P. P. Keyno

Unifying Types of Access Permissions to Information Resources of Cloud Web-services

Key words: Web, access permissions, unification

The article describes the technique of generalization of access permissions to different parts of Web-pages The permissions of each part divides into create, read, update and delete ones for each user or group. It is based on the principles of organizing permissions in UNIX-like operating systems and CRUD.

П. П. Кейно

УНИФИКАЦИЯ ТИПОВ ПРАВ ДОСТУПА К ИНФОРМАЦИОННЫМ РЕСУРСАМ ОБЛАЧНЫХ WEB-СЕРВИСОВ

За свою короткую историю Глобальная сеть обросла огромным количеством крупных интернет-сервисов различной степени сложности. Появились многочисленные интерактивные порталы с самой разнообразной архитектурой и наполнением. В таком информационном многообразии требуется проработка эффективной структурированной методики разграничения доступа к данным. Существует большое количество фреймворков, а также систем управления содержимым, в которых так или иначе реализованы механизмы такого разграничения [1]. Недостаток большинства предлагаемых решений заключается в отсутствии универсальности. Например, права доступа в большинстве систем устанавливаются на строго специфичные сущности, такие как статья или категория форума. При этом алгоритм работы, разграничивающий права доступа, может значительно отличаться в зависимости от сущности, не говоря уже об отличиях от системы к системе.

В представленном подходе предлагается унифицировать права доступа по элементам, типам, пользователям и группам. Унификация выполнена как подзадача в рамках разработки проекта BlockSet — специализированного инструментария, включающего собственный язык декларативного программирования BML на основе XML [2]. Методология BlockSet подразумевает разделение логики работы динамической web-страницы (локации) на наборы и блоки. Роль наборов выполняют специализированные области на странице, генерируемые динамически. Это может быть меню, лента новостей, список статей и многое другое. Если проводить параллели, то набор — это эквивалент таблицы в базе данных, имеющий индивидуальное представление (шаблон) и модификаторы — различные атрибуты, задающие поведение набора. Блоки — составляющие наборов также имеют собственные атрибуты-модификаторы. По аналогии с набором и таблицей, блок представляет собой поле в БД. В примере с меню блоком может выступать свойство пункта меню: название, ссылка, дата посещения и т. д. Конфигурация блоков и наборов задается на языке декларативного программирования BML (англ. BlockSet Modelling Language).

Исходя из описанного выше подхода можно говорить о высоком уровне абстракции данных в нем. Это означает, что роль блоков и наборов может выполнять абсолютно любая сущность динамической web-страницы, а таких сущностей может быть достаточно много, и они, в свою очередь, могут содержать данные, доступные ограниченному количеству пользователей. Кроме того, некоторые сущности необходимо редактировать (обновлять новости на сайте, например), а это дополнительные индивидуальные права доступа, в данном случае на модификацию. Разумеется, при таком подходе необходимо внедрение системы разграничения прав доступа. В методологии BlockSet права доступа устанавливаются на следующие сущности:

1. Локация (динамическая web-страница);
2. Набор;
3. Экземпляр набора.

Предполагается, что все пользователи входят в группы. Один пользователь может входить в множество групп. Используя группы, становится возможным объединять пользователей по общим критериям, устанавливая им одинаковые права доступа на те или иные объекты.

Разрешение доступа на динамическую страницу задается с помощью атрибутов “allowedgroups” и “allowedusers” элемента Location для групп и пользователей соответственно. Существуют атрибуты и для выполнения обратной операции — запрещения доступа к странице: “disallowedusers” и “disal- lowedgroups”. Во всех четырех атрибутах через пробел перечисляются имена групп и пользователей. В атрибуте “restricted” задается, является ли страница закрытой. Если страница является закрытой, то приоритетными будут атрибуты запрещения доступа. В противном случае — разрешения. В случае если пользователю (или его группе) запрещен доступ на страницу, со стороны сервера будет возвращен код HTTP 403 (Запрещено).

Права доступа на экземпляры набора (то есть, по сути, на записи в базе данных) имеют более сложную архитектуру и могут быть следующих типов: создание, чтение, изменение, удаление, что полностью соответствует термину CRUD (от англ.: Create, Read, Update, Delete). Термин CRUD используется в области персистентных хранилищ данных [3]. Примечательно, что каждый тип может иметь всего два значения: «истина» или «ложь», то есть на хранение одного типа разрешения отводится 1 бит. Так как мы предусмотрели четыре типа прав доступа, получаем 4 бита, которые удобно записывать в шестнадцатеричной системе счисления. Это будет старший полубайт. Однако для экземпляров наборов, созданных самим пользователем, необходимо ввести дополнительные права, действующие только для создателя (владельца) данного экземпляра. Их мы запишем в младший полубайт. Владелец обозначается буквой “o” (англ. owner). Полученная битовая карта показана на рис. 1. Таким образом, один байт используется для установки прав на один или несколько экземпляров. В итоге права 0x00 не разрешают ничего, 0xFF разрешают всё, зато 0x0F разрешит полный доступ только создателю (владельцу) набора, тогда как 0x8F разрешит просмотр всем, а любое изменение или добавление — только владельцам. Выборка экземпляров набора для установки прав доступа задается по ключу в базе данных. Ключ может быть как первичный, так и вторичный. В случае вторичного ключа права устанавливаются на все экземпляры и на создание новых экземпляров с этим же ключом. Установленные в наборе права доступа могут наследоваться дочерними наборами. Таким же методом допустимо устанавливать ограничения и на целый набор. Для этого достаточно не указывать ключ вовсе.

7	6	5	4	3	2	1	0
C	R	U	D	Co	Ro	Uo	Do

Рис. 1. Битовая карта прав доступа экземпляра набора

В основе методики хранения разрешений лежит принцип разграничения прав доступа, используемый в UNIX-подобных операционных системах для файлов [4], но с уклоном в CRUD- принцип. Первое отличие заключается в том, что в файловых системах UNIX используется 3 типа прав: на чтение, запись и исполнение. Каждый тип прав, таким образом, задается в восьмеричной системе счисления. Особенности построения динамического web диктуют свои условия, поэтому тип «запись» раздроблен на три составляющие: создание, модификация, удаление. Такое деление оправдано особенностями взаимодействия пользователей с интерактивной web-средой. Например, каждый пользователь может написать сообщение на форуме (создать новый экземпляр набора «сообщения»), однако изменить это сообщение или удалить его администрация форума может не разрешить. Тип «чтение» был оставлен без изменений и подразумевает разрешение на просмотр набора. Тип «исполнение» в среде web не имеет смысла.

Термин CRUD оказался полностью оправдан в представленной архитектуре. Второе отличие заключается в классах. У файлов есть три класса прав доступа: владелец, группа владельца, другие. В нашей архитектуре классов всего два: владелец и общий. Однако если в системе UNIX для одного файла существует один пакет разграничений, в нашем случае для одного экземпляра набора может существовать множество групп и пользователей, которым установлены индивидуальные права на этот набор.

Подведем итоги вышесказанного. Разработанная методика может быть использована для разграничения прав доступа к самым различным частям динамической web-страницы, а также к ней самой. Эта архитектура имеет высокий уровень гибкости, ведь динамическая страница может содержать большое количество разнообразных частей, а на выходе генерироваться документ совершенно индивидуальной конфигурации.

СПИСОК ЛИТЕРАТУРЫ:

1.Обыденных Д. Система разделения прав доступа в веб-приложении // Хабрахабр. 2009. URL: http://habrahabr.ru/post/ 51231/.

2.Bray T. at al. Extensible Markup Language (XML) 1.1 (Second Edition) // World Wide Web Consortium (W3C). 2006. URL: http:/ /www.w3.org/TR/xml11/ (дата обращения: 03.12.2014).

3.Martin J. Managing the Data-base Environment. Englewood Cliffs, New Jersey: Prentice-Hall, 1983. — 381 p.

4.Bach M. J. The Design of the Unix Operating System. New Jersey: PHI Learning Private Limited, 1986. — 110 p.

REFERENCES:

1.Obydennyh D. Sistema razdelenija prav dostupa v veb-prilozhenii // Habrahabr. 2009. URL: http://habrahabr.ru/post/51231/.

2.Bray T. at al. Extensible Markup Language (XML) 1.1 (Second Edition) // World Wide Web Consortium (W3C). 2006. URL: http://www.w3.org/TR/xml11/ (date of access: 03.12.2014).

3.Martin J. Managing the Data-base Environment. Englewood Cliffs, New Jersey: Prentice-Hall, 1983. — 381 p.

4.Bach M. J. The Design of the Unix Operating System. New Jersey: PHI Learning Private Limited, 1986. — 110 p.