× Internals: SympaSession.pm

- This document was obsoleted.

This module provides objects to create, load, store and remove sessions.

How to use sessions object ?

Session are initialized at the beginning of the main wwsympa loop. They are updated at the end of this loop. So basic extension of sessions usage in wwsympa just needs to set new session vars anywhere in wwsympa.fcgi actions. Imagine you want to introduce a menu in Sympa to change the user interface skin. You will probably introduce a new action associated with a form to choose skins; this form will introduce in wwsympa an incoming parameter named $in{'skin_name'}. This parameter will become a property of the current session just by the following instruction :

```
$session->{'skin_name'} = $in{'skin_name'};
```

That's all !

Session and authentication

At first access in wwsympa.fcgi a new session is created. The session will continue until a period of unactivity longer then $Conf{'session_table_ttl'}. Note that sessions don't start just when users logs in but a session is created even for anonymous users and sessions don't stop when users logs out. So an important property of a session is $session→{'email'}. This var is initialized with 'nobody' if user is not identified.

Because of this possibility of anonymous sessions we can carry some sessions properties for anonymous users. As an example of features that are possible with anonymous sessions : listmaster can change the log level for his sessions. Ones this is done he can log out and test some part of the application as an anonymous user still with a log_level associated with his current session.

new()

Creates an object Session.

Parameters

- robot : sessions are related to a specific robot
- cookie : the sympa_session cookie value which is a random value without any semantic

If cookie is undef or not in *session_table*, new() will create a new session by allocation of a random. If cookie is defined and found in the sympa_session table new check if the session is valid (mainly : the

http client host (remote_addr) must be the same as the host that initialyse the session). Fix Me! : may be usefull to ignore old sessions. If the session is valid it is loaded : load() .

load()

- load() read session attributes from session_table using cookie value as the primary key. All attributes are returned in a hash. Load() use tools::string_2_hash in order to convert the varchar *data_session* column in a set of vars <var>=“<value>”;<var>=“<value>”;

store()

- store() store session attributes from curernt objet into session_table. Store update the *date_session* attributes which is used by &SympaSession::purge_session() to remove old sessions.

store() convert a set of vars into a *data_session* string which can be stored in the database using tools::hash_2_string

store() renew the session_id. This make the session hijaking much more diffcult. [1)]

purge_old_sessions()

Parameters

- robot

This proc is not object oriented. It is deseign to remove from session_table old session. It is controled by the task_manager depending on sympa.conf parameter *session_expiration_period*

list_sessions()

Use to list current session (feature from sympa admin page/listmaster only)

Parameters

delay : number of second to select the age of listed sessions robot : connected_only : off|on in order to limit or not teh listed sessions to connected users

return an array of hash that describe sessions

1)
http://en.wikipedia.org/wiki/Session_hijacking#Prevention

From:
https://www.sympa.org/ - **Sympa mailing list server**

Permanent link:
https://www.sympa.org/internals/internals-session

Last update: **2018/03/29 03:47**