

# MEETS SaaS LTI Launch Manual

## Table of Contents

MEETS SaaS LTI Launch Manual.....	1	<i>lis_person_name_full</i> .....	7
Overview.....	1	<i>context_title</i> .....	7
Assistance.....	2	<i>launch_presentation_locale</i> .....	7
Technology.....	2	<i>tool_consumer_info_product_family_code</i> .....	7
Security.....	2	<i>tool_consumer_info_version</i> .....	8
Credentials.....	2	<i>launch_presentation_return_url</i> .....	8
Access Endpoint.....	3	Other parameters.....	8
LTI.....	3	Custom Parameters.....	8
iframe.....	4	Endpoints.....	8
Required Parameters.....	4	<i>endpoint</i> .....	9
<i>user_id</i> .....	4	Themes.....	10
<i>lis_person_name_given</i> .....	4	<i>theme</i> .....	10
<i>lis_person_name_family</i> .....	4	Auxiliary data.....	10
<i>lis_person_contact_email_primary</i> .....	5	<i>auxiliary_user_&lt;user field name&gt;</i> .....	10
<i>context_id</i> .....	5	<i>auxiliary_context_&lt;context field name&gt;</i> .....	11
<i>roles</i> .....	5	Overrides.....	11
Signature Details.....	6	<i>override_&lt;LTI parameter&gt;</i> .....	11
<i>oauth_signature_method</i> .....	6	LMS Template Parameters.....	12
<i>oauth_consumer_key</i> .....	6	LTI Launch Example.....	12
<i>oauth_timestamp</i> .....	6	Required functions.....	12
<i>oauth_nonce</i> .....	6	MAC computation.....	12
<i>oauth_signature</i> .....	7	LTI launcher function.....	13
Optional Parameters.....	7		

## Overview

[Learning Tools Interoperability](#) (LTI) is a specification for launching one platform from another in a user's browser with [federated identity](#) between them. This pseudo-standard has implementations in a wide array of popular [Learning Management Systems](#) (LMSs) and tool providers. The authentication and basic federation aspects of LTI are based upon [RFC 5849 Signature](#), whereas the identity and context specific subject being federated are unique to the LTI specification. The LTI specification itself consists of many optional components that are not always implemented, and across both the required and optional components, different implementations have different levels of compliance, to say nothing about ambiguous aspects of the specification. This document covers LTI details required by MEETS SaaS, implementation notes, as well as non-standard features MEETS offers.

# Assistance

Assistance on connecting popular LTI compliant LMSs with MEETS is included with every MEETS subscription. If you need assistance and guidance connecting a compliant LMS, please contact [CirQlive Tech Support](#). If you are trying to connect a non-compliant LMS, you will need to speak to your LMS contacts and ask them to implement compliant LTI support.

If you are the developer of a product looking to implement MEETS integration with your platform, please refer to this document on how to perform basic federation to MEETS. Please also refer to the [MEETS SaaS API Usage Manual](#) in order to learn how to achieve a tighter integration. If you require a MEETS instance to develop with, please contact [CirQlive Sales](#).

# Technology

MEETS SaaS is accessed via a modern web browser and makes use of the following technology:

- [HTTPS](#) via [TLS v1.2+](#) for [encryption](#) and integrity protocol.
  - [Rijndael \(AES\)](#) and [ChaCha20](#) for the encryption algorithms.
  - [Elliptic Curve Diffie Hellman](#) as the [key exchange](#) algorithm.
- [HMAC](#) based authentication codes according to [RFC 5849 Signature](#) for authentication for federation.
- [DHTML](#) including [JavaScript](#) for application rendering and functionality.
- Software developed by the [OpenSSL Project](#) for use in the OpenSSL Toolkit for some of the encryption and security primitives.

# Security

MEETS SaaS is developed by a team of security professionals and aims to provide one of the most secure application experiences available over The Internet. Please refer to [MEETS Security](#) for more details.

# Credentials

The LTI [URL](#) and credentials to perform federation are accessed from within the administration section of your MEETS platform. Please refer to the appropriate documentation:

- [MEETS Administration for Cisco WebEx](#).
- [MEETS Administration for \(Citrix / LogMeIn\) GoTo](#).
- [MEETS Administration for Zoom](#).

In terms of platform setup and LTI information, all the aforementioned manuals are identical. Only their documentation on web conferencing integration details differ.

MEETS SaaS provides the ability to connect multiple platforms to MEETS, each with their own distinct LTI connection credentials. MEETS offers some streamlined support for various popular LMSs, however MEETS can work with any platform that supports enough of the LTI specification.

## Access Endpoint

Every MEETS platform has its own [domain](#), and is generally a [subdomain](#) of *meets.cirqlive.com*. The access endpoint in typical cases is **<your sub domain>.meets.cirqlive.com/lti.exe**.

In some special cases, an alternate URL is used. However, whatever the URL is, it will always be documented from the MEETS administration section for your specific MEETS instance, and can be obtained together with the credentials required for a specific LTI connection.

## LTI

The way LTI works, one must perform a [POST](#) operation to the LTI endpoint URL. The POST operation must minimally contain several authentication, user identity, and context information parameters in a [query string](#). This POST operation has a *Content-Type* of *application/x-www-form-urlencoded*, which is typically how [HTML forms](#) are submitted to a server. The parameters must be sent to the LTI endpoint shortly after they are generated, and the same exact parameters may not be sent more than once.

Although there is more than one way to implement the semantics just described, the best practice approach is that after the user indicates they want to launch an LTI application, the following steps are performed:

1. On the [server-side](#), compile the necessary user and context data required.
2. On the server-side, generate the authentication code for the data.
3. Present the user's browser with a client-side form containing the data and authentication code.
4. Have the user's browser immediately submit this form to the LTI endpoint URL.

The actual page in the user's browser which causes the form to be generated and submitted can be presented in an [iframe](#) to make it appear that the remote platform (MEETS) is part of the originating platform. The actual page could also be launched in a new tab by adding *target="\_blank"* to the [anchor element](#) used to generate the launch button for MEETS which references the page that causes the form to be generated and submitted.

The various steps can all occur as a single operation from the user's perspective. All but the last step occurs as part of the URL being loaded on the server. For the final step, the user's browser can auto-submit the form to the appropriate URL as follows:

```
<script type="text/javascript">
/**/
var f = document.getElementById("lti_launch_form"); //Replace lti_launch_form with the actual HTML id assigned to the form
f.style.display = "none"; //Make the form disappear, so from the user's perspective they just see a page loading while the form
submits
f.submit(); //Submit the actual form, which will load MEETS.
/*]]&gt;*/
&lt;/script&gt;</pre></div><div data-bbox="17 891 238 915" data-label="Text"><p>The form itself generally begins with:</p></div><div data-bbox="17 926 764 948" data-label="Text"><pre>&lt;form id="lti_launch_form" action="https://whatever-subdomain.meets.cirqlive.com/lti.exe" method="post"&gt;</pre></div>
```

## iframe

Having the remote platform displayed inside an *iframe* is very popular. If you decide to place MEETS inside of an *iframe*, be sure to allow it to go into full screen mode.

```
<iframe src="..." ... allowfullscreen="allowfullscreen"></iframe>
```

Without the [allowfullscreen](#) attribute, videos MEETS can display will not be able to offer full screen viewing without launching the video player in a new tab.

## Required Parameters

The parameters enumerated throughout this section must exist and contain non-empty values within the query string being POSTed, or MEETS will reject the federation attempt. Other third party platforms (tool providers) may have other requirements.

### *user\_id*

A unique user ID for the currently logged in user in the originating platform (LTI tool consumer). Each time a particular user is federated into MEETS, the same *user\_id* should be used. If one user happens to be assigned a *user\_id* of another user, that user will be able to see and access the data of the other user, as from the perspective of MEETS, they are one and the same. If on a subsequent visit a user gets a new *user\_id*, MEETS will consider them a new user, and will not give them access to material they previously had access to with the prior *user\_id*.

*user\_id* should not be unique for each *context\_id* (see below). If it is, users will need to reenter their settings for every *context\_id*, and they will never have access to all their material at one time.

It should be noted that while a user name or some user number can be used, few LMSs actually send anything recognizable for this value.

MEETS requires the value passed to be valid [ASCII](#) and 128 [octets](#) (bytes) or less. If your LTI launch implementation requires larger values please contact [CirQlive Tech Support](#).

### *lis\_person\_name\_given*

The first name to be displayed for the user being federated. The user will see the value provided as their own first name, and this name will also be what other users see when being shown information about this user. If this value changes for the same user upon subsequent accesses, MEETS will update the value.

MEETS requires the value passed to be valid [UTF-8](#) and 128 characters or less.

### *lis\_person\_name\_family*

The last name to be displayed for the user being federated. The user will see the value provided as their own last name, and this name will also be what other users see when being shown information about this user. If this value changes for the same user upon subsequent accesses, MEETS will update the value.

MEETS requires the value passed to be valid [UTF-8](#) and 128 characters or less.

### ***lis\_person\_contact\_email\_primary***

The default e-mail address to use for e-mailing this user when necessary, as well as for sharing with other integrated platforms that require to know the user's e-mail address. When MEETS is setup to use *Singular Authentication* to integrate a web conferencing service, this e-mail address must match that as recognized by the web conferencing service for this user, otherwise, the user's pre-existing web conferencing service accounts will not be federated (if applicable). If this value changes for the same user upon subsequent accesses, MEETS will update the value.

MEETS requires the value passed to be a valid e-mail address.

### ***context\_id***

A unique context ID for the current context the user is in that launched this MEETS session. Typically this is a course ID, and allows MEETS to be aware of which course the user is originating from, and will group and organize data according to this ID. However this ID can represent whatever is appropriate within the originating hierarchy, such as the corporate department or office region. It is important to specify a unique value which represents the correct granularity to connect with MEETS, which will in turn be organizing events and content within this context. When a user revisits MEETS from the same context, the *context\_id* needs to remain the same, otherwise it will appear as if material has vanished. Similarly, different contexts should not share the same *context\_id*, otherwise events and content from an unrelated context will be presented in MEETS.

*context\_id* should not be unique for each *user\_id*. If it is, users will never see events, content, and other data created by other users.

It should be noted that while some course number can be used, few LMSs actually send anything recognizable for this value.

MEETS requires the value passed to be valid [ASCII](#) and 128 [octets](#) (bytes) or less. If your LTI launch implementation requires larger values please contact [CirQlive Tech Support](#).

### ***roles***

A comma (,) separated list of roles the current user has in the current context. If this value changes for the same user upon subsequent accesses to this *context\_id*, MEETS will update the value.

The official specifications for LTI lists [several dozen possible roles](#) under the heading *LIS vocabulary for Context Role*. As LMSs and other originating platforms cannot even remotely begin to agree what all these roles are, and whether to use a descriptive handle or a [Uniform Resource Name](#) (URN), MEETS uses some heuristics to identify the roles being presented.

LTI roles which are for Learners or Students end up being recognized by MEETS as a *student* (typically a user who is expected to attend online events and complete assignments). LTI roles for Instructors and Teaching Assistants are recognized by MEETS as a *teacher* (typically a user who can schedule and host events and create context-wide content). LTI roles for Administrators, Managers, and Content Developers are recognized by MEETS as an *admin* (typically a user who can schedule events and create context-wide content). Various capabilities and requirements of the three canonical roles that MEETS supports within contexts can be configured via the MEETS administration section.

Users who are recognized as an *admin* in a context are not considered an *admin* in the MEETS administration section.

If you are generating your own LTI launcher, CirQlive suggests using the following roles as necessary: *urn:lti:role:ims/lis/Learner*, *urn:lti:role:ims/lis/Instructor*, *urn:lti:role:ims/lis/Administrator*, and combining them when required.

## Signature Details

The following parameters are all required, and form the authentication aspect of the federation attempt. More details regarding these parameters and the authentication method as a whole can be found in the [RFC 5849 Signature](#) section of the [MEETS SaaS API Authentication Manual](#). The *key* and *secret* required in order to compute the values for *oauth\_consumer\_key* and *oauth\_signature* are described in the *Credentials* section above.

### *oauth\_signature\_method*

Specifies the algorithm used to sign/MAC the parameters. Officially for LTI, the algorithm is *HMAC-SHA1*. MEETS SaaS supports *HMAC-SHA1*, *HMAC-SHA256*, and *HMAC-SHA512* for LTI authentication.

### *oauth\_consumer\_key*

Specifies the *key* (alternatively, LTI connection ID) for the federation attempt in question. MEETS allows clients to create many LTI connections each with their own ID (*oauth\_consumer\_key*). Each platform (such as multiple LMS platforms, or testing and production instances) a client wishes to integrate should use a different connection. Sharing the same connections across multiple platforms will cause *user\_id* and *context\_id* to conflict across platforms causing all kinds of problems. These conflicts are irreparable.

This value together with the LTI connection secret represent the credentials used for authentication. The values for both of these can be found inside the MEETS administration section. Each LTI connection in MEETS will have its own *key* and *secret*.

### *oauth\_timestamp*

A [UNIX timestamp](#) for the server time the signature is generated at. Specifying this using an inaccurate clock or some other kind of timestamp (with a different [epoch](#), unit quantity other than seconds, or non-UTC timezone) will either fail to work or not provide the appropriate security qualities. MEETS requires that the time specified be within a few minutes of the actual time. Leeway is given to allow for minor clock drift as well as the time it takes from signature generation till the data is actually passed over The Internet via the user's browser to MEETS. Generating the *oauth* parameters and then waiting some time before sending to the endpoint may cause it to be considered too old, and therefore rejected. Setting *oauth\_timestamp* and generating the signature should only occur immediately before they are intended to be used.

### *oauth\_nonce*

A random server-side [nonce](#) (string of data) which along with the *oauth\_timestamp* should be unique for each federation attempt. Using a duplicate *oauth\_timestamp* and *oauth\_nonce* pair with a signature will be rejected by MEETS SaaS. Please see the relevant information in the MEETS SaaS API Authentication Manual for more details.

### ***oauth\_signature***

A cryptographic signature generated server-side over all the other parameters being passed for this federation attempt, together with the LTI connection secret - which is not passed along. The signature should only be generated and the *oauth* parameters should only be sent immediately before you intend to send it to MEETS SaaS. MEETS SaaS will reject data that is identical to data it received earlier, in order to prevent [Capture-Replay attacks](#).

The cryptographic algorithm to use is described by the one selected for the *oauth\_signature\_method* parameter, with the parameters arranged according to the [RFC 5849 Signature](#) specification. Please see the relevant information in the MEETS SaaS API Authentication Manual for more details.

It is crucial to note that the LTI connection secret should never be shared with anyone, otherwise they will be able to impersonate other users and gain access to other contexts and roles within MEETS. Therefore, signatures should NEVER be generated [client side](#). If an LTI connection secret was accidentally shared, it can be regenerated from the MEETS administration section. In order to prevent malicious people from just guessing the secret, all secrets generated by MEETS are randomly selected using a [cryptographically secure pseudo-random string generator](#) from a [key space](#) of  $2^{384}$  possibilities. All secrets generated by MEETS are 64 [octets](#) (bytes) long.

## **Optional Parameters**

These parameters are not required, but they are strongly recommended, and would be foolish to not implement.

### ***lis\_person\_name\_full***

The combined first and last name to be displayed for the user being federated. Somewhat redundant.

MEETS requires the value passed to be valid [UTF-8](#).

### ***context\_title***

The name or title for the current context (*context\_id*). This is displayed as the human-readable value for data organized into this context within MEETS. If this value changes upon subsequent accesses to this *context\_id*, MEETS will update the value.

MEETS requires the value passed to be valid [UTF-8](#) and 255 characters or less.

### ***launch\_presentation\_locale***

The language to display MEETS in. Languages can be specified using an [ISO 639-1](#) two letter [language code](#). Extended language + locale codes such as *en\_us* and *en\_gb* are allowed as well. Case does not matter. If this value changes for the same user upon subsequent accesses, MEETS will update the value.

MEETS does not support every possible language. Unrecognized languages will default to English.

### ***tool\_consumer\_info\_product\_family\_code***

The name of the originating platform (tool consumer). For example *moodle*, *WordPress*, *Sakai*, *Joe's Supreme Virtual Education*. If this value changes for the same user upon subsequent accesses to this *context\_id*, MEETS will update the value.

MEETS may enable additional functionality if it recognizes the originating platform.

MEETS requires the value passed to be valid [UTF-8](#) and 255 characters or less.

### ***tool\_consumer\_info\_version***

The version of the *tool\_consumer\_info\_product\_family\_code*. For example *2.1* or *11*. If this value changes for the same user upon subsequent accesses to this *context\_id*, MEETS will update the value.

MEETS may enable additional functionality if it recognizes the originating platform.

### ***launch\_presentation\_return\_url***

A URL MEETS should navigate to when returning this user to the originating platform for this *context\_id*, when applicable. If this value changes for the same user upon subsequent accesses to this *context\_id*, MEETS will update the value.

## **Other parameters**

The LTI specification requires several parameters that MEETS does not, nor does MEETS even look at them. However MEETS in the future may require them, and in order to ensure maximum compliance with other tool providers, one designing a new LTI launcher should also send the following parameters:

- *lti\_message\_type* — Set to *basic-lti-launch-request*.
- *lti\_version* — Set to *LTI-1p0*, *LTI-1p1*, *LTI-1p1p1*, or a similar LTI specification.
- *resource\_link\_id* — A value which should be unique for every LTI launch link present in the originating platform.

It is advisable to also specify every other parameter marked as *recommended* in LTI specifications.

## **Custom Parameters**

MEETS supports a number of custom (non-standard) LTI parameters to overcome certain difficulties as well as provide some interesting integration possibilities. All LTI custom parameters are prefixed with *custom\_*. This prefix must be present in the actual raw POST data that is transmitted to MEETS, but often is not present when adding custom parameters via interfaces that appear in popular LMSs. For the sake of brevity, this prefix will not appear in this section below, but must be supplied in the actual raw POST data.

## **Endpoints**

When MEETS loads the LTI endpoint, what normally appears is one of the following:

- *Account Settings* — The first time a user joins MEETS with the role of a *teacher*, and course-wide events or appointment booking is available. On this page the user can configure their web conferencing account settings that MEETS will use for them when creating events or booking appointments.
- *Event Calendar* — When not the above, users will end up on the event calendar page if course-wide events is enabled. On this page users can see scheduled events, and those with appropriate permissions can schedule additional events.



- *Appointment Booking* — When not the above, users will end up on the appointment booking page if appointment booking is enabled. On this page those with appropriate permissions can offer appointments, and students can reserve and join appointment events.
- *Recordings* — When not the above, users will end up on the recordings page if conferencing recordings is enabled. On this page users can view and download recordings as well as access additional information about past events.
- *Files (content)* — When not the above, users will end up on the files page if file management is enabled and your organization purchased this feature. On this page those with appropriate permissions can upload files, and all users can view and download those files.
- If none of the above options are available, then users will end up on an empty page and be given the option to navigate to whatever is available.

An additional LTI parameter can specify what MEETS should display when loading its endpoint instead of the above.

#### *endpoint*

This parameter (*custom\_endpoint*) should be passed one of the following values:

- *page:account* — Display the account settings page. If the user does not have permission to access this page, they will encounter an error.
- *page:appointments* — Display the appointment booking page for this *context\_id*.
- *page:attendance* — Display the attendance page for this *context\_id*. If the user does not have permission to access this page for this *context\_id*, they will encounter an error.
- *page:calendar* — Display the event calendar page for this *context\_id*.
- *page:content* — Display the content page (files page) for this *context\_id*.
- *page:notes* — Display the user notes page for this *context\_id*. If user notes is disabled, or the user does not have permission to access this page for this *context\_id*, they will encounter an error.
- *page:recordings* — Display the recordings page for this *context\_id*. If the user does not have permission to access this page for this *context\_id*, they will encounter an error.
- *page:synq* — Display the Webex Teams Synq page for this *context\_id*. If your organization does not have to access this page, the user will encounter an error.
- *event:<event ID>* — Example: *event:54321* — Display the event information/launch page for a particular event. The event ID is the numeric MEETS ID for an event. This ID is displayed next to the name of events in MEETS, and can also be obtained via the MEETS SaaS API. Please refer to the [MEETS SaaS API Usage Manual](#) for more information. If the specified event does not exist within this *context\_id*, the user will encounter an error.
- *content:<content ID>* — Example: *content:2468* — Download, launch, or play the specified content as appropriate for it. The content ID is the numeric MEETS ID for a piece of content. This ID is displayed next to the file name of content in MEETS' content page (files page) including the one in the administration section, and can also be obtained via the MEETS SaaS API. Please refer to the [MEETS SaaS API Usage Manual](#) for more information. If the specified content does not exist within this *context\_id*, the user will encounter an error.
  - If your platform typically displays MEETS in an *iframe*, and does not use the *allowfullscreen* attribute described above in the *iframe* section, then video content will not be able to be played back in full screen mode. If you are linking to content representing a video in this situation, it is advisable to have your platform launch the MEETS content in a new tab if possible.

To achieve tighter integrations between the originating platform and MEETS, each page in an LMS course could link directly to multiple MEETS pages, such as the calendar, recordings, and content.

If you're using the MEETS APIs to process events and display them within your own interface, the *event:<event ID>* endpoint can be used to directly link to them within MEETS.

If you're using the MEETS APIs to process content and display them within your own interface, the *content:<content ID>* endpoint can be used to directly link to them within MEETS.

## Themes

When MEETS loads, the theme in use is the default one specified by the administrator or overwritten by the user. However an additional LTI parameter can specify which MEETS theme should be used.

### *theme*

This parameter (*custom\_theme*) should be passed one of the following values:

- *contour* — a theme with a three dimensional look and considerable contrast.
- *smooth* — a theme with a plain and clean look.

Note: You should not force a theme on users overriding their settings using this parameter. If you want to use this parameter in a user-friendly way, do so in cases where you offer to launch MEETS with different themes. For example, next to your launch button you can display a theme drop down selector where users can choose "default" or one of the other themes.

## Auxiliary data

MEETS supports collecting additional data regarding users and contexts which are used as part of information collation. This information can then be retrieved alongside user and context information when pulling information via the MEETS APIs. Please refer to the [MEETS SaaS API Usage Manual](#) for more information regarding data retrieval.

### *auxiliary\_user\_<user field name>*

For *<user field name>* you may specify any field name you'd like for user data (it may not contain an equal symbol). The name used will appear alongside user information in information reports.

You may specify as many of these as you'd like, with only one restriction. The combined user data may each not exceed 4096 octets (bytes) when percent encoded and merged for a query parameter as defined in [RFC 3986](#).

Some LMSs, such as [Blackboard Learn allows one to use template placeholders](#) whose content change for each user, thereby allowing per-user information to be added to MEETS. An example of usage for the user's Batch ID would be to add the following line to the customer parameters in Blackboard Learn:

*auxiliary\_user\_batch\_id=@X@user.batch\_uid@X@*. This will store along side user data a field named *batch\_id* using the value Learn stores in *user.batch\_uid* and can be retrieved via the MEETS SaaS API. In practice, this will transmit something like: *custom\_auxiliary\_user\_batch\_id%3D5423-3242*.

***auxiliary\_context\_<context field name>***

For ***<context field name>*** you may specify any field name you'd like for context data (it may not contain an equal symbol). The name used will appear alongside context information in information reports.

You may specify as many of these as you'd like, with only one restriction. The combined context data may each not exceed 4096 octets (bytes) when percent encoded and merged for a query parameter as defined in [RFC 3986](#).

Some LMSs, such as [Blackboard Learn allows one to use template placeholders](#) whose content change for each context (course), thereby allowing per-context information to be added to MEETS. An example of usage for the Learn Course ID (which differs from the Learn generated *context\_id*) would be to add the following line to the customer parameters in Blackboard Learn: *auxiliary\_context\_course\_id=@X@course.id@X@*. This will store along side context data a field named *course\_id* using the value Learn stores in *course.id* and can be retrieved via the MEETS SaaS API. In practice, this will transmit something like: *custom\_auxiliary\_context\_course\_id%3D24\_2*.

## Overrides

MEETS allows any of the main LTI parameters to be overridden. These overrides are provided for several purposes:

- To possibly fix broken values that some LMSs normally send, and cannot be otherwise altered to do the right thing.
- To allow for linking across different contexts, perhaps to pull in information from a course taught in a prior semester.
- To generate completely new contexts manually, perhaps to offer multiple sections within a particular segment within your organization hierarchy.
- To change display text so that something different appears in MEETS than appears in the originating platform.

Since parameters can be overridden, this is also a potential avenue for abuse. Ensure that only authorized users are able to set/modify custom parameters in LTI launchers. If your LMS supports white-listing custom parameters that users can individually alter, do not include *override\_user\_id* and *override\_context\_id*. It should be noted that in most popular LMSs, only administrators or only those who have access to the LTI connection secret are able to set/modify custom parameters.

***override\_<LTI parameter>***

***<LTI parameter>*** can be any parameter which appears in the non-signature-details required parameters section above, as well as any that appear in the optional parameters section.

Blackboard Learn has been known to switch the values for *user\_id* and *context\_id* for the same user and context between different versions of their LMS. This means that upon upgrading Blackboard Learn, there is the possibility that users will lose access to their data, or worse, conflicts can ensue. Therefore specifying the following two overrides is crucial:

- *override\_user\_id=@X@user.pk\_string@X@*
- *override\_context\_id=@X@course.pk\_string@X@*

This ensures that the *user\_id* and *context\_id* are stable in that they also refer to the same piece of user and course data between versions, and that a variable which maintains consistency across versions is chosen.

If your LMS supports using multiple e-mail addresses, and only one of them corresponds to the user's web conferencing service, setting `override_lis_person_contact_email_primary` to use the correct one is useful, if your LMS offers features to allow setting a custom parameter to a specific piece of data for each user.

Please note that `@X@`, `user.pk_string`, and `course.pk_string` are all [Learn specific template syntax and variables](#). Not all LMSs will support templates, nor are they likely to use the same syntax or have variables with the same names.

## LMS Template Parameters

Some LMSs support template parameters which will pass along the current value of a particular setting to an LTI variable. See above for examples of how to use these with auxiliary data and overrides.

- [Blackboard Learn](#)
- [Instructure Canvas](#)

## LTI Launch Example

The [MEETS SaaS API Authentication Manual](#), in the section [RFC 5849 Signature](#), contains information and examples in various languages such as Java, C#, and others on how to generate an [RFC 5849 Signature](#). Adding to that the appropriate LTI variables and sending it all via POST is all that is needed to launch MEETS. What follows is a simple and straight forward example on how to put all that together in PHP. The logic here should be simple enough to port to other languages.

### Required functions

The functions `array_sort`, `array_to_string`, and `hmac_sha512` are required from the PHP *Method 2* example in the [RFC 5849 Signature](#) section of the MEETS SaaS API Authentication Manual. Additionally, if your PHP version does not have it built in, the `random_bytes` function is required as well.

### MAC computation

The MEETS SaaS API Authentication Manual, in the [RFC 5849 Signature](#) section, contains a somewhat bulky function to compute the message authentication code (MAC). Since the way it's used in LTI already will have all the POST parameters in their own array, and variables are set directly via POST and not an authorization header, the following simplified version will be used instead:

```
<?php
function mac_compute($method, $uri, array $params, $key_client = '', $key_token = '')
{
    $params = array_to_string(array_sort($params));
    $str = rawurlencode($method).'&'.rawurlencode($uri).'&'.rawurlencode($params);
    $key = rawurlencode($key_client).'&'.rawurlencode($key_token);
    return(base64_encode(hmac_sha512($key, $str)));
}
//$key_client here refers to the LTI connection secret
//$key_token is not actually used by the LTI specification
```

If the LTI URL included a query string (which MEETS SaaS currently does not use), a more complicated compute function like that found in the MEETS SaaS API Authentication Manual would be required.

### LTI launcher function

The launcher function requires details about the user, the course that MEETS SaaS is being launched from, the user's role in that course, and details regarding the LTI connection. Details regarding appropriate values for the user, course, and role related parameters can be found above in the *Required Parameters* section. Details regarding the LTI credentials is described in the *Signature Details* section above, while what to use for the LTI connection parameters themselves is described in the *Credentials* section near the beginning of the document.

```
<?php
function lti_launcher($user_id, $user_name_first, $user_name_last, $user_email,
                    $course_id, $course_name,
                    array $user_role_in_course, //Possible roles are student, teacher, and admin
                    $lti_url, $lti_key, $lti_secret, $lti_language = 'en_US')
{
    function lti_role_array_to_string(array $user_roles)
    {
        $roles = array
        (
            'student' => 'urn:lti:role:ims/lis/Learner',
            'teacher' => 'urn:lti:role:ims/lis/Instructor',
            'admin' => 'urn:lti:role:ims/lis/Administrator'
        );
        $user_roles_real = array();
        foreach ($user_roles as $role)
        {
            if (isset($roles[$role])) { $user_roles_real[] = $roles[$role]; }
        }
        return(implode(',', $user_roles_real));
    }

    //Calculate LTI variables
    $variables = array();
    $variables['user_id'] = $user_id;
    $variables['lis_person_name_given'] = $user_name_first;
    $variables['lis_person_name_family'] = $user_name_last;
    $variables['lis_person_name_full'] = $variables['lis_person_name_given'] . ' ' . $variables['lis_person_name_family'];
    $variables['lis_person_contact_email_primary'] = $user_email;
    $variables['context_id'] = $course_id;
    $variables['context_title'] = $course_name;
    $variables['roles'] = lti_role_array_to_string($user_role_in_course);
    $variables['launch_presentation_locale'] = $lti_language;
    $variables['tool_consumer_info_product_family_code'] = 'CirQlive LTI Launcher'; //You could rename this to the name of the product
    used (eg: WordPress)

    $time = gettimeofday();
```

```

$variables['oauth_signature_method'] = 'HMAC-SHA512';
$variables['oauth_consumer_key'] = $lti_key;
$variables['oauth_timestamp'] = $time['sec'];
$variables['oauth_nonce'] = base64_encode(pack('H*', sprintf('%05x', $time['usec'])) . random_bytes(21));
$variables['oauth_signature'] = mac_compute('POST', $lti_url, $variables, $lti_secret);

//Construct autolaunching form
$form = '<form id="lti_launch_form" action="' . htmlspecialchars($lti_url) . '" method="post">
<div>
';
foreach ($variables as $key => $value)
{
    $form .= '    <input type="hidden" name="' . htmlspecialchars($key) . '" value="' . htmlspecialchars($value) . '" />
';
}
$form .= '    <input id="lti_form_launch_button" type="submit" value="Launch" />
</div>
</form>
<script type="text/javascript">*<![CDATA[*/*var f=document.getElementById("lti_launch_form");f.style.display="none";f.submit();/*
*]]>*/</script>
';
return($form);
}

```

After performing authentication and validating the user's permission, the above function can be called. Use along the following lines:

```

<?php
echo lti_launcher
(
    $user->id(),
    $user->name_first(),
    $user->name_last(),
    $user->email_address(),
    $user->context_current()->id(),
    $user->context_current()->name(),
    $user->context_current()->is_teacher() ? array('teacher') : array('student'),
    $lti_url,
    $lti_key,
    $lti_secret
), "\n";

```

This will inject something like the following into the HTML being output:

```

<form id="lti_launch_form" action="https://example.meets.cirqlive.com/lti.exe" method="post">
<div>
    <input type="hidden" name="user_id" value="u123" />
    <input type="hidden" name="lis_person_name_given" value="Jane" />
    <input type="hidden" name="lis_person_name_family" value="Dough" />

```

```
<input type="hidden" name="lis_person_name_full" value="Jane Dough" />
<input type="hidden" name="lis_person_contact_email_primary" value="jdough@example.com" />
<input type="hidden" name="context_id" value="c321" />
<input type="hidden" name="context_title" value="Baking 101" />
<input type="hidden" name="roles" value="urn:lti:role:ims/lis/Instructor" />
<input type="hidden" name="launch_presentation_locale" value="en_US" />
<input type="hidden" name="tool_consumer_info_product_family_code" value="CirQLive LTI Launcher" />
<input type="hidden" name="oauth_signature_method" value="HMAC-SHA512" />
<input type="hidden" name="oauth_consumer_key" value="25" />
<input type="hidden" name="oauth_timestamp" value="1402759365" />
<input type="hidden" name="oauth_nonce" value="01zgy9baE5w5wTgE5cnFtZCPHUhoFT2P" />
<input type="hidden" name="oauth_signature"
value="SCqcp4iEkB3W0vOp+bfszMcauY5JcVEgPRA10W2/QEcenImLdavvA+2DyM6+SSi53IYjHeU+VcFBhvZkicMk4g==" />
  <input id="lti_form_launch_button" type="submit" value="Launch" />
</div>
</form>
<script type="text/javascript">/*! [CDATA[* /var f=document.getElementById("lti_launch_form");f.style.display="none";f.submit();/
*]]>*/</script>
```

This example aims to be minimal for MEETS. It is advisable to add support for parameters listed above in the optional parameters section.