Java is a trademark of Sun Microsystems, Inc.

Using REST and WS-* in the Cloud

Doug Tidwell
IBM Corporation
dtidwell@us.ibm.com
ibm.com/cloud

Agenda

- A quick overview of cloud computing, REST and WS-*
- Accessing cloud services with REST and SOAP [Demo]
- Deploying REST services with JSR-311 [Demo]
- Accessing a cloud service w/single sign-on [Demo]
- Wrap-up

Java is a trademark of Sun Microsystems, Inc.

JavaOne℠

A quick overview…

…of cloud computing,
REST, WS-*

The cloud

- Cloud computing . . . is a style of computing where IT-related capabilities are provided ‘as a service,’ allowing users to access technology-enabled services ‘in the cloud’ without knowledge of, expertise with or control over the technology infrastructure that supports them.
  - From Wikipedia
- Everybody has a slightly different idea of what cloud computing really is.
  - *This definition has been edited 350+ times in 2009.*

The cloud is here to stay

- Extremely stupid idea of assuming the entire planet just can not be bothered with their own data (nor the security thereof). As always there will be some who think they 'need' this. I hope this whole cloud [stuff] just goes away. Logistically speaking it will never be anything but a waste of money.
  - Posted by ___*@gmail.com*, who apparently just can not be bothered with their own email (nor the security thereof).

REpresentational State Transfer

- Again, from Wikipedia:
  - Application state and functionality are resources
  - Every resource can be accessed via a URI
  - Simple interface: POST, GET, PUT, DELETE (think create, read, update, delete)
  - Cacheable, stateless protocol
- Based on Roy Fielding's seminal Ph.D. dissertation
- Everybody has a slightly different idea about what REST really is.
  - *The definition above has been edited 5,000+ times since I started reading this slide.*

WS-*

- We'll refer to WS-* as SOAP, WSDL and standards such as WS-Security, WS-ReliableMessaging, WS-Conversation, etc.
  - Proponents of REST often call this "Big Web Services."
- This means packaging data in a SOAP envelope, using WSDL to determine the service's interface and endpoint, and using the WS-* standards to add headers to the SOAP envelope as needed.

A plea for sanity

- It's easy to find avid proponents of REST or WS-* out there.
- As always, **the sane answer is to examine the two technologies and figure out which meets your needs.**
  - Maybe it's both.
  - David Chappell's blog post **"REST vs. WS-*: War is Over (If You Want It)"** is a good read. As are the responses.

Java is a trademark of Sun Microsystems, Inc.

Java

JavaOne℠

The state of the cloud

The state of the cloud

- Here are some common cloud services:
  - Amazon's Simple Storage Service (S3): SOAP and REST APIs
  - Google Search: REST API only, SOAP was discontinued in 2006
  - Flickr: Supports SOAP, REST, XML-RPC
  - Backpack: REST-ish
  - Twitter: REST only
  - Facebook: REST only
- If you're going to use cloud services, you'll probably have to deal with REST.

Demo

- We'll look at a couple of client applications that access data and applications running in the cloud.
  - One uses WS-*, one uses REST.
- These applications use services hosted by Amazon and Backpack.

Java is a trademark of Sun Microsystems, Inc.

JavaOne℠

JSR-311

Making it easy to deploy
a REST service

JSR-311/JAX-RS

- As the cloud matures, chances are you'll need to share data and services with partners.
- Chances are you'll need to deploy something with a REST interface.
- JSR-311, also known as JAX-RS, is a way of deploying Java code as a REST service.
- The spec makes it easy to deploy an annotated Java class as a resource accessible via HTTP.
- Annotations in your code define how different methods are addressed from a URL.

A sample REST service

- Defining the path for the service:

```
@Path("/warehouse/")
public class RestWarehouseServiceImpl
  implements WarehouseService {
```

- The URL for the service is `[baseURL]/warehouse/`.

A sample REST service

- Defining a method and parameters:

```
@GET
@Path("/query/{partNo}/")
public int queryPart(
   @PathParam("partNo") String partNo) {
   return warehouseService.
          queryPart(partNo);
}
```

- `[baseURL]/warehouse/queryPart/NC-3882/` returns the quantity of part NC-3882 in stock.

Running the service

```
JAXRSServerFactoryBean sf = new
  JAXRSServerFactoryBean();
sf.setResourceClasses
  (RestWarehouseServiceImpl.class);
sf.setResourceProvider
  (RestWarehouseServiceImpl.class,
   new SingletonResourceProvider(new
         RestWarehouseServiceImpl()));
sf.setAddress("http://ec2...aws.com/");
sf.create();
```

Demo

- A quick look at our warehouse service
  - Written using JSR-311
  - Deployed to an Amazon Machine Image running in the cloud
- We should get an integer when we load **`http://ec2...aws.com/warehouse/queryPart/NC-3882/`**.

Java is a trademark of Sun Microsystems, Inc.

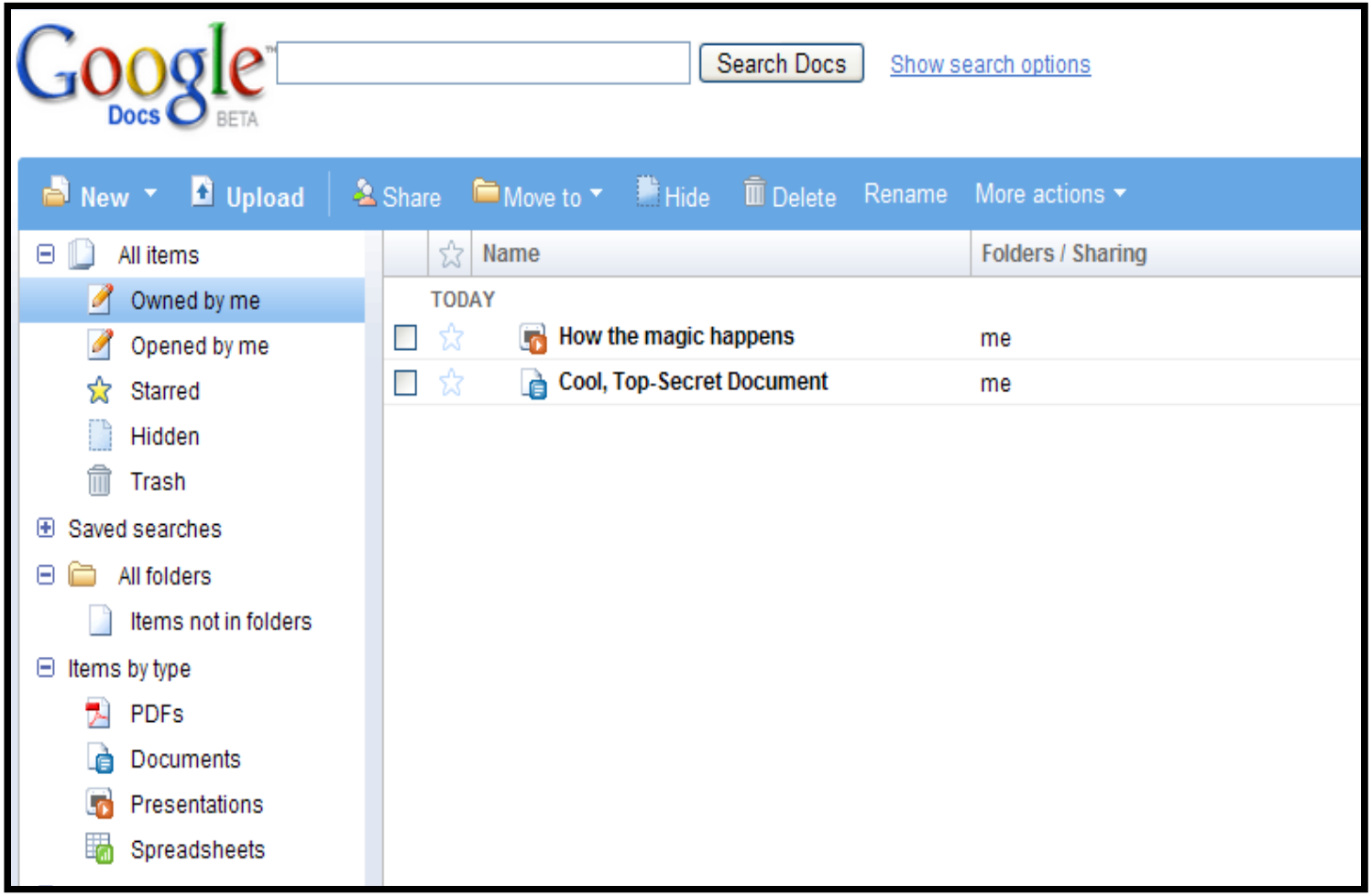
JavaOne℠

Accessing a cloud service with single sign-on

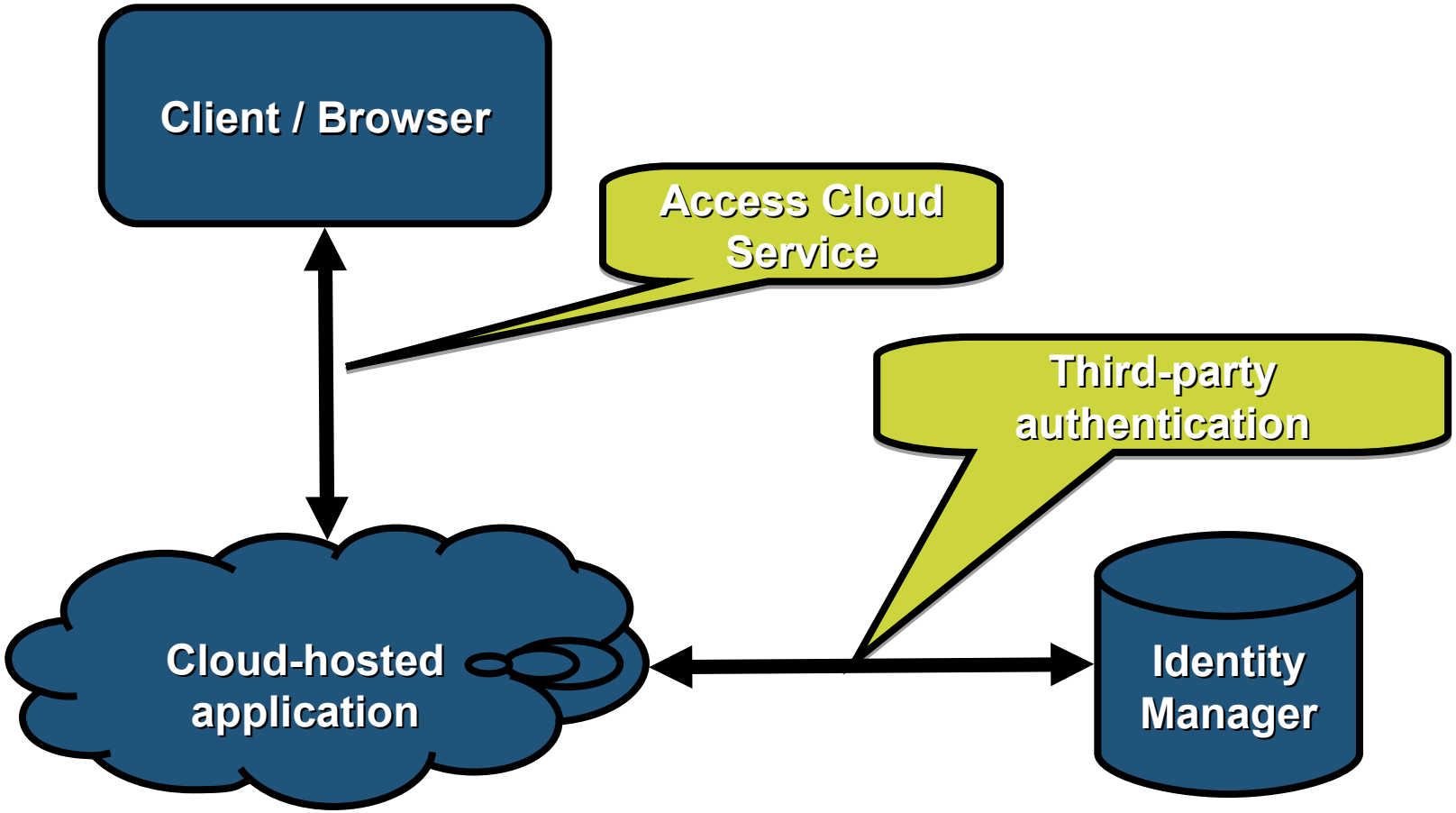
Demo

- We'll look at a demo of a private Google Docs application hosted in the cloud.
  - We need single sign-on
  - We have an existing identity infrastructure
  - We want to use standards where we can (SAML tokens, in this case)
- The technique here uses HTTP, so it could be used with either REST or WS-*.

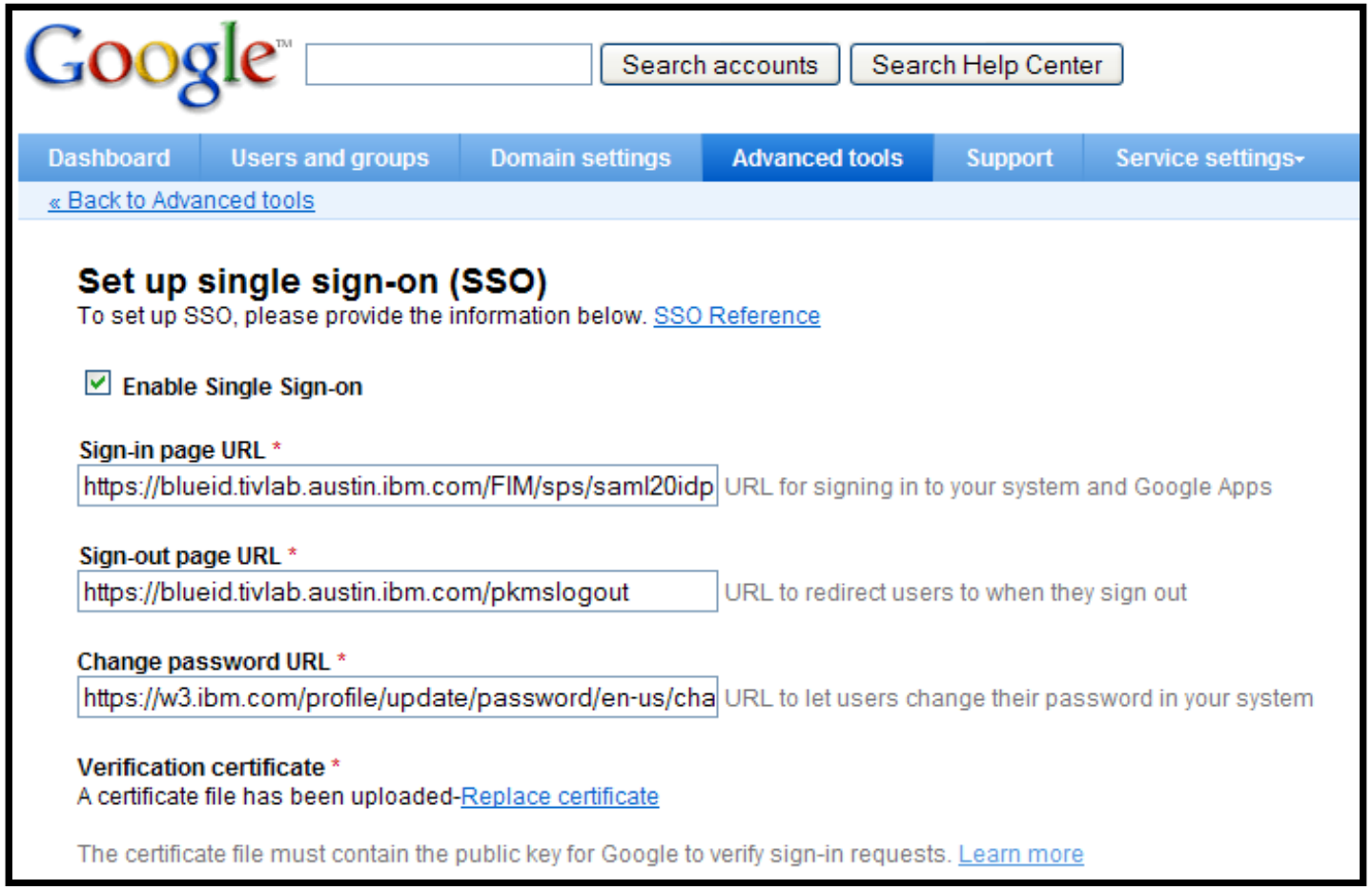
A private Google Apps site



Federated identity/single sign-on



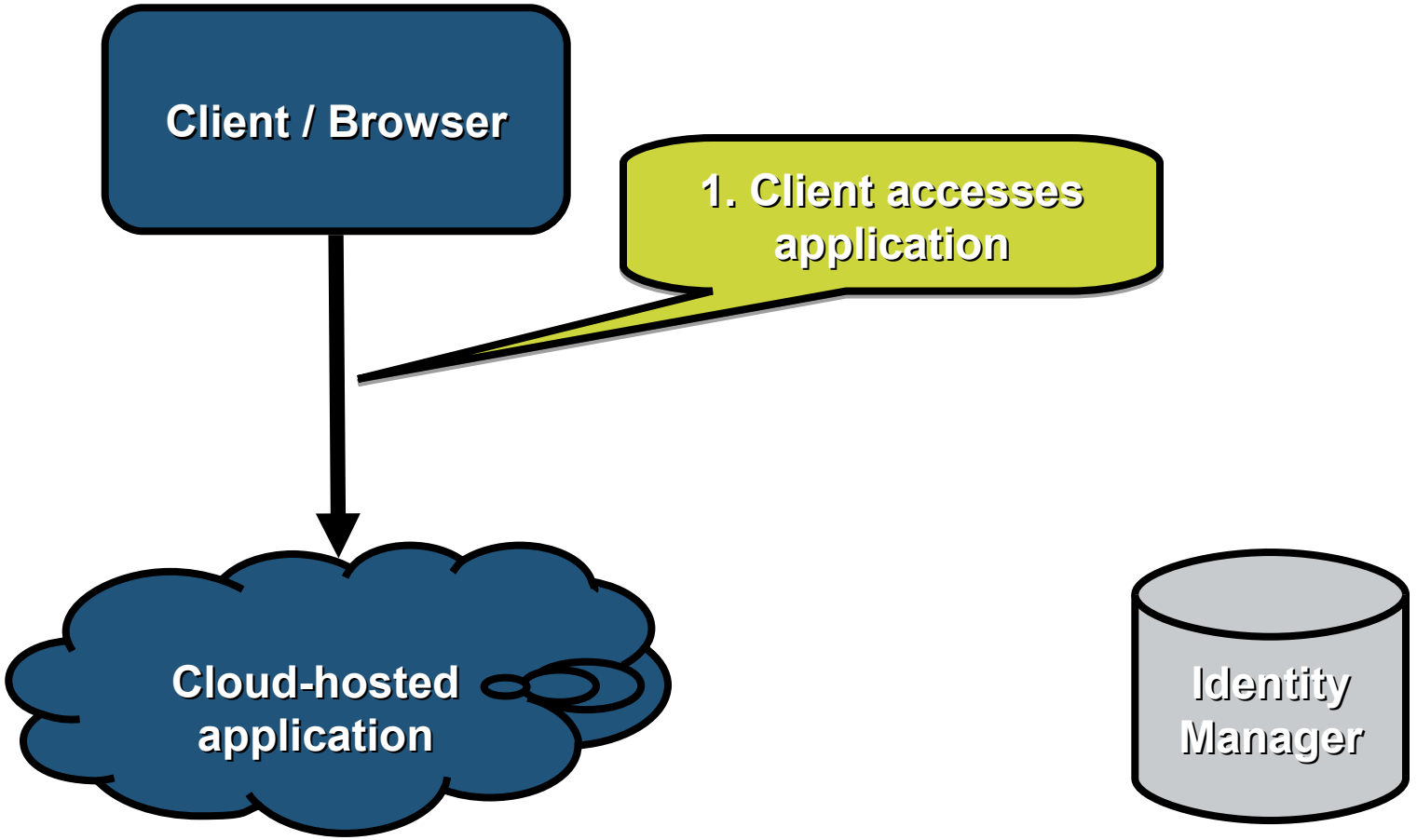
Setting up SSO with Google Apps



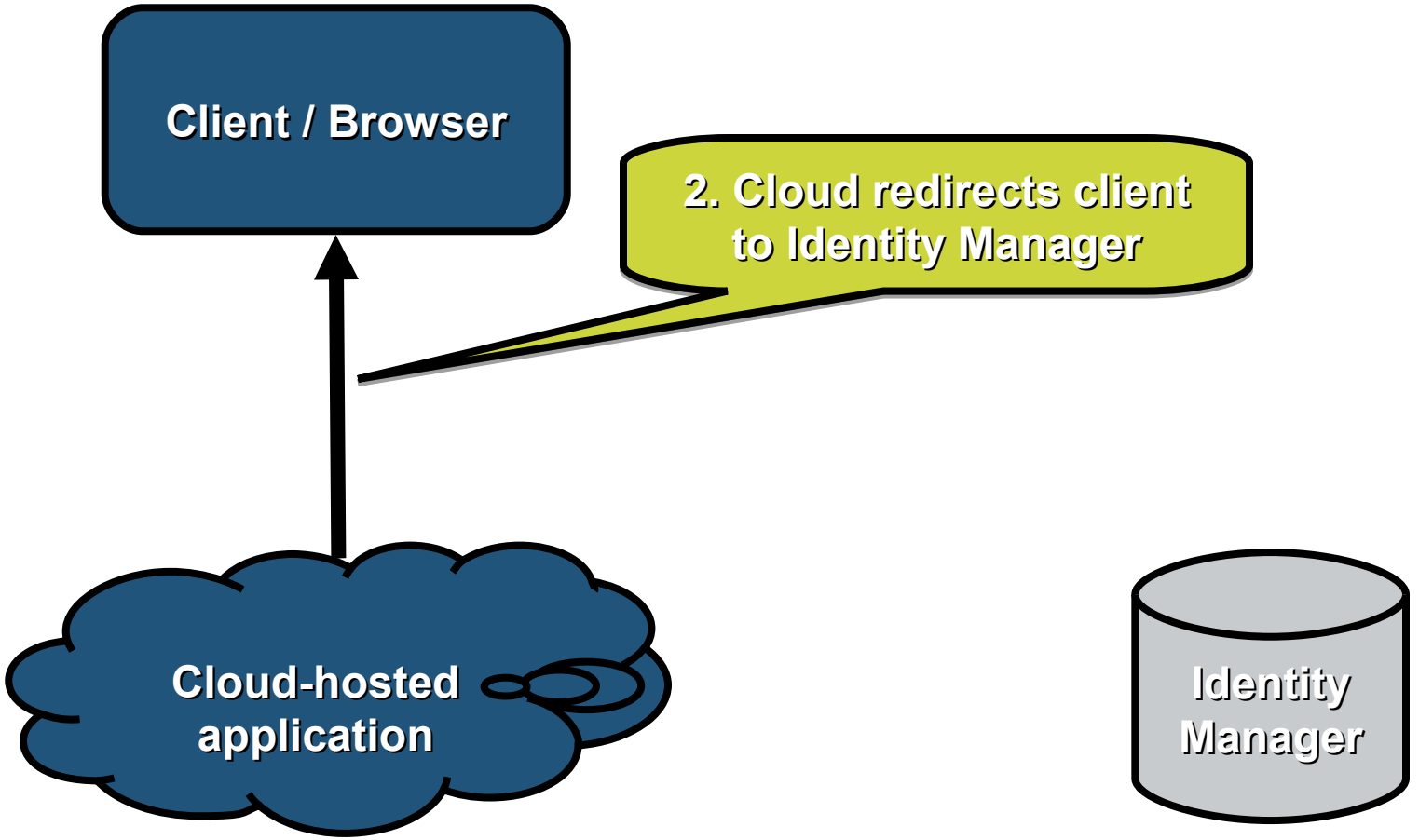
The infrastructure

- We've set up Google Apps to redirect the user to an identity manager behind a corporate firewall.
- The identity manager examines our credentials. If we're trustworthy, it generates a signed SAML token that's passed back to Google Apps.
- This lets us reuse our existing authentication infrastructure.

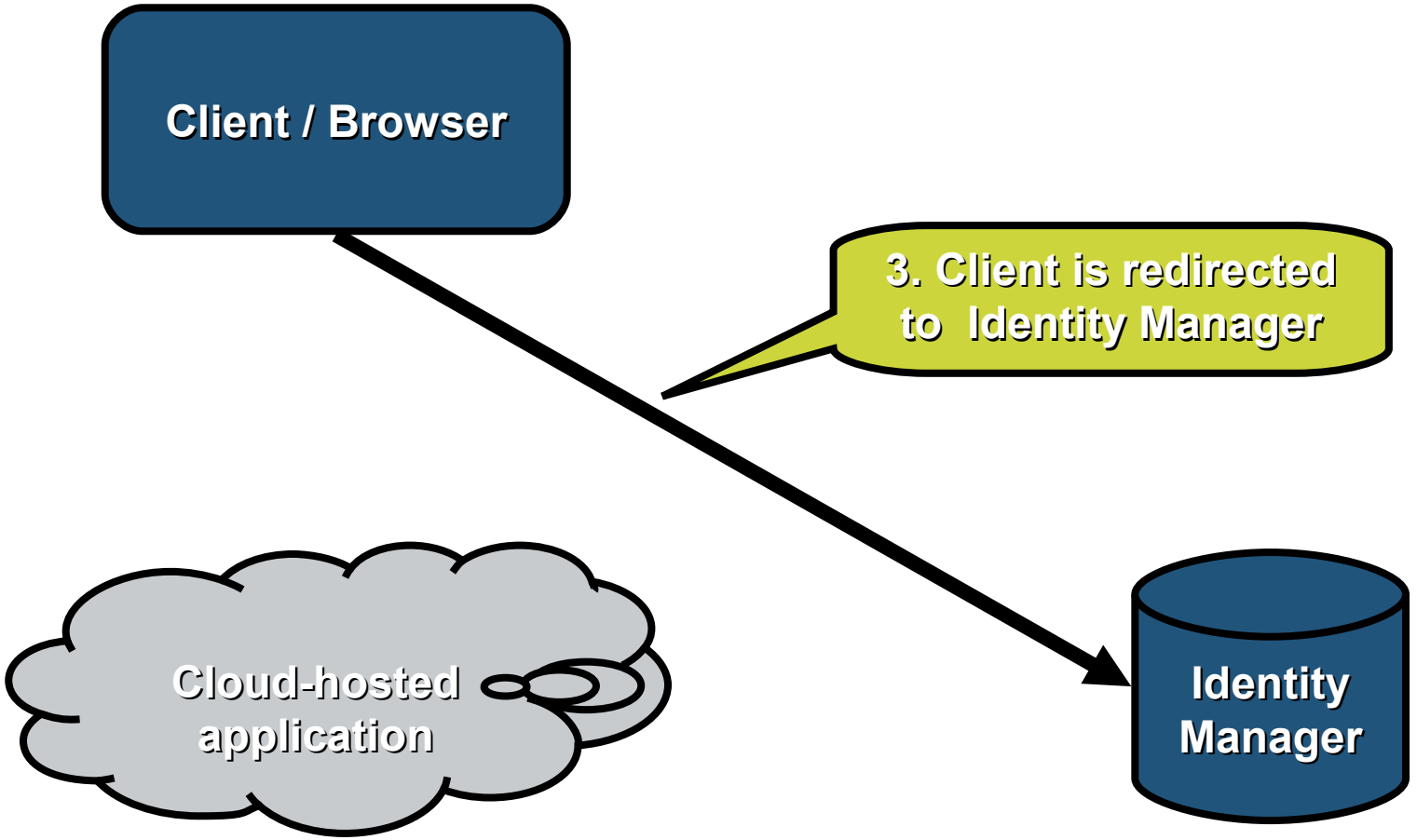
How the magic happens



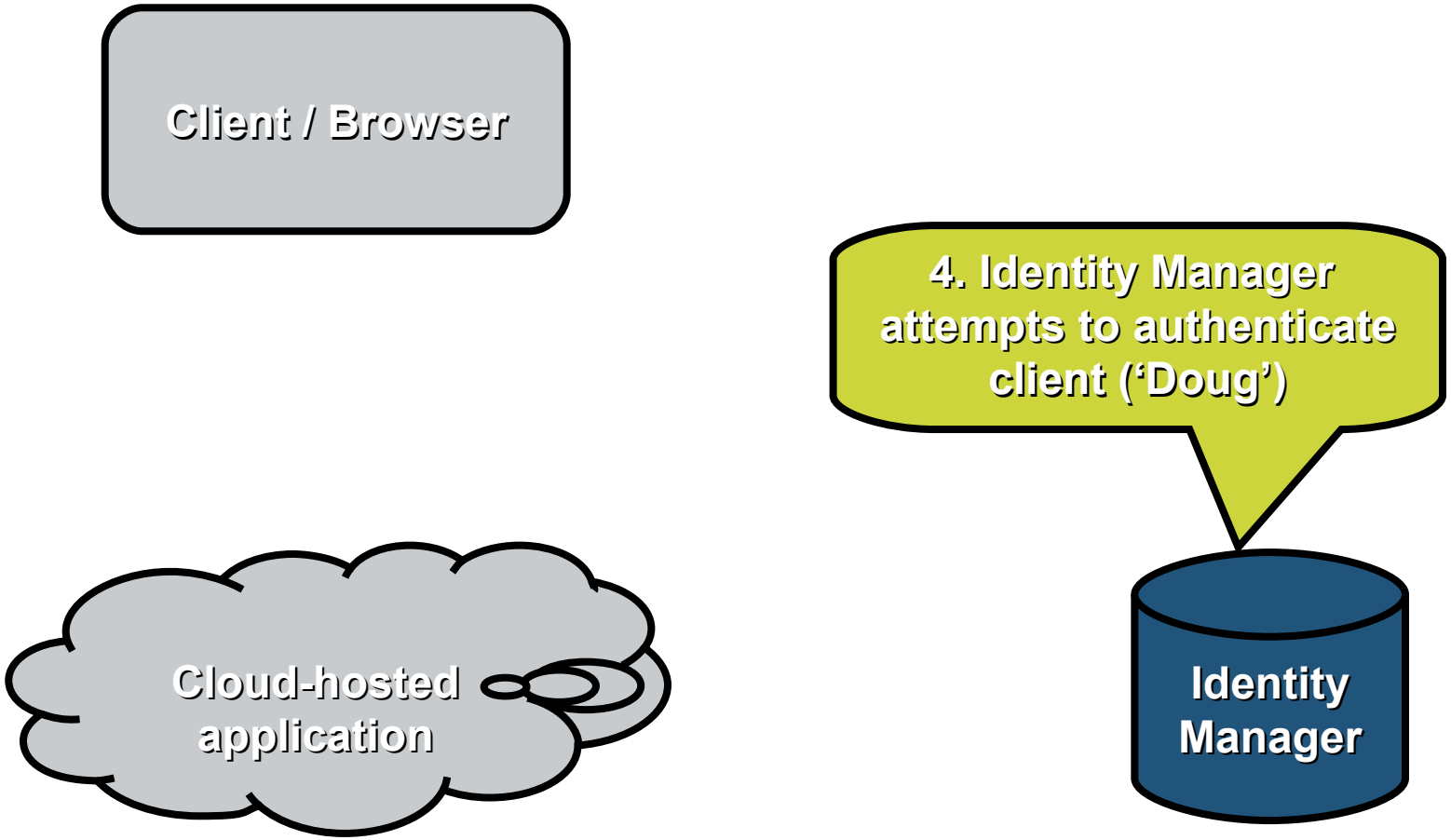
How the magic happens



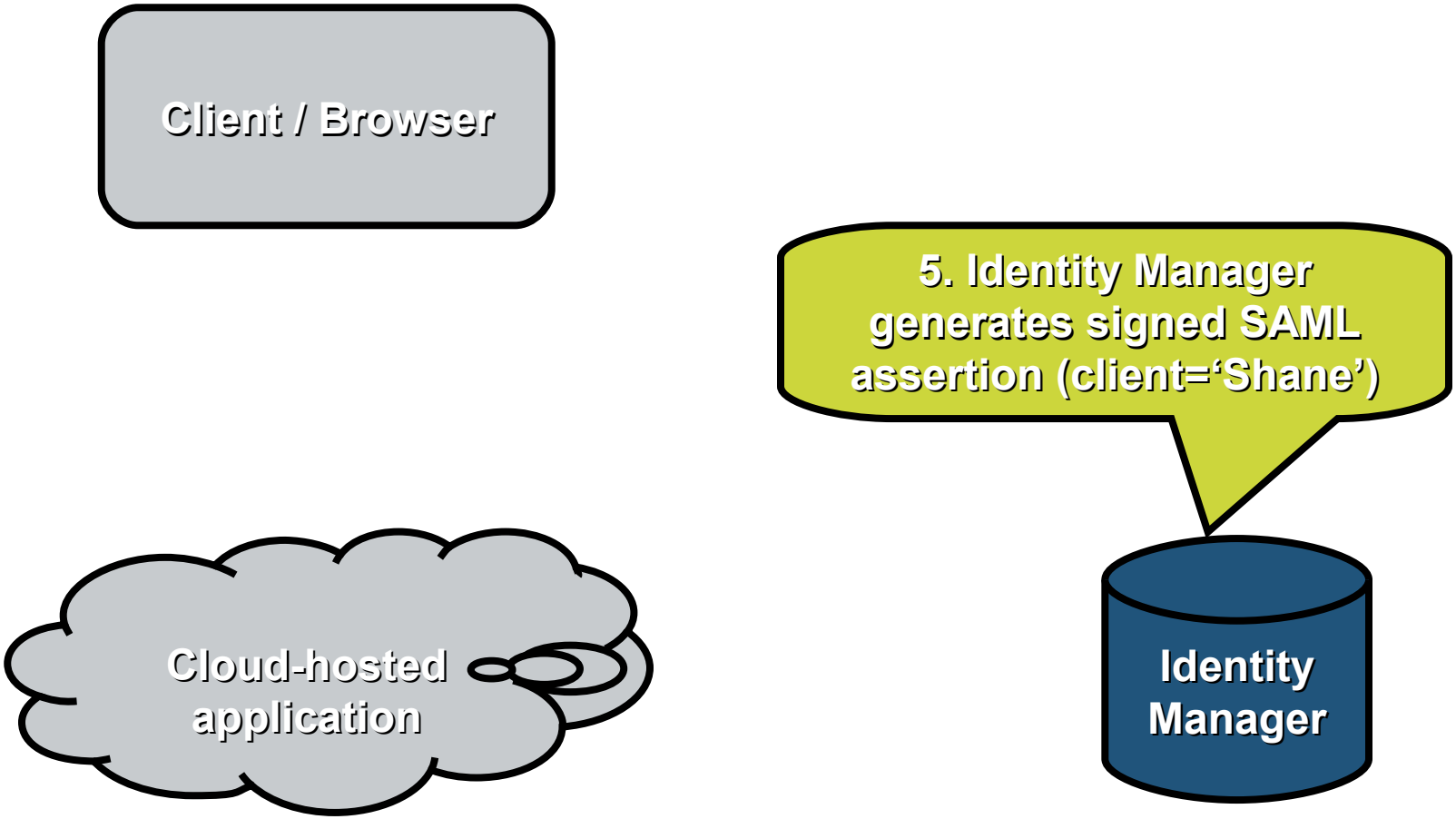
How the magic happens



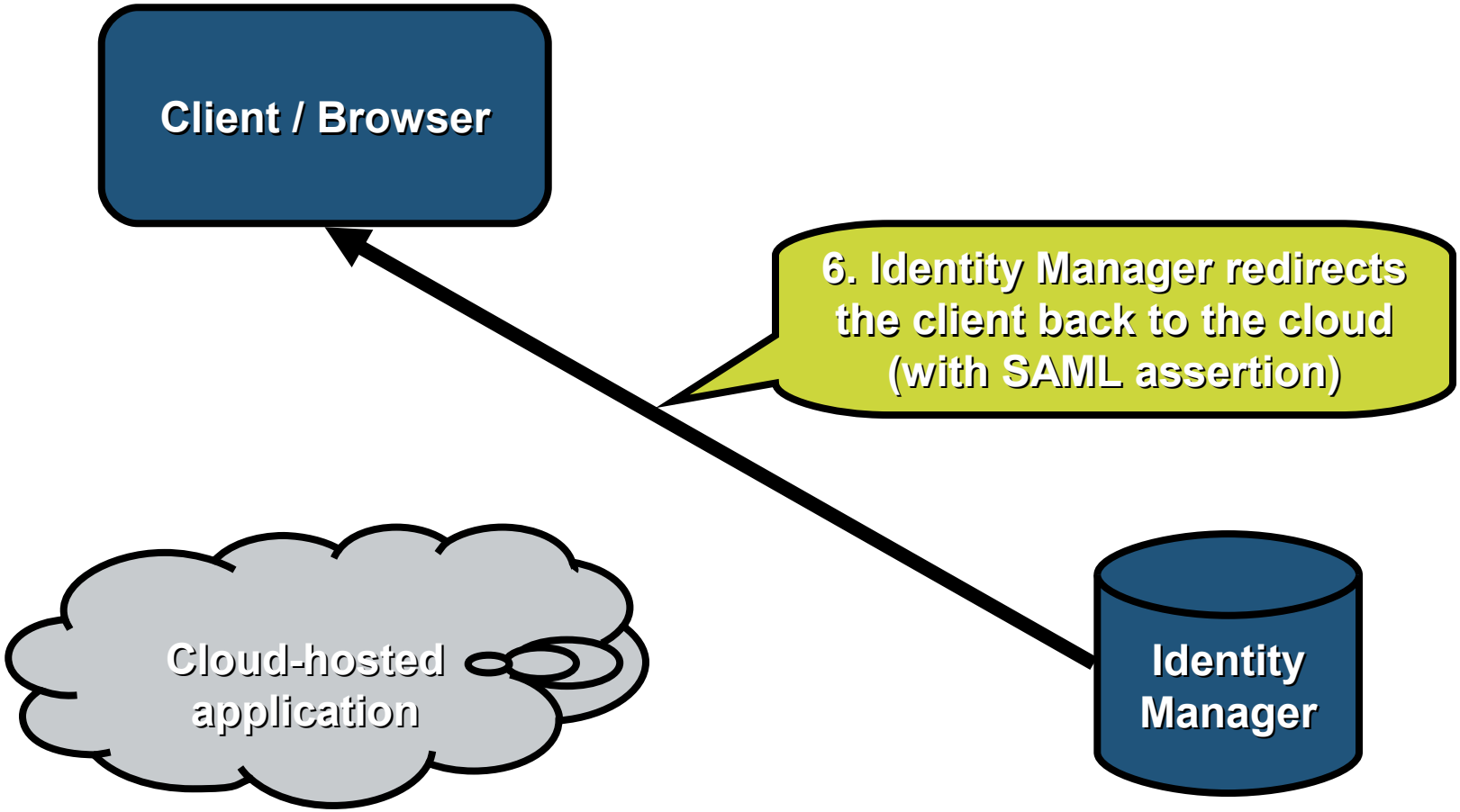
How the magic happens



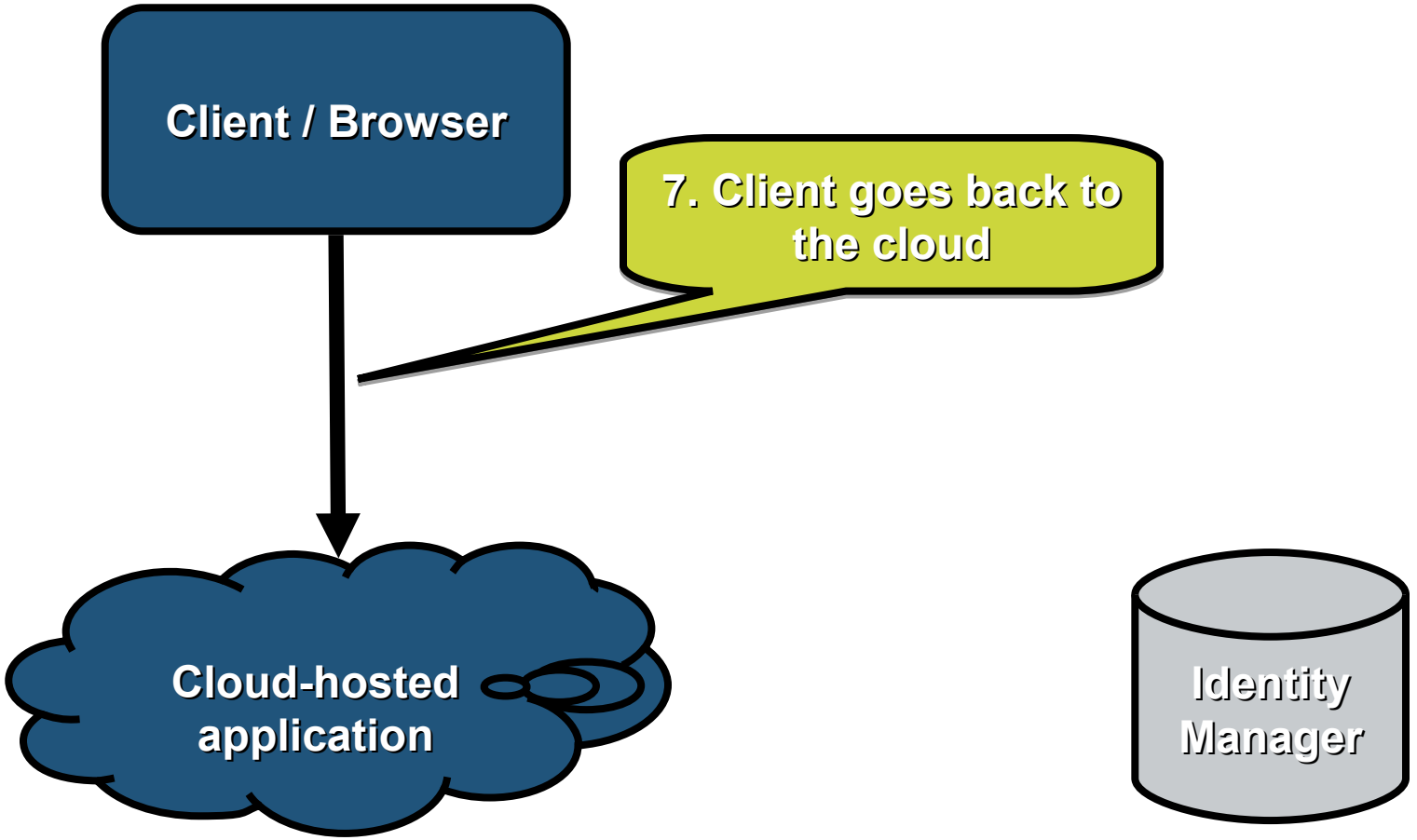
How the magic happens



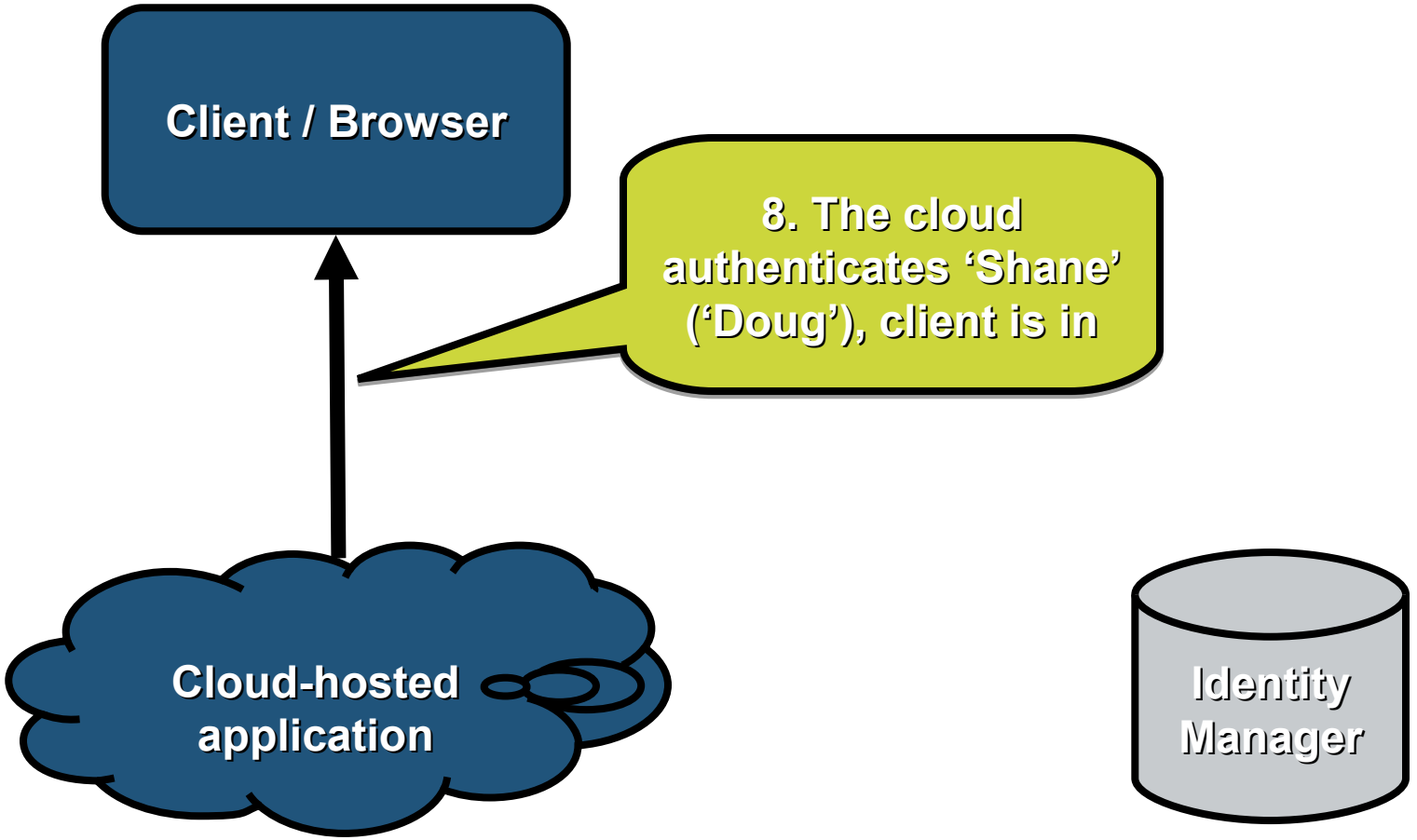
How the magic happens



How the magic happens



How the magic happens



Other techniques

- The technique we've used here applies to Google. Google accepts a signed SAML token to authenticate user requests.
- If we want to make the authentication process part of the request, WS-* and REST are different:
  - In the WS-* world, we can standardize on WS-Security and its related specs. This lets us use protocols other than HTTP(S).
  - In the REST world, each service has a different way of doing things. Amazon DevPay requires the HTTP header to have an **`x-amz-security-token`**, for example. REST also requires HTTP.

Java is a trademark of Sun Microsystems, Inc.

JavaOne℠

Summary

Summary

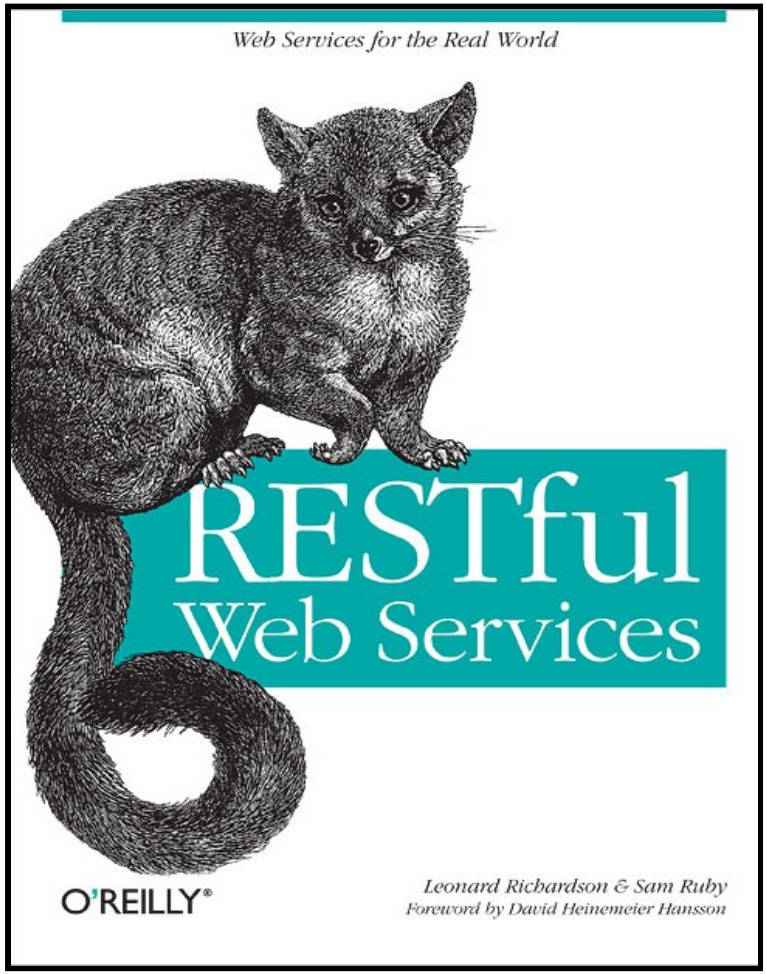
- Cloud computing has significant benefits, but all the hard problems still apply.
  - Encryption, transactions, reliable messaging....
  - **Proceed with caution.**
- Use standards wherever you can.
  - SAML, for example.
- REST and WS-* are two useful ways of accessing services.
  - Many cloud services support only one, but if you have a choice, use whichever meets your needs.

REST and WS-*

- WS-* defines standards for authentication, signatures, non-repudiation, guaranteed delivery, conversations
- WS-* can use protocols other than HTTP
- REST can be much simpler, although WS-* has better tooling (today).
- If you're going to use REST, use it correctly.
  - Don't use POST for everything, for example.
- To repeat: **The sane answer is to examine the two technologies and figure out which meets your needs.**

A really useful book

- Written by Leonard Richardson and IBM's Sam Ruby
- If you're new to REST or want to know more about REST vs. WS-*, get a copy of this book.
- ISBN 0-596-52926-0
- `oreilly.com/catalog/ 978(`



Some other resources

- Roy Fielding's dissertation:
  `roy.gbiv.com/pubs/dissertation/top.htm`
- W3C's SOAP primer:
  `w3.org/TR/soap12-part0`
- David Chappell's REST vs. WS-* blog posting:
  `tinyurl.com/6hmemh`

Thank You

Doug Tidwell, IBM
dtidwell@us.ibm.com

ibm.com/cloud