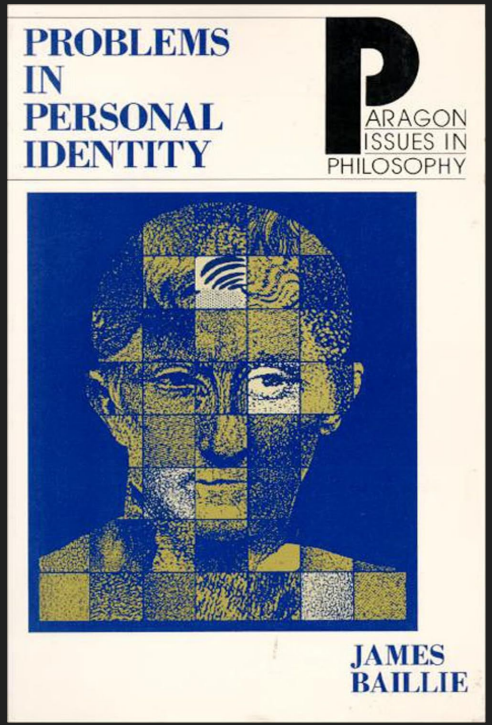
Authenticating Primo Users for 3rd-party Integrations

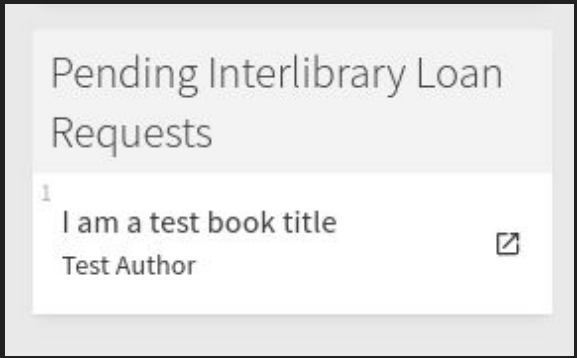
Jeff Peterson, University of Minnesota

Problem

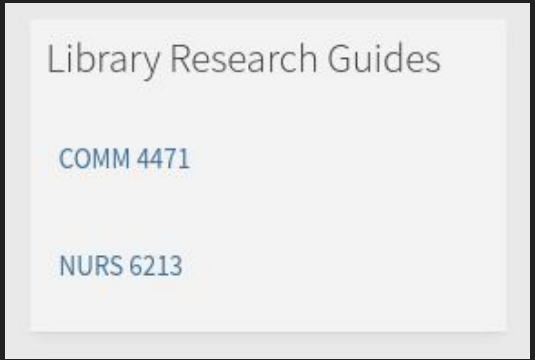
We want to add a custom feature to Primo that depends on a Primo user's identity.



Example: Custom account page tiles



ILLiad transactions in the Primo account page



Research guides for a student's enrolled courses

How do we:

1. Determine the user's identity?
2. Verify the user's identity claim?

Uh… thanks DALL-E 3?

Solution (for Primo VE)

JSON Web Tokens (JWT) [1]

JSON Web Signatures (JWS) [2]

JSON Web Keys (JWK) [3]

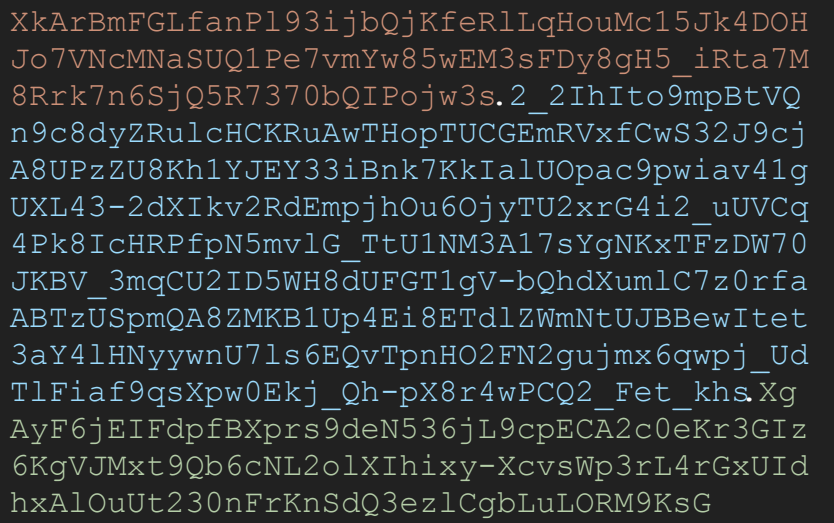
Thanks DALL-E 3

[1] https://datatracker.ietf.org/doc/html/rfc7519
[2] https://datatracker.ietf.org/doc/html/rfc7515
[3] https://datatracker.ietf.org/doc/html/rfc7517

JSON Web Token (JWT)

Three base64url-encoded strings, separated by periods:

- Header
- Payload
- Signature



JSON Web Token: Header

Identifies the signing key and signing algorithm

```
{
   "kid": "primaPrivateKey-SOME_INST",
   "alg": "ES256"
}
```

JSON Web Token: Payload

Contains claims, notably:

- When the token was issued (iat)
- When it expires (jti)
- User's Alma primary identifier (userName)
- User's Alma group code (userGroup)

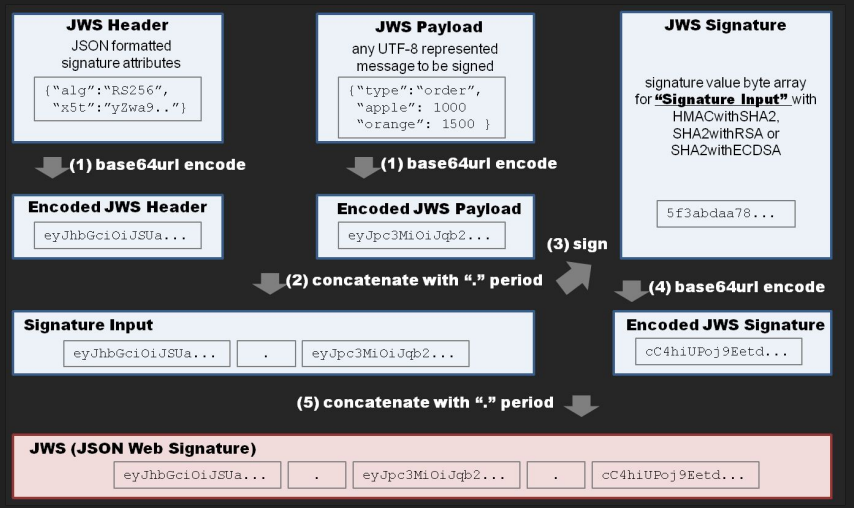

```
{
 "iss": "Prima",
 "exp": 1713565171,
 "iat": 1713478771,
 "userName": "anonymous-0123_456789",
 "displayName": null,
 "user": "anonymous-0123_456789",
 "userGroup": "GUEST",
 "institution": "SOME_INST",
 "userIp": "10.42.42.42",
 "authenticationProfile": "",
 "authenticationSystem": "",
 "language": "en",
 "samlSessionIndex": "",
 "samlNameId": "",
 "onCampus": "false",
 "signedIn": null,
 "viewId": "SOME_INST:VIEW"
}
```

JSON Web Token: Signature

Signed hash of the header and payload

Can be used to verify that:

- The token was issued / signed by Primo
- The token has not been modified
- The token has not expired



Urushima, Kenji. JWS Signature Generation Flow, 2017. http://kjur.github.io/jsrsasign

JSON Web Keys

In order to verify the signature, we need the issuer's public key.

All Alma/Primo institutions expose public keys in a JSON Web Key Set (JWKS) endpoint.

Production:

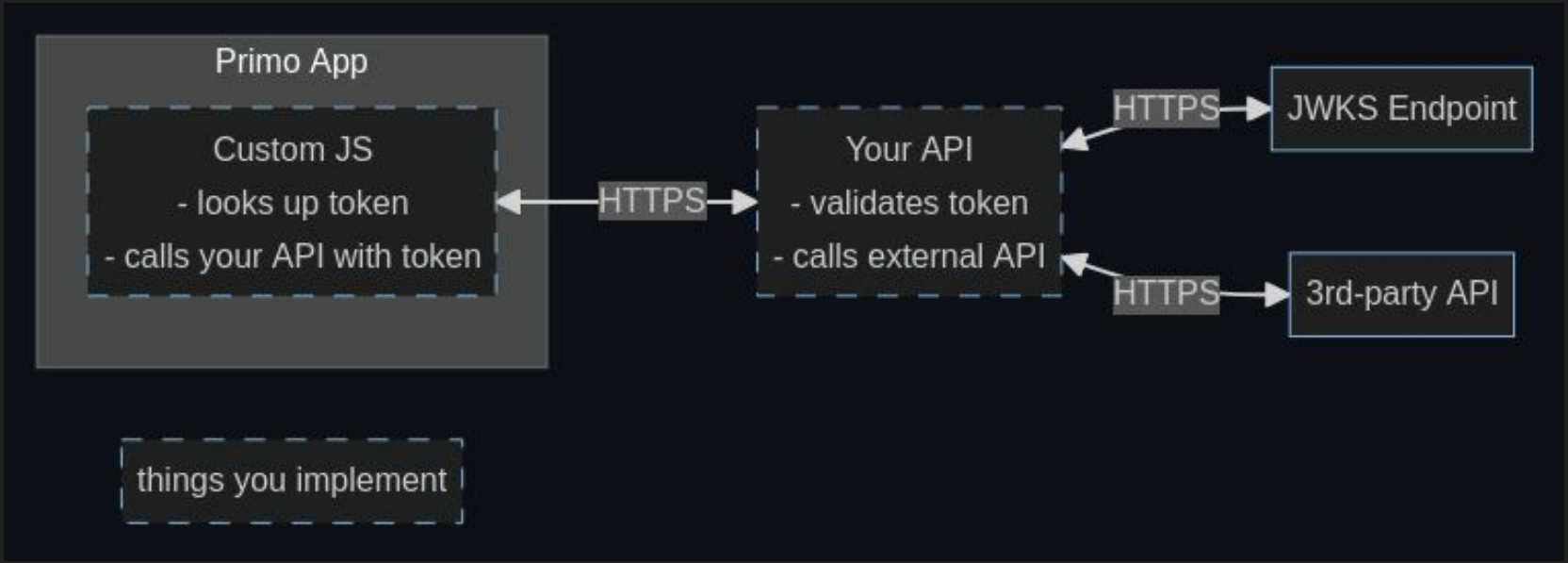
`https://api-{REGION}.hosted.exlibrisgroup.com/auth/{INST_CODE}/jwks.json`

Sandbox:

`https://api-{REGION}.hosted.exlibrisgroup.com/auth/{INST_CODE}/jwks.json?env=sandbox`


```
{"keys": [
   {
       "kty": "RSA",
       "n": "o0h874QlymQoEhLZ...",
       "e": "AQAB",
       "alg": "RS256",
       "kid": "exlhep-01"
   },
   {
       "kty": "EC",
       "kid": "primaPrivateKey-SOME_INST",
       "use": "sig",
       "x": "h6uopk5OwYk6sXrVhwocag...",
       "y": "qockcUQCmNU62-JZUweHOj...",
       "crv": "P-256",
       "alg": "ES256"
   }
]}
```

Putting it all together



A similar approach can be used call external services from Alma CloudApps

JWT lookup

Use Primo's `UserSessionManagerService` or retrieve it from `sessionStorage` directly

```
// assuming you've injected the $rootScope service
const jwt = $rootScope.$$childHead.$ctrl.
userSessionManagerService.getJwt();

// or just grab it from sessionStorage
const jwt =
sessionStorage.getItem("primoExploreJwt") ?? "";
```

Sending the JWT to the API

Now just send the token to your endpoint in an authorization header as: `Bearer {token}`

You could also opt to handle this in a custom AngularJS HTTP interceptor.

Note: the token string is stored with enclosing quotes, so you might need to remove them.

```
$http.get("https://example.edu/some-service", {
  headers: {
    Authorization:
      `Bearer ${jwt.replace(/^"(.*)"$/, "$1")}`
  }
});
```

Validating the JWT

Varies depending on language. Here's an Express.js middleware example.

Once the token is validated, you might want to check the `signedIn` or `userGroup` claim before proceeding.

```
app.use(
 expressjwt({
   secret: jwks.expressJwtSecret({
     jwksUri: YOUR_JWKS_ENDPOINT,
   }),
   algorithms: ["ES256"],
 })
);
// guest tokens are still valid tokens
app.use((req, res, next) => {
 if (req.auth && req.auth.userGroup !== "GUEST") {
   next();
 } else {
   res.sendStatus(403);
 }
});
```

Token validation: JWT/JWK libraries

Java

https://github.com/auth0/jwks-rsa-java

https://github.com/auth0/java-jwt

Node.js

https://github.com/auth0/node-jwks-rsa

https://github.com/auth0/node-jsonwebtoken

https://github.com/auth0/express-jwt

PHP

https://github.com/firebase/php-jwt

Python

https://pyjwt.readthedocs.io/en/stable

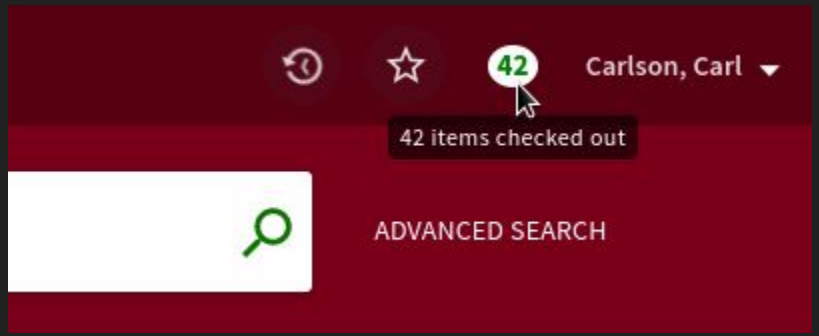
Ruby

https://github.com/jwt/ruby-jwt

Loan Counter

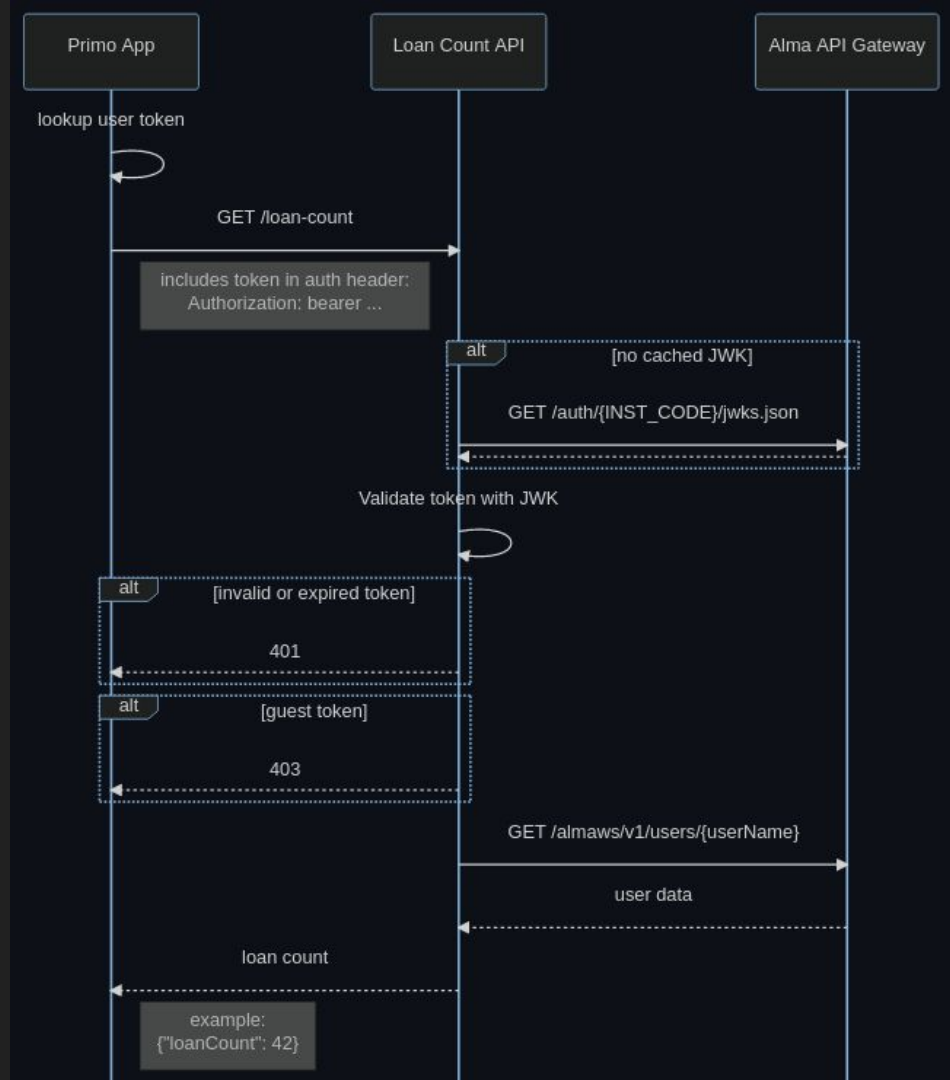
A somewhat pointless but hopefully instructive Primo token validation example

https://github.com/gpeterso/primo-jwk-example



Loan Counter: Details

NOTE: This is a purely didactic example. If you actually wanted to implement this feature, just use Primo's built-in `LoanService`. It'll handle all the JWT stuff.



Loan Counter: Setup

1. Make sure Node.js 18+ is installed
2. Clone the repository
3. From the project's root directory, run `npm install`
4. Create a file named `.env` in the project's root directory
5. Edit the .env file to look something like this:

```
INST_CODE="..."      # your Alma institution code
ALMA_API_KEY="..."   # key should have read-only user permissions
PROXY_TARGET="..."   # Primo base URL (e.g. https://foo.primo.exlibrisgroup.com)
REGION="na"          # change this if you're not in North America
ALMA_API_BASE_URL="https://api-${REGION}.hosted.exlibrisgroup.com/almaws/v1"
JWKS_URI="https://api-${REGION}.hosted.exlibrisgroup.com/auth/${INST_CODE}/jwks.json"
```

6. Run `npm start`
7. You should now be able to visit `http://localhost:3000/discovery/search?vid={YOUR_VIEW_ID}`, and the loan count badge should appear when you sign in

What about Primo Back Office?

Primo Back Office JWTs

Public key / JWK not available?

Technically possible to validate tokens on your FE server(s):

- entails deploying custom JSPs / servlets that reach into Primo internals
- not recommended

Other options for Primo Back Office

Maybe use your institution's SSO platform? (e.g. Shibboleth / SAML)

Shibboleth* passive login setup

- Put a shib-protected service between Primo and the 3rd-party service
- Configure** the service provider to accept "isPassive" login requests
- Service provider attempts to log the user in without prompting (if the user already has an active session with the identity provider)
- Similar to Primo's "silent login" feature

* should work with other SAML 2.0 implementations too

** see https://shibboleth.atlassian.net/wiki/spaces/SP3/pages/2065335036/isPassive

Shibboleth passive login example

1. Assume 2 Shib-protected endpoints (e.g. /login-callback & /some-api) with passive auth enabled
2. In Primo custom JS, open a hidden iframe with target = `https://your-server/Shibboleth.sso/Login?isPassive=true&target=https://your-server/login-callback`
3. After the SP-IdP SAML dance, browser redirects to /login-callback
4. If a session is found, login-callback returns HTML with a `parent.postMessage("...")` script, signaling login success
5. Primo custom JS receives the message can now call /some-api (with cookies)

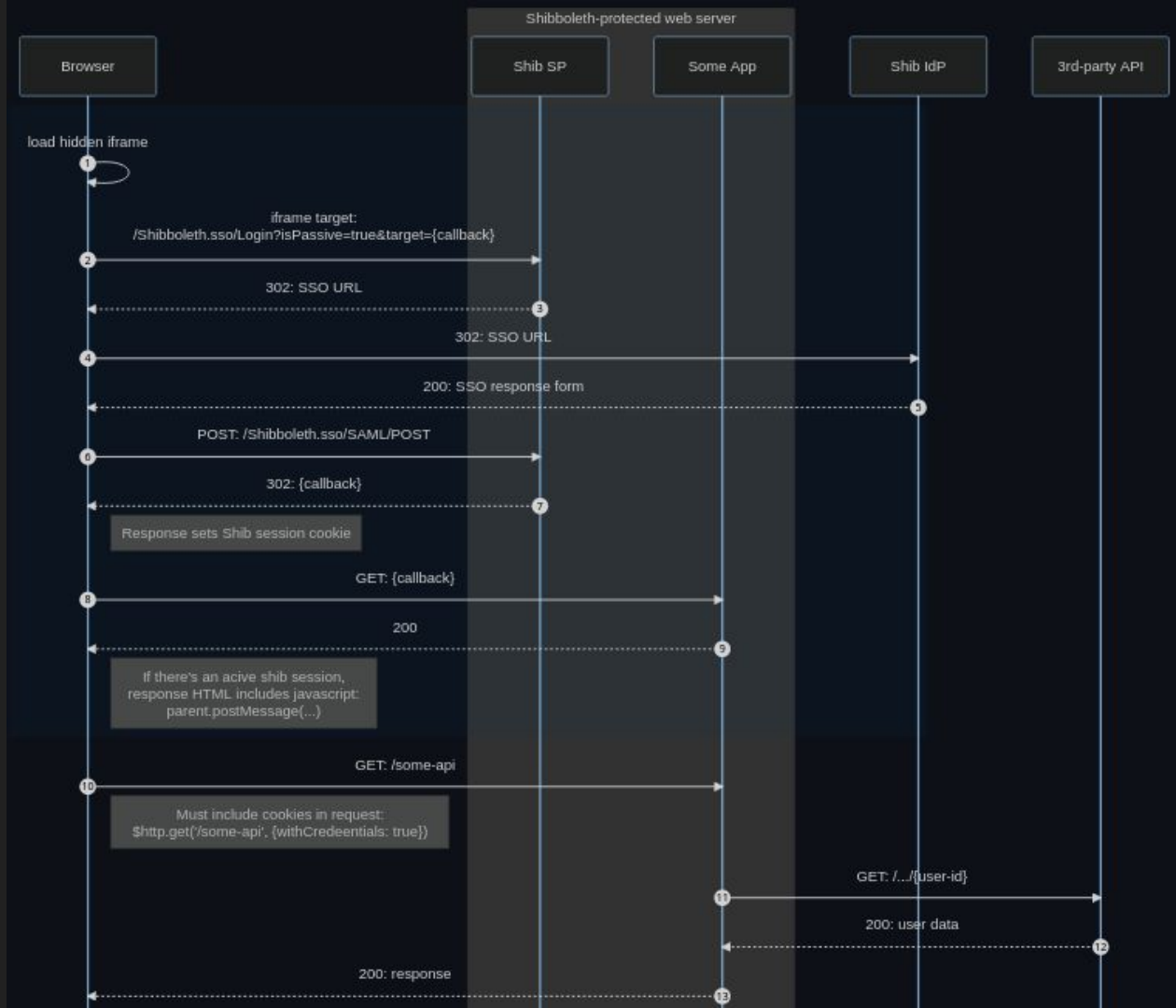
Passive login flow

Only the "happy path" is show here

Uses cookies, so SP and Primo CNAME should have the same domain

Primo timeout may be > IdP timeout

User ID in request #11 is a hypothetical SAML attribute



Shib Auth component

If you wanted to get fancy, you could handle the passive login logic in a custom component that calls an `on-auth` hook when it receives a successful login message from the iframe.

Example:
https://github.com/UMNLibraries/primo-ve-customization/tree/main/src/views/01UMN_INST-TWINCITIES/components/shib-auth

```
<!-- shib-auth template -->
<iframe
  style="display: none"
  ng-src="{{$ctrl.passiveAuthUrl}}">
</iframe>
<ng-transclude></ng-transclude>

<!-- shib-auth usage example -->
<shib-auth on-auth="$ctrl.loadCourses()">
 <!-- tanscluded content here -->
 <md-list>
   <md-list-item ng-repeat="c in $ctrl.courses">
     {{c.subject}} {{c.number}}
   </md-list-item>
 </md-list>
</shib-auth>
```

Summary

Summary: Primo VE

- Use your institution's JSON Web Keys to validate Primo tokens
- "Validation" is an assertion that the token is:
  - issued/signed by Primo
  - unaltered
  - unexpired
- Unauthenticated guest tokens are still valid tokens
- Protect user tokens
  - Use HTTPS
  - Don't share with 3rd parties

Summary: Primo Back Office

- Technically possible to use JWTs, but might be more trouble than it's worth
- Consider ways to use your institution's SSO platform (e.g. Shib passive login)
  - Can be complicated to set up
- Consider moving to VE? ¯_(ツ)_/¯

Questions?