**Student’s Individual Study #2**

**Full Name:** Bekenov Temirlan  
**Educational Program:** Faculty of IT

**Topic: Users and Permissions**

**Target**

The purpose of this study is to define a complete role and permission model for the TaskBoard project, including system users, groups, and automation accounts.  
Additionally, the goal is to configure user and group management in Linux, apply proper file and sudo permissions, and enable secure SSH access using key-based authentication.

**Tasks**

* Declare a role table;
* Create all groups and at least one user for each role;
* Assign permissions for all groups;
* Configure SSH access using key pairs.

**1. Role Table Definition**

The TaskBoard system defines several user roles and service accounts to ensure clear separation of privileges and secure system operation.

| **Role Name** | **Description** | **Permissions** | **Superuser Permissions** |
| --- | --- | --- | --- |
| Administrator | Manages the entire system and user access | Full read, write, delete, create, and service management rights | sudo access for system configuration |
| Developer | Develops and maintains backend and frontend code | Read/write access to application directories and databases | None |
| Database Admin (DBA) | Manages PostgreSQL database | Full control of PostgreSQL instance | sudo -u postgres |
| Auditor | Reviews logs and verifies security configurations | Read-only access to logs and configurations | None |
| Automation Bot | Executes automated tasks and deployments | SSH access with key authentication and limited service control | sudo for restarting services |
| TaskBoard App User | Runs backend Node.js service | Read/write within /var/taskboard/ | None |
| Nginx User | Runs Nginx proxy and serves static files | Read-only access to /var/www/taskboard | None |
| Postgres System User | Internal PostgreSQL user | Database-level administration | sudo -u postgres |

**2. Creating Groups and Users**

Each defined role corresponds to a Linux group, and each group has one or more users assigned to it.  
User and group creation was automated using dedicated setup scripts.

Service accounts such as taskboard, nginx, and postgres are configured as system users with no login shell to maintain system security, while standard users such as admin1 or dev1 have normal access and home directories.

**3. Assigning Permissions**

Proper directory and file permissions were configured to ensure system security and controlled access between components.  
Ownership and access rights for application, database, and web directories were assigned to the appropriate service accounts.  
Access to sensitive directories, such as /var/log, was limited to authorized users only.

Sudo privileges were configured through system-level configuration files, providing controlled administrative access.  
For example, administrators have full privileges, while the automation account can only restart necessary services.  
This approach maintains both flexibility and security within the environment.

**4. SSH Configuration for Automation User**

To improve security and support automated deployments, the automation account uses SSH key-based authentication instead of passwords.  
This configuration establishes a secure connection channel for automation tasks without exposing credentials.

The SSH configuration was adjusted to disable password-based login and enforce the use of public/private key pairs.  
This enhances security while enabling automated maintenance and deployment scripts to execute commands remotely.

**5. Step-by-Step Report Summary**

1. Defined roles and permissions for all users and system services.
2. Created Linux groups and users for each defined role.
3. Assigned directory and sudo privileges according to role requirements.
4. Configured secure SSH access for the automation user using key-based authentication.
5. Verified that all permissions and user configurations function correctly and align with security best practices.

**Conclusions**

In this study, a complete user and permission model for the TaskBoard project was implemented.  
All users and service accounts were assigned defined roles, permissions, and access levels.  
Automation was introduced to simplify setup and reduce configuration errors.  
The integration of SSH key-based authentication ensures secure and reliable access, forming a strong foundation for future project deployment and scalability.