Item 82
git_comments:

1. this permissions HTTP method does not match this rule. try other rules
2. check global permissions.
3. * * This checks for the defaults available other rules for the keys
4. * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.
5. this resource needs a principal but the request has come without any credential.
6. * * read a key value as a set. if the value is a single string , * return a singleton set * * @param m the map from which to lookup * @param key the key with which to do lookup
7. this is to do optimized lookup of permissions for a given collection/path
8. check permissions for each collection
9. no role is assigned permission.That means everybody is allowed to access
10. * Licensed to the Apache Software Foundation (ASF) under one or more * contributor license agreements. See the NOTICE file distributed with * this work for additional information regarding copyright ownership. * The ASF licenses this file to You under the Apache License, Version 2.0 * (the "License"); you may not use this file except in compliance with * the License. You may obtain a copy of the License at * * http://www.apache.org/licenses/LICENSE-2.0 * * Unless required by applicable law or agreed to in writing, software * distributed under the License is distributed on an "AS IS" BASIS, * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. * See the License for the specific language governing permissions and * limitations under the License.

git_commits:

1. **summary:** SOLR-7838: An authorizationPlugin interface where the access control rules are stored/managed in ZooKeeper
   message: SOLR-7838: An authorizationPlugin interface where the access control rules are stored/managed in ZooKeeper git-svn-id: https://svn.apache.org/repos/asf/lucene/dev/trunk@1694553 13f79535-47bb-0310-9956-ffa450edef68

github_issues:

github_issues_comments:

github_pulls:

github_pulls_comments:

github_pulls_reviews:

jira_issues:

1. **summary:** Implement a RuleBasedAuthorizationPlugin
   description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-

type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}

2. **summary:** Implement a RuleBasedAuthorizationPlugin
   description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
3. **summary:** Implement a RuleBasedAuthorizationPlugin
   description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
4. **summary:** Implement a RuleBasedAuthorizationPlugin
   description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
5. **summary:** Implement a RuleBasedAuthorizationPlugin
   description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection":

"mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
label: code-design

6. **summary:** Implement a RuleBasedAuthorizationPlugin
description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
7. **summary:** Implement a RuleBasedAuthorizationPlugin
description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
8. **summary:** Implement a RuleBasedAuthorizationPlugin
description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user

solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}

9. **summary:** Implement a RuleBasedAuthorizationPlugin
   description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
10. **summary:** Implement a RuleBasedAuthorizationPlugin
    description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
11. **summary:** Implement a RuleBasedAuthorizationPlugin
    description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
12. **summary:** Implement a RuleBasedAuthorizationPlugin
    description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions

{code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}

13. **summary:** Implement a RuleBasedAuthorizationPlugin
description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
14. **summary:** Implement a RuleBasedAuthorizationPlugin
description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
15. **summary:** Implement a RuleBasedAuthorizationPlugin
description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
16. **summary:** Implement a RuleBasedAuthorizationPlugin
description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit",

"role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}

17. **summary:** Implement a RuleBasedAuthorizationPlugin
    description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}
18. **summary:** Implement a RuleBasedAuthorizationPlugin
    description: h2. authorization plugin This would store the roles of various users and their privileges in ZK sample authorization.json {code:javascript} { "authorization": { "class": "solr.ZKAuthorization", "user-role" :{ "john" : ["admin", "guest"] "tom" : 'dev' } "permissions": [{"name":"collection-edit", "role": "admin" }, {"name":"coreadmin", "role":"admin" }, {"name": "mycoll_update", "collection": "mycoll", "path":["/update/*"], "role": ["guest","admin"] }] } } } {code} This also supports editing of the configuration through APIs Example 1: add or remove roles {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-user-role": {"tom":["admin","dev"}, "set-user-role": {"harry":null} }' {code} Example 2: add or remove permissions {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json'-d '{ "set-permission": { "name":"a-custom-permission-name", "collection":"gettingstarted", "path":"/handler-name", "before": "name-of-another-permission" }, "delete-permission":"permission-name" }' {code} Use the 'before' property to re-order your permissions Example 3: Restrict collection admin operations (writes only) to be performed by an admin only {code} curl --user solr:SolrRocks http://localhost:8983/solr/admin/authorization -H 'Content-type:application/json' -d '{ "set-permission" : {"name":"collection-admin-edit", "role":"admin"}}' {code}

jira_issues_comments:

1. Commit 1694553 from [~noble.paul] in branch 'dev/trunk' [https://svn.apache.org/r1694553] SOLR-7838: An authorizationPlugin interface where the access control rules are stored/managed in ZooKeeper
2. Commit 1694554 from [~noble.paul] in branch 'dev/trunk' [https://svn.apache.org/r1694554] SOLR-7838: CHANGES
3. Commit 1694557 from [~noble.paul] in branch 'dev/branches/branch_5x' [https://svn.apache.org/r1694557] SOLR-7838: An authorizationPlugin interface where the access control rules are stored/managed in ZooKeeper
4. Commit 1694559 from [~noble.paul] in branch 'dev/branches/branch_5x' [https://svn.apache.org/r1694559] SOLR-7838: Predicate is not available in Java 7
5. **body:** What's this? A JIRA without description. Commits without fixVersion and Assignee. No clue on how to use it... No attached patches or discussion of solution before commit. PS: I have not looked at the

code, so the committed stuff may be perfectly OK and wanted. Just felt like giving a friendly reminder about working "[The Apache-Way|http://theapacheway.com/]".
label: code-design

6. Sorry for the trouble. The description is same as that in SOLR-7692 (the parent ticket) other committers insisted on splitting this into multiple pieces. So I just created a ticket for committing stuff. I'll copy the description over
7. Cool, I did not notice the parent when I wrote the comment, it all makes sense now. Sorry for jumping to conclusions :(
8. Commit 1694864 from [~noble.paul] in branch 'dev/branches/lucene_solr_5_3' [https://svn.apache.org/r1694864] SOLR-7757: Improved security framework where security components can be edited/reloaded, Solr now watches /security.json. Components can choose to make their config editable, SOLR-7838: An authorizationPlugin interface where the access control rules are stored/managed in ZooKeeper , SOLR-7837: An AuthenticationPlugin which implements the HTTP BasicAuth protocol and stores credentials securely in ZooKeeper
9. Sorry to be so late in reviewing this. One thing that I noticed was that the "permissions" section (both in security.json and in the output of /admin/authorization) is a JSON object but order is very important here. Now some JSON parsers adhere to JSON standard (keys in map/object are unordered) and some do not. But if we do not change this from a object/map to an array before the release, there'd be no way to change it in a back-compatible manner later. I'd vote to delay the release by a day or two and fix this now.
10. need to fix the syntax as per the feedback
11. Commit 1695308 from [~noble.paul] in branch 'dev/trunk' [https://svn.apache.org/r1695308] SOLR-7838: changed the permissions froma map to an array so that order is obvious
12. Commit 1695324 from [~noble.paul] in branch 'dev/branches/branch_5x' [https://svn.apache.org/r1695324] SOLR-7838: changed the permissions froma map to an array so that order is obvious
13. Commit 1695325 from [~noble.paul] in branch 'dev/branches/branch_5x' [https://svn.apache.org/r1695325] SOLR-7838: changed the permissions froma map to an array so that order is obvious
14. Commit 1695330 from [~noble.paul] in branch 'dev/branches/lucene_solr_5_3' [https://svn.apache.org/r1695330] SOLR-7838: changed the permissions from a map to an array so that order is obvious
15. Commit 1695331 from [~noble.paul] in branch 'dev/branches/lucene_solr_5_3' [https://svn.apache.org/r1695331] SOLR-7838: changed the permissions from a map to an array so that order is obvious
16. Bulk close for 5.3.0 release