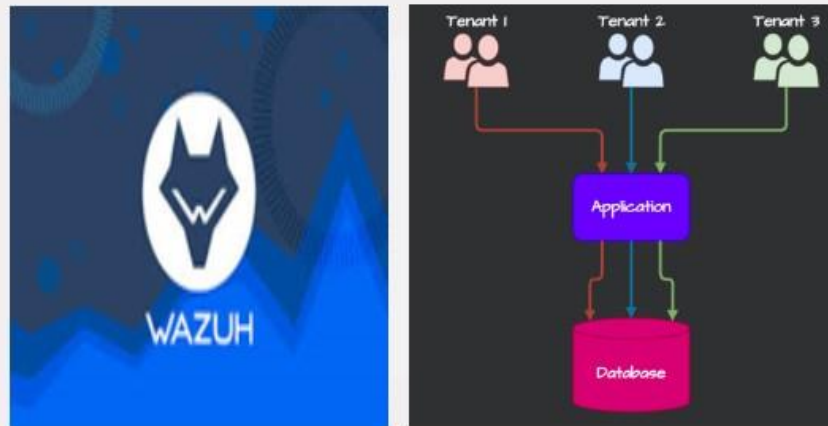




# MULTI-TENANCY IN WAZUH

## Multi-Tenant Wazuh



### Created By:

Muhammad Akhtar Nadeem  
Department of Information Technology  
Akhwat College University Kasur.

### Instructor:

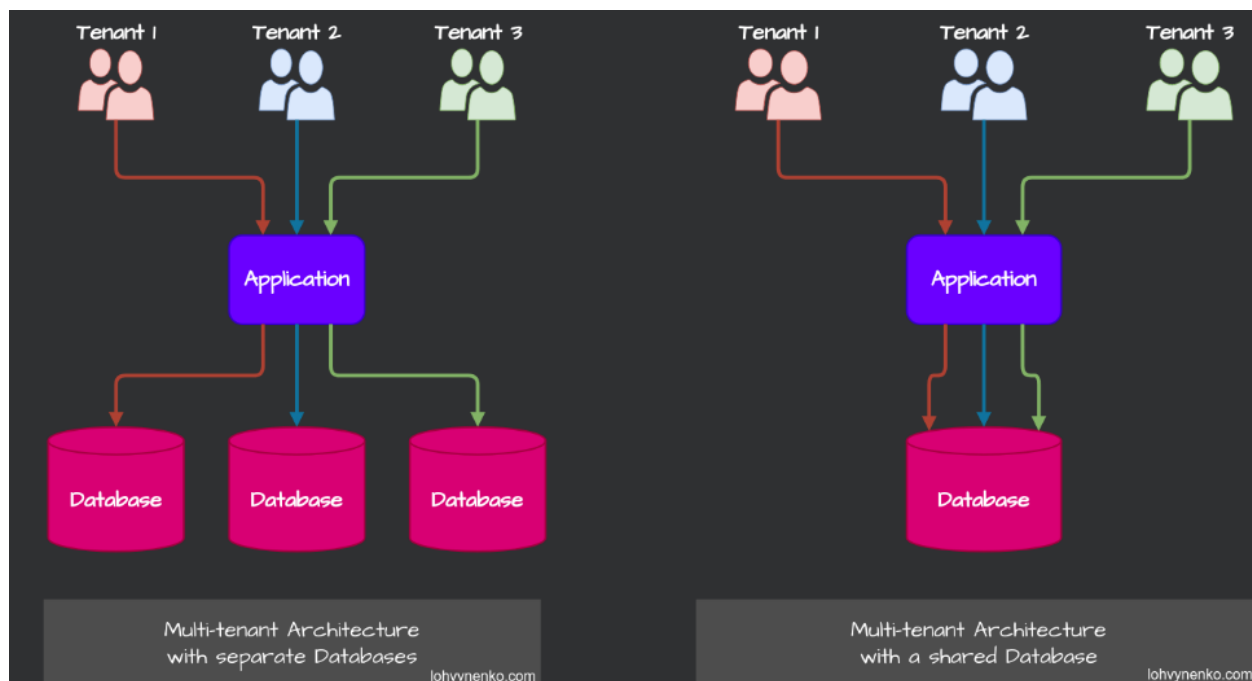
MUHAMMAD MOIZ UD DIN RAFAY  
IT Fortress  
Computer and Network Security

## What is Multi-Tenancy?

Multi-tenancy is an architecture in which a single instance of a software application serves multiple customers. In this environment, each customer is called a tenant. In a multi-tenant system, multiple instances of an application operate in a shared environment.

However, each tenant's data is isolated and remains invisible to other tenants. This is made possible because each tenant is integrated physically but logically separated. This means that a single instance of the software can run on one server and can serve multiple tenants.

## Multi-tenant Architecture



## How Does Multi-Tenancy Work?

### Single Application, Single Database

In this configuration, all tenants share a single application instance along with a single database. Each tenant's data is separated and isolated within the same database using schemas or tenant-specific identifiers.

## **Single Application, Multiple Database**

The single application, multiple database model involves one application instance connected to multiple databases. Each tenant has its own database, ensuring data isolation at the storage level.

## **Multiple Application, Multiple Database**

In the multiple application, multiple database model, each tenant has their own dedicated application instance as well as a separate database. This setup provides the highest level of isolation and security among the multi-tenant architectures.

## **Features of a multi-tenant architecture**

- **Costs:** Multi-tenant operation has a lower cost per user due to resources being shared.
- **Efficiency:** Since the environment is the same for each customer, onboarding new customers is a faster process.
- **Maintenance:** Updates to a multi-tenant system include all customers, so system upgrades and maintenance can be managed by the software provider rather than the individual customers. This leads to fewer issues with multiple versions of the application running for different customers.

## **Limitations to multi-tenant architectures:**

- **Vulnerability:** A multi-tenant environment allows multiple access points for users, which can increase the threat of a security breach.
- **Complexity:** Serving multiple clients in one instance of an application/database adds an extra level of complexity to the codebase and database maintenance.
- **Backup:** A multi-tenant environment makes backup and restoration more complex, so not all providers offer reliable restoration services.
- **Less customization:** A multi-tenant environment offers fewer customizations, so users have less control over the quality. There can be pressure to add customizations for individual tenants, especially if they are large organizations.
- **Global problems:** If a technical problem occurs on the provider's end, it can lead to issues for all users. This may apply to uptime, system upgrades and other global processes.

## Multi-tenancy in Wazuh server

Tenants in the Wazuh dashboard are containments for saving index patterns, visualizations, dashboards, and other objects. Tenants are useful for safely sharing your work with other users. You can control which roles have access to a tenant and whether those roles have read or write access. By default, all the Wazuh dashboard users have access to two independent tenants.

### How Multi-Tenancy Works in Production Environment (Wazuh)

In a real working environment (like a company or security center), multi-tenancy means using one Wazuh system to manage and monitor logs for many users or teams, but each team only sees their own data.

#### Example:

A company has 3 departments IT, Finance, and HR. All departments send their logs to the same Wazuh server. Using multi-tenancy, each department logs into the same Wazuh dashboard. Only sees their own agents, logs, alerts, and dashboards. Cannot see other departments' data.

#### How is the data safe?

Even though everyone uses the same Wazuh system, they can only see their own data. This is done using security rules that hide data from other users.

#### Why use it in production?

- One Wazuh system can be used for many teams or clients.
- Saves cost, time, and hardware.
- Good for security companies who watch many clients from one place.
- Helps keep data private and secure.

### How enable Multi-Tenant in Wazuh

Perform the following instructions below on the Wazuh dashboard to enable multi-tenancy.

1. Edit the [/etc/wazuh-dashboard/opensearch\\_dashboards.yml](#) configuration file and make the following changes:

- Set the [opensearch\\_security.multitenancy.enabled](#) setting to [true](#).

- Add the line `opensearch_security.multitenancy.tenants.preferred: ["Global", "Private"]`
- Additionally, you can edit the `uiSettings.overrides.defaultRoute` to set a default tenant, for example, global, each time a user logs in.

`uiSettings.overrides.defaultRoute:/app/wz home?security_tenant=global`

```
GNU nano 7.2 /etc/wazuh-dashboard/opensearch_dashboards.yml
server.host: 0.0.0.0
opensearch.hosts: https://127.0.0.1:9200
server.port: 443
opensearch.ssl.verificationMode: certificate
# opensearch.username: kibanaserver
# opensearch.password: kibanaserver
opensearch.requestHeadersAllowlist: ["securitytenant","Authorization"]
opensearch_security.multitenancy.enabled: true
opensearch_security.multitenancy.tenants.preferred: ["Global", "Private"]
opensearch_security.readonly_mode.roles: ["kibana_read_only"]
server.ssl.enabled: true
server.ssl.key: "/etc/wazuh-dashboard/certs/wazuh-dashboard-key.pem"
server.ssl.certificate: "/etc/wazuh-dashboard/certs/wazuh-dashboard.pem"
opensearch.ssl.certificateAuthorities: ["/etc/wazuh-dashboard/certs/root-ca.pem"]
uiSettings.overrides.defaultRoute: /app/wz-home?security_tenant=global
opensearch_security.cookie.secure: true
```

2. Restart the Wazuh dashboard so changes can take effect.

`# systemctl restart wazuh-dashboard`

## Multi-Tenancy Setup in Wazuh using OpenSearch Dashboards – Step-by-Step Guide

This process involves adding a label in the agent group's centralized configuration to identify the Wazuh alerts coming from this group of agents, creating an internal user, and giving it reading permission only for those documents that correspond to the group of authorized agents.

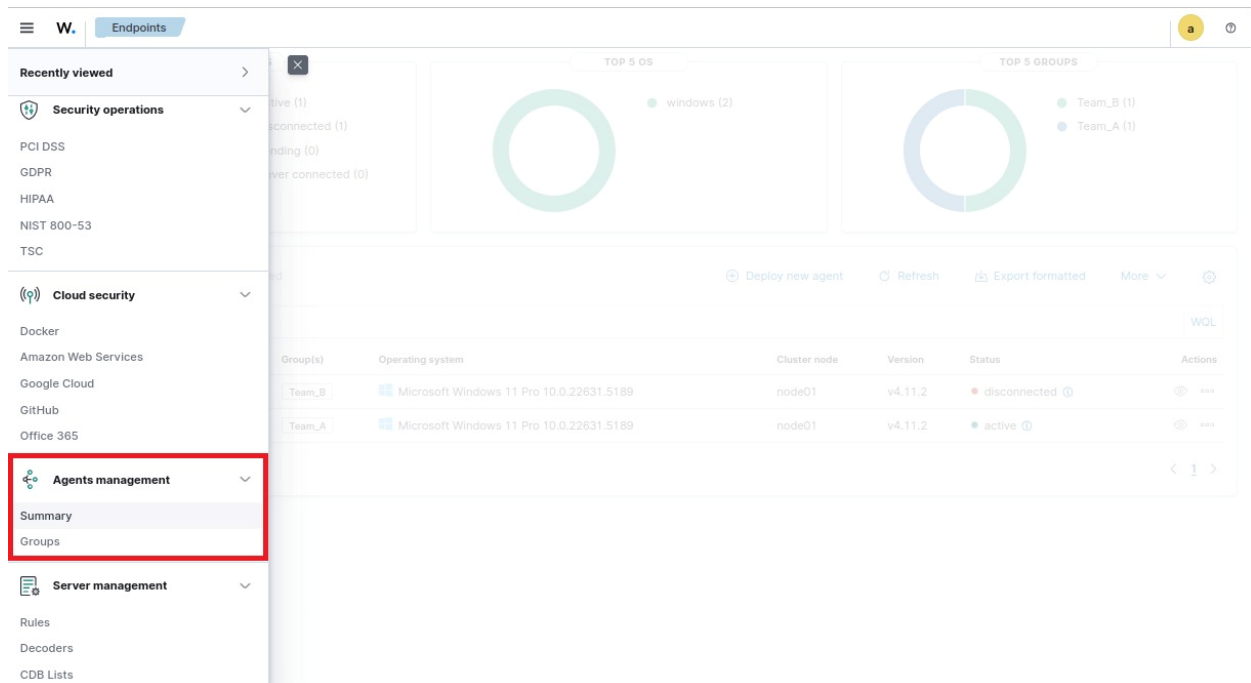
It also involves mapping this user with the Wazuh API, creating a custom policy that includes permissions to read, restart, upgrade, among other actions over a group of agents, and finally creating a custom role and mapping it to our internal user.

As a final result, we will have a new user with permission to manage a group of agents and read the documents regarding the said group.

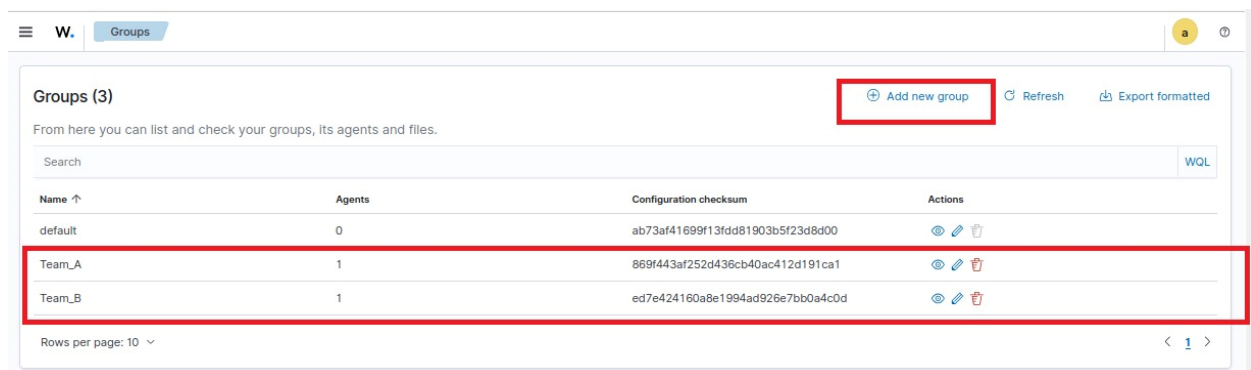
### Task 1:

Create the two groups Team\_A and Team\_B and In the group configuration, you must use the LABEL to distinguish the groups. Add the agent to these two group.

- i. Log into the Wazuh dashboard as administrator.
- ii. Select Agents management > Endpoint Groups to open the page.



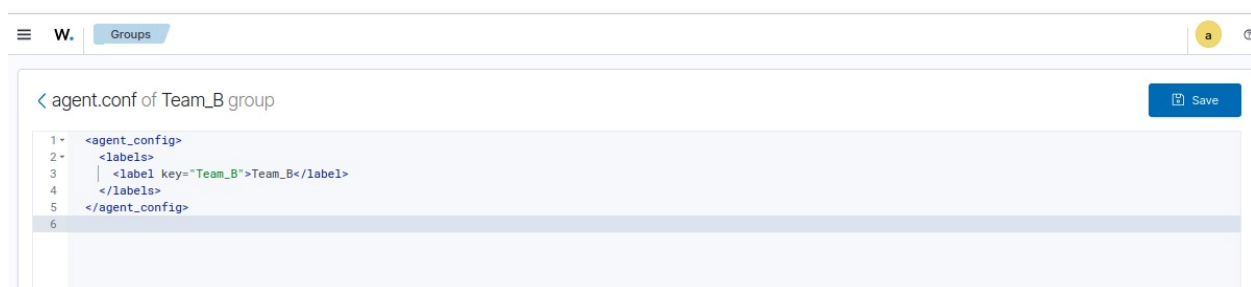
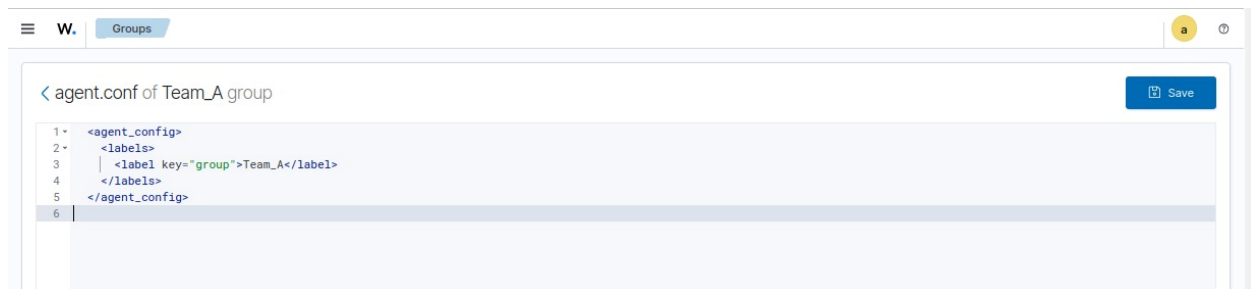
- iii. Create the two group Team\_A and Team\_B.



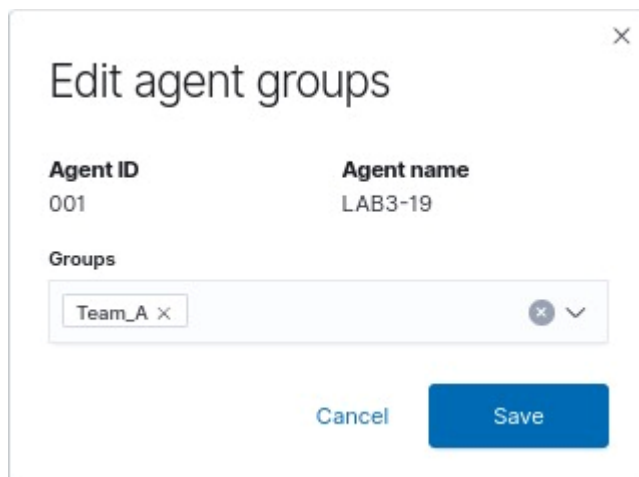
- iv. Select group and click Edit group configuration.
- v. Add a label to identify the group, for example.
 

```
<agent_config>
  <labels>
    <label key="group">Team_A</label>
```

</labels>  
</agent\_config>




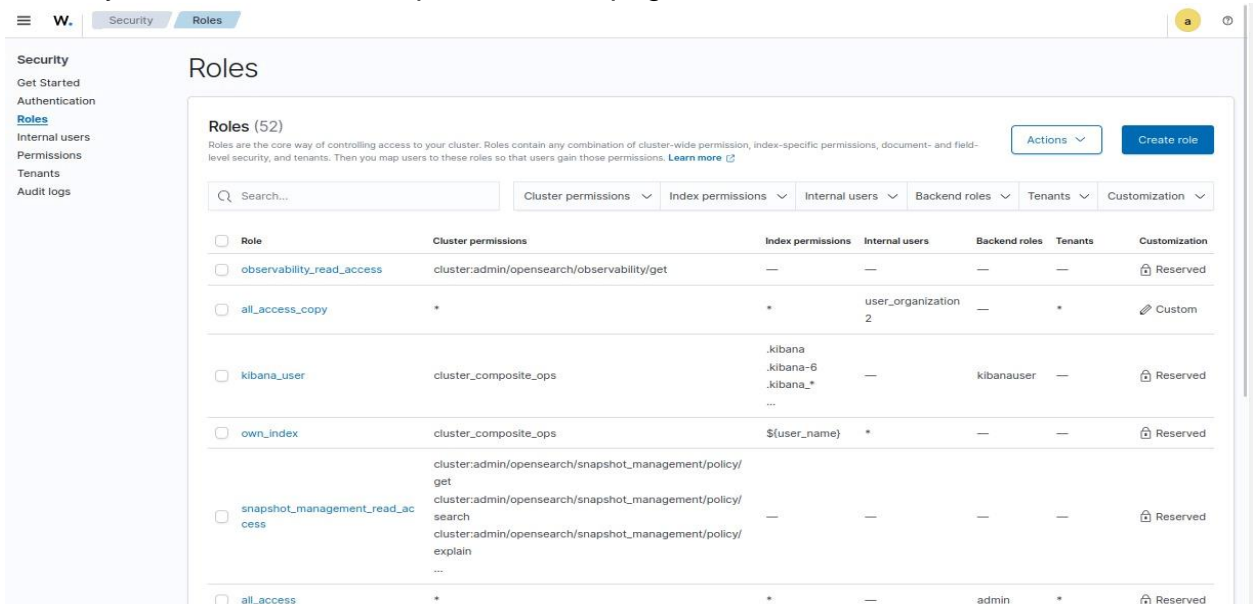
- vi. Select the agent and assign them to the previously created group.



## Task 2:

Define roles for both group in the OpenSearch security section and then map these roles to internal user.

- i. Click  to open the menu on the Wazuh dashboard, go to index management > Security, and then Roles to open the roles page and click on create role.



- Name: Assign a name to the role.
  - Name: Assign a name to the role.(read\_Team\_A)
  - Cluster permissions: [cluster\\_composite\\_ops\\_ro](#)
  - Index: \*
  - Index permissions: [read](#)
  - Click Add another index permission and unfold the new section Add index permission. Complete the empty fields with the following parameters and make sure to replace your group name accordingly:
  - Index: [wazuh-alerts\\*](#)
  - Index permissions: [read](#)
  - Document level security:

```
{
  "bool": {
    "must": {
      "match": {
        "agent.labels.group": "Team_A"
      }
    }
  }
}
```



```

    }
  }
}

```

- Click Add another index permission and unfold the new section Add index permission. Complete the empty fields with the following parameters and make sure to replace your group name accordingly:

- Index: [wazuh-monitoring\\*](#)
- Index permissions: [read](#)
- Document level security:

```

{
  "bool": {
    "must": {
      "match": {
        "group": "Team_A"
      }
    }
  }
}

```

- Under Tenant permissions, select Tenant: [global\\_tenant](#) and the Read only option.
- Click Create to complete the task.

The screenshot displays the 'Index permissions' configuration interface in a web browser. It features three identical, vertically stacked sections for adding new index permissions. Each section includes a 'Name' input field, a 'Permissions' dropdown menu currently set to 'read', and a 'Document level security' field containing a JSON query: `{ "bool": { "must": { "match": { "group": "Team_A" } } } }`. Below the JSON field, there is a 'Tenant' dropdown menu set to 'global\_tenant' and a 'Create' button. The interface is clean with a light blue header and a white main content area.

- ii. Click Select the **Mapped users** tab, click Manage mapping, and create new internal user.

The screenshot shows the 'Create internal user' page. At the top, there are tabs for 'Security', 'Internal users', and 'Create internal user'. The 'Credentials' section includes a 'Username' field with the value 'ECO\_USER' and a 'Password' field with masked characters. A password strength indicator shows 'Very strong'. Below the password field is a 'Re-enter password' field. The 'Backend roles - optional' section has a 'Backend role' field with the value 'Type in backend role' and a 'Remove' button. There is also an 'Add another backend role' button.


- iii. Go to map user page and map the user you created in previous step.

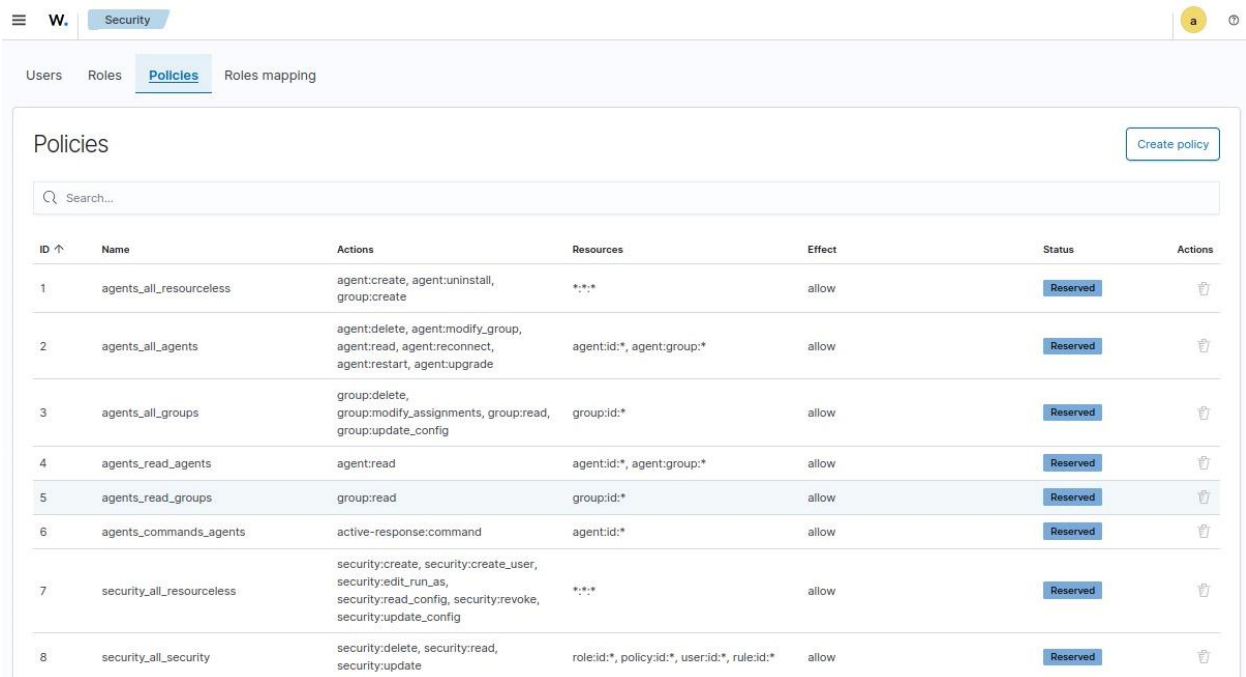
The screenshot shows the 'Map user' page. At the top, there are tabs for 'Security', 'Roles', 'Team\_A', and 'Map user'. The 'Map user' section includes a 'Users' field with the value 'ECO\_USER' and a 'Create new internal user' button. Below the 'Users' field is a 'Backend roles' section with a 'Backend role' field containing the value 'adminiostrator' and a 'Remove' button. There is also an 'Add another backend role' button. At the bottom right, there are 'Cancel' and 'Map' buttons.









I created role and mapped user for group one (Team\_A) now you can create role and mapped user for second group (Team\_B) using previous same steps.

### Task 3:

Create new policies for each group and then assign it to each role that is we created in previous task.

- i. Click  to open the menu on the Wazuh dashboard, go to server management > Security, and then policies to open the policies page.



ID ↑	Name	Actions	Resources	Effect	Status	Actions
1	agents_all_resourceless	agent:create, agent:uninstall, group:create	*.*.*	allow	Reserved	
2	agents_all_agents	agent:delete, agent:modify_group, agent:read, agent:reconnect, agent:restart, agent:upgrade	agent:id:*, agent:group:*	allow	Reserved	
3	agents_all_groups	group:delete, group:modify_assignments, group:read, group:update_config	group:id:*	allow	Reserved	
4	agents_read_agents	agent:read	agent:id:*, agent:group:*	allow	Reserved	
5	agents_read_groups	group:read	group:id:*	allow	Reserved	
6	agents_commands_agents	active-response:command	agent:id:*	allow	Reserved	
7	security_all_resourceless	security:create, security:create_user, security:edit_run_as, security:read_config, security:revoke, security:update_config	*.*.*	allow	Reserved	
8	security_all_security	security:delete, security:read, security:update	role:id:*, policy:id:*, user:id:*, rule:id:*	allow	Reserved	

- ii. Click **Create policy** and complete the empty fields with the requested information.
  - Policy name: Assign a name to the new policy (Read\_Team\_A).
  - Action: Select the actions that the user is allowed to perform, for example, [agent:read](#), and click [Add](#). Select as many actions as needed.
  - Resource: Select [agent:group](#).
  - Resource identifier: Write the name of the agents' group, for example, Team\_A, and click Add. You can add as many resources as needed.
  - Select an effect: [Select Allow](#).
  - Click **Create policy** to complete the action

W.

Security

Users

Roles

Policies

Roles mapping

Policies

Search...

ID	Name	Actions	Resources
1	agents_all_resourceless	agent:create, agent:uninstall, group:create	*.*.*
2	agents_all_agents	agent:delete, agent:modify_group, agent:read, agent:reconnect, agent:restart, agent:upgrade	agent.id.*
3	agents_all_groups	group:delete, group:modify_assignments, group:read, group:update_config	group.id.*
4	agents_read_agents	agent:read	agent.id.*
5	agents_read_groups	group:read	group.id.*
6	agents_commands_agents	active-response:command	agent.id.*
7	security_all_resourceless	security:create, security:create_user, security:edit_run_as, security:read_config, security:revoke, security:update_config	*.*.*
8	security_all_security	security:delete, security:read, security:update	role.id.*, policy.id.*
9	users_all_resourceless	security:create_user, security:edit_run_as, security:revoke	*.*.*
10	users_all_users	security:delete, security:read, security:update	user.id.*

New policy

Policy name

Read\_Team\_A

Introduce a name for this new policy.

Action

Set an action where the policy will be carried out.

Actions

agent:read

Resource

agent:group

Select the resource to which this policy is directed.

Resource identifier

Introduce the resource identifier. Type \* for all.

Resources

agent:group.Team\_A

Select an effect

Allow

Select an effect.

Create policy

- iii. Click **Roles** to open the tab, click **Create Role**, and fill in the empty fields with the requested information.
- Role name: Assign a name to the new role (read\_Team\_A).
  - Policies: Select the policy created previously.
  - Click Create role to confirm the action.

W.

Security

Users

Roles

Policies

Roles mapping

Roles

Search...

ID	Name	Policies
1	administrator	agents_all_resourceless agents_all_agents agents_all_groups agents_commands_agents cluster_all_nodes cluster_read_policy decoders_all_files decoders_all_resourceless rules_all_resourceless role_read_name soc_read_soc syscheck_all_syscheck events_agent_resourceless
2	readonly	agents_read_agents agents_read_group soc_read_policy cluster_read_resource syscheck_read_syscheck rules_read_rules rules_read_name soc_read_soc
3	users_admin	users_all_resourceless users_all_users
4	agents_readonly	agents_read_agents agents_read_groups
5	agents_admin	agents_all_resourceless agents_all_agents agents_all_groups
6	cluster_readonly	cluster_read_resourceless cluster_read_nodes
7	cluster_admin	cluster_all_resourceless cluster_all_nodes
100	read_organization1	read_organization1
101	read_organization2	read_organization2
102	IT_Team	Read_Team_A

Rows per page: 10 / -

New role

Role name

read\_Team\_A

Introduce a name for this new role.

Policies

Read\_Team\_A X

Assign policies to the role.

Create role

- iv. Click on Create Role mapping and complete the empty fields with the requested information.
- Role mapping name: Assign a name (read\_Team\_A\_mapping) to the role mapping.
  - Roles: Select the role created previously, for example Team\_A\_role, and the cluster\_readonly role. The latter assigns the user basic configuration reading permissions.
  - Internal users: Select the internal user created previously.
  - Click Save role mapping to confirm the action.

The screenshot displays the Wazuh dashboard's 'Roles mapping' section. On the left, a table lists existing mappings:

ID	Name	Roles
1	wui_elastic_admin	administrator
2	wui_opensearch_admin	administrator
100	read_organization1	read_organization1
101	read_organization2	read_organization2
102	IT_Team	IT_Team
103	Read_Team_B_Mapping	read_Team_B

On the right, the 'Create new role mapping' modal is open. It contains the following fields and sections:

- Role mapping name:** A text input field containing 'read\_Team\_A\_mapping'.
- Roles:** A dropdown menu labeled 'Select roles'.
- Mapping rules:** A section titled 'Map internal users' with a dropdown menu showing 'ECO\_USER'.
- Custom rules:** A section with a dropdown menu showing 'Any are true' and a link to 'Add new rule'.
- Buttons:** A 'Save role mapping' button at the bottom right.

I created policy, role and then mapping the role for group one (Team\_A) now you can create for second group (Team\_B) through these same steps.

## Task 4:

For Wazuh App permissions, you will have to do the following:

Make sure that `run_as` is set to true in the `/usr/share/wazuh-dashboard/data/wazuh/config/wazuh.yml` configuration file. Restart the Wazuh dashboard service and clear your browser cache and cookies.

```

GNU nano 7.2 /usr/share/wazuh-dashboard/data/wazuh/config/wazuh.yml
# url: https://env-1.example
# # Host / API port
# port: 55000
# # Host / API username
# username: wazuh-wui
# # Host / API password
# password: "fGL0D0l2+*vSm3s0RAePk..40gyaYcVg"
# # Use RBAC or not. If set to true, the username must be "wazuh-wui".
# run_as: true
# - env-2:
# url: https://env-2.example
# port: 55000
# username: wazuh-wui
# password: "fGL0D0l2+*vSm3s0RAePk..40gyaYcVg"
# run_as: true

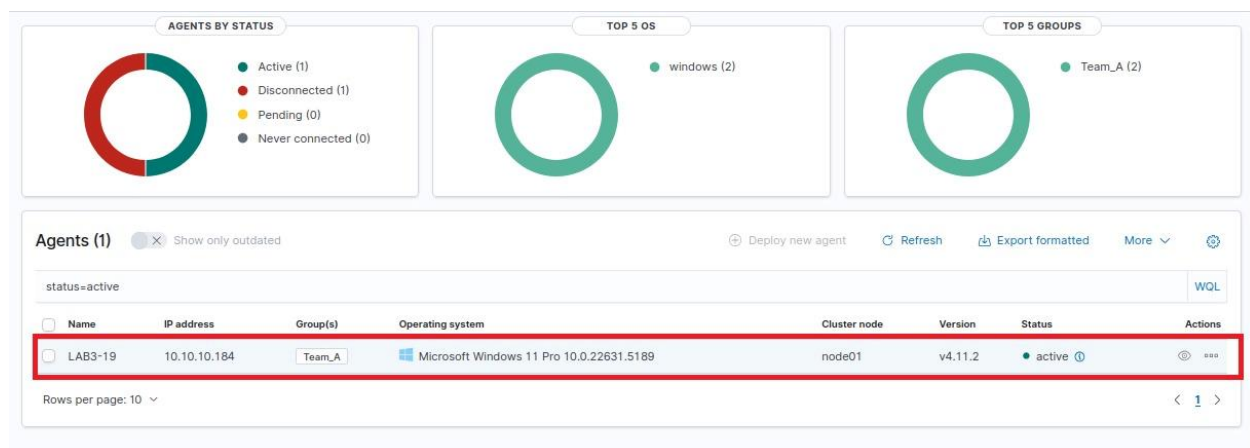
hosts:
- default:
  url: https://127.0.0.1
  port: 55000
  username: wazuh-wui
  password: "fGL0D0l2+*vSm3s0RAePk..40gyaYcVg"
  run_as: true

```

Finally, we enabled the multi-tenancy in wazuh now we will verify it through login the both group's users. Logout from administrator account and login from Team\_A user.



Now we only see and manage the Team\_A group agents that is verify we created two different multi-tenant successfully.



## Conclusion

In this document, we explained what multi-tenancy is, how it works, its benefits and limitations, and how to enable and configure it in Wazuh using OpenSearch Dashboards.

Multi-tenancy is very useful in real-world environments where different teams or clients need to use the same system but with isolated data and access. It allows organizations to manage multiple users or groups securely, all from a single Wazuh server.

By following the step-by-step guide, we successfully created separate users and roles for different groups and verified that each user can only see their own data. This confirms that multi-tenancy was configured correctly in Wazuh.

This setup is especially useful for enterprises and security providers who want to save resources while keeping their environments secure and organized.