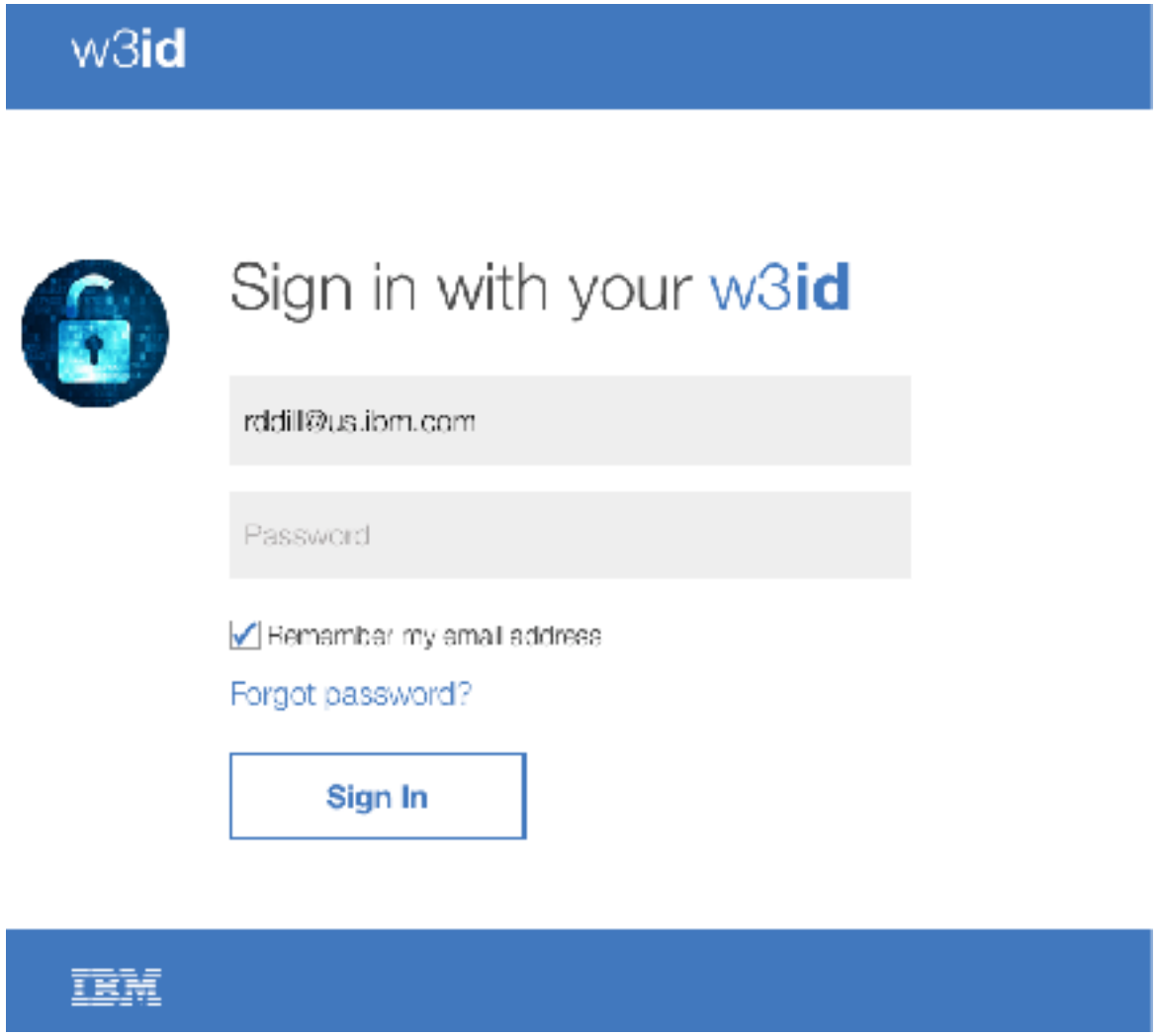
Chapter 7-2: Authentication with IBM w3-id

Learning Bluemix & Cognitive

Bob Dill, IBM Distinguished Engineer, CTO Global Technical Sales

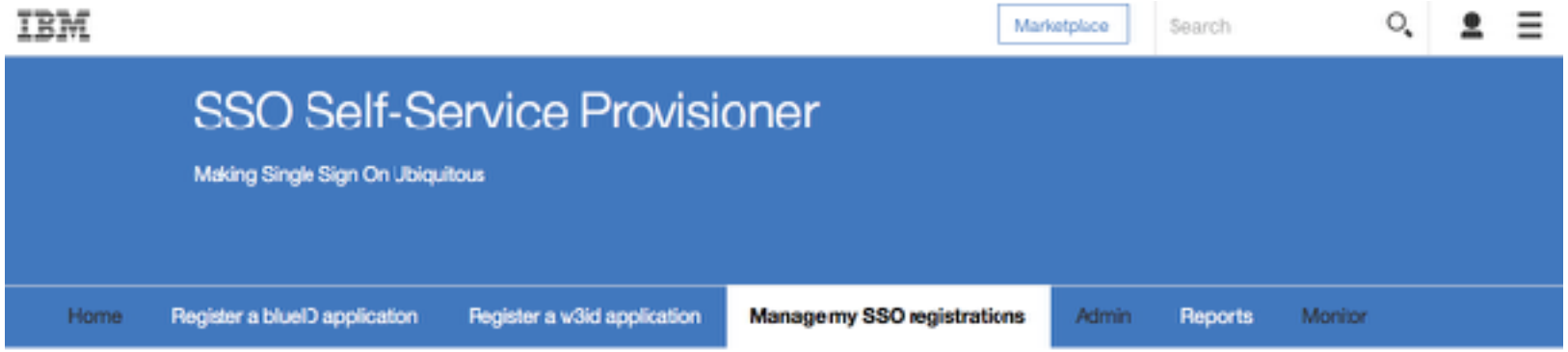
Chapter 7: Securing your application with IBM w3-id

- Authorizing your application to use w3-id
- Enabling your Bluemix application to use https
- Implementing w3-id



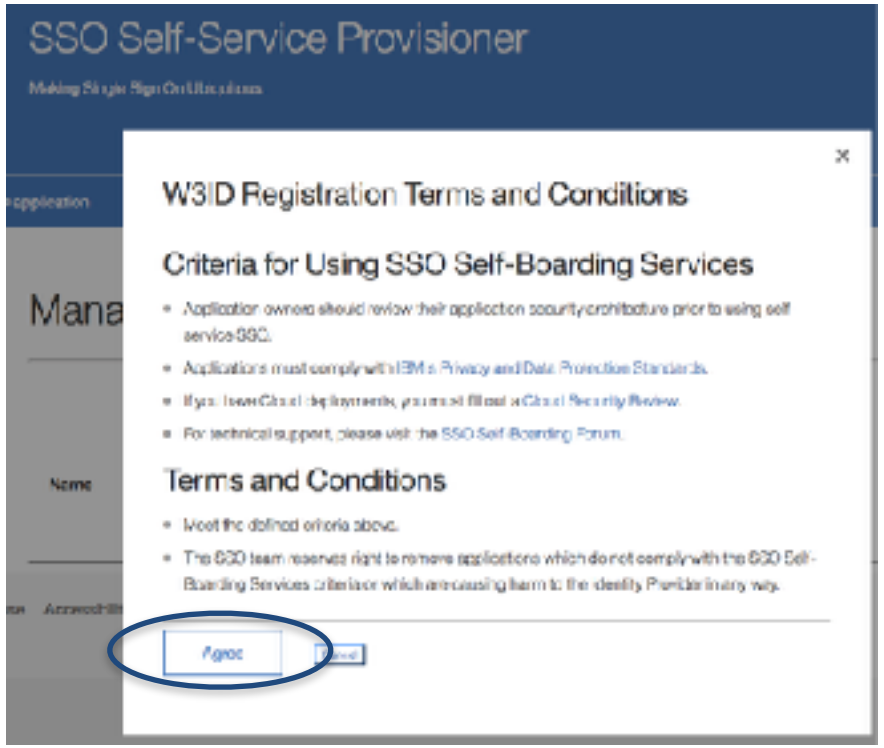
Enabling your application to use w3-id

- **Start here**: https://w3.innovate.ibm.com/tools/sso/application/list.html



- To create an internal app, click on "Register a w3id application"
- To create a client-facing app, click on "Register a blueID application"
- Sample NodeJS app from IBM CIO organization using w3-id (SAML)
  - https://w3-connections.ibm.com/wikis/home?lang=en-us#!/wiki/W7d38ca49a9b1_4048_8833_53626980363f/page/Node%20SSO%20with%20SAML

1 Register a w3-id application



1. Agree to the terms and conditions

2 Register a w3-id application

Application Details

Specify details about your application and contacts. This application registration will show up in the Management SSO registration tab for future changes.

Friendly Name:* Z2C Chapter07-2 SSO

Application Description:* Tutorial on how to use w3-id in Dedicated Bluemix

Planned Production Deployment Date: 2016-12-16

Approximate Number Of Users:* 200

Approximate Max Logins Per Hour:* 20

Home Page: https://z2c-chapter7-sso.w3i

Contact Details

Registered By: rddill@us.ibm.com

Technical Owner:* rddill@us.ibm.com

Business Owner:* rddill@us.ibm.com

Support Contact: rddill@us.ibm.com

Other Contact:

Next Cancel

2). Provide a reference (friendly) name for this request and a description of what this app will do. This simplifies your management later on.

- Enter a planned deployment time in the immediate future
- Put in a reasonable number of users. For this tutorial, use small numbers. I used 200 and 20 for the demo page for this chapter.
- Put in the bluemix homepage url for your application. This is the ROUTE from the app dashboard in bluemix.

z2c-chapter7-sso z2c-chapter7-sso.w3ibm.mybluemix.net

- Put in your intranet id for the last 4 fields and press Next

3 Register a w3-id application

w3id Protocol Selection

Choose one of the following SSO protocols for your application. The choice you make depends on many factors including which protocol(s) your application supports, the types of clients you have, the attributes you need returned, the expertise you have with SSO, the platform you are running on, whether or not your application has firewall access to connect directly to w3id, and much more. Check here for more guidance on selecting a w3id protocol.

If you have a large volume 3rd party application, you should use SAML instead of OpenID Connect given the nature of direct connectivity from an external site.

- ○ Use w3ID (BluePages) OpenID Connect 1.0.
- ◉ Use w3ID (BluePages) SAML 2.0.
- ○ Use w3ID (BluePages) OpenID 2.0.

Back | Next | Cancel

3). Select w3-id SAML2

4 Register a w3-id application

Select identity provider

Choose an identity provider for this instance of your application. If you are still testing your application or have not fully worked out the settings for your SSO configuration, then choose a test/stage provider to use. Once you've worked out all of the settings and it's working, then you can choose a production provider. Never perform stress testing using the production SSO providers. Check here for guidance on selecting the correct identity provider.

	Provider Name	Phase	Approval Required	Activation Type
○	w3id Pilot (only for apps piloting future upcoming w3id features/function - not to be used for most applications)	Production (uses production LDAP)	Yes	IMMEDIATE
○	w3 SSO (AHE/EHS - Heritage) Production	Production (uses production LDAP)	Yes	SCHEDULED
○	w3 SSO (AIT/EHS - Heritage) Test	Test (uses test LDAP)	Yes	SCHEDULED
◉	w3id Staging (for dev/test of apps heading to production)	Test (uses production LDAP)	No	MANUAL
○	w3id Production (typical choice for most applications)	Production (uses production LDAP)	Yes	MANUAL
○	w3id Test (for initial setup of w3id with test LDAP)	Test (uses test LDAP)	No	MANUAL

Back | Next | Cancel

4). Select test (not production)

5 Register a w3-id application

Identity provider metadata download

The following identity provider was selected to be used by your service provider. Use the associated metadata file when configuring your service provider. Follow instructions in the SAML Wiki for assistance.

Identity Provider Name: **w3id Staging (for dev/test of apps heading to production)**

Identity Provider Metadata File: samlidp_IBM_metadata_CIS.xml

Select the response file template you would like to use to board your Service Provider.

Application Template: Bluemix Template

Target Application URL: https://z2c-chapter7-sso.w3ibm.mybluemix.net/

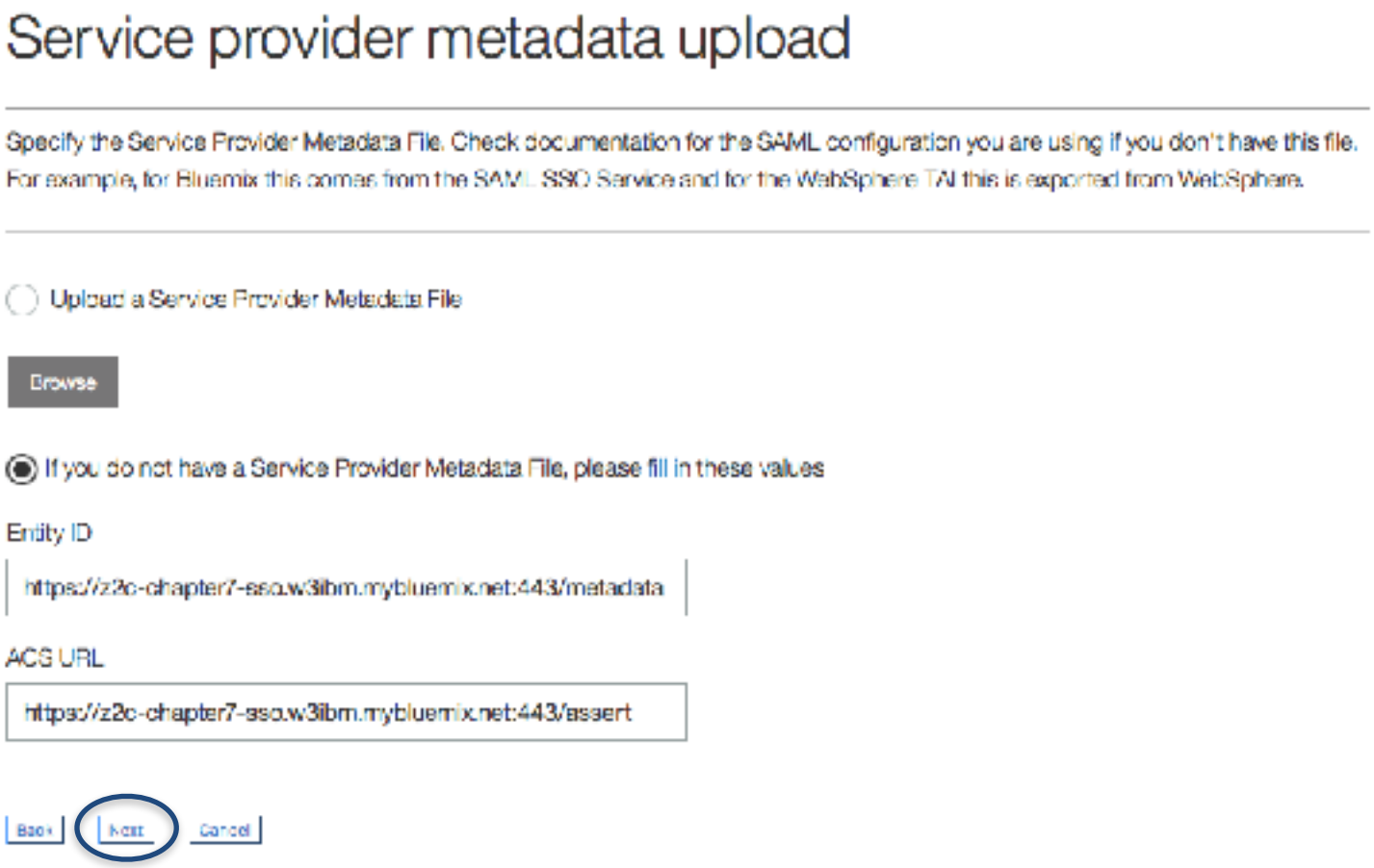
Back | Next | Cancel

5). Scroll down the application template and select the Bluemix template.

Enter your application URL as https

z2c-chapter7-sso | z2c-chapter7-sso.w3ibm.mybluemix.net

6 Register a w3-id application



6). Enter your bluemix route as https:// in both fields.

Entity ID in this example is :443/metadata. The only part you can change is “metadata”. If you change this, you’ll have to change the related code in your authenticate.js file

ACS URL is your callback path from the w3-id authentication process. The url ends with :443/assert. The only part you can change is “assert”. If you change this, you’ll have to change the related code in your authenticate.js file

7 Register a w3-id application

SAML Information For SP Setup

Use this information to complete your SP setup

Partner ID:	https://z2c-chapter7-sso.w3ibm.mybluemix.net:443/metadata.xml
Issuer:	https://w3id.alpha.sso.ibm.com/auth/sps/samlidp/saml20
IDP-Initiated Login URL:	https://w3id.alpha.sso.ibm.com/auth/sps/samlidp/saml20/logininitial?RequestBinding=HTTPPost&PartnerId=https://z2c-chapter7-sso.w3ibm.mybluemix.net:443/metadata.xml&NameIdFormat=email&Target=https://z2c-chapter7-sso.w3ibm.mybluemix.net/
SP-Initiated Login URL:	https://w3id.alpha.sso.ibm.com/auth/sps/samlidp/saml20/login
IDP Metadata File	samlidp_IBM_metadata_CIS.xml

| Back | Next | | Cancel |

7). Use this information in the index.js file and in the authenticate.js file in the var sections where you build your login url.

Set up index.js

```
var https = require('https');
var myKey = path.join(path.dirname(require.main.filename),"cert","key.pem");
var myCert = path.join(path.dirname(require.main.filename),"cert","cert.pem");
var myKeyFile = fs.readFileSync(myKey);
var myCertFile = fs.readFileSync(myCert);
var env = require('./controller/env.json');
var sessionSecret = env.sessionSecret;

// start of security
var serverURLb = "z2c-chapter7-sso.w3ibm.mybluemix.net";
var serverURL = "http://" + serverURLb;
var serverURLs = "https://" + serverURLb;
var partnerIDURL = serverURLs+":443/metadata";
var entityURL = partnerIDURL + ".xml";
var loginpage = serverURLs + "/login";

// create a new express server
var app = express();
app.use(cookieParser(sessionSecret));
```

To create key and certification files, use the following commands

```
openssl genrsa -out key.pem
openssl req -new -key key.pem -out csr.pem
openssl x509 -req -days 9999 -in
    csr.pem -signkey key.pem -out cert.pem
```

- On the left, we establish the variables which will be used in this chapter. the key file and certificate file are the same as the ones created for Chapter 7.
- Below, we add a series of app.get paths. The sequence IS important and all of the new paths must precede the app.use statement for the router and the app.use statement for express.static.
- res.cookies.authenticated is only visible to the server and is only present when the

```
app.get('/js*', function(req, res) {
  if ((typeof(req.signedCookies.authenticated) != 'undefined') &&
  (req.signedCookies.authenticated == req.cookies.email )) {loadSelectedFile(req, res);}
  else {res.redirect(loginpage);}
});
app.get('/in*', function(req, res) {
  if ((typeof(req.signedCookies.authenticated) != 'undefined') &&
  (req.signedCookies.authenticated == req.cookies.email )) {loadSelectedFile(req, res);}
  else {res.redirect(loginpage);}
});
app.get('/', function(req, res) { res.redirect(loginpage); });
app.use('/', require("./controller/restapi/router"));
app.use(express.static(__dirname + '/HTML'));
```

Creating the authenticate routines: the variables

```
var path = require('path');
var fs = require('fs');
var express = require('express');
var saml2 = require('saml2-js');
var Saml2js = require('saml2js') ;
var path = require('path');
var cookieParser = require('cookie-parser');
var env = require('../../env.json');
var sessionSecret = env.sessionSecret;

var myKey = path.join(path.dirname(require.main.filename),"cert","key.pem");
var myCert = path.join(path.dirname(require.main.filename),"cert","cert.pem");
var myKeyFile = fs.readFileSync(myKey);
var myCertFile = fs.readFileSync(myCert);

var serverURLb = "z2c-chapter7-sso.w3ibm.mybluemix.net";
var serverURL = "http://" + serverURLb;
var serverURLs = "https://" + serverURLb;
var partnerIDURL = serverURLs+":443/metadata";
var entityURL = partnerIDURL + ".xml";
var loginpage = serverURL + "/login";
var app = express();
app.use(cookieParser(sessionSecret));

var loginURL =
"https://w3id.alpha.sso.ibm.com/auth/sps/saml idp/saml20/loginini
Id="+partnerIDURL+"&NameIdFormat=email&Target="+serverURLs;

var sp_options = {
    entity_id: entityURL,
    private_key: myKeyFile.toString(),
    certificate: myCertFile.toString(),
    assert_endpoint: serverURLs + ":443/assert"
};
var sp = new saml2.ServiceProvider(sp_options);
var idp_options = {
sso_login_url: loginURL,
    certificates: fs.readFileSync("cert/w3id.sso.ibm.com").toString()
};
var idp = new saml2.IdentityProvider(idp_options);
```

SAML to get security basics

cookie-parser to send info back to server

Retrieve key file and certificates

Match to paths from w3-id request

Build log in URL from parts

Create SAML options for callback

Create SAML options for certificate

Creating the authenticate routines: the functions

```
// Endpoint to retrieve metadata
exports.metadata = function(req, res) {
  console.log("metadata entered");
  res.type('application/xml');
  res.send(sp.create_metadata());
}

// Starting point for login
exports.login = function(req, res) {
  //console.log(idp);
  console.log("login entered");
  sp.create_login_request_url(idp, {}, function(err, login_url, request_id) {
    if (err != null)
      return res.send(500);
    res.redirect(login_url);
  });
}

// Assert endpoint for when login completes
exports.assert = function(req, res) {
  console.log("assert entered");
  var options = {request_body: req };
  var response = new Buffer(req.body.SAMLResponse || req.body.SAMLRequest, 'base64');
  var parser = new Saml2js(response);
  var userFromW3 = parser.toObject();
  var email = userFromW3.emailaddress;
  console.log("assert completed for "+email);
  res.cookie('authenticated', email,{ maxAge: 43200, httpOnly: true, signed: true });
  res.cookie('email',email,{  maxAge: 43200 });
  res.status(302).redirect('/index.html');
}
```

- Metadata dynamically creates information for w3-id which identifies your application on bluemix.
- login initiates the authentication request to w3-id.
  - idp holds your certificate
  - login_url is returned based on the loginURL provided in dip
- assert is called only on successful completion of the w3-id authentication request.
  - The userFromW3 object has a great deal of useful information and is worth inspecting
  - two cookies are set. authenticated is only visible to the server where email is visible to the client (so you can use the email address with things like the blue pages API to retrieve a picture of the person who's logged in.

The Plan: 30 minute Chapters with an hour or two of practice

1. The Story, Architecture for this app	
2. Setting up Bluemix	
3. Building your first Watson App	(Watson Speech to Text)
4. Getting Watson to talk back	(Watson Text to Speech)
5. Understanding Classifiers	(Watson NLC)
6. Creating a custom dialog with Watson	(custom Q&A, session management)
7. Authentication	(puts C2 thru 6 together)
8. Alchemy News	(Watson Alchemy)
9. Visual Recognition and Images	(Watson Visual Recognition)
10. Watson Conversations	(Watson Conversations)
11. Retrieve & Rank	(Watson Alchemy + Retrieve & Rank)
12. Getting started on my first client prototype	Design Thinking, Stories, Architecture, Keeping it simple

Chapter 8: Understanding Alchemy News

Learning Bluemix & Cognitive

Bob Dill, IBM Distinguished Engineer, CTO Global Technical Sales