8.7. Property-level access control deprecated

This section describes how to configure property-level access control in Neo4j.

Please note that the methods for defining property-level access control, as described in this section, have been deprecated and will be removed in a future release. New methods for this functionality will be provided in an upcoming release.

Ths section describes the following:

- Introduction
- Implement a property blacklist

8.7.1. Introduction deprecated

You can use role-based, database-wide, property blacklists to limit which properties a user can read.

The table below lists the configuration parameters that control this feature:

Parameter name	Default value	Description

Parameter name	Default value	Description
`dbms.security.property_level.enabled` (../../reference/configuration-settings/#config_dbms.security.property_level.enabled)	`false`	Enable property level access control.
`dbms.security.property_level.blacklist` (../../reference/configuration-settings/#config_dbms.security.property_level.blacklist)		An authorization mapping for property level access for roles. The map should be formatted as a semicolon-separated list of key-value pairs, where the key is the role name and the value is a comma-separated list of blacklisted properties. The blacklisted properties for a given user is the union of the blacklist for all the roles that user is part of.

Considerations

The property blacklist prevents users from reading properties. A user querying for a blacklisted property will get the same results as if the property did not exist on the node/relationship.

Blacklisting a property will only affect the reading of that property, not the writing. It is therefore recommended to only add users that are assigned the `reader` role to roles that have a property blacklist.

Limitations

All properties with a name corresponding to the ones in the blacklist will be affected. This is regardless of whether it is associated with a node or a relationship, and regardless of node labels and relationship types.

8.7.2. Implement a property blacklist deprecated

To enable this feature, the following steps must be taken:

1. Enable property-level access control through the setting `dbms.security.property_level.enabled`.
2. Configure the setting `dbms.security.property_level.blacklist` to restrict specific roles from reading the named properties.
3. Create one, or several, custom roles, which will have restricted property access.
4. Add user to appropriate roles.

Example 8.23. Implement a property blacklist

First, we enable property-level access control and create the blacklist:

```
dbms.security.property_level.enabled=true
dbms.security.property_level.blacklist=\
  roleX=propertyA;\
  roleY=propertyB,propertyC
```

Then, we create the custom roles and assign users to them:

```
CALL dbms.security.createRole('roleX')
CALL dbms.security.createRole('roleY')
CALL dbms.security.addRoleToUser('roleX', 'user-1')
CALL dbms.security.addRoleToUser('roleY', 'user-2')
CALL dbms.security.addRoleToUser('roleX', 'user-3')
CALL dbms.security.addRoleToUser('roleY', 'user-3')
```

This will have the following effects:

- The user `user-1` will be unable to read the property `propertyA`.
- The user `user-2` will be unable to read the properties `propertyB` and `propertyC`.
- The user `user-3` will be unable to read the properties `propertyA`, `propertyB` and `propertyC`.