Here's how to set up access control on MySQL securely on Ubuntu in accordance with the CIS benchmarks:

1. User Accounts and Privileges:

- **Principle of Least Privilege:** Follow the principle of least privilege when creating user accounts. Grant users only the minimum permissions required to perform their specific tasks within the database. Avoid assigning excessive permissions like *.* which allows access to all databases and privileges.
- **Dedicated Users:** Avoid using the root user for everyday database operations. Create dedicated user accounts with restricted privileges for your applications. This minimizes potential damage if the user credentials are compromised.
- **Strong Passwords:** Enforce strong password policies when creating user accounts. Use a combination of upper and lowercase letters, numbers, and special characters. Consider a minimum password length of at least 12 characters.
- **Authentication Methods:** Use secure authentication plugins like caching_sha2_password or the improved mysql_native_password plugin. These provide stronger encryption compared to older methods.

2. Remote Access Control:

- **Restrict Remote Access (Optional):** The CIS benchmarks might recommend restricting remote access to the MySQL server by default. This can be achieved by setting the bind-address option in the MySQL configuration file (/etc/mysql/my.cnf) to only listen on localhost (127.0.0.1). If remote access is necessary, restrict it to specific IP addresses using firewall rules.

3. Account Lockouts (Optional):

- **Implement Account Lockouts:** Consider implementing account lockouts after a certain number of failed login attempts. This helps prevent brute-force attacks where attackers try to guess user passwords.

4. Password Expiration (Optional):

- **Enforce Password Expiration:** Enforce password expiration policies to require users to change their passwords periodically. This reduces the risk associated with compromised passwords being used for an extended period.

5. Database and Object Permissions:

- **Grant Specific Permissions:** When granting permissions to users, focus on specific databases, objects (tables, views, etc.), and operations (SELECT, INSERT, UPDATE, DELETE) relevant to their needs. Avoid granting unnecessary access to entire databases or schemas.

Here's an example command for creating a secure user account with limited privileges:

SQL

```
CREATE USER 'username'@'localhost' IDENTIFIED BY 'strong_password'
REQUIRE NONE;
GRANT SELECT, INSERT ON database.specific_table TO
'username'@'localhost';
FLUSH PRIVILEGES;
```

Explanation:

- This command creates a user named username with access only from the localhost.
- strong_password is a placeholder for your actual strong password.
- REQUIRE NONE removes unnecessary privileges like GRANT.
- The GRANT statement assigns specific permissions (SELECT, INSERT) for the specific_table within the database to the user.
- FLUSH PRIVILEGES ensures the changes take effect immediately.

Additional Resources:

- CIS MySQL Security Benchmarks (reference security guides that mention these benchmarks): While not directly available online, security guides referencing CIS benchmarks can offer details. Search for "CIS MySQL Security Benchmarks Ubuntu".
- MySQL documentation on user accounts and privileges: https://dev.mysql.com/doc/refman/8.3/en/create-user.html
- MySQL documentation on GRANT statement: https://dev.mysql.com/doc/refman/8.0/en/grant.html

Remember:

- Regularly review and update user accounts and their privileges to ensure they align with changing needs.
- Consider the specific recommendations in the CIS benchmarks for your Ubuntu version for a comprehensive access control strategy.