

# Migrating a B2 to LTI/REST

*How to build an LTI Advantage Tool*

Eric Preston, Staff Engineer, Blackboard Inc.

# Forward-Looking Statement

Statements regarding our product development initiatives, including new products and future product upgrades, updates or enhancements represent our current intentions, but may be modified, delayed or abandoned without prior notice and there is no assurance that such offering, upgrades, updates or functionality will become available unless and until they have been made generally available to our customers.

# AGENDA

## How to Build an LTI Advantage Tool

### What is LTI?

Platforms and tools; security; services

### Building and Deploying an LTI Application

B2 Example – UI and Java API

LTI Launch

LTI Deep Linking

LTI Services

Application developer makes it available

How to get a new app installed

How to configure placements and where they appear



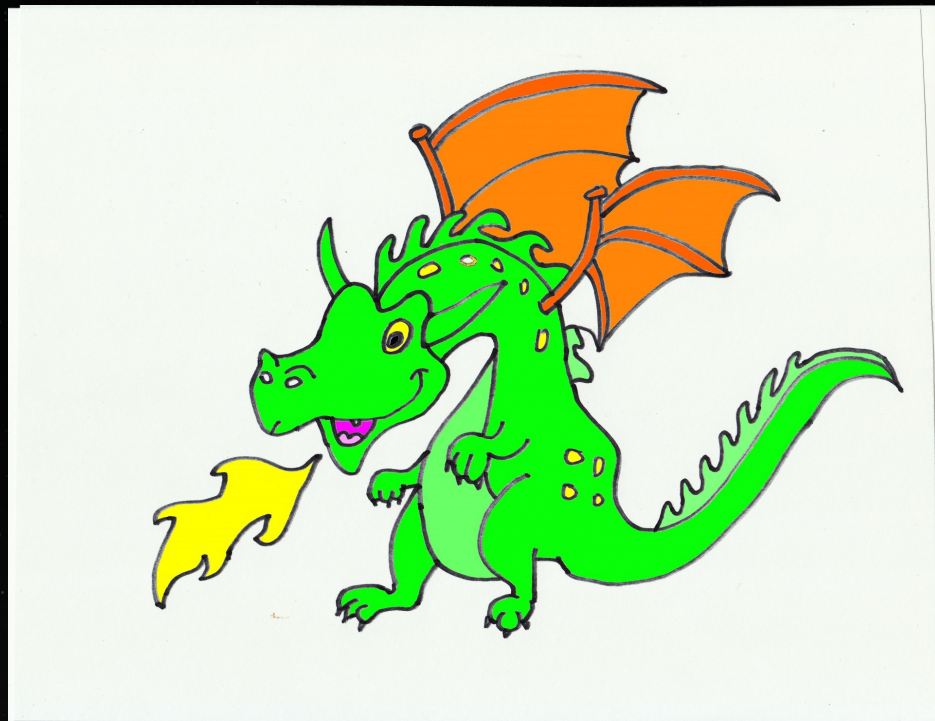
## About Me

---

- With Blackboard in Product Development > 13 years
- Member of IMS Global LTI and Caliper Technical Working Groups
- On the Developer Relations team with Scott Hurrey
- Dad, classical guitar, nature photography

# What is a B2?

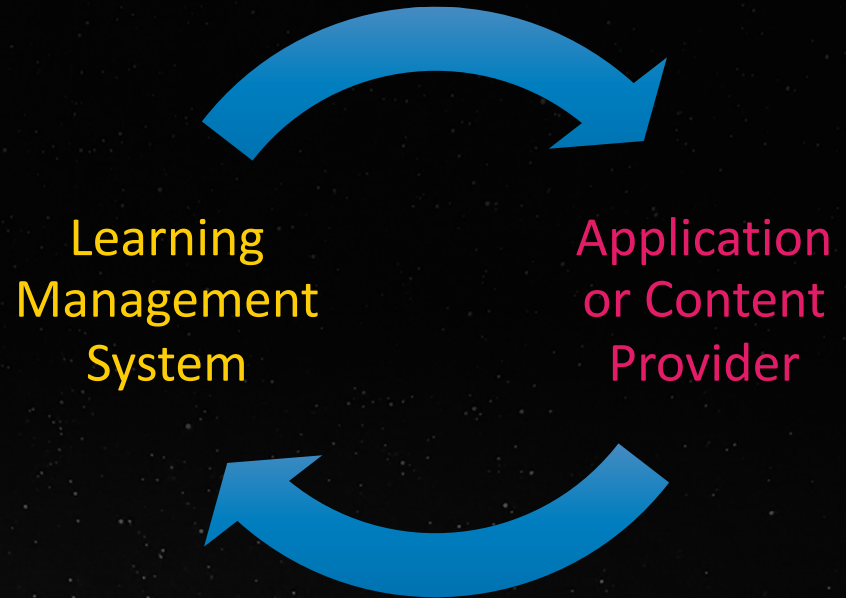
- Blackboard Building Block
- Java Web Application
- Access to Learn Java APIs
- Access to Learn database (ouch)
- Runs inside Learn JVM (super ouch)
- UI of choice?
- XML configuration files
- Not supported in Ultra
- There be DRAGONS



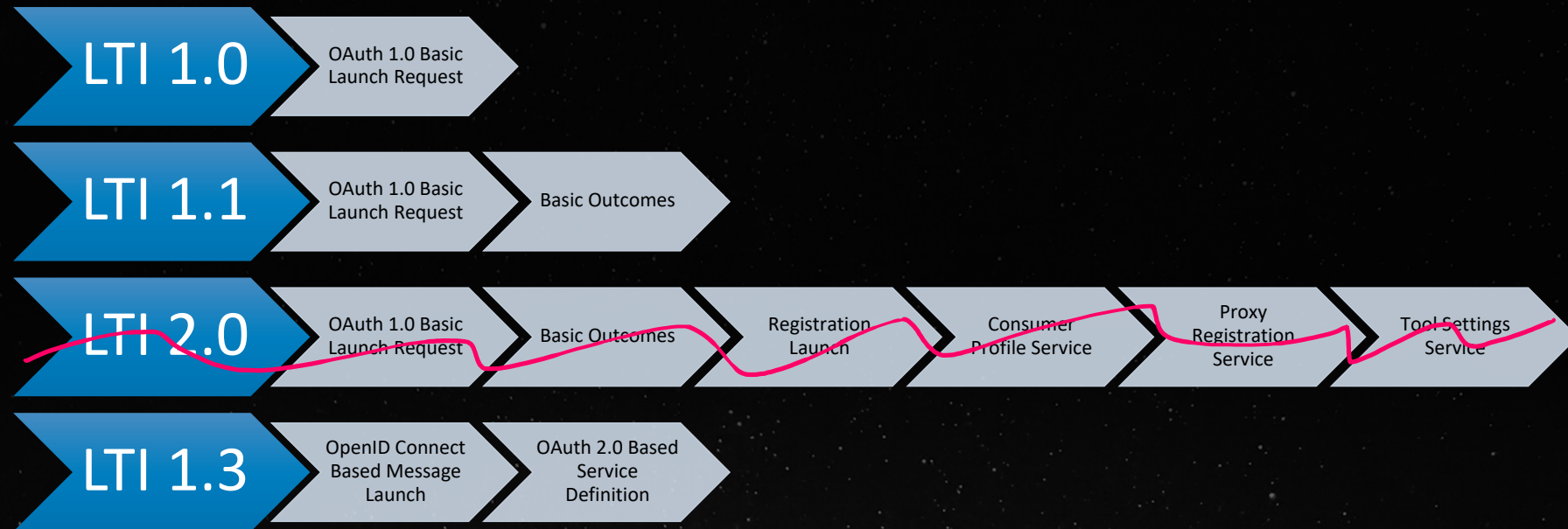
# What is LTI?

- A UI flow to connect LMS/VLE (Platform) to an Application (Tool)
- Provides Single-sign On (SSO) capabilities – a **TRUST** relationship between Platform and Tool
- Services to get data into and from a Platform
  - Content, e.g., Video, Assignment, Book
  - Roster
  - Grades

**TWO VERSIONS:** 1.1 and 1.3/Advantage

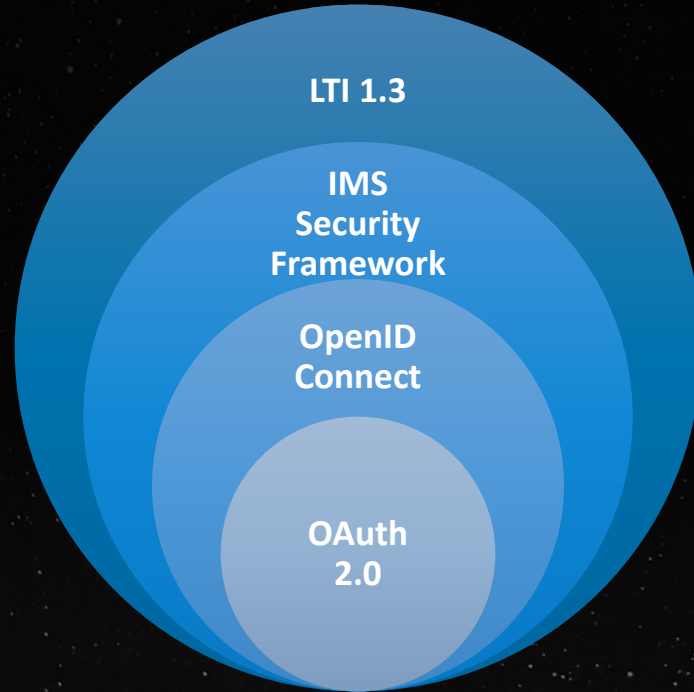


# A Brief History of LTI



# LTI 1.3/Advantage Security

---

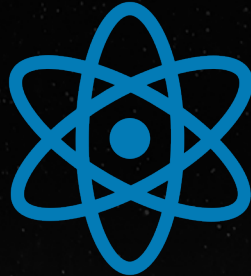




# Meet LTI Advantage

---

LTI 1.3 Core



Deep Linking

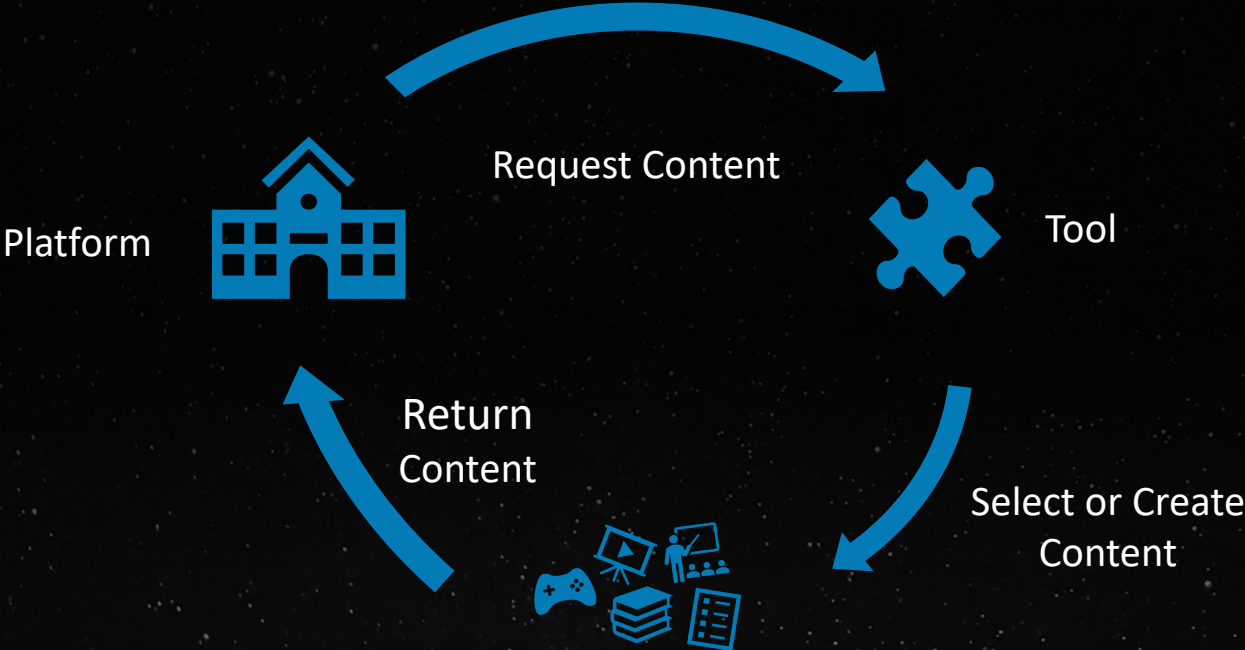


Names and  
Roles



Assignments  
and Grades

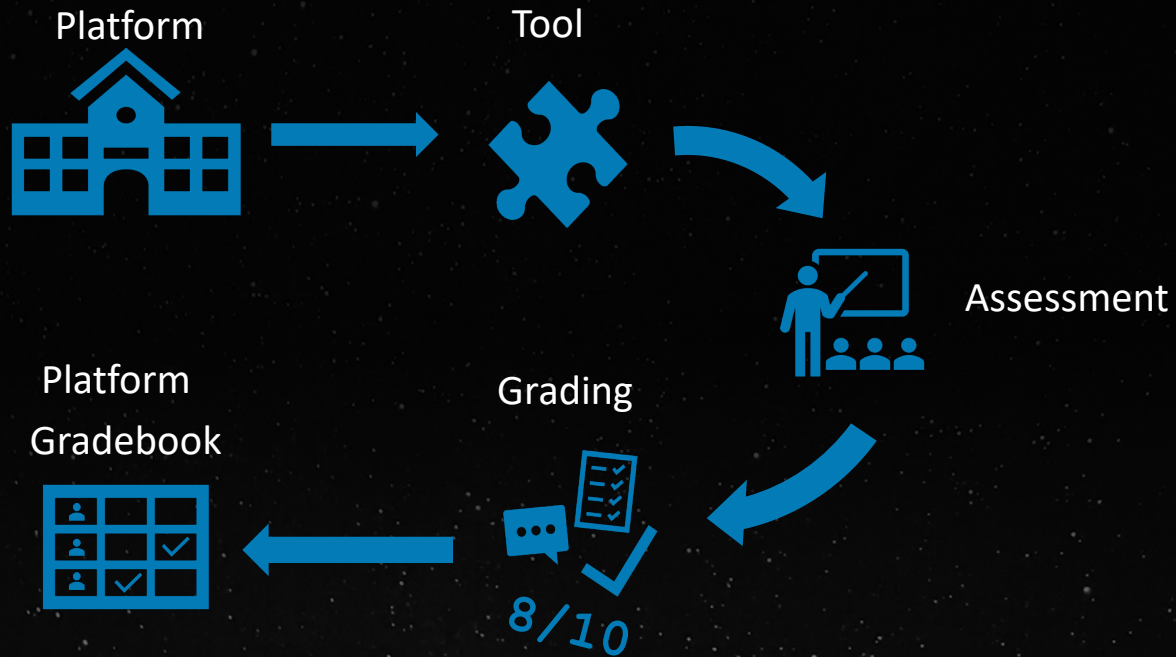
# Deep Linking



# Names and Roles



# Assignments and Grades



# Application Developer

---

- Build an awesome web application
  - Language and UI framework of choice
- Support LTI Launch
- Get IMS Global certification
- Deploy it somewhere (Azure, AWS, Heroku, ...)
- Register it with Blackboard **ONCE**
- Deploy it to Learn (Admin)
- Use it in a course or however you like (Instructors & Students)

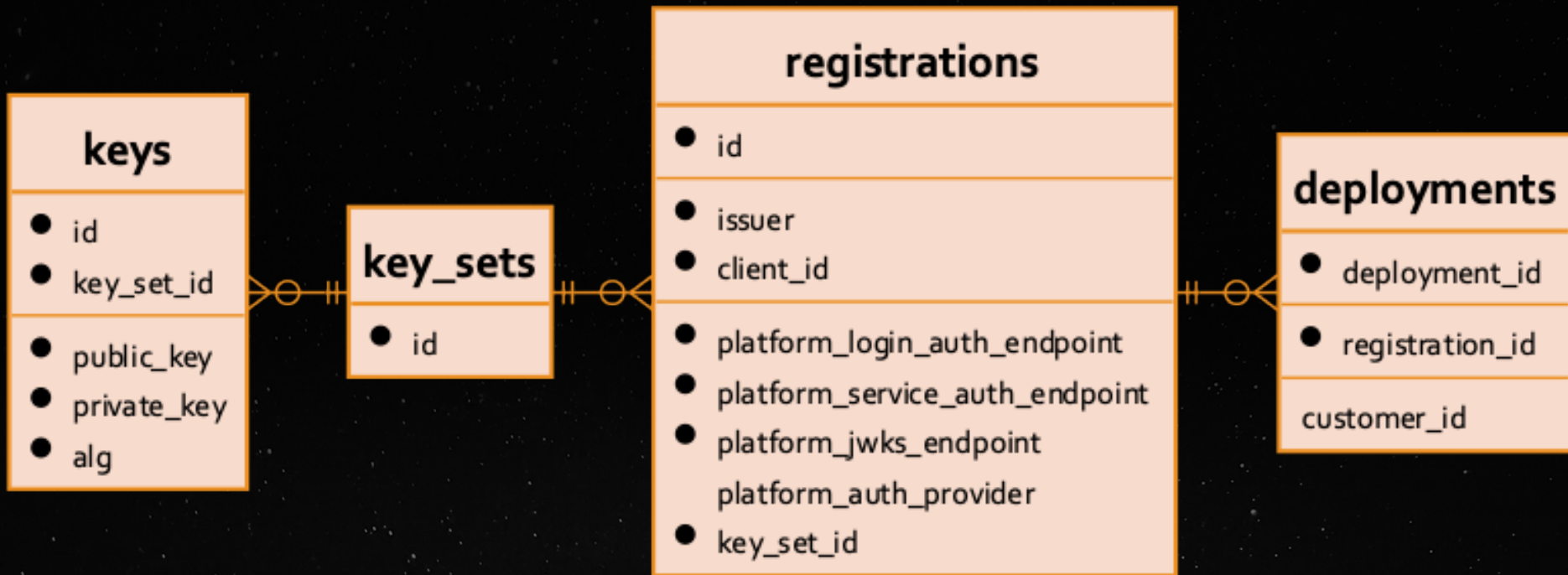


# Bad News: Specs you should read

---

- OAuth 2:
  - <https://oauth.net/2/> (Client Credentials flow)
  - <https://www.digitalocean.com/community/tutorials/an-introduction-to-oauth-2>
- OpenID Connect:
  - [https://openid.net/specs/openid-connect-core-1\\_0.html#ThirdPartyInitiatedLogin](https://openid.net/specs/openid-connect-core-1_0.html#ThirdPartyInitiatedLogin)
- JWT:
  - <https://tools.ietf.org/html/rfc7519>
  - <https://jwt.io>
- LTI Advantage:
  - <https://www.imsglobal.org/ims-security-framework>
  - <https://www.imsglobal.org/activity/learning-tools-interoperability>

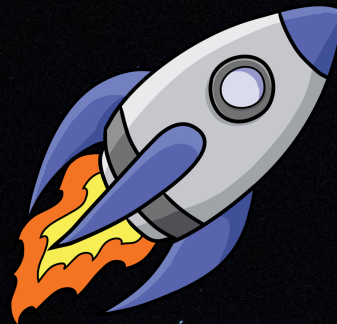
# Sample Tool Data Model



# LTI 1.3 Launch

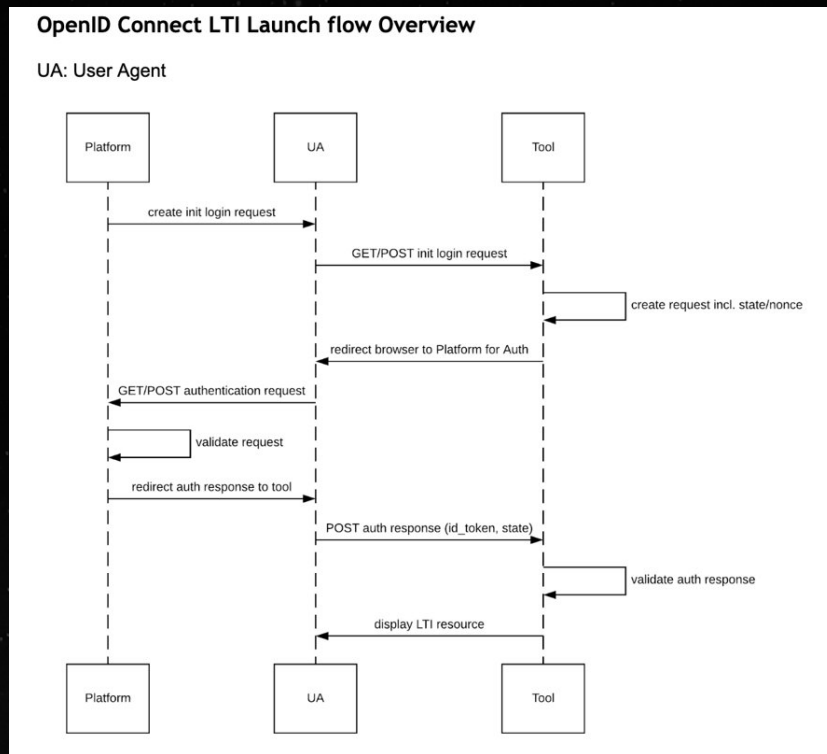
---

- OIDC login
- Parse JWT – header.body.signature
  - The launch is an OpenID id\_token (JWT)
- Validate header
  - Get kid and public key from JWKS URL
  - Get algorithm and validate it is what you expect
- Validate payload & signature
- Get payload





# Open ID Connect 3<sup>rd</sup>-party-initiated Login Flow



# OIDC Login GET or POST – create **state** and **nonce**

```
155 exports.oidcLogin = function(req, res) {
156   let state = uuid.v4();
157   let nonce = uuid.v4();
158   let url =
159     config.oidcAuthUrl +
160     "?response_type=id_token" +
161     "&scope=openid" +
162     "&login_hint=" +
163     req.query.login_hint +
164     "&lti_message_hint=" +
165     req.query.lti_message_hint +
166     "&state=" +
167     state +
168     "&redirect_uri=" +
169     encodeURIComponent(`${config.frontendUrl}/lti13`) +
170     "&client_id=" +
171     config.bbClientId +
172     "&nonce=" +
173     nonce;
174
175   // Per the OIDC best practices, save the state in a cookie, and check it on the way back in
176   res.cookie("state", state, { sameSite: 'none', secure: true, httpOnly: true });
177
178   console.log("LTI JWT login init; redirecting to: " + url);
179   res.redirect(url);
180 };
```



Get from registration

# Receive Launch POST – id\_token and state

Algorithm RS256

**Encoded** PASTE A TOKEN HERE

```
eyJraWQiOiJlNGExMzd1MS1lNWMyLTQxNWUtYmY4  
OC0yZDZiMWMxOWI0Y2QiLCJhbGciOiJSUzI1NiJ9  
.eyJzdWIiOiJlZDE3N2ZlZk1NGQ0YzEyYjA4MWMw  
5ZTAyM2Y4ODkxNSIsImh0dHBzOiIwXC9wdXJsLm1  
tc2dsb2JhbC5vcmdcL3NwZWNCe2x0aVwvY2xhaW1  
cL2RlcGxveW11bnRfaWQiOiJhMTYyOWUyYi1mYjY  
yLTQ3ODgtOGUzYi1lMDQ4YTFiMTE5MzEiLCJodHR  
wczpcL1wvchVybC5pbXNnbG9iYWwub3JnXC9zcGV  
jXC9sdGlcL2NsYWltXC9Z2XJzaW9uIjoiaMS4zLjA  
iLCJodHRwczpcL1wvchVybC5pbXNnbG9iYWwub3J  
nXC9zcGVjXC9sdGktYWdzXC9jbGFpbVwvZW5kcG9  
pbmQiOnsibGluZW10ZW1zIjoiaHR0cHM6XC9cL21  
5bGVhcm4uaW50LmJicGQuaW9cL2x1YXJucXc9hcG1  
cL3YxXC9sdGlcL2NvdXJzZXNcL180XzFcl2xpbmV  
JdGVtcyIsImxpbnVpdGVtIjoiaHR0cHM6XC9cL21  
5bGVhcm4uaW50LmJicGQuaW9cL2x1YXJucXc9hcG1  
cL3YxXC9sdGlcL2NvdXJzZXNcL180XzFcl2xpbmV  
JdGVtcyIsImxpbnVpdGVtIjoiaHR0cHM6XC9cL21
```

**Decoded** EDIT THE PAYLOAD AND SECRET

**HEADER: ALGORITHM & TOKEN TYPE**

```
{  
  "kid": "e4a137e1-e5c2-415e-bf88-2d2b1c19b4cd",  
  "alg": "RS256"  
}
```

**PAYLOAD: DATA**

```
{  
  "sub": "ed177fee954d4c12b081c9e023f88915",  
  
  "https://purl.imsglobal.org/spec/lti/claim/deployment_id":  
    "a1629e2b-fb62-4788-8e3b-e048a1b11931",  
  "https://purl.imsglobal.org/spec/lti/claim/version":  
    "1.3.0",  
  "https://purl.imsglobal.org/spec/lti-  
ags/claim/endpoint": {  
    "lineitems":  
      "https://mylearn.int.bbpd.io/learn/api/v1/lti/courses/_4_1  
/lineItems",  
    "lineitem":  
      "https://mylearn.int.bbpd.io/learn/api/v1/lti/courses/_4_1  
/lineItems/_1481_1",  
    "scope": [
```

# Validate State & JWT

```
app.post('/lti13', (req, res) => {
  console.log('-----\nlti13');

  // Per the OIDC best practices, ensure the state parameter passed in here matches the one in our cookie
  const cookieState = req.cookies['state'];
  if (cookieState !== req.body.state) {
    res.send(`The state field is missing or doesn't match.`);
    return;
  }

  jwtPayload = ltiAdv.verifyToken(req.body.id_token);
  if (!jwtPayload || !jwtPayload.verified) {
    res.send('An error occurred processing the id_token.');
```

# Verify JWT Client ID

```
30 // Validates the signature and content of the JWT
31 exports.verifyToken = function(id_token) {
32     let parts = id_token.split('.');
33
34     // Parse and store payload data from launch
35     let jwtPayload = new JWTPayload();
36     jwtPayload.header = JSON.parse(Buffer.from(parts[0], encoding: 'base64').toString());
37     jwtPayload.body = JSON.parse(Buffer.from(parts[1], encoding: 'base64').toString());
38     jwtPayload.verified = false;
39
40     // Verify launch is from correct party
41     // aud could be an array or a single entry
42     let clientId;
43     if (jwtPayload.body.aud instanceof Array) {
44         clientId = jwtPayload.body.aud[0];
45     } else {
46         clientId = jwtPayload.body.aud;
47     }
48
49     if (clientId !== config.bbClientId) {
50         console.log('Client ID passed in does not match configured client ID');
51         return null;
52     }
53 }
```

Verify issuer

# Verify JWT Signature

```
49
50 // Get the public keys from the platform JWKS URL
51 try {
52   console.log(`LMS JWKS URL ${config.jwksUrl}`);
53   const response = await axios.get(config.jwksUrl);
54   console.log(`Public keys ${JSON.stringify(response.data)}`);
55
56   const key = response.data.keys.find(k => k.kid === jwtPayload.header.kid);
57
58   jwt.verify(id_token, jwk2pem(key));
59   jwtPayload.verified = true;
60 } catch (err) {
61   console.log(`Get public keys failed: ${JSON.stringify(err)}`);
62 }
63
64 return jwtPayload;
65 };
66
```

# Parse the JWT

```
49   const targetLinkUri = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti/claim/target_link_uri' ];
50   let returnUrl;
51   let deepLinkData;
52   let isDeepLinking = false;
53
54   if (targetLinkUri.endsWith('deepLink')) {
55     returnUrl = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti-dl/claim/deep_linking_settings' ].deep_link_return_url;
56     deepLinkData = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti-dl/claim/deep_linking_settings' ].data;
57     isDeepLinking = true;
58   } else if (targetLinkUri.endsWith('lti13')) {
59     returnUrl = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti/claim/launch_presentation' ].return_url;
60   } else {
61     res.send('We don't recognize that targetLinkUri ${targetLinkUri}');
62   }
63
64   const roles = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti/claim/roles' ];
65   console.log('Roles are ${JSON.stringify(roles)}');
66
67   let isStudent = false;
68   for (let i = 0; i < roles.length; i++) {
69     if (roles[i] === 'http://purl.imsglobal.org/vocab/lis/v2/membership#Learner') {
70       isStudent = true;
71       break;
72     }
73   }
74
75   const learnServer = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti/claim/tool_platform' ].url;
76   const lmsType = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti/claim/tool_platform' ].product_family_code;
77   const custom = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti/claim/custom' ];
78   const nrpsUrl = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti-nrps/claim/roleservice' ].context_memberships_url + '&groups=true';
79   const agsUrl = jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti-ags/claim/endpoint' ].lineitem;
80
81   const learnInfo = {
82     userId: jwtPayload.body[ 'sub' ],
83     courseUUID: jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti/claim/context' ].id,
84     courseId: jwtPayload.body[ 'https://purl.imsglobal.org/spec/lti/claim/context' ].label, // Learn's Course ID
85     learnHost: learnServer,
```

# Get OAuth Tokens and Redirect to Tool UI

```
146 // If we have a 3LO auth code, let's get us a bearer token here.
147
148 const nonce = uuid.v4();
149
150 const restToken = restService.getLearnRestToken(learnUrl, nonce);
151 console.log(`Learn REST token ${restToken}`);
152
153 // Now get the LTI OAuth 2 bearer token (shame they aren't the same)
154 const ltiToken = await ltiTokenService.getLTIToken(config.bbClientId, config.oauthTokenUrl, scopes);
155
156 // Cache the LTI token
157 await ltiTokenService.cacheToken(ltiToken, nonce);
158
159 // Now finally redirect to the IVS app
160 res.redirect(`/?nonce=${nonce}&returnurl=${returnUrl}&cname=${courseName}&student=${isStudent}&dl=${isDeepLinking}&setLang=${learnLocale}#/viewAssignment`);
161 });
162
```



# Handle Deep Linking

- Just like LTI 1.3 launch, but different
- Build a UI to create/select content
- Build payload to send back to LMS
  - Can be links or embedded
- Sign payload (JWT)
- Post back to LMS return URL

An Ultra Course  
Assignment

Name

Description

| User name    | Email                | Grade |
|--------------|----------------------|-------|
| Suzy Queue   | [REDACTED]@gmail.com | 0     |
| Joe Coole    | [REDACTED]@gmail.com | 0     |
| Eric Preston | [REDACTED]@gmail.com | 0     |

Cancel Submit

# Deep Link Launch

```
48
49  const targetLinkUri = jwtPayload.body['https://purl.imsglobal.org/spec/lti/claim/target_link_uri'];
50  let returnUrl;
51  let deepLinkData;
52  let isDeepLinking = false;
53
54  if (targetLinkUri.endsWith('deeplink')) {
55    returnUrl = jwtPayload.body['https://purl.imsglobal.org/spec/lti-dl/claim/deep_linking_settings'].deep_link_return_url;
56    deepLinkData = jwtPayload.body['https://purl.imsglobal.org/spec/lti-dl/claim/deep_linking_settings'].data;
57    isDeepLinking = true;
58  } else if (targetLinkUri.endsWith('lti13')) {
59    returnUrl = jwtPayload.body['https://purl.imsglobal.org/spec/lti/claim/launch_presentation'].return_url;
60  } else {
61    res.send(`We don't recognize that targetLinkUri ${targetLinkUri}`);
62  }
63
```

# Deep Link Response

```
171 app.post('/sendAssignment', async (req, res) => {
172     let body = req.body;
173     console.log('sendAssignment called: ${JSON.stringify(body)}');
174
175     // Get the OAuth2 bearer token from our cache based on the nonce. The nonce serves two purposes:
176     // 1. Protects against CSRF
177     // 2. Is the key for our cached bearer tokens
178     const nonce = body.nonce;
179     const cachedNonce = await redisUtil.redisGet(nonce);
180
181     if (!cachedNonce) {
182         console.log(`Couldn't find nonce...exiting`);
183         res.send('Could not find nonce');
184     }
185
186     // Remove the nonce so it can't be replayed
187     redisUtil.redisDelete(nonce);
188
189     const restToken = restService.getCachedToken(nonce);
190     console.log('sendAssignment got bearer token ${restToken}');
191
192     const learnInfo = await ltiAdv.getLearnInfo(req.cookies[LEARN_INFO_KEY]);
193
194     const deepLinkReturn = await deepLinkService.createDeepContent(body.assignment, learnInfo, restToken);
195     console.log('sendAssignment got deep link return ${JSON.stringify(deepLinkReturn)}');
196     res.send(deepLinkReturn);
197 }
```

# Deep Link Response JSON

```
60 const createDeepLinkJwt = function (assignment, learnInfo) {
61   const assignmentId = uuid.v4();
62
63   // Save this assignment
64   assignmentService.saveAssignment(assignmentId, assignment);
65
66   const contentItems = [{
67     type: "ltiResourceLink",
68     title: assignment.name,
69     text: assignment.submission,
70     url: `${config.frontendUrl}/lti13`,
71     available: {
72       endDateTime: assignment.endDateTime,
73     },
74     submission: {
75       endDateTime: assignment.endDateTime,
76     },
77     iframe: {
78       width: 600,
79       height: 400,
80     },
81     lineItem: {
82       scoreMaximum: 100,
83       label: assignment.name,
84       resourceId: assignmentId,
85       tag: "essay"
86     },
87     custom: {
88       resourceId: assignmentId,
89       userName: `${User.username}`
90     }
91   }];
92
```

Embed

# Deep Link Response JWT

```
92
93     const now = moment.now() / 1000;
94     const deepLinkResponse = {
95         iss: config.bbClientId,
96         aud: learnInfo.iss,
97         sub: config.bbClientId,
98         iat: now,
99         exp: now + 5 * 60,
100        locale: "en_US",
101        "https://purl.imsglobal.org/spec/lti/claim/deployment_id": learnInfo.deployId,
102        "https://purl.imsglobal.org/spec/lti/claim/message_type": "LtiDeepLinkingResponse",
103        "https://purl.imsglobal.org/spec/lti/claim/version": "1.3.0",
104        "https://purl.imsglobal.org/spec/lti-dl/claim/data": learnInfo.deepLinkData,
105        "https://purl.imsglobal.org/spec/lti-dl/claim/content_items": contentItems
106    };
107
108     console.log(`Deep link creator returned: ${JSON.stringify(deepLinkResponse)}`);
109
110     return ltiAdv.signJwt(deepLinkResponse);
111 }
112
```

# Sign Response JWT

```
109
110 exports.signJwt = function(json) {
111   try {
112     let privateKey = jwk2pem(config.privateKey);
113     const signedJwt = jwt.sign(json, privateKey, options: {algorithm: 'RS256', keyid: '12345'});
114     console.log(`signedJwt ${signedJwt}`);
115     return signedJwt
116   } catch (exception) {
117     console.log(`Something bad happened in signing ${exception}`);
118   }
119 };
120
```

## Send the Response as a form POST

```
88
89
90
91
92 // The LTI Deep Linking spec requires a form POST back to the Platform
93 const form = document.createElement( tagName: 'form' );
94 form.setAttribute( qualifiedName: 'action', params.getReturnUrl() as string );
95 form.setAttribute( qualifiedName: 'method', value: 'POST' );
96 const jwtParam = document.createElement( tagName: 'input' );
97 jwtParam.setAttribute( qualifiedName: 'name', value: 'JWT' );
98 jwtParam.setAttribute( qualifiedName: 'value', response.data );
99 form.appendChild( jwtParam );
100 document.body.appendChild( form );
101 form.submit();
102 });
```

# Provide URL to your Public Keys

---

- JSON Web Keyset or JWKS URL
- JWK format
- PEM2JWK converter  
<https://8gwifi.org/jwkconvertfunctions.jsp>

```
"publicKeys": {
  "keys": [
    {
      "kty": "RSA",
      "e": "AQAB",
      "use": "sig",
      "kid": "12345",
      "alg": "RS256",
      "n": "sB3jz6IZB0uergkZ-RUpCoZuNeaL2A2"
    },
    {
      "kty": "RSA",
      "e": "AQAB",
      "use": "sig",
      "kid": "abcdefg",
      "alg": "RS256",
      "n": "sB3jz6IZB0uergkZ-RUpCoZuNeaL2A2"
    }
  ]
}
```



# Big Sigh of Relief

The screenshot shows the Blackboard Ultra Course interface for 'ULTRA1 An Ultra Course'. At the top right, there is a notification for 'digest 10ms 5ms 9ms'. The course title is 'An Ultra Course'. The instructor is Eric Preston, with a profile picture and the title 'Instructor'. A vertical sidebar on the left contains navigation icons. The main content is divided into two columns: 'Details & Actions' and 'Course Content'. The 'Details & Actions' column lists various management tasks with links. The 'Course Content' column lists course items, including 'Assignment 1', 'LTI Dev Test 1.1 Content', 'Local LTI 1.3 Content', and 'Bobcat Discussion', each with its due date and visibility settings.

ULTRA1

## An Ultra Course

digest 10ms 5ms 9ms

Eric Preston  
Instructor

### Details & Actions

- Roster  
[View everyone in your course](#)
- Course Groups  
[Create and manage groups](#)
- Course is open  
[Students can access this course](#)
- Attendance  
[Mark attendance](#)
- Announcements  
[Create announcement](#)
- Books & Tools  
[View course & institution tools](#)
- Question Banks  
[Manage banks](#)
- Student Preview  
[Enter student preview mode](#)

### Course Content

- Assignment 1  
Due date: 7/15/20, 1:00 PM  
Conditional availability: [Available based on date](#)  
Do this
- LTI Dev Test 1.1 Content  
Hidden from students
- Local LTI 1.3 Content  
Hidden from students
- Bobcat Discussion  
Due date: 9/30/20, 2:43 PM  
Visible to students  
Talk about Bobcats

# Break Time



# Get LTI OAuth 2 Bearer Token

```
7 const oauth2JWT = (clientId, tokenUrl) => {
8   let now = Math.trunc( * new Date().getTime() / 1000);
9   let json = {
10    iss: "lti-tool",
11    sub: clientId,
12    aud: [tokenUrl, 'foo'],
13    iat: now,
14    exp: now + 5 * 60,
15    jti: crypto.randomBytes( size: 16).toString( encoding: "hex")
16   };
17
18   return ltiAdv.signJwt(json);
19 };
20
21 exports.getLTIToken = async (clientId, tokenUrl, scope) => {
22   console.log('getLTIToken client ${clientId} tokenUrl: ${tokenUrl} scope: ${scope}');
23   const clientAssertion = oauth2JWT(clientId, tokenUrl);
24
25   const options = {
26     method: "POST",
27     url: tokenUrl,
28     headers: {
29       'Content-Type': 'application/x-www-form-urlencoded'
30     }
31   };
32
33   const body = {
34     grant_type: "client_credentials",
35     client_assertion_type:
36       "urn:ietf:params:oauth:client-assertion-type:jwt-bearer",
37     client_assertion: clientAssertion,
38     scope: scope
39   };
40
41   try {
42     const response = await axios.post(tokenUrl, qs.stringify(body), options);
43     const token = response.data.access_token;
```

# Get Course Roster – Names & Roles Provisioning Service

```
4 exports.loadUsers = async (courseId, resourceId, url, token) => {
5   if (!courseId) return [];
6
7   const body = {
8     method: 'GET',
9     uri: url,
10    headers: {
11      'content-type': "application/vnd.ims.lti-npns.v2.membershipcontainer+json",
12      Authorization: "Bearer " + token
13    }
14  };
15
16  console.log(`loadUsers for ${resourceId}, request: ${JSON.stringify(body)}`);
17  try {
18    const response = await axios.get(url, body);
19
20    const members = response.data.members;
21    console.log(`loadUsers returning ${JSON.stringify(response.data)}`);
22
23    let users = [];
24    for (let i = 0; i < members.length; i++) {
25      const submission = await redisUtil.redisGet(`key: ${resourceId}:${members[i].user_id}`);
26      console.log(`Submission for ${members[i].user_id}: ${JSON.stringify(submission)}`);
27
28      const user = {
29        id: members[i].user_id,
30        name: members[i].name,
31        email: members[i].email,
32        grade: submission ? submission.grade : '0'
33      }
34
35      users.push(user);
36    }
37    return users;
38  } catch (exception) {
```

# Names & Roles (and Groups) Response

```
1 - {
2   "id": "https://mylearn.int.bbpd.io/learn/api/v1/lti/external/namesandroles/_4_1?placement_id=_537_1",
3   "context": {
4     "id": "b21a35dbbcb49e29b96d67ad3e77e5a",
5     "label": "ULTRAI",
6     "title": "An Ultra Course"
7   },
8   "members": [
9     {
10      "status": "Active",
11      "name": "Suzy Queue",
12      "given_name": "Suzy",
13      "family_name": "Queue",
14      "middle_name": "",
15      "email": "sue@gmail.com",
16      "user_id": "ed177fee954d4c12b081c9e023f88915",
17      "lis_person_sourcedid": "sue",
18      "roles": [
19        "http://purl.imsglobal.org/vocab/lis/v2/membership#Learner"
20      ],
21      "group_enrollments": []
22    },
23    {
24      "status": "Active",
25      "name": "Joe Coole",
26      "given_name": "Joe",
27      "family_name": "Coole",
28      "middle_name": "",
29      "email": "joe@gmail.com",
30      "user_id": "bcd30d0868464d859a714d2b597e1131",
31      "lis_person_sourcedid": "joe",
32      "roles": [
33        "http://purl.imsglobal.org/vocab/lis/v2/membership#Learner"
34      ],
35      "group_enrollments": []
36    },
37    {
38      "status": "Active",
39      "name": "Eric Preston",
40      "given_name": "Eric",
41      "family_name": "Preston",
42      "middle_name": "",
43      "email": "eric@gmail.com",
44      "user_id": "d7443496cba24f1b96cc97e6473b9265",
45      "lis_person_sourcedid": "eric",
46      "roles": [
47        "http://purl.imsglobal.org/vocab/lis/v2/membership#Instructor"
48      ],
49      "group_enrollments": []
50    }
51  ]
52 }
```

# Post a Score to Assignment & Grades Service

```
3
4 exports.sendGrade = async (courseId, userId, score, url, token) => {
5   const scoresUrl = `${url}/scores`;
6
7   const options = {
8     method: 'POST',
9     uri: scoresUrl,
10    headers: {
11      'content-type': 'application/vnd.ims.lis.v2.lisitem+json',
12      Authorization: 'Bearer ' + token
13    }
14  };
15
16  const scoreBody = {
17    userId: userId,
18    scoreGiven: score ?? null,
19    scoreMaximum: 100.0,
20    comment: 'This is exceptional work.',
21    timestamp: moment().toISOString(),
22    activityProgress: 'Completed',
23    gradingProgress: 'FullyGraded'
24  };
25
26  console.log(`sendGrade body: ${JSON.stringify(scoreBody)} options: ${JSON.stringify(options)}`);
27
28  try {
29    const response = await axios.post(scoresUrl, scoreBody, options);
30
31    const grade = response.data;
32    console.log(`sendGrade returning ${JSON.stringify(grade)}`);
33
34    return grade;
35  } catch (exception) {
```

## Some “Gotchas”

---

- In an `id_token`, the “`aud`” can be either an array or a string.
- Launch URLs are pre-registered so cannot change per-launch. For backwards compatibility, use the target link URI in the `https://www.imslobal.org/spec/lti/v1p3/#target-link-uri` claim.
- There are three ways a tool’s public key can be shared – don’t use the other two
  - Recommended – A tool provides a JWKS URL with their public keys.
- In a deep linking launch the “`data`” value given by the platform must be returned unchanged.
- When requesting an access token, the “`aud`” must be the platform auth provider if given. If no platform auth provider is given the auth token endpoint must be used instead.
- The OpenID Connect login request can be a get or a post.
- A registration can have many deployments, never assume there is only one.

# Public REST API – When LTI is not Enough

---

- Decide whether to use 3-legged OAuth (act on behalf of user)
- Get a bearer token
  - Different than LTI token
- Construct a payload
- Call the APIs



## 3-legged OAuth – get auth code

```
106
107 // At this point we want to get the 3LO auth code, and then OAuth2 bearer token, and THEN we can send the user
108 // to the MS Teams Meeting app UI.
109
110 const redirectUri = `${config.frontendUrl}/tldcode&scope=*&response_type=code&client_id=${config.appKey}&state=${cookieState}`;
111 const authcodeUrl = `${learnServer}/learn/api/public/v1/oauth2/authorizationcode?redirect_uri=${redirectUri}`;
112
113 console.log(`Redirect to get 3LO code ${authcodeUrl}`);
114 res.redirect(authcodeUrl);
115 });
116
117 // The 3LO redirect route
118 app.get('/tldcode', async (req, res) => {
119   console.log(`tldcode called with code: ${req.query.code} and state: ${req.query.state}`);
120
121   const cookieState = req.cookies['state'];
122   if (cookieState !== req.query.state) {
123     res.send(`The state field is missing or doesn't match.`);
124     return;
125   }
126 }
```

## 3-legged OAuth – receive auth code

```
144     const redirectUri = `${config.frontendUrl}/tlocode`;
145     const learnUrl = learnHost + `/learn/api/public/v1/oauth2/token?code=${req.query.code}&redirect_uri=${redirectUri}`;
146
147     // If we have a 3LO auth code, let's get us a bearer token here.
148     const nonce = uuid.v4();
149
150     // Cache the nonce
151     redisUtil.redisSave(nonce, 'nonce');
152
153     const restToken = restService.getLearnRestToken(learnUrl, nonce);
154     console.log(`Learn REST token ${restToken}`);
155
156     // Now get the LTI OAuth 2 bearer token (shame they aren't the same)
157     const ltiToken = await ltiTokenService.getLTIToken(config.bbClientId, config.oauthTokenUrl, scopes, nonce);
158     console.log(`Learn LTI token ${ltiToken}`);
159
160     // Now finally redirect to the UI
161     res.redirect(`/?nonce=${nonce}&returnurl=${returnUrl}&cname=${courseName}&student=${isStudent}&dl=${isDeepLinking}&setLang=${learnLocale}#/viewAssignment`);
162   });
163
164
```

# Get REST OAuth 2 Bearer Token

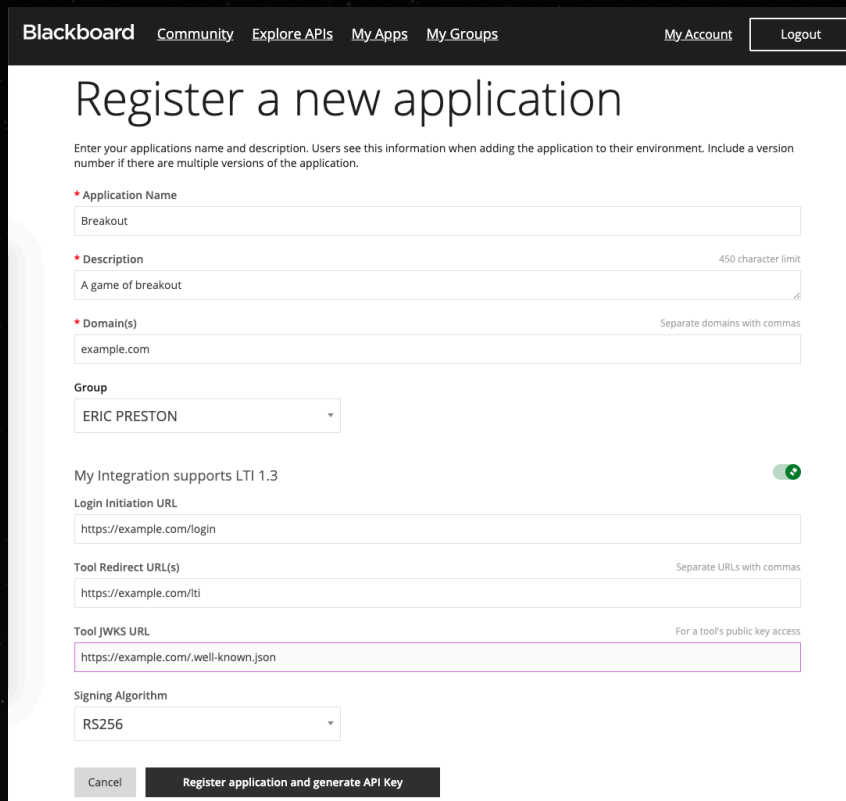
```
5 exports.getLearnRestToken = async (learnUrl, nonce) => {
6   const auth_hash = new Buffer.from( obj: `${config.appKey}:${config.appSecret}`).toString( encoding: 'base64');
7   const auth_string = `Basic ${auth_hash}`;
8   console.log(`Auth string: ${auth_string}`);
9   const options = {
10    headers: {
11      Authorization: auth_string,
12      'Content-Type': 'application/x-www-form-urlencoded'
13    }
14  };
15
16  console.log(`Getting REST bearer token at ${learnUrl}`);
17  try {
18    const response = await axios.post(learnUrl, {data: 'grant_type=authorization_code', options});
19    const token = response.data.access_token;
20    console.log("Got bearer token");
21
22    // Cache the nonce
23    redisUtil.redisSave(nonce, 'nonce');
24
25    // Cache the REST token
26    redisUtil.redisSave(`${nonce}:rest`, token);
27    return token;
28  } catch (exception) {
29    console.log(`Failed to get token with response ${JSON.stringify(exception)}`);
30    return '';
31  }
32};
```

# Get Courses & Create Calendar

```
23 const createCalendarItem = async (parameter, token: any, learnInfo, token) => {
24   const xhrConfig = {
25     headers: {Authorization: `Bearer ${token}`}
26   };
27
28   try {
29     // First we need to get what type of course we've got so we can get the calendar ID
30     const courseResponse = await axios.get(`url: ${learnInfo.learnHost}/Learn/api/public/v2/courses/uuid:${learnInfo.courseUUID}`, xhrConfig);
31
32     console.log(`Got course; Ultra status is ${courseResponse.data.ultraStatus}, and PK1 is: ${courseResponse.data.id}`);
33
34     // We need the course PK1 for the Calendar API
35     const calendarOptions = {
36       calendarId: courseResponse.data.id,
37       type: 'Course',
38       title: assignment.name,
39       location: assignment.submission,
40       description: assignment.submission,
41       start: moment().startOf('unitOfTime: 'hour').add( amount: 1, unit: 'hour').toISOString(),
42       end: moment().startOf('unitOfTime: 'hour').add( amount: 2, unit: 'hour').toISOString()
43     };
44
45     console.log(`Calendar create options: ${JSON.stringify(calendarOptions)}`);
46
47     const learnUrl = `${learnInfo.learnHost}/Learn/api/public/v1/calendars/items`;
48
49     // Create the calendar item
50     const response = await axios.post(learnUrl, calendarOptions, xhrConfig);
51     console.log(`Created calendar item!!! ${JSON.stringify(response.data)}`);
52   } catch (exception) {
53     console.log(`Error creating calendar item: ${JSON.stringify(exception)}, from: ${learnInfo.learnHost}`);
54   }
55 };
56
```

# Register an LTI 1.3 Application

- Go to <https://developer.blackboard.com>
- Register your email to create an account
- Create an Application
  - Supports both REST and LTI Advantage
- Enter tool information:
  - OIDC Login URL
  - Redirect URI(s)
  - JWKS URL for your public key(s)



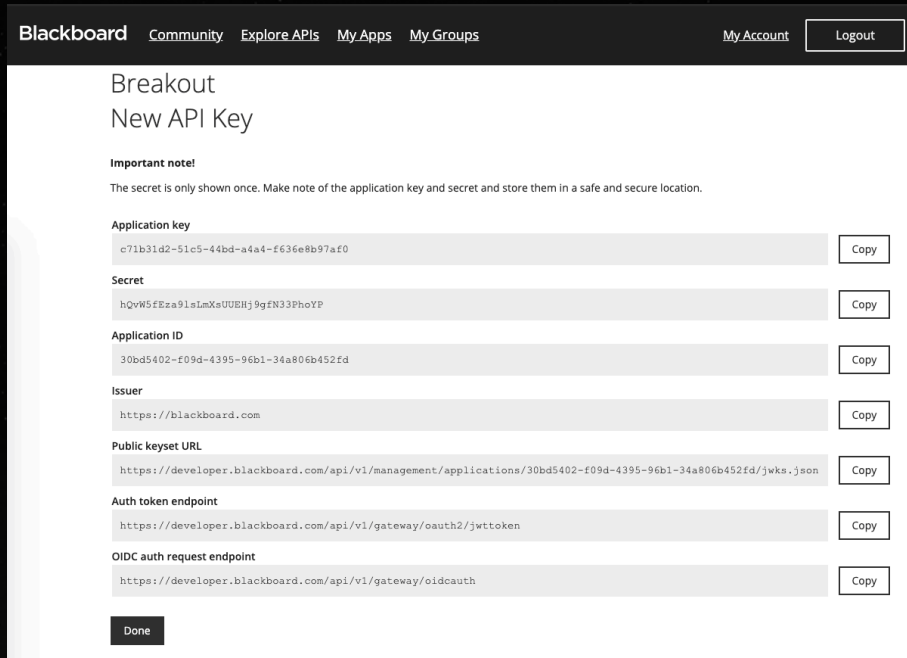
The screenshot shows the 'Register a new application' page on the Blackboard developer portal. The page has a dark header with 'Blackboard' and navigation links: 'Community', 'Explore APIs', 'My Apps', and 'My Groups'. On the right of the header are 'My Account' and 'Logout' buttons. The main content area is white and contains the following form fields:

- Application Name:** A text input field containing 'Breakout'.
- Description:** A text input field containing 'A game of breakout'. A note indicates a '450 character limit'.
- Domain(s):** A text input field containing 'example.com'. A note says 'Separate domains with commas'.
- Group:** A dropdown menu with 'ERIC PRESTON' selected.
- My Integration supports LTI 1.3:** A green checkmark icon.
- Login Initiation URL:** A text input field containing 'https://example.com/login'.
- Tool Redirect URL(s):** A text input field containing 'https://example.com/lti'. A note says 'Separate URLs with commas'.
- Tool JWKS URL:** A text input field containing 'https://example.com/.well-known.json'. A note says 'For a tool's public key access'.
- Signing Algorithm:** A dropdown menu with 'RS256' selected.

At the bottom of the form are two buttons: a grey 'Cancel' button and a dark grey 'Register application and generate API Key' button.

# Save Important Learn Values

- Application Key & Secret for REST
- Application ID == Client ID in LTI
- Issuer is always <https://blackboard.com>
- Learn JWKS URL
- Learn OAuth2 token endpoint (for grades, etc.)
- Learn OIDC auth endpoint (for launch flow)
- Update your tool's configuration



The screenshot shows the Blackboard interface for creating a new API key. The page title is "Breakout New API Key". It includes a navigation bar with "Blackboard", "Community", "Explore APIs", "My Apps", "My Groups", "My Account", and "Logout". An "Important note" states: "The secret is only shown once. Make note of the application key and secret and store them in a safe and secure location." Below this, several fields are displayed with their values and a "Copy" button next to each:

- Application key:** c71b31d2-51c5-44bd-a4a4-f636e8b97af0
- Secret:** hQvW5fEz91sLmXaUUEHj9gFN33PhoYP
- Application ID:** 30bd5402-f09d-4395-96b1-34a806b452fd
- Issuer:** https://blackboard.com
- Public keyset URL:** https://developer.blackboard.com/api/v1/management/applications/30bd5402-f09d-4395-96b1-34a806b452fd/jwks.json
- Auth token endpoint:** https://developer.blackboard.com/api/v1/gateway/oauth2/jwttoken
- OIDC auth request endpoint:** https://developer.blackboard.com/api/v1/gateway/oidcauth

A "Done" button is located at the bottom left of the content area.

# Learn Administrator

- Get the Application ID (Client ID in LTI-speak)
- Register REST Application - **OPTIONAL**
- Register LTI 1.3 Tool
- Create one or more Placements

### Administrator Tools

**TOOL STATUS**

*The following fields are read-only, but you can toggle the status of this tool*

|                    |  |
|--------------------|--|
| Client ID          | 1a3a0c34-fd56-4462-a743-e3e72554c3b5                     |
| Name               | LTI JWKS Test Tool                                       |
| Description        | A JWKS URL testing tool                                  |
| Deployment ID      | 4e479bda-dcde-4928-ab2f-0ee742aae260                     |
| Initiate Login URL | https://ltool.int.bbpd.io/login                          |
| Tool Redirect URLs | https://ltool.int.bbpd.io/lti3,https://ltool.int.bbpd.io |
| JWKS URL           | https://ltool.int.bbpd.io/.well-known/jwks.json          |
| Domains            | ltool.int.bbpd.io  |

Tool Status  
 Approved  
 Excluded

Tool Provider Custom Parameters  
assignment\_pk=@X@content.pk\_string@X@  
course\_pk=@X@course.pk\_string@X@  
userSysRoles=@X@user.role@X@  
courseStart=@X@courseSection.timeFrame.begin  
dueDate=@X@resourceLink.submission.endDateTime  
userEmail=@X@person.email.primary  
customRedirect=@X@custom.redirect@X@

Enter any custom parameters required by the tool provider. Parameters must each be on their own line and be entered in "name=value" format.

---

### INSTITUTION POLICIES

*You can change the following settings for this tool. The fields use global values by default.*

User Fields to Send  
 Role in Course  
 Name  
 Email Address

Allow grade service access  Yes  No

# LTI Placements in Learn

- Currently Support 6 types
  - Deep Linking
  - Content
  - Course
  - System
  - Admin
  - Ultra extension\*
- Usually specified by Application Developer
- \* A topic for another day, e.g., EasySoft

## Administrator Tools

### PLACEMENT INFORMATION

• Label   
The label that displays in the course

### Description

:

• Handle   
Uniquely identifies the placement

• Availability  Yes  No  
Make placement available to course builders and instructors

Type Placement Type determines where this tool appears in Blackboard Learn. The tool can be placed in a course or made available for specific users. [Learn more about placement types.](#)

Deep Linking content tool

Allow student access

Course content tool

Allows grading

Course tool

Allow student access

System tool

Administrator tool

Ultra extension

Not all Ultra extensions are visible to your users

Launch in New Window

Icon  Upload a custom icon that will be shown in the course. The size of the icon should be 50 by 50 pixels.

### TOOL PROVIDER INFORMATION

Enter the Tool Provider information. The Tool Provider URL must be located on one of the configured host names.

• Target Link URI

Tool Provider Custom

Click **Submit** to proceed.



# Demo Time

Bobcat Discussion

Discussion

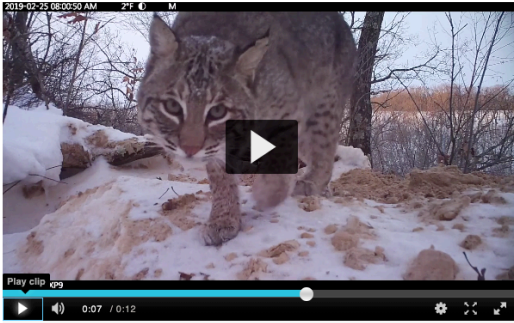
Discussion Topic

Talk about Bobcats

Responses (1)

Type a response

Eric Preston  
0 seconds ago, at 2:50 PM



2019-02-25 08:00:50 AM 2F 0 M

Play clip 0:07 / 0:12

Discussion Topic

Discuss

Participate

Find participants

## Recent and Future

---

- Ability for students to embed LTI content (e.g., videos)
- Course Groups with Names & Roles
- Embed in Original editor\*
- Tool developers define LTI Placements\*
- Submission Review spec so can launch to the right place in the tool from gradebook\*

\*Coming soon(ish)?



# LTI Resources

---

- <https://github.com/blackboard/BBDN-LTI-Adv-Node>
- <https://github.com/IMSGlobal/ltibootcamp>
- <https://pypi.org/project/PyLTI1p3/>
- <https://www.imslobal.org/ims-security-framework>
- <https://www.imslobal.org/activity/learning-tools-interoperability>
- <https://docs.blackboard.com/learn/REST/Getting%20Started%20With%20REST.html>
- <https://docs.blackboard.com/dvba/Using%20the%20Blackboard%20Learn%20AMI%20for%20REST%20and%20LTI%20Development.html>

Questions?



# DEVCON 20

Blackboard

#BbWorld20