说说Zookeeper中的ACL - 就是你的博客 - 博客频道

Access Control在分布式系统中重要性是毋庸置疑的，今天这篇文章来介绍一下Zookeeper中的Access Control(ACL)。

- **1. 概述**
  传统的文件系统中，ACL分为两个维度，一个是属组，一个是权限，子目录/文件默认继承父目录的ACL。而在Zookeeper中，node的ACL是没有继承关系的，是独立控制的。Zookeeper的ACL，可以从三个维度来理解：一是scheme; 二是user; 三是permission，通常表示为scheme:id:permissions, 下面从这三个方面分别来介绍：
  （1）**scheme**: scheme对应于采用哪种方案来进行权限管理，zookeeper实现了一个pluggable的ACL方案，可以通过扩展scheme，来扩展ACL的机制。zookeeper-3.4.4缺省支持下面几种scheme:

  - **world**: 它下面只有一个id, 叫anyone, world:anyone代表任何人，zookeeper中对所有人有权限的结点就是属于world:anyone的
  - **auth**: 它不需要id, 只要是通过authentication的user都有权限（zookeeper支持通过kerberos来进行authencation, 也支持username/password形式的authentication)
  - **digest**: 它对应的id为username:BASE64(SHA1(password))，它需要先通过username:password形式的authentication
  - **ip**: 它对应的id为客户机的IP地址，设置的时候可以设置一个ip段，比如ip:192.168.1.0/16, 表示匹配前16个bit的IP段
  - **super**: 在这种scheme情况下，对应的id拥有超级权限，可以做任何事情(cdrwa)

  另外，zookeeper-3.4.4的代码中还提供了对sasl的支持，不过缺省是没有开启的，需要配置才能启用，具体怎么配置在下文中介绍。

  - **sasl**: sasl的对应的id，是一个通过sasl authentication用户的id，zookeeper-3.4.4中的sasl authentication是通过kerberos来实现的，也就是说用户只有通过了kerberos认证，才能访问它有权限的node.

  （2）**id**: id与scheme是紧密相关的，具体的情况在上面介绍scheme的过程都已介绍，这里不再赘述。

  （3）**permission**: zookeeper目前支持下面一些权限：

  - CREATE(c): 创建权限，可以在在当前node下创建child node

- DELETE(d): 删除权限，可以删除当前的node
- READ(r): 读权限，可以获取当前node的数据，可以list当前node所有的child nodes
- WRITE(w): 写权限，可以向当前node写数据
- ADMIN(a): 管理权限，可以设置当前node的permission

- **2. 实现**

  如前所述，在zookeeper中提供了一种pluggable的ACL机制。具体来说就是每种scheme对应于一种ACL机制，可以通过扩展scheme来扩展ACL的机制。在具体的实现中，每种scheme对应一种AuthenticationProvider。每种AuthenticationProvider实现了当前机制下authentication的检查，通过了authentication的检查，然后再进行统一的permission检查，如此便实现了ACL。所有的AuthenticationProvider都注册在ProviderRegistry中，新扩展的AuthenticationProvider可以通过配置注册到ProviderRegistry中去。下面是实施检查的具体实现：

```
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
```

```
25
26
27
28
29
30
31
32
33
34
35
```

1. void
    checkACL
   (ZooKeeperServer zks, List
   <acl>
    acl, int perm,
2. List
   <id>
    ids)
    throws
    KeeperException.
   NoAuthException <span style="color:rgb(
   0,153,0)">{
3. if

   (skipACL)
    {
4. return
   ;
5. }
6. if

   (acl ==
    null
    || acl.

size
()
 ==
 0
)
{

7. return
;
8. }
9. for

(Id authId :
 ids)
 {
10. if
 (authId.
getScheme
()
.equals
("super"
))
 {
11. return
;
12. }
13. }
14. for

(ACL a :
 acl)
 {
15. Id id = a.
getId
()
;
16. if

 (
(a.getPerms
()
 &
 perm)
 !=
 0
)
{

17. if
 (id.
getScheme
()
.equals
("world"
)
18. && id.
getId
().
equals
("anyone"
))
 {
19. return
;
20. }
21. AuthenticationProvider ap =
 ProviderRegistry.getProvider
(id
22. .getScheme
()
);

23. if

(ap !=

 null
)
{

24. for

(Id authId :
 ids)
 {
25. if
 (authId.
getScheme
()
.equals
(id.
getScheme
()
)
26. && ap.
matches
(authId.getId
()
, id.getId
()
))
 {
27. return
;
28. }
29. }
30. }
31. }
32. }
33. throw
 new
 KeeperException.
NoAuthException

()
;

34. }
35.
</id>
</acl>

- **3. server配置**
  可以通过下面两种方式把新扩展的AuthenticationProvider注册到ProviderRegistry:
  配置文件：在zookeeper的配置文件中，加入authProvider.$n=$classname即可
  JVM参数：启动Zookeeper的时候，通过-Dzookeeper.authProvider.$n=$classname的方式，把AuthenticaitonProvider传入
  在上面的配置中, $n是为了区分不同的provider的一个序号，只要保证不重复即可，没有实际的意义，通常用数字1，2，3等
- **4. 管理ACL**
  可以通过zookeeper client来管理ACL, zookeeper的发行包中提供了一个cli工具zkcli.sh，可以通过它来进行acl管理，下面通过一些例子来说明acl管理的基本方法：

-

```
[zk: bjsd-zk-tst.hadoop.srv:11000(CONNECTED) 2] create -s /test data sasl:test:cdr
Created /test0000000370
[zk: bjsd-zk-tst.hadoop.srv:11000(CONNECTED) 3] getAcl /test0000000370
'sasl,'test
: cdr
[zk: bjsd-zk-tst.hadoop.srv:11000(CONNECTED) 4] create -s /test data ip:192.168.1.2:cdrw
Created /test0000000371
[zk: bjsd-zk-tst.hadoop.srv:11000(CONNECTED) 5] getAcl /test0000000371
'ip,'192.168.1.2
: cdrw
[zk: bjsd-zk-tst.hadoop.srv:11000(CONNECTED) 6] create -s /test data ip:192.168.1.2:cdrw,sasl:test:cdrwa
Created /test0000000372
[zk: bjsd-zk-tst.hadoop.srv:11000(CONNECTED) 7] getAcl /test0000000372
'ip,'192.168.1.2
: cdrw
'sasl,'test
: cdrwa
[zk: bjsd-zk-tst.hadoop.srv:11000(CONNECTED) 8] create -s /test data
Created /test0000000373
[zk: bjsd-zk-tst.hadoop.srv:11000(CONNECTED) 9] setAcl /test0000000373 ip:192.168.1.3:cdrwa
cZxid = 0x1b00024ab5
ctime = Sun Jun 02 20:23:54 CST 2013
mZxid = 0x1b00024ab5
mtime = Sun Jun 02 20:23:54 CST 2013
pZxid = 0x1b00024ab5
cversion = 0
dataVersion = 0
aclVersion = 1
ephemeralOwner = 0x0
dataLength = 4
numChildren = 0
[zk: bjsd-zk-tst.hadoop.srv:11000(CONNECTED) 10] getAcl /test0000000373
'ip,'192.168.1.3
: cdrwa
```

顶
1