0.1 Security Framework in Sakai

The Sakai portal framework has implemented a sophisticated internal security mechanism with role-based access control at a fairly fine grained function based level. We here summarise how this is implemented and the procedures adopted in applying the rules. Sakai not only provides a portal client interface rendered as HTML, but also has kernel components which expose a Web services interface or WebDav. It also has a WSRP producer integrated in the kernel.

Users: Each user of the portal framework has an account. Information stored includes [userid — first name — last name — e-mail — passwd — type]. We often use e-mail for userid, but it could also be the FedId. Validation of the userid/ passwd pair is the primary authentication mechanism. Clearly any other authenticatio can be used provided the user can then be mapped onto a Sakai account. [The uses of the type field are yet to be understood, but may include any string such as "friend", "guest", "registered".] The account is identified in the DB with an InternalId, either the userid or by a unique string, depending on how it was created.

Types: A User can have a Type which is defined when the user is created. This has system-wide scope. The user's account is pre-populated using the template `!user.template.type`. Examples include "guest", "registered".

Sites: A site, or "worksite" is a specific DB area for related content. It can be "public", which means it can be joined by any quthenticated user, or it is "private" which means it is moderated by one or more users with "maintain" or equivalent role for addint participants to the site. Participants can be added with any of the roles which have been defined for that site; "access" and "maintain" are the default ones. A site is identified in the DB with a unique SiteId string. A site can also have a type, e.g. "project" or "course" are defaults.

Pages: A page within a site has an associated left hand menu button and is a view onto a set of tools and corresponding data. A page has a PageId which is a unique string.

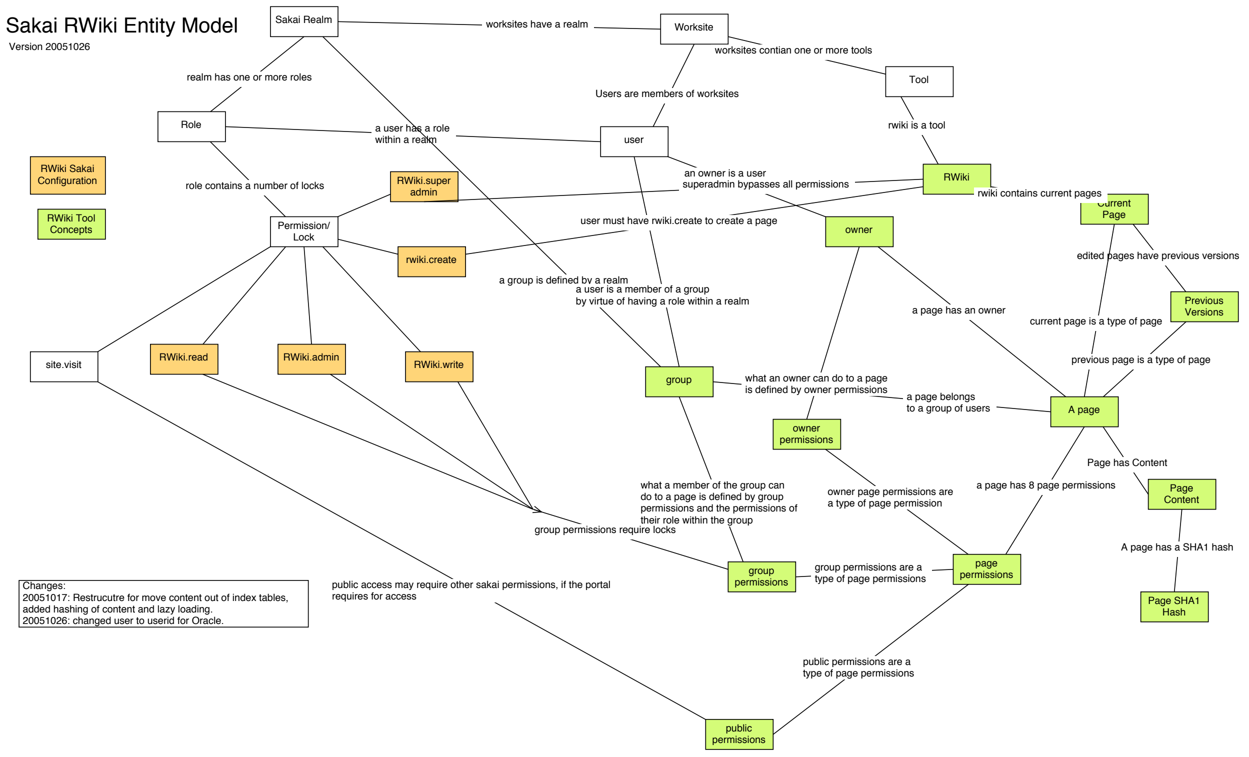
Tools: Each page on a site contains one or more tools arrange in one- or two-column layout A specific tool has a ToolId which is a unique string. For instance "sakai.resources" is a resource management tool which can be configured for each site. Tools can be developed to apply permissions based on users' roles.

Role: A role is identified by a RoleId which is a name. Users' base roles are ".anon" or ".auth" depending if the have not or have logged into Sakai. Other roles can be defined (see below) and have scope within a site.

Realms: Realms are used to associate roles with users for a particular activity. A site realm is identifies as `/site/SideId`. The definition of the role applies within a Realm. A Realm may contain many different roles. "access" and "maintain" are the default ones. Other types of realm are `/content/user`, `/content/group`. In addition to these specific realms, there are defaults such as `!group.template`, `!site.template`, `!user.template`, `!site.user`, `!pubview`, etc.

Function: Functions (or permissions) within a set for each role in a realm can be switched on or off. For instance a user with role ".anon", i.e. the public, will probably only have function "content.read" allowed.

Group:


Sakai RWiki Entity Model
Version 20051026
Sakai Realm
worksites have a realm
Worksite
worksites contian one or more tools
Tool
realm has one or more roles
Users are members of worksites
Role
a user has a role within a realm
user
rwiki is a tool
RWiki Sakai Configuration
RWiki Tool Concepts
role contains a number of locks
RWiki.super admin
an owner is a user
superadmin bypasses all permissions
RWiki
rwiki contains current pages
Current Page
Permission/ Lock
user must have rwiki.create to create a page
owner
rwiki.create
edited pages have previous versions
a group is defined by a realm
a user is a member of a group by virtue of having a role within a realm
Previous Versions
a page has an owner
current page is a type of page
site.visit
RWiki.read
RWiki.admin
RWiki.write
previous page is a type of page
group
what an owner can do to a page is defined by owner permissions
a page belongs to a group of users
A page
owner permissions
Page has Content
what a member of the group can do to a page is defined by group permissions and the permissions of their role within the group
owner page permissions are a type of page permission
a page has 8 page permissions
Page Content
group permissions require locks
A page has a SHA1 hash
group permissions
group permissions are a type of page permissions
page permissions
Page SHA1 Hash
public access may require other sakai permissions, if the portal requires for access
public permissions are a type of page permissions
public permissions

Changes:
20051017: Restrucutre for move content out of index tables, added hashing of content and lazy loading.
20051026: changed user to userid for Oracle.

Figure 1: Rwiki Entity Model for Sakai

Alias: In the system, an alias can be used to provide a user-friendly name for an Id. This is mainly for presentation purposes, as aliases can be ambiguous and referencing content by alias will not work. For instance `r.j.allan@dl.ac.uk` which is my UserId can be aliased to "Rob". Sites and other objects can be given an alias.

:

To illustrate some of the complexity of the system we use the entity model diagram developed for the Rwiki tool at Cambridge, see Figure 1. Similar models could be developed for each tool and used to ensure consistency and appropriate behaviour within the framework. Note also that the upper part of this figure illustrates the relationships between realm, worksite, tool, user and role as already discussed. The concepts of group, owner, permission, page, content, lock are also shown.

Table 1 shows a subset of the functions associated with a few Sakai worksite roles. A list (incomplete) of functions is given on Sakaipedia `http://bugs.sakaiproject.org/confluence/display/ENC/Permissions+list`.

In an implementation for a teaching extablishement the roles could be extended to be: Affiliate, Assistant, Candidate, Instructor, Member, Observer, Student in addition to access and maintain (example from Chuck Severance).

Site templates can be defined; the defaults are "course" and "project". Each template, such as `!site.template` has an associated real containing a number of pre-defined roles, e.g. maintain

Table 1: Example Functionality for Tools based on Sakai Worksite Roles.

Function Role	Maintain	Member *	Access
calendar.all.groups	yes		
calendar.delete.any	yes		
calendar.delete.own	yes	yes	
calendar.import			
calendar.new	yes	yes	
calendar.read	yes	yes	yes
calendar.revise.any	yes		
calendar.revise.own	yes	yes	
content.all.groups	yes		
content.delete.any	yes		
content.delete.own	yes	yes	
content.hidden			
content.new	yes	yes	
content.read	yes	yes	yes
content.revise.any	yes		
content.revise.own	yes	yes	
disc.delete.any	yes		
disc.delete.own	yes		
disc.new	yes	yes	yes
disc.new.topic	yes		
disc.read	yes	yes	yes
disc.revise.any	yes		
disc.revise.own	yes	yes	yes
rwiki.admin	yes		
rwiki.create	yes	yes	yes
rwiki.read	yes	yes	yes
rwiki.superadmin			
rwiki.update	yes	yes	yes

* “Member” is not a default role, but has been added to illustrate how it can be used.

and access. `!site.template.course` has “Instructor”, “Student” and “Teaching Assistant” pre defined. `!site.template.portfolio` has “CIG Coordinator”, “CIG Participant”, “Evaluator” and “Reviewer” pre defined. When a site is first created an appropriate template is chosen for it and these roles with all their permissions are inherited for the tools it will contain. Very similar templates apply to groups.

User templates can also be defined according to type, as shown above. With each template comes a realm and associated roles. There are two which are slightly difference to site roles, namely “.auth” and “.anon”. The “.auth” role, for an authenticated user, contains the possible “side.add” function which controls whether the user can create new worksites. This is included by default only in the `!user.template.maintain` realm.

How are the rules applied? Hierarchy?

[more]

Publically viewable content and the Gateway site

Currently, only a small number of tools have the option to make the content they control publically viewable. These include Site Info, Announcement, Resources, Syllabus. Links to viewable content are automatically included in the output of the Search tool which can be configured into the public-facing Gateway site. Other tools can be added to this site and pages are configurable as normal. For instance an iFrame can be used to display a Web page to users not yet logged in using the Web Content tool. [We are currently investigating how to extend this for other tools.]

Defaults, setting up a new site as admin or as a power user

The default roles for users added to the system and included as members in new sites are “admin” and “maintain”. All Sakai tools understand these roles and should behave appropriately. A user cannot have a default role, but one can be set for all users joining a site with the Membership tool or for multiple participant additions with the Site Info tool. Once a participant is added to a site, their role can be changed by the site owner or admin to any of the roles defined for that site, e.g. “member” as above.

Importing Content from other sites

It is possible to select another site from which to import content, e.g. when configuring the Resources tool. The use of this facility, and its behaviour under the security rules desribed here, requires further investigation.

Sakai API for using functions and roles

API needs a better description for TPP developers.

Authorisation in Sakai is controlled by “providers”. There are User Provider, Role (Realm) Provider and Site Provider which access data stored in the underlying DB. The Provider architecture is designed to take in a wide range of enterprise information sources.

The User (UserDirectory) Provider can fully populate new user objects and responds to API queries about UserId or passwd, e.g. at log-in time. JLDAP, OpenLDAP, Kerberos, and IMS Enterpries can be plugged into this provider. Future work is to support the existence of proxy credentials inside Sakai.

The Site Provider is invoked when a new site is created. The type is chosen and the new site populated with information, such as start date, end date, title. APIs are available to check this information.

The Realm Provider is also checked at log-in time. It defines what sites and roles within each site the user will have. This information is held in Sakai internal tables. The Realm Provider can take in roles from external enterprise sources, but will apply rules to ensure there are no security breaches possible.