# Ticket based authentication : Integrator Guide

# Ticket Direct Access (to Moodle) plugin

## Introduction

This guide is provided for :

- Integrators and administrators using the auth_ticket as part of other feature
- Developers that want to send self-authenticated notifications

## Administrators and integrators

This plugin provides the way to some other plugins to send pre-authenticated tickets to users. This addresses a set of security considerations :

### Ticket encryption methods

This component provides either DES or RSA encryption methods to encode/decode tickets. Administrators must be aware of :

- DES (AES) uses internal Mysql/Maria DB AES_ENCRYP/AES_DECRYPT function that is not available (nor similar) in PostgreSQL.
- RSA is a more compatible method based on openssl underlying layer. Moodle needs have MNET network enabled and a valid local keypair available. The key length should be of 2048 bytes to ensure the ticket payload has sufficiant length to hold the targer urls. Openssl functions claim about length restrictions of the input payload when using 1024 rsa key length. (See http://php.net/manual/en/function.openssl-private-encrypt.php for more information).

### Ticket validity time

Tickets have a validity time that obsoletise the ticket after some delay. If tickets may be used in a non HTTPS secured environment, the timeguard of the tickets should be kept as little as possible. There is a potential risk of stealing a ticket somewhere on the network by misintentionned people that might use it to enter moodle into a user account.

Usually it will be recomended tickets be used in a secured HTTPS context, thus the ticket data sequence is protected all amonth the path from sender to server.

## Developers

### Ticket pre-authenticated user

The ticket holds the user identity it has been created for. Thus it will be quite a bad idea to send admin pre-authenticated tickets to non desired people. The developpers using the auth_ticket API should care about this risk.

#### Long, short and persistant tickets

As increasing the ticket validity time enlarges the risky time window, developpers should care sending likely short tickets rather than long tickets. Although persistant tickets has been given as a possibility, this validity time should not be used, or assigned to trashbin fake users with very restricted capabilities.

## Ticket API

The ticket API is contained in the lib.php. The actual API offers four functions. The former pair are high level notification sending functions using tickets. The latter are low encode/decode function.

#### function ticket_notify($recipient, $sender, $title, $notification, $notificationhtml, $url, $purpose = '', $term = 'short')

Simple sending to user with return ticket. The return ticket allows auser receiving amail to enter immediately the platform being connected automatically during a hold time.Tthe ticket is catched by a custom auth module that decodes generated ticket and let user through.

Only recipients that have a valid Moodle account can use an access tickets. The ticket is only valid on the given return URL and cannot be used for going to another location, unless user's profile other mention.

```
 @param object $recipient
 @param object $sender
 @param string $title mail subject
 @param string $notification raw content of the mail
 @param string $notification_html html content of the mail
 @param string $url return url of the ticket
 @param string $purpose some textual comment on what the ticket was for
 @param bool $term the ticket validity duration, may be 'short', 'long' or
'persistant'.
```

#### function ticket_notifyrole($roleid, $context, $sender, $title, $notification, $notificationhtml, $url, $purpose = '', $checksendall = false, $term = 'short')

Send a notification message to all users having the role in the given context.

```
 @param int $roleid id of the role to search users on
 @param object $context context in which find users with the role
 @param object $sender user identity of the sender
 @param string $title mail subject
 @param string $notification raw content of the mail
 @param string $notification_html html content of the mail
 @param string $url return url of the ticket
 @param string $purpose some textual comment on what the ticket was for
 @param bool $checksendall if true, the function returns true if all the
recipients were sucessfull
 @param bool $term the ticket validity duration, may be 'short', 'long' or
'persistant'.
 @return true if at least one email could be sent or all are sent depending
on $checksendall.
```

**function ticket_generate($user, $reason, $url, $method = 'des', $term = 'short')**

Generates a direct access ticket for this user.

```
 @param int $userid the ID of the user to whom the ticket must be made for
 @param string $reason the reason of the ticket
 @param string $url the access URL the user will be redirected to after
validating his return ticket.
 @return the encoded ticket
```

**function ticket_decode($encrypted, $method = 'des')**

Decodes a direct access ticket for this user.

```
 @param string $encrypted the received ticket
 @param string $method the decrypt method. Supports 'des' using DB internal
function or 'rsa' using openssl layer.
 @return an object containing ticket information.
```

---

## Credits

- Valéry Frémaux (valery@activeprolearn.com)) Developments, Documentation
- Florence Labord (florence@activeprolearn.com) Documentation

Back to componant index - Plugins index - Back to catalogue