TikiWrapper_gmuslera

This is a mix of thoughs on several topics, but I think it all could be related to doing a wrapper to unifies all tiki funcionality in simpler URLs.

Permission names

Actually tiki permission names are a bit heterogeneous, as can be seen in PermissionAdmin, some are in the form the prefix tiki_p plus a resource/object name, plus an action, in others the action is first and then the object, and as the permission system seem to be started with wiki, some of the permissions related to tiki are just the prefix and the action. Also the action naming is not the equal for all... for accessing resource permissions sometimes is read, sometimes is play, or vote, or things like that.

What I propose here is to have a common naming in permissions, thats it, a prefix (it could be tiki_p like now), then the resource type (wiki, forum, blog, even system for systemwide activities like administration), then the action. Also I would try to unify the way to name some kind of actions, like using access instead of anything that mean see the content of that resource, be a game, a poll, etc. Also "access" is very generic, and could be used when evaluating **other** permissions, i.e. before checking if there an explicit permission for editing or other kind of interaction, I could check if I can accessin a very generic way.

Upgrading from a system to another of naming is not so bad, just doing a query to do a replacement for the new name for the records that had the old name for the names that will be changed, and could be embedded in the tiki_N.MtoN.N+1.sql scripts.

Why all the trouble? If permission names have a regular naming, then the name of the permission can be built dinamically, and the permission system not need to know previously about that kind of permissions, just check that for the passed kind of object and action the permission exist.

Program names

Like permissions, the program names not follow a convention, or at least, not all of them. The URL I'm looking at now is tiki-editpage.php?page=TikiWrapper_gmuslera. Following the same idea, they could be called in the same way, a prefix, a object/resource type, and an action.

What are the benefits? Again, the program name could be built instead of knowing it previously, could be a match between permission and programs, maybe not alaways will be a 1-to-1 relationship for cases, but that have some potential.

Wrapper

Now that I was thinking in permissions, and my old love WYSIWYCA, focused a bit in the problem to do that in the "custom menu", after all, the user put there URLs, that point or not to Tiki content, and anyway, it depend on the called module to check if the user can access, and in sites with layered or complex security could be a lot of links that lead to the no access page.

How that could be managed? The first idea I have is to link the internal content in a common, unique way, not with URLs that could or not point to the same site, so, why not have a wrapper that hides the internal URLs and just points to the right content and how to access it?

In fact, a bit in this site that is done. When you access http://tikiwiki.org/PageName or /artNN or things like that, the URL (thru rewriting of URLs) points directly to the content, by the default action of accesing to it. If i.e. tiki-index.php is the wrapper to access all content and actions on the site, internal links could be checked for permissions even before showing them in "custom" menus, and call the appropiate actions to that content.

As there is no a common name system for all kind of resources, the resource name should be added in the URL and the same for the action if its not just access. Could be used the PATHINFO variable to make nice URLs, like http://tikiwikiorg/tiki-index.php/wiki/TikiWrapper_gmuslera or even with a minimum of rewrite rules, something like http://tikiwiki.org/wiki/TikiWrapper_gmuslera. If names gets hierarchical, adding the

category or the owner's name, it could even be http://tikiwiki.org/wiki/Documentation/CssStyles or http://tikiwiki.org/Usergmuslera/TikiWrapper (the pages could be renamed if the path already say the page kind) or http://tikiwiki.org/articles/26?action=edit&otheroptions.

With a bit of renaming of programs even now some of this can be done. http://tikiwiki.org/wiki/PageName?action=edit could be mapped to http://tikiwiki.org/tiki-edit-wiki.php?page=PageName, but having a wrapper could have a lot of benefical effects

- The individual modules could not be directly executables, that only works called from the wrapper. Could be possible to have even the entire Tiki tree out of the web tree, leaving only the index.php(and maybe the resources that should be accesed from other programs, like jgraphpad and illustrations) that could have good effects on site security
- Less requeriments for modules, intead that all and each one call tiki-setup and things like that, just call it from the wrapper.
- The permission checking, or at least, a first level of it, could be put in the wrapper itself, the what is needed to check is in the passed object type,category,action and the current user.
- What about translations? Maybe wiki is the same in every language, but other words, like forums, have other names in other languages. Could be a translation for action or object kind names, but if all fails, always could be done a translation at rewrite rules level by the admin of the site that want translated URLs.
- Implementation details are mostly hidden in the URLs. You don't know how the data is accessed, what program, module, library etc do the job. That could enable future reestructurations of programs without affecting data accesibility from internal or external links.
- A Wrapper also could implement names for numeric named resources, as in TikiNames, to have something like http://tikiwiki.org/forum/Developers/

Even without the PATHINFO trick, having a common way to call internal resources (like i.e. http://mysite/index.php?data=/wiki/CategoryName/PageName) could put a little closer the WYSIWYCA ideal, a way to link internal resources in a predectible way could check accesibility for that resources.

The standard naming of objects, resources, programs and even tables in the database could make easier to understand the system, and more capable to grow, could give a lot of initial work to do reach the final naming scheme, the conversion process and doing the migration scripts (that more than just renaming tables or permissions in there, should also try to fix the internal links in the content of current sites) but I think is a step in the right direction, enabling an easier path for making deeper changes in the system.