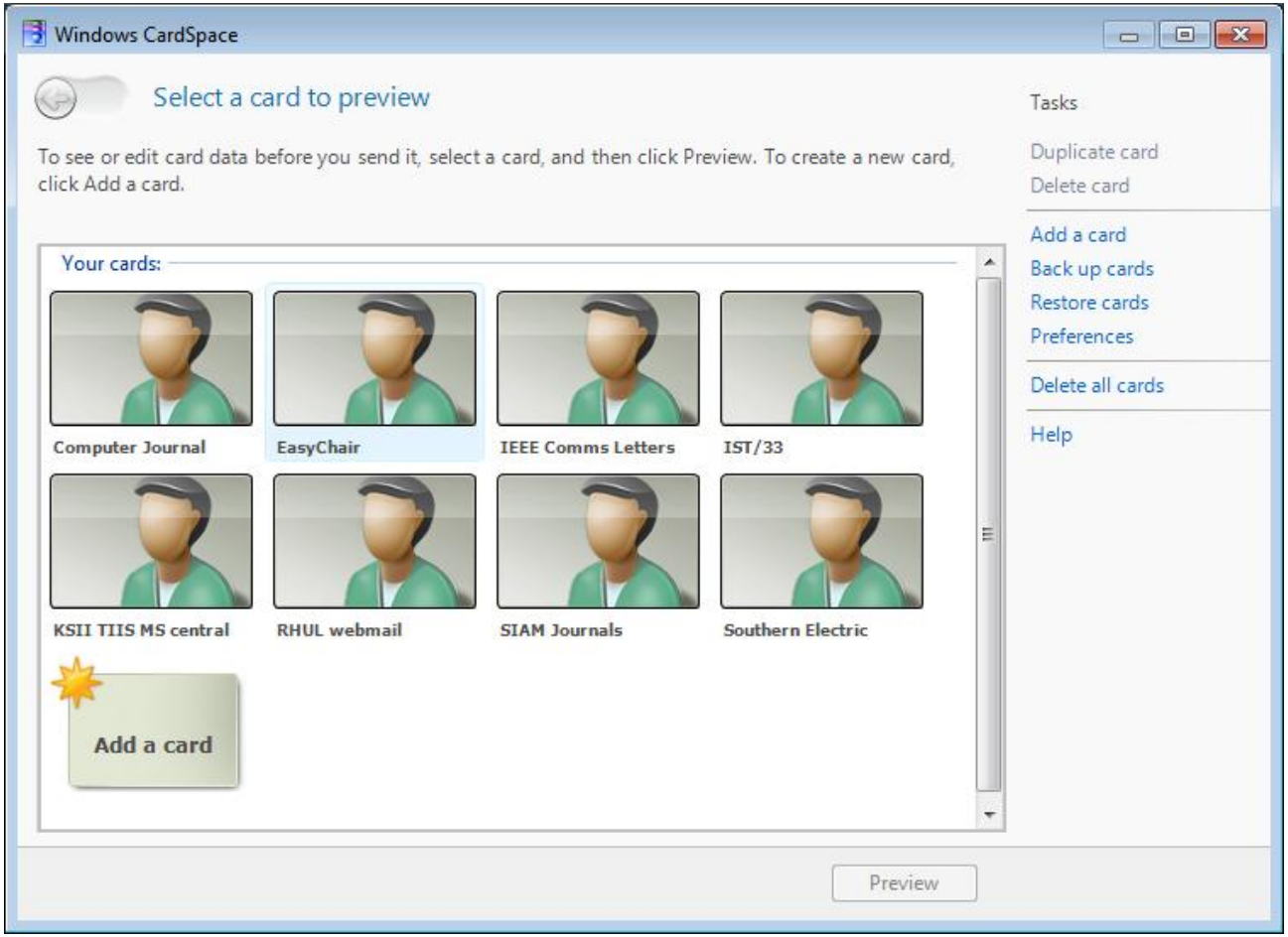
CardSpace: a brief description

- CardSpace acts as client-based agent, and provides a consistent card-based user interface.
- That is, sets of user credentials (relationships with IdPs) are represented to users as cards.
- CardSpace also defines a set of protocols for interactions between IdPs, Clients (user machines) and SPs.

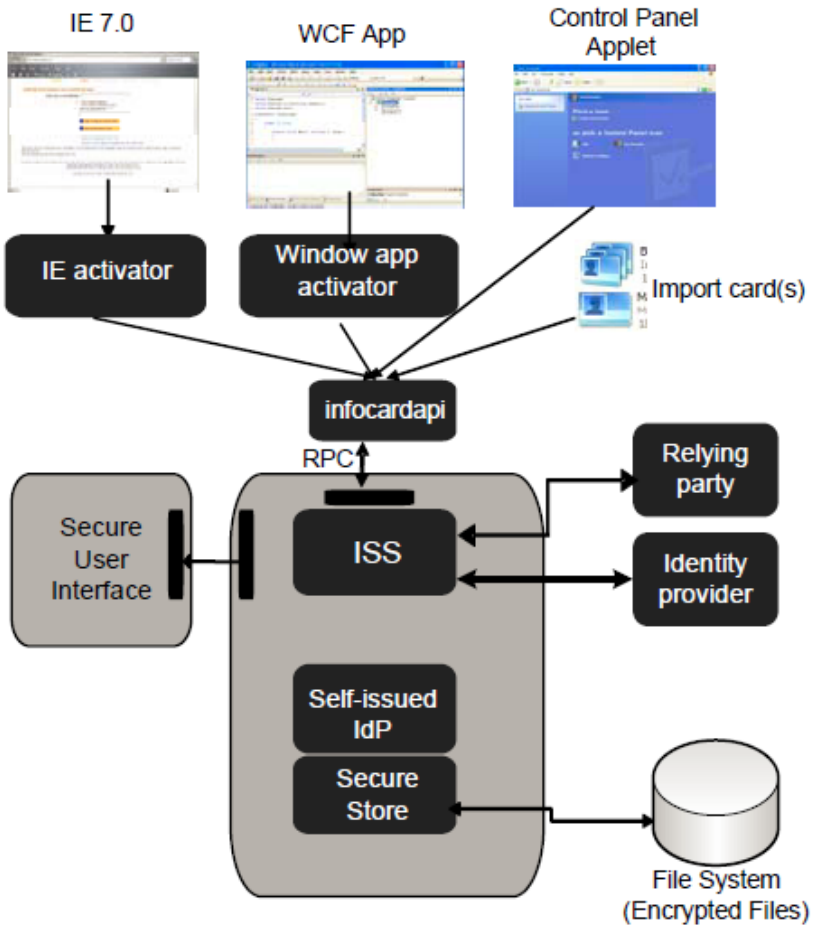
CardSpace operation

- The user, interacting with the browser via the *identity selector*, may have identities issued by one or more IdPs.
- Each identity is represented by an *InfoCard* held by the identity selector, and this InfoCard is the means by which the user interacts with the identity selector to choose which identity to use.
- Each IdP runs a Security Token Service (STS), to generate security tokens.
- A *Self-issued Identity Provider* may be provided by a client platform to allow use of self-issued tokens.

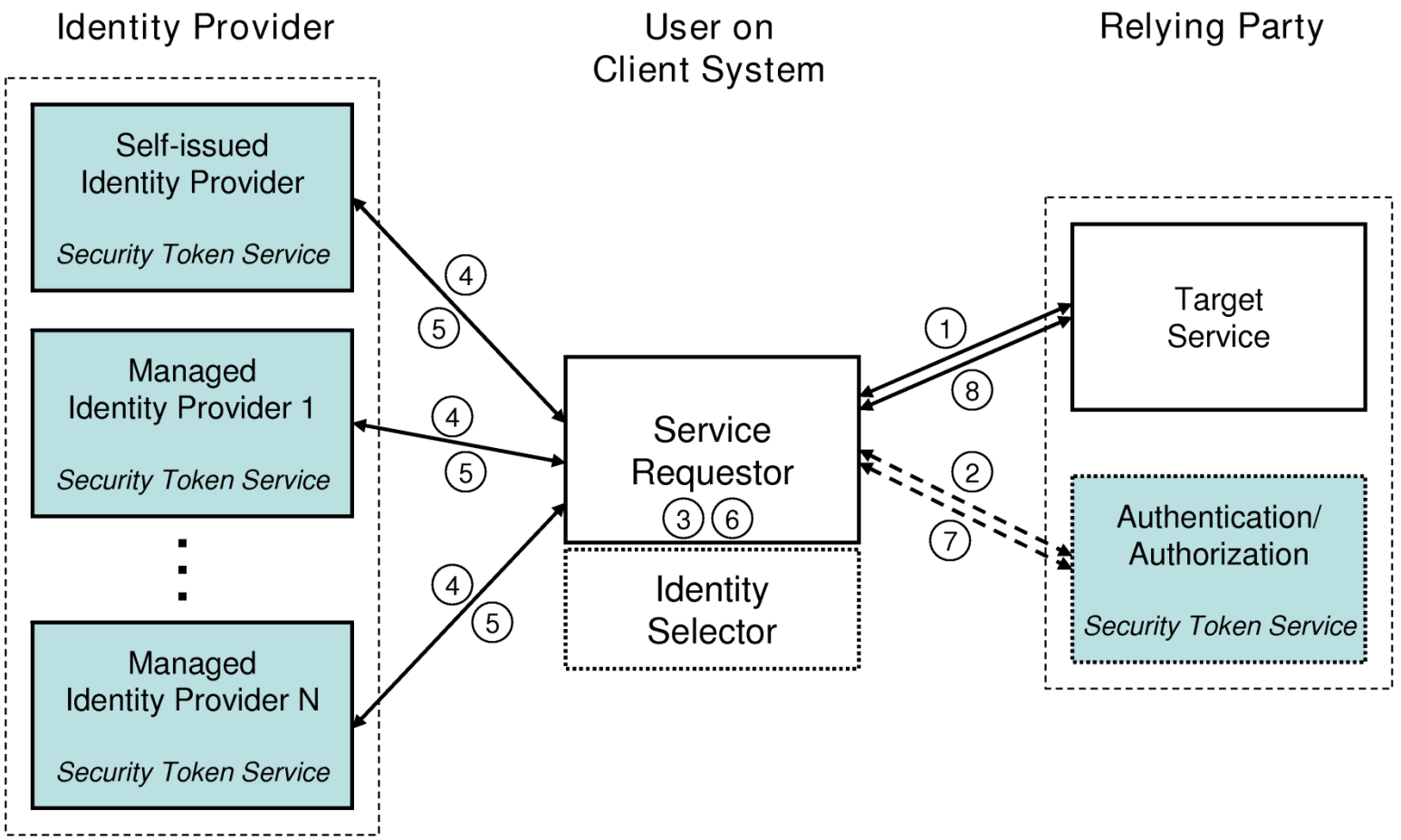
CardSpace Identity Selector



CardSpace architecture



CardSpace interaction model



Operation I

1. Service requester gets the security policy of the target service. We suppose that the policy requires the requester to get a token issued by an IdP's STS.
2. (optional) The service requester gets the policy of the authentication/authorisation STS (to determine properties of required token).
3. The requester asks the identity selector to provide a security token meeting the policy of the target service.
4. The identity selector first gets the user to choose an InfoCard capable of meeting the target service requirements, and then gets the policy of the selected IdP's STS.

Operation II

5. The InfoCard indicates the method to be used to authenticate the user to the IdP STS; the user sends an appropriate credential to the IdP STS, and the identity selector gets back a token.
6. The token is given to the service requester.
7. (optional) The service requester presents the token to the STS, which generates a token for the target service.
8. The service requester presents the token to the target service to get access.

User authentication

- Before issuing a token, an IdP will typically need to authenticate the user.
- This user authentication takes place via the local CardSpace software
- Two key advantages:
  - provides consistent user experience;
  - limits possibility of phishing attacks.

An observation

- The user interface of CardSpace and the underlying communications protocols are not inherently tied together.
- Why not keep the simple/intuitive user interface, and use it as the front end for a tool which manages user credentials in a consistent way regardless of the underlying identity management system?

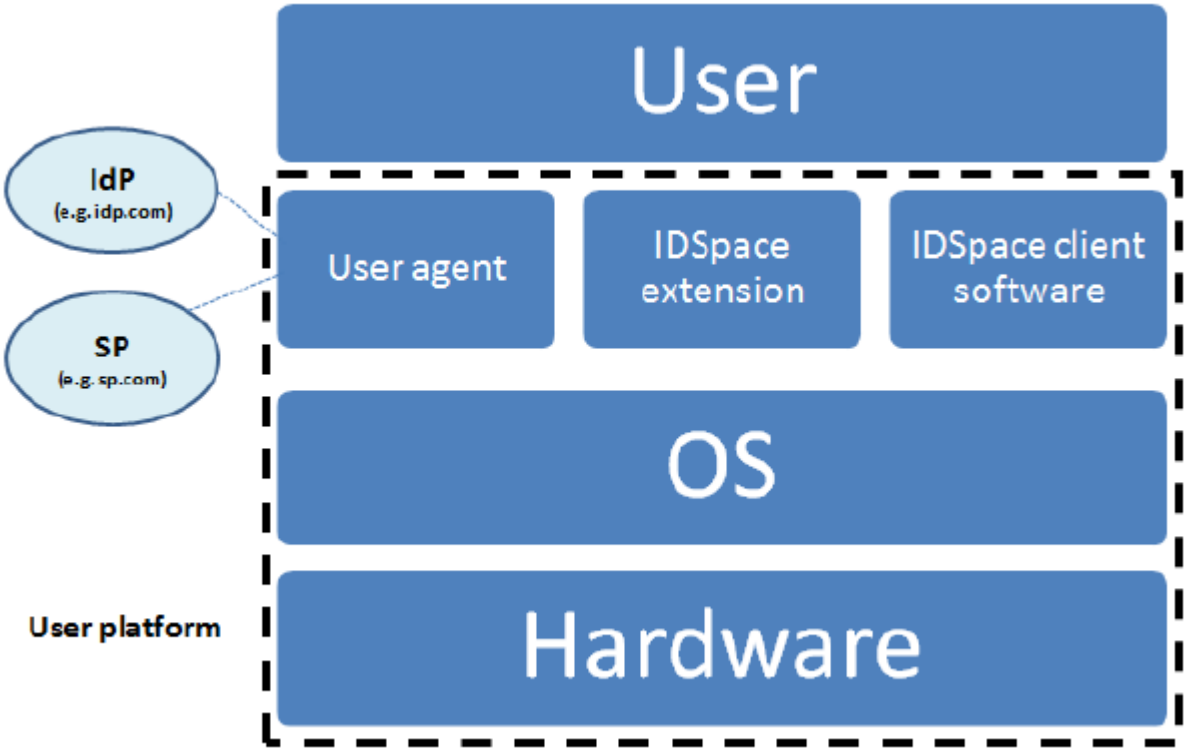
An observation (continued)

- Credential sets could identify with which identity management system (or systems) they should be used.
- For example, each credential set could be stored as a self-describing XML document.
- Indeed, these credential sets could include username/password pairs.

A universal client adapter

- We now describe our scheme, called **IDSpace** (in homage to role CardSpace played in developing the idea).
- IDSpace has two main components – a browser plugin (the **IDSpace extension**) and a separate piece of software (the **IDSpace client software**).
- Both execute on the user platform.

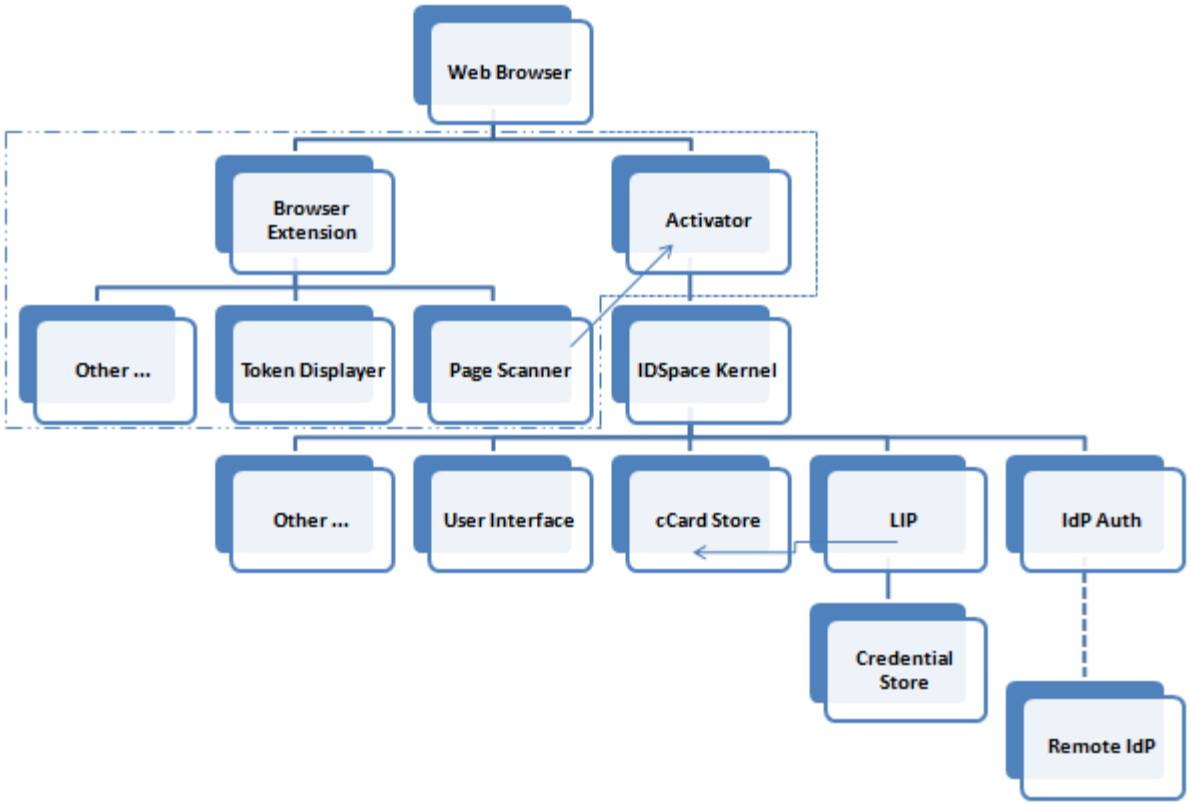
IDSpace high level architecture



IDSpace components

- The IDSpace system possesses a number of components, as shown on the next slide:
  - **Card Selector**: presents a card-based interface to user to enable choice of IdP and credentials;
  - **cCard store**: stores credential cards (cCards) containing credential info (used by Card Selector);
  - **Credential store**: separate secure storage for keys, passwords, attributes, etc., associated with cCards;
  - **Kernel**: core component controlling system operation;
  - **Page Scanner**: scans web pages;
  - **Activator**: activates the Card Selector.

IDSpace client architecture



Sketch of protocol I

- The IDSpace works as follows.
  1. User browses an SP login page.
  2. The **IDSpace Page Scanner** examines the page to see which identity systems are supported.
  3. The user is offered a choice (e.g. via right clicking) of systems to use. [There are many options for implementing this step.]
  4. The **IDSpace Activator** activates the **IDSpace Card Selector**.

Sketch of protocol II

5. The **IDSpace Data Transporter** passes metadata (e.g. selected identity system, SP identity, SP policy) to the **IDSpace Kernel**.
6. The **Kernel** interacts with the **Card Selector**, which allows the user to choose a cCard (and possibly an identity system).
7. The **Kernel** interacts with the selected IdP to obtain a token for use by the SP. If necessary the IdP authenticates the user via the **Card Selector**.
8. The **Token Displayer** asks the user for permission to send the token to the SP.

User experience

- The user interacts with a single piece of software (the **Card Selector**) regardless of which underlying system is in use.
- This enables the user to use a single simple interface to:
  - choose (and manage) credentials;
  - be authenticated to an IdP;
  - give consent for release of PII to an SP.

Agenda

- Introduction
- A problem – not another new protocol
- Active-client identity management
- A solution – not another new protocol!
- Mappings to specific systems
- Other functionality
- Concluding remarks

Mappings

- Each identity system operates differently, and hence each system maps slightly differently onto IDSpace.
- Main relevant characteristic is whether an identity system is:
  - passive-client based, or
  - active-client based.
- We look at these two cases.

Role of IDSpace (active-client case)

- In such a case the IDSpace client software plays the role of the active client.
- IDSpace acts as a type of ‘universal’ client, integrating the various systems and handling credential information and user authentication in a unified and consistent way.
- Thus, for example, IDSpace can transparently replace the Microsoft CardSpace software.

Role of IDSpace (passive-client case)

- The IDSpace client software essentially converts a passive-client (redirect-based) system into an active-client system.
- Redirects are no longer under the control of the SP (and IdP).
- The IDSpace client also manages authentication of the user to the IdP.
- The operation of IDSpace is completely transparent to the IdP and SP.

Features

- Regardless of the ID system protocols supported by the SP and IdP, IDSpace is transparent to both parties.
- That is, no parties (except the user who installs and uses the software) need to be aware of its presence.
- As long as the SP and IdP share at least one identity system, then IDSpace operation is possible.

Agenda

- Introduction
- A problem – not another new protocol
- Active-client identity management
- A solution – not another new protocol!
- Mappings to specific systems
- Other functionality
- Concluding remarks

Password management

- Password managers are commonplace.
- However, apart from schemes built into browsers, they do not appear to be widely used.
- **PassCard** is a browser-plugin-based scheme we have described previously which allows CardSpace to be used as a password manager.
- The idea behind PassCard could readily be extended to allow IDSpace to provide password management facilities (with username/password pairs being represented as cCards).

Moving into the cloud

- Cloud-based identity management systems offer some advantages over client-based schemes (not least portability).
- Indeed, cloud-based variants of CardSpace have been proposed in which InfoCards are cloud-based.
- One possible extension of IDSpace would be to make it cloud-based.

Identity system interoperation

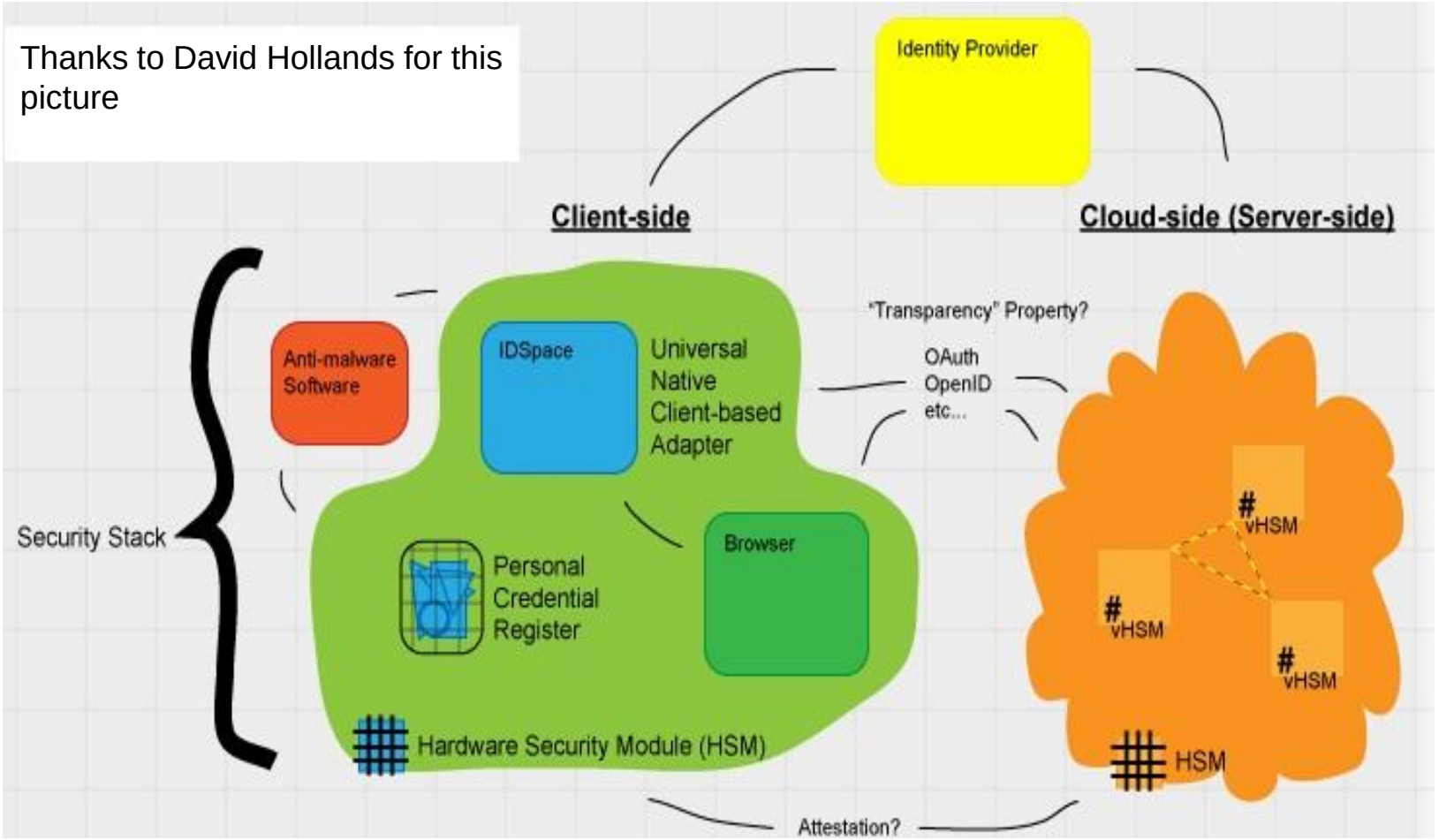
- In other work, we have proposed and prototyped a series of client-based (browser plug-in based) schemes to support interoperation between an IdP and an SP supporting different identity systems.
- This functionality could also be supported by an IDSpace client.

Role of client agents

- IDSpace is just one example of the potential power of a client-based security agent.
- There are many other ways in which client-hosted software might be used to assist users in making difficult security-relevant decisions when using Internet services.
- Indeed, this paper is really intended to encourage the research community to think more about using client-based schemes to improve user security.

IDSpace – the bigger picture

Thanks to David Hollands for this picture



Agenda

- Introduction
- A problem – not another new protocol
- Active-client identity management
- A solution – not another new protocol!
- Mappings to specific systems
- Other functionality
- Concluding remarks

IDSpace works!

- A preliminary prototype of IDSpace has recently been built by my co-author (Haitham Al-Sinani), and is still under development.
- Unfortunately it is not yet in a demonstrable state.
- However, we soon hope to make available a usable prototype.

Related work

- Copies of published papers on PassCard and the various identity management interoperation schemes can be found on my home page:

  www.chrismitchell.net

- Many are also available as RHUL technical reports:

  www.ma.rhul.ac.uk/tech

Questions?

- For further information please contact:
  - Haitham Al-Sinani
    Haitham.Al-Sinani.2009@live.rhul.ac.uk
  - Chris Mitchell
    me@chrismitchell.net
- Address:

Information Security Group
Royal Holloway
University of London
Egham TW20 0EX
UK