ACE Syntax

An ACE is defined by a combination of user, group, or *role* definitions. You can combine these definitions using the following syntax:

Operator	Description
u	Username or user ID, as they appear in `/etc/passwd`, of a specific user. Usage: `u:<username or user ID>`
g	Group name or group ID, as they appear in /etc/group, of a specific group. Usage: `g:<group name or group ID>`
r	Name of a specific role. Usage: `r:<role name>`.
p	Public. Specifies that this operation is available to the public without restriction. Cannot be combined with any other operator.
!	Negation operator. Usage: `!<operator>`.
&	AND operation.
|	OR operation
()	Delimiters for subexpressions.
""	The empty string indicates that no user has the specified permission.

An example definition is `u:1001 | r:engineering`, which restricts access to the user with ID 1001 or to any user with the role `engineering`.

In this next example, members of the group `admin` are given access, and so are members of the group `qa`:

```
g:admin | g:qa
```

For another example, suppose that you have this list of groups to which you want to give read permissions:

- The `admin` group as a whole, but not the admins for a particular cluster (which is named `cl3`).
- Members of the `qa` group who are responsible for testing the two applications (named `app2` and `app3`).
- The business analysts (group `ba`) in department 7A (group `dept_7a`)
- All of the data scientists (group `ds`) in the company.

To grant the read permission, you construct this boolean expression:

```
u:cfkane | (g:admin & !g:cl3) | (g:qa & (g:app2 | g:app3)) | (g:ba & g:dept_7a) | g:ds
```

This expression is made up of five subexpressions which are separated by OR operators.

- - The first subexpression `u:cfkane` grants the read permission to the username `cfkane`.
  - The subexpression `(g:admin & !g:cl3)` grants the read permission to the admins for all clusters except cluster `cl3`. The operator `g` is the group operator, the value `admin` is the name of the group of all admins. The `&` operator limits the number of administrators who have read permission because

only those administrators who meet the additional condition will have it.

The condition `!g:cl3` is a limiting condition. The operator `!` is the NOT operator. Combined with the group operator, this operator means that this group is excluded and does not receive the read permission.

Warning: Be careful when using the NOT operator. You might exclude fewer people than you intended. For example, suppose that you do not want anyone in the group `group_a` to have access. You therefore define this ACE: `!g:group_a` You might think that the data is now protected because members of `group_a` do not have access to it. However, you have not restricted access for anyone else except the members of `group_a` . The rest of the world can access the data. You should not define ACEs through exclusion by using the NOT operator. You should define them by inclusion and use the NOT operator to limit further the access of the groups or roles that you have included.
In the subexpression `(g:admin & !g:cl3)` , the NOT operator limits the number of members within the admin group who have access. The `admin` group is included, and all users who are also part of the `cl3` group are excluded.

- The subexpression `(g:qa & (g:app2 | g:app3))` demonstrates that you can use a subexpression within a subexpression. The larger subexpression means that only members of group `qa` who are also members of group `app2` or `app3` have read access to the data. The smaller subexpression limits the number of people in the `qa` group have have this permission.
- - The next two subexpressions -- `(g:ba & g:dept_7a)` and `g:ds` -- grant the read permission to the members of group `ba` who are also in the group `dept_7a` . It also grants permission to the members of the group `ds` .