SSH Key Authentication & Ansible Access

Overview

This project documents the setup and verification of SSH key-based authentication to enable secure, passwordless access for Ansible-managed systems.

Summary

Generated an RSA SSH key pair with ssh-keygen.

Verified the public key in ~/.ssh/id_rsa.pub.

Deployed the key to a remote host using ssh-copy-id.

Confirmed passwordless SSH login.

Validated network configuration after successful access.

Verification

```
ssh egarrido@dev-app-eg3.procore.prod1
ip a
```

Outcome

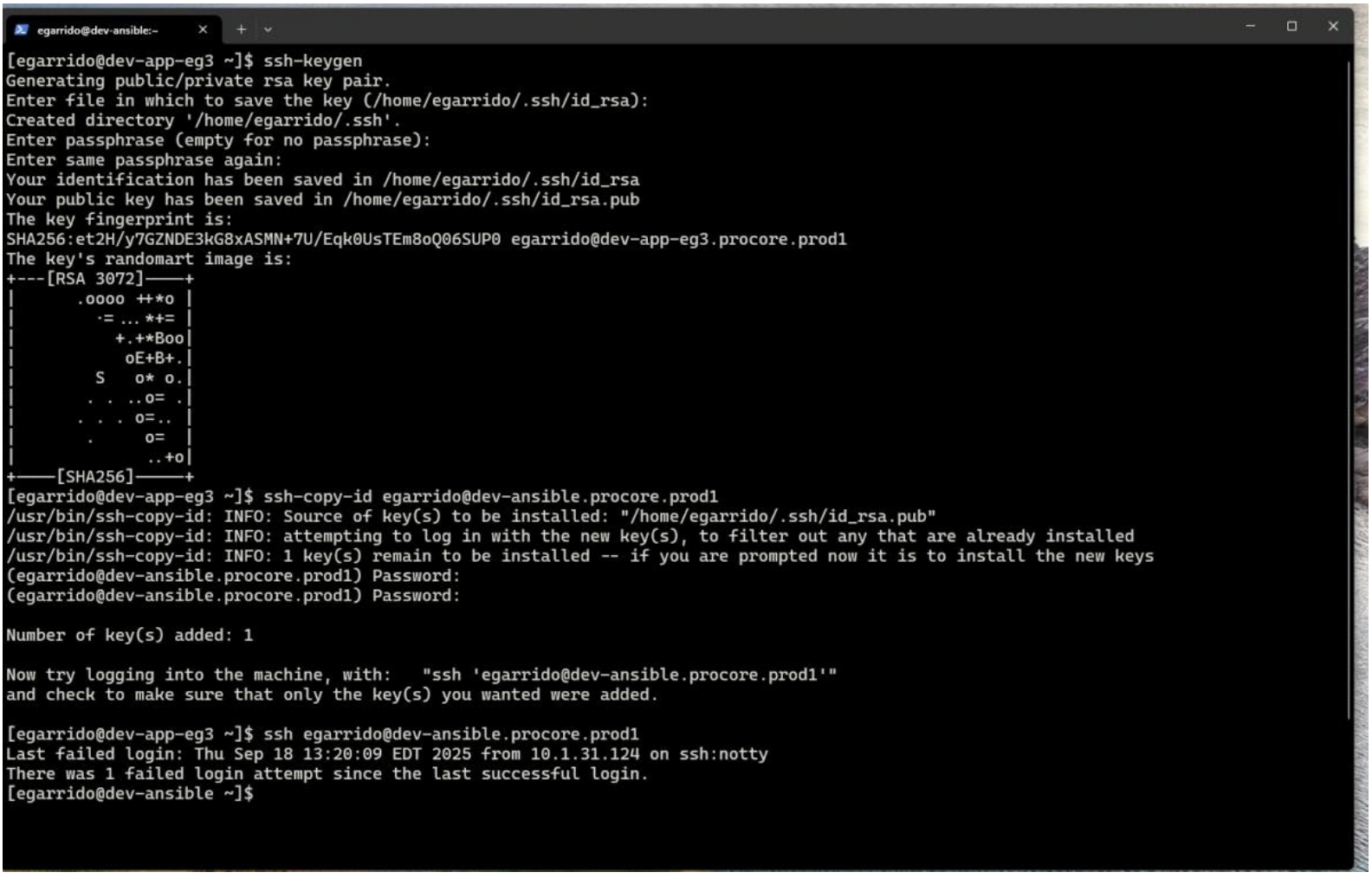
Secure SSH access is established and ready for Ansible automation.

An RSA SSH key pair is generated for the user egarrido using ssh-keygen.

The public key is copied to the remote host dev-ansible.procore.prod1 using ssh-copy-id.

Key-based authentication is successfully installed and confirmed.

Passwordless SSH access to the Ansible host is verified with a successful login.

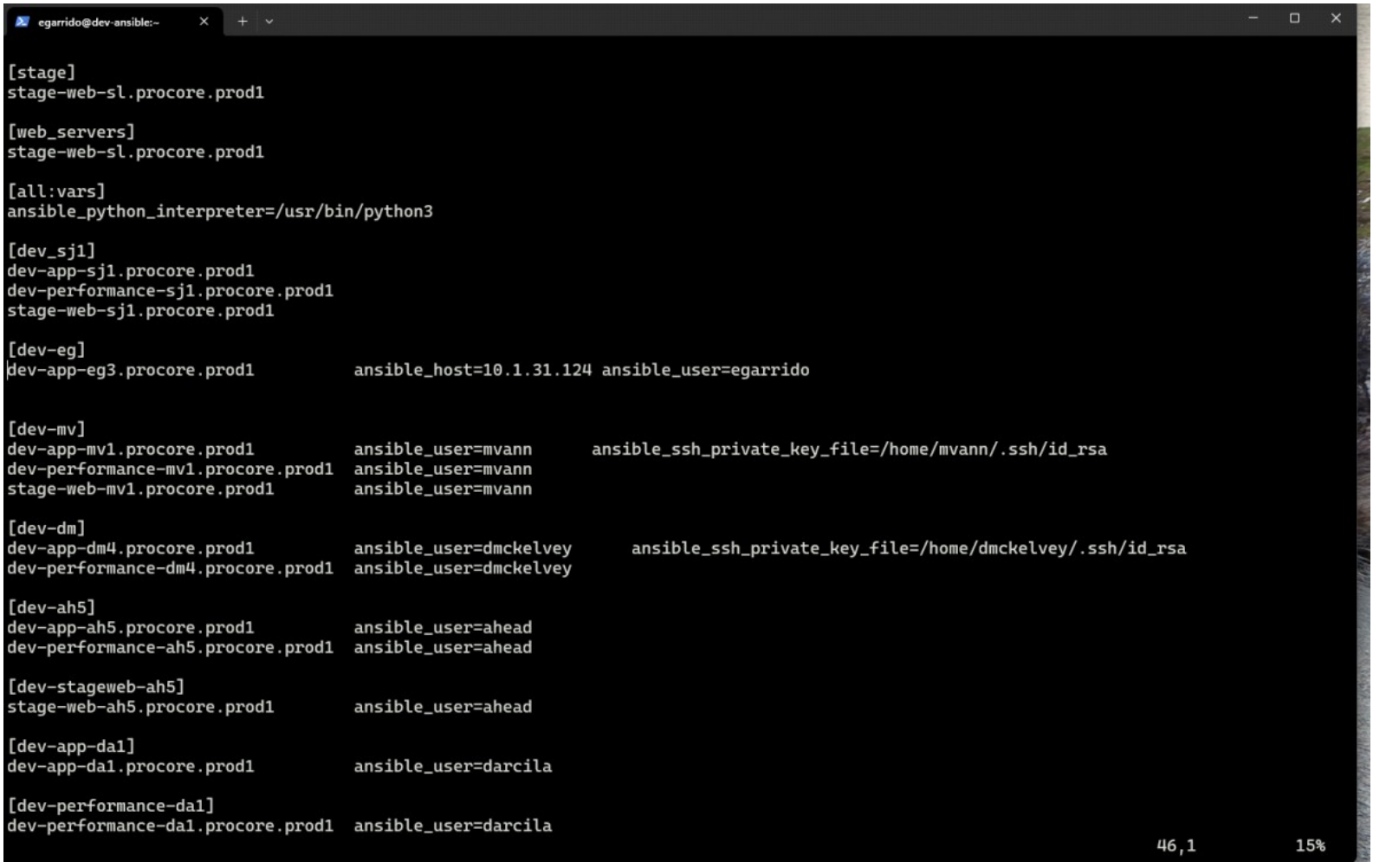


An Ansible inventory file is configured with multiple host groups for stage, dev, and performance environments.

Group variables define the Python interpreter for Ansible execution.

Hosts are mapped with explicit ansible_user, ansible_host, and SSH private key paths where required.

The inventory supports key-based authentication and role separation across environments, enabling consistent Ansible management at scale.

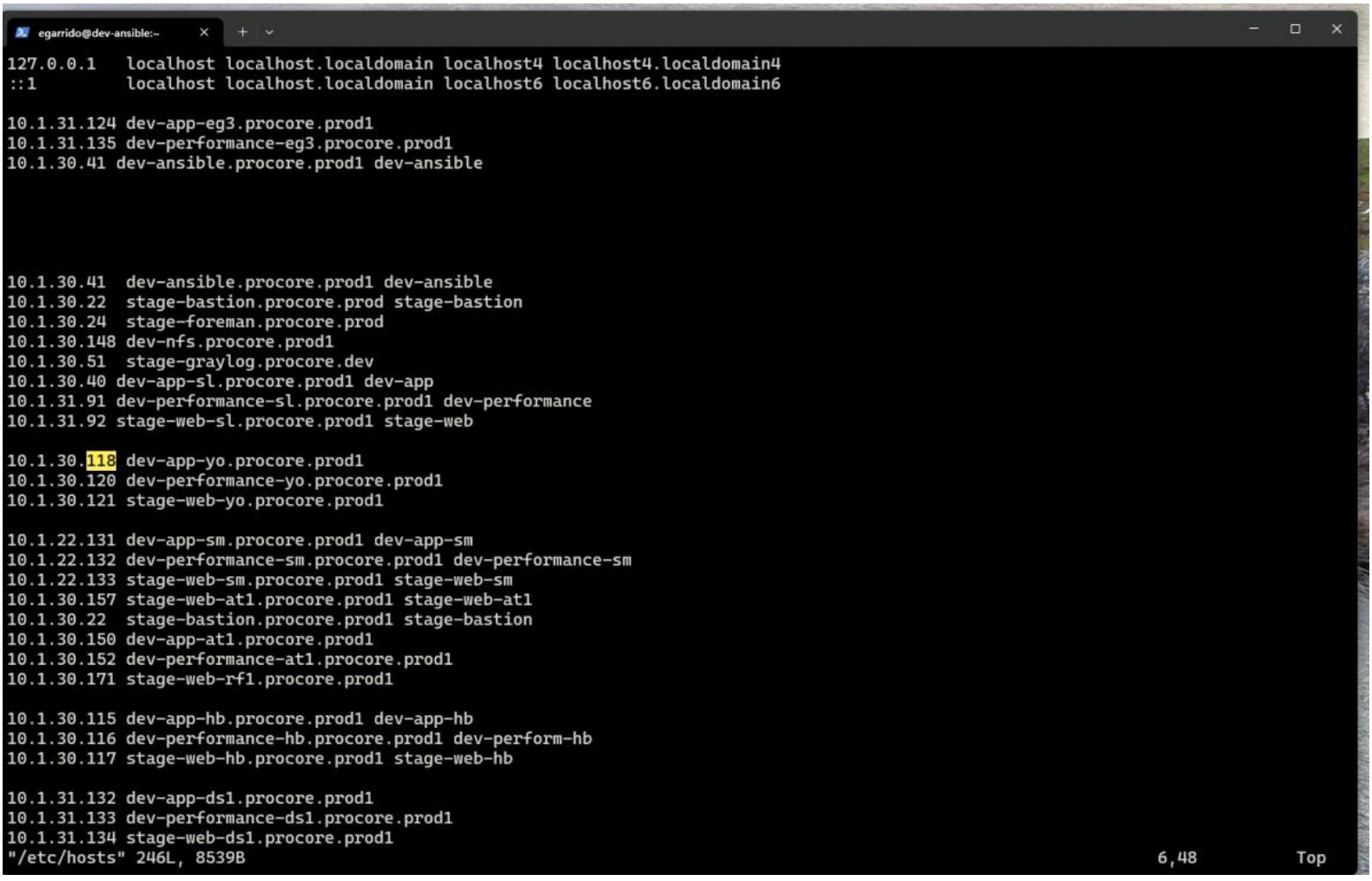


The /etc/hosts file is configured with static hostname-to-IP mappings for dev, stage, and performance systems.

Entries include application servers, web servers, bastion hosts, and supporting infrastructure.

Host aliases are defined to simplify SSH access and Ansible inventory resolution.

Local name resolution ensures reliable connectivity in environments without relying solely on DNS.

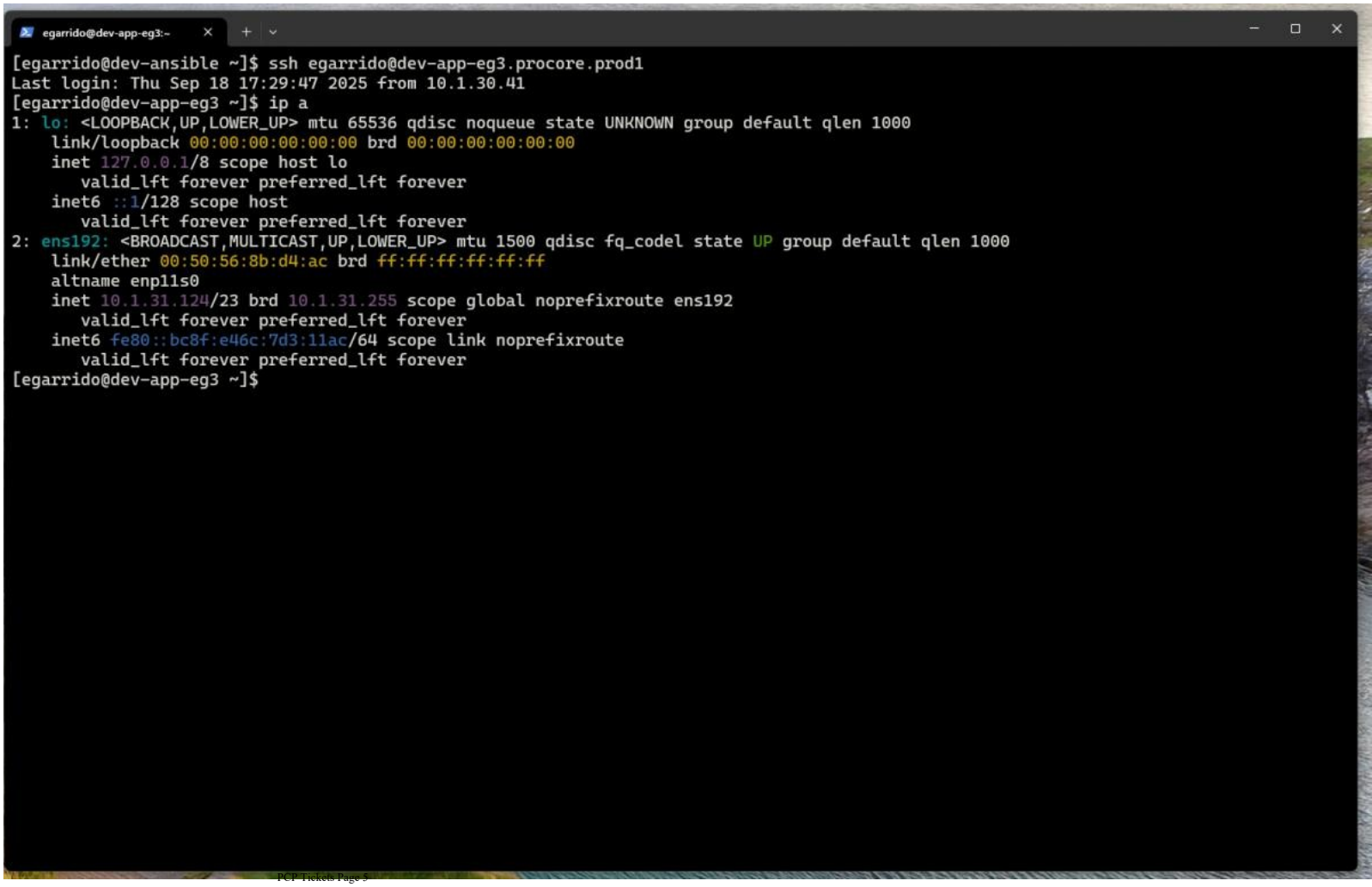


SSH access to dev-app-eg3.procore.prod1 is successful using key-based authentication.

Network interfaces are verified with ip a, confirming the primary interface is UP.

The system is configured with a static IPv4 address 10.1.31.124/23 on interface ens192.

Network configuration is validated, confirming proper connectivity and host accessibility

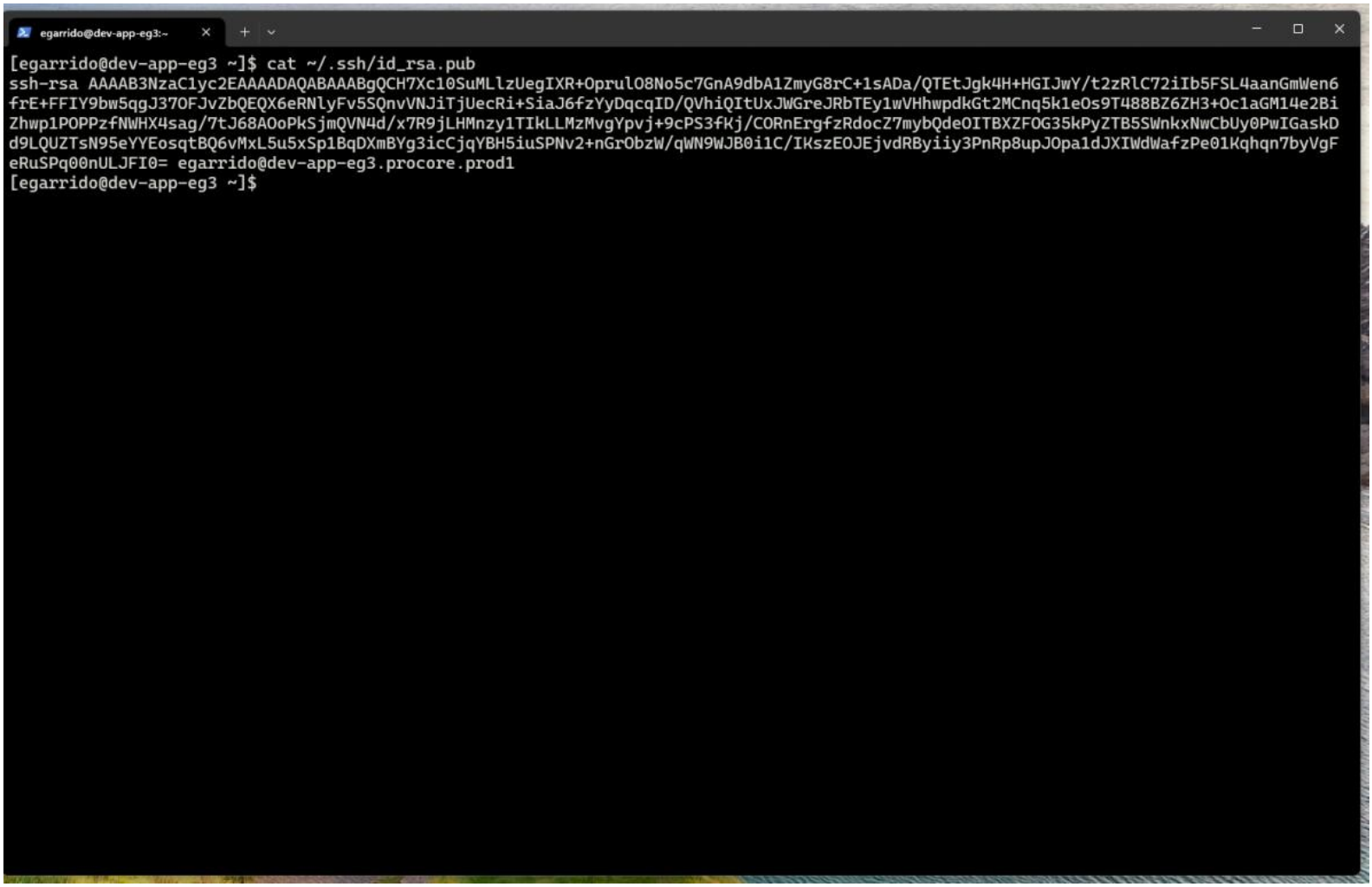


The public SSH key is displayed from ~/.ssh/id_rsa.pub.

The key confirms successful generation of an RSA key pair for the user egarrido.

This public key is used for key-based authentication with remote systems.

Verifies readiness for passwordless SSH access and Ansible automation.

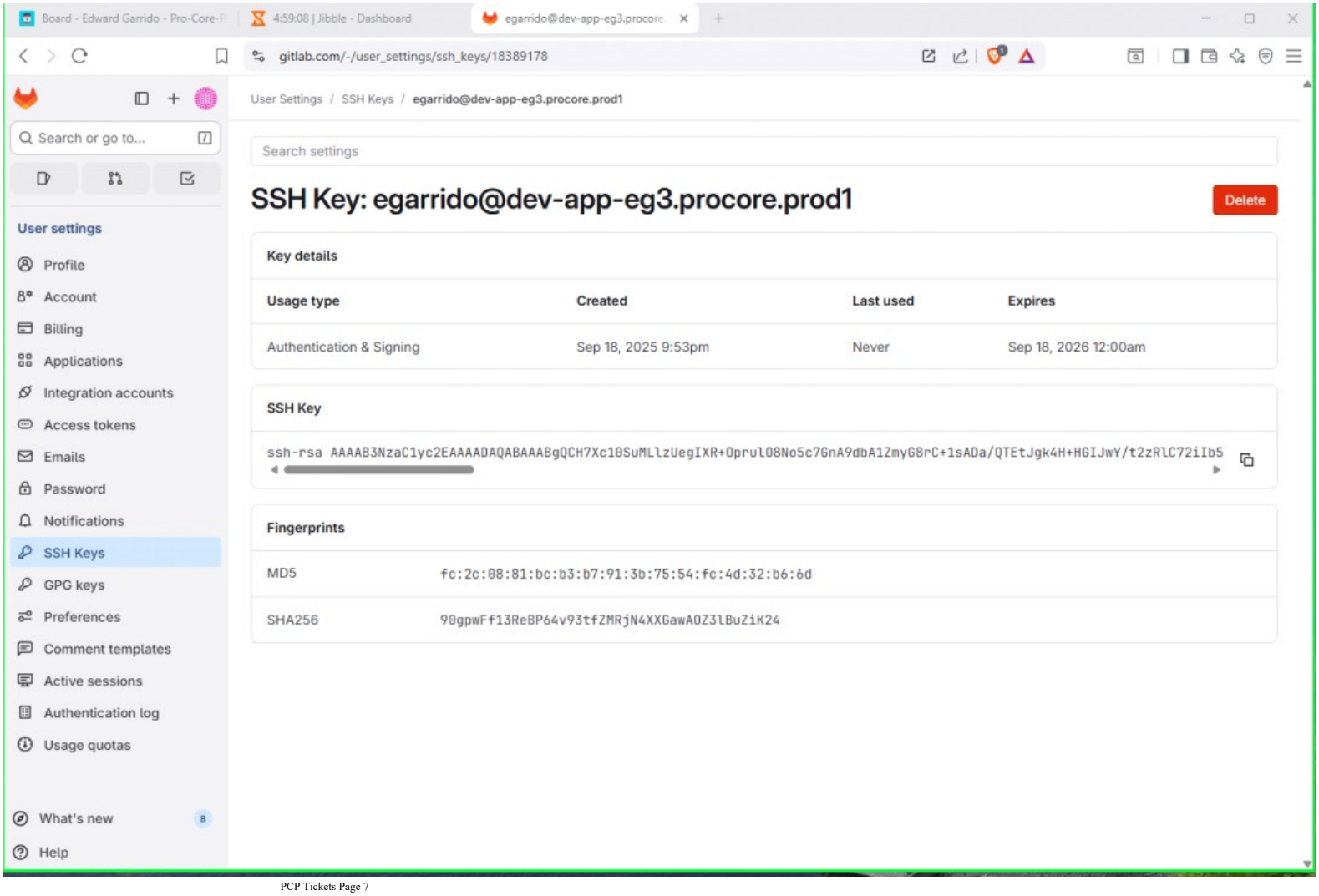


An SSH public key is successfully added to GitLab under user settings.

The key is configured for authentication and signing and is linked to the host dev-app-eg3.procore.prod1.

Fingerprints (MD5 and SHA256) confirm key integrity.

This enables secure, key-based access for Git operations without using passwords.

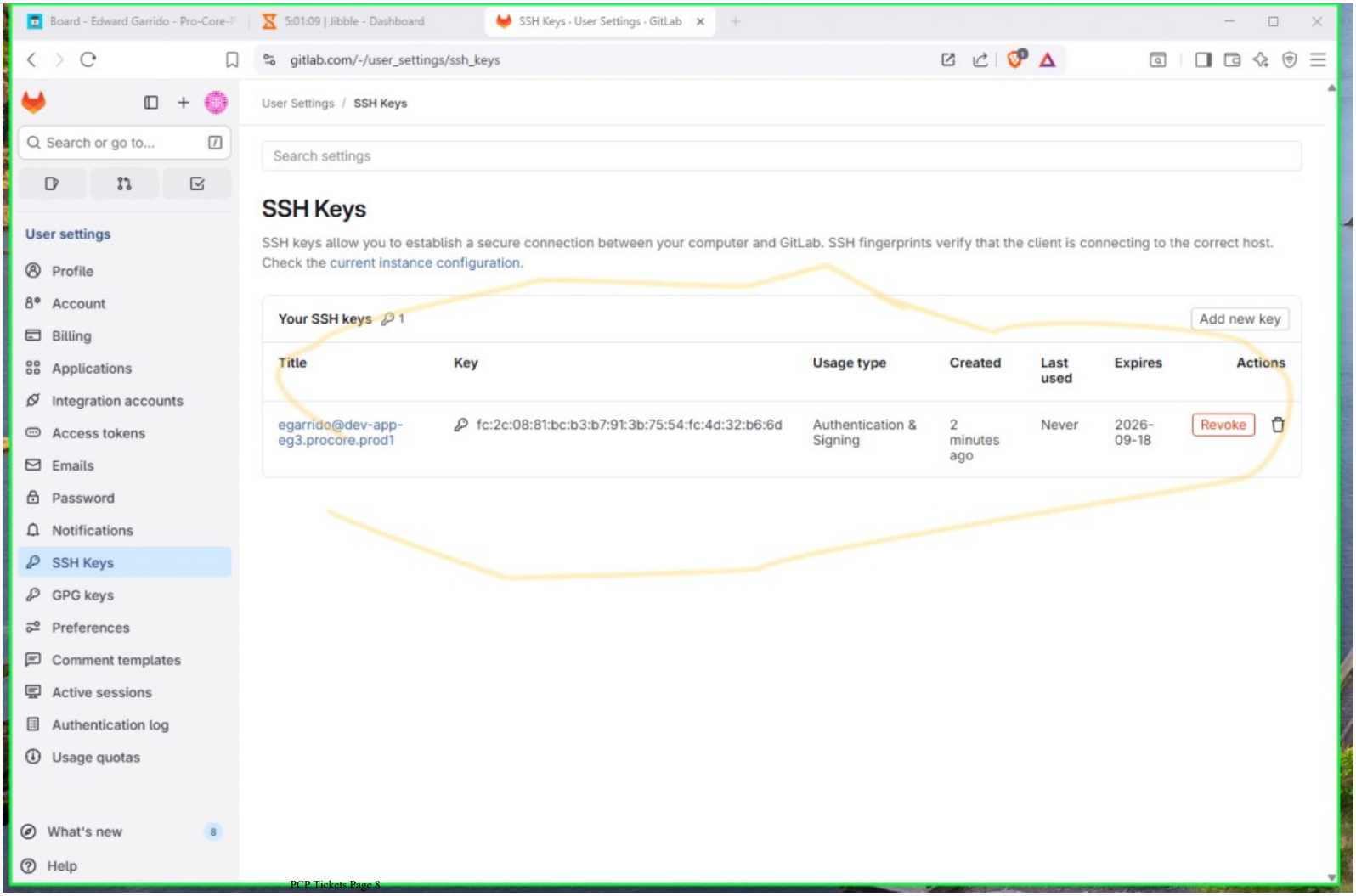


The SSH public key is successfully registered in GitLab under User Settings → SSH Keys.

The key is enabled for Authentication & Signing and associated with egarrido@dev-app-eg3.procore.prod1.

Creation time, expiration date, and fingerprint are visible, confirming proper key registration.

GitLab is now configured for secure, passwordless Git operations using SSH.

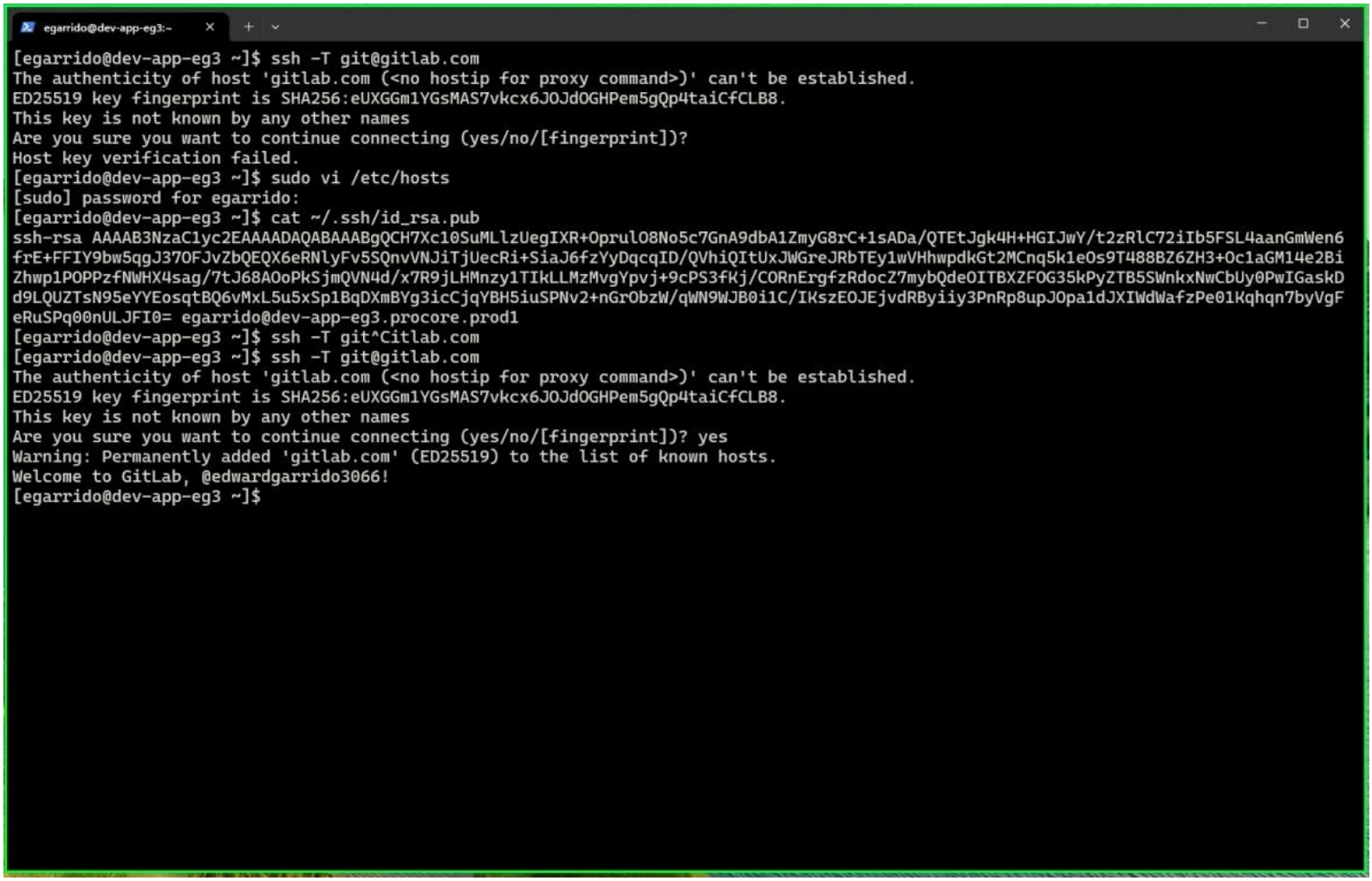


SSH connectivity to GitLab is tested using ssh -T git@gitlab.com.

The GitLab host fingerprint is reviewed and accepted, adding it to the local known_hosts file.

The SSH public key is recognized by GitLab, confirming successful authentication.

A welcome message confirms secure, passwordless SSH access for Git operations.



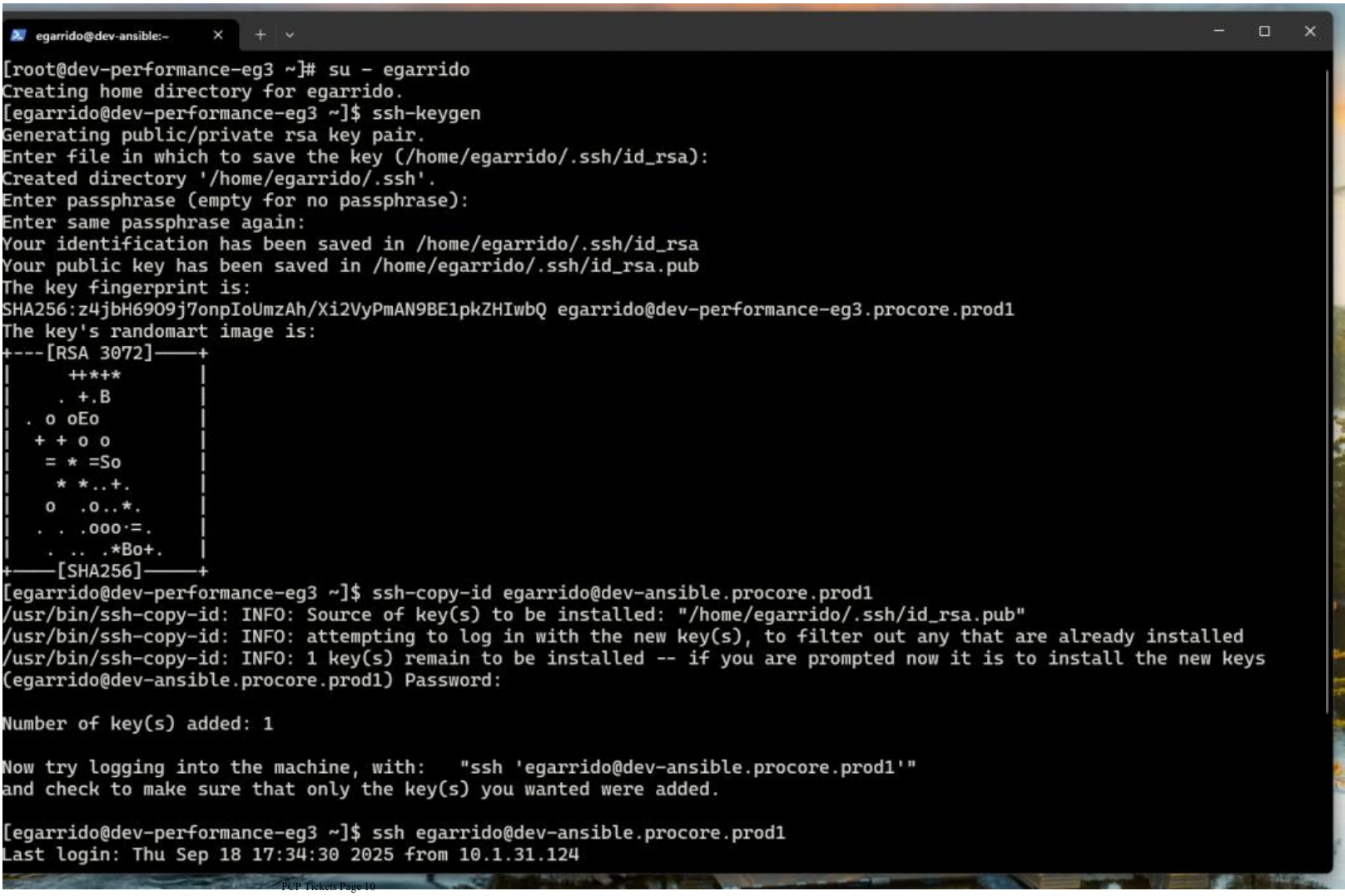
Switched to the egarrido user and created a home directory on the performance host.

Generated a new RSA SSH key pair using ssh-keygen.

Copied the public key to the Ansible control node with ssh-copy-id.

Successfully authenticated to dev-ansible.procore.prod1 using key-based SSH.

Confirms passwordless access from the performance system to the Ansible host, enabling cross-host automation



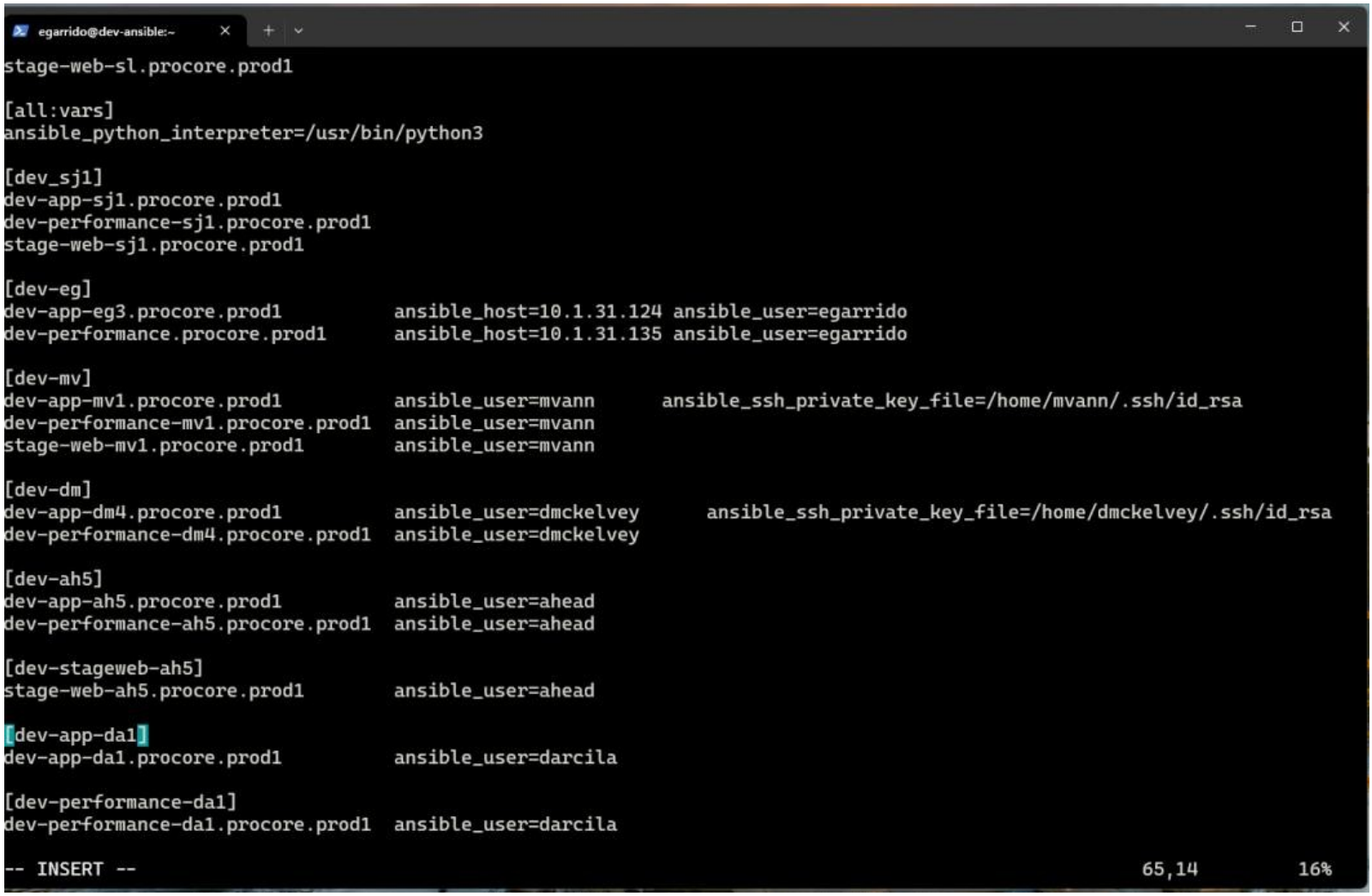
The Ansible inventory is updated to include both development and performance hosts.

Hosts are grouped by environment and role for organized targeting.

User-specific SSH settings (ansible_user, ansible_host, and private key paths) are defined per group.

A global Python interpreter is set to ensure consistent Ansible execution.

The inventory supports secure, key-based access across multiple environments for automation.

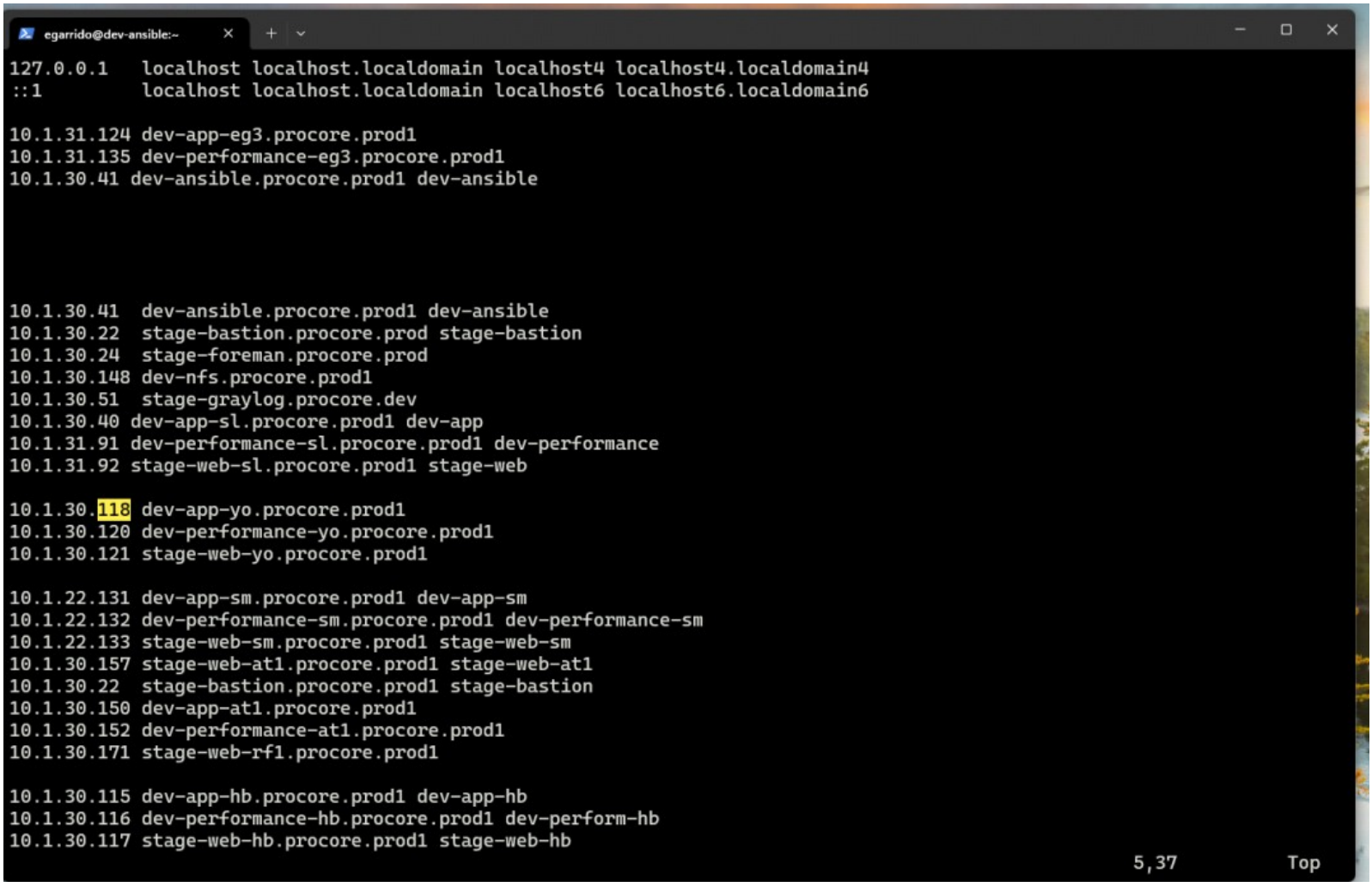


The /etc/hosts file is updated with static IP-to-hostname mappings for dev, stage, and performance environments.

Entries include application servers, web servers, bastion hosts, and supporting infrastructure.

Host aliases are defined to simplify SSH access and Ansible targeting.

Local name resolution is ensured without relying solely on DNS, supporting consistent automation and connectivity

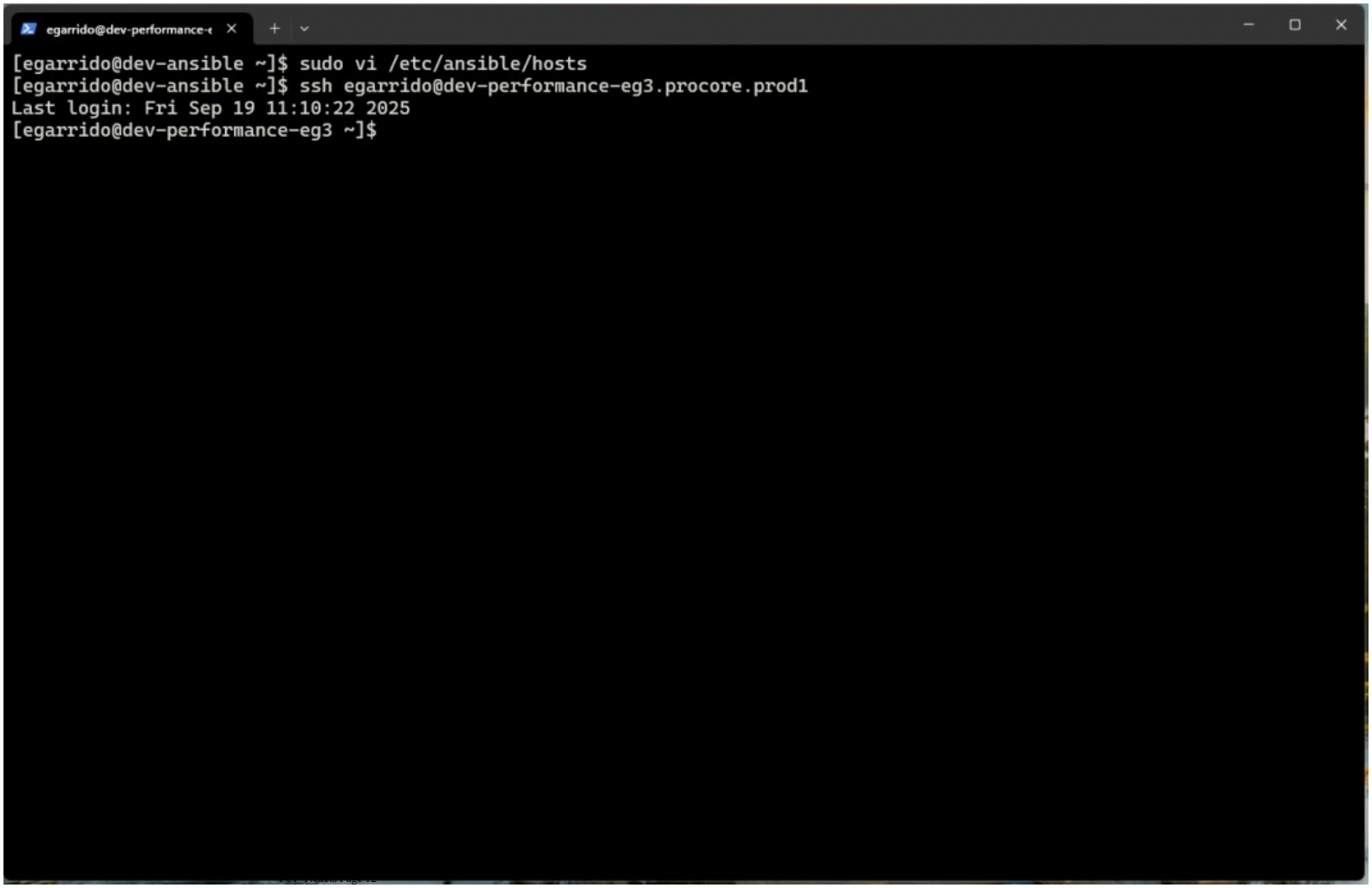


The Ansible inventory file is edited on the control node at /etc/ansible/hosts.

SSH access to dev-performance-eg3.procore.prod1 is tested and succeeds.

Successful login confirms correct inventory configuration and key-based authentication.

Validates end-to-end connectivity from the Ansible control node to a performance host.

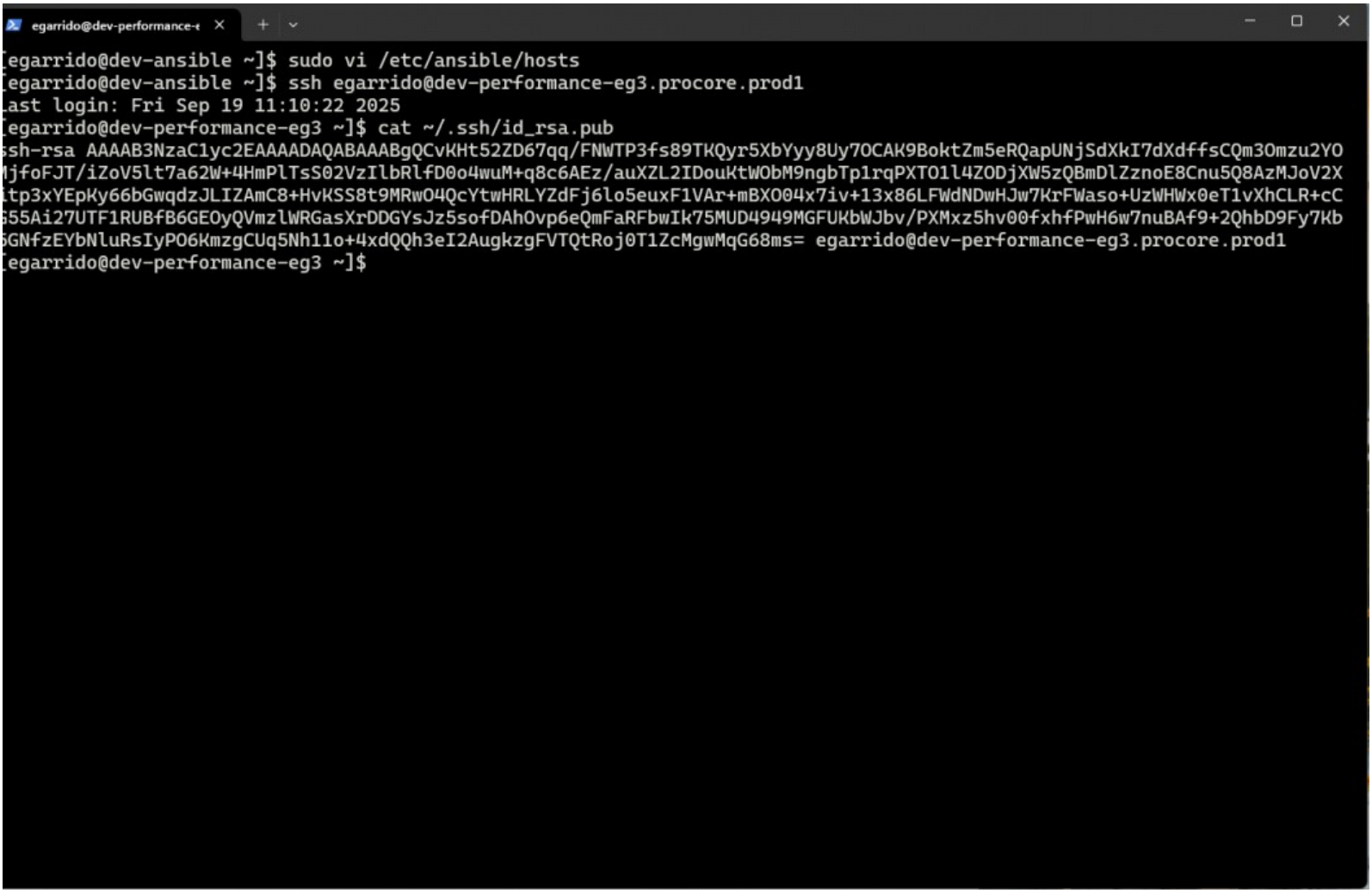


SSH access to dev-performance-eg3.procore.prod1 is confirmed from the Ansible control node.

The public SSH key is verified on the performance host at ~/.ssh/id_rsa.pub.

Key-based authentication is successfully in place for the egarrido user.

Confirms the host is fully prepared for Ansible-managed automation without password prompts

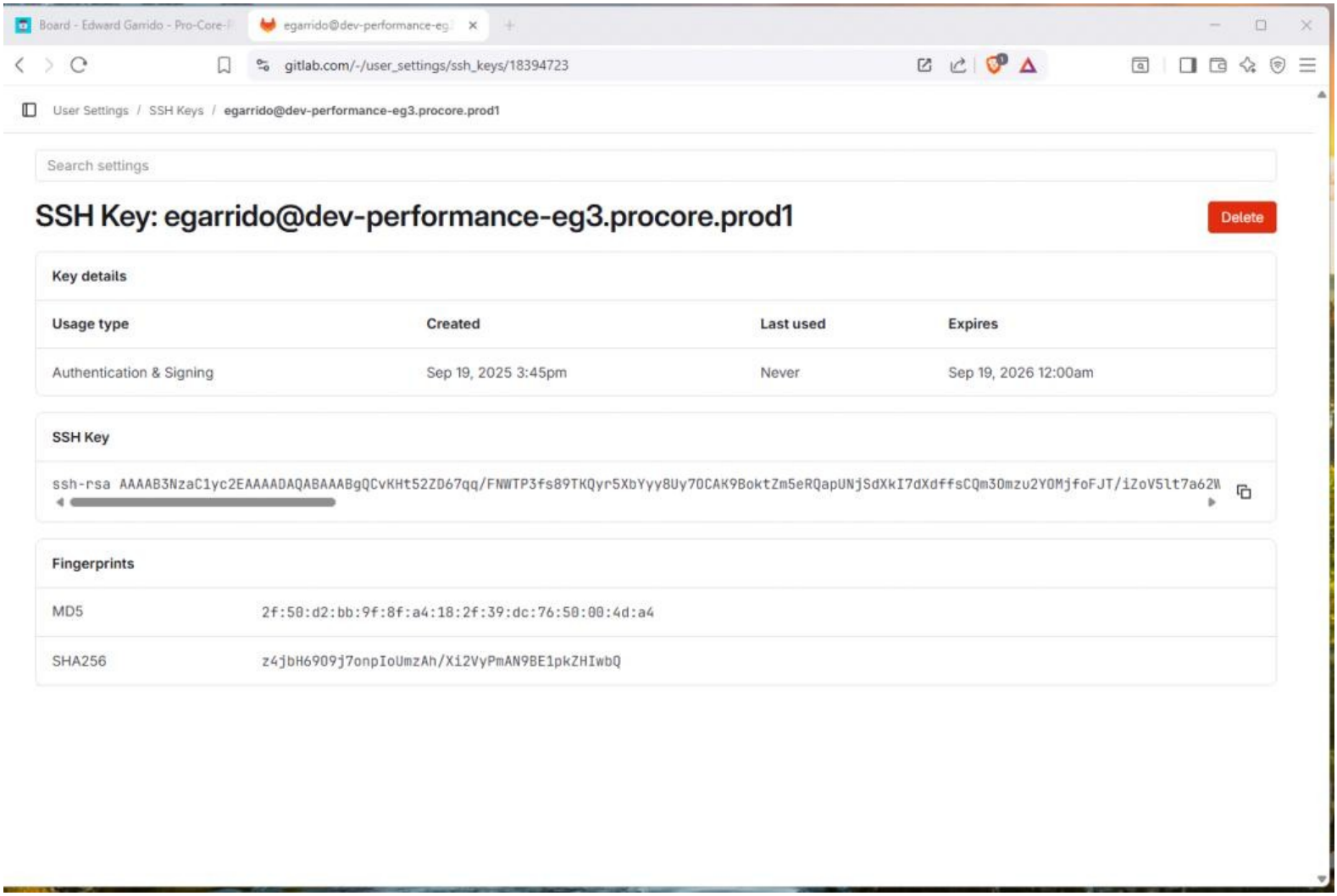


A second SSH public key is successfully added to GitLab for the performance host dev-performance-eg3.procore.prod1.

The key is enabled for Authentication & Signing with a defined expiration date.

Fingerprints (MD5 and SHA256) confirm the key’s integrity.

This completes secure, passwordless Git access from the performance environment, supporting multi-host automation workflows.

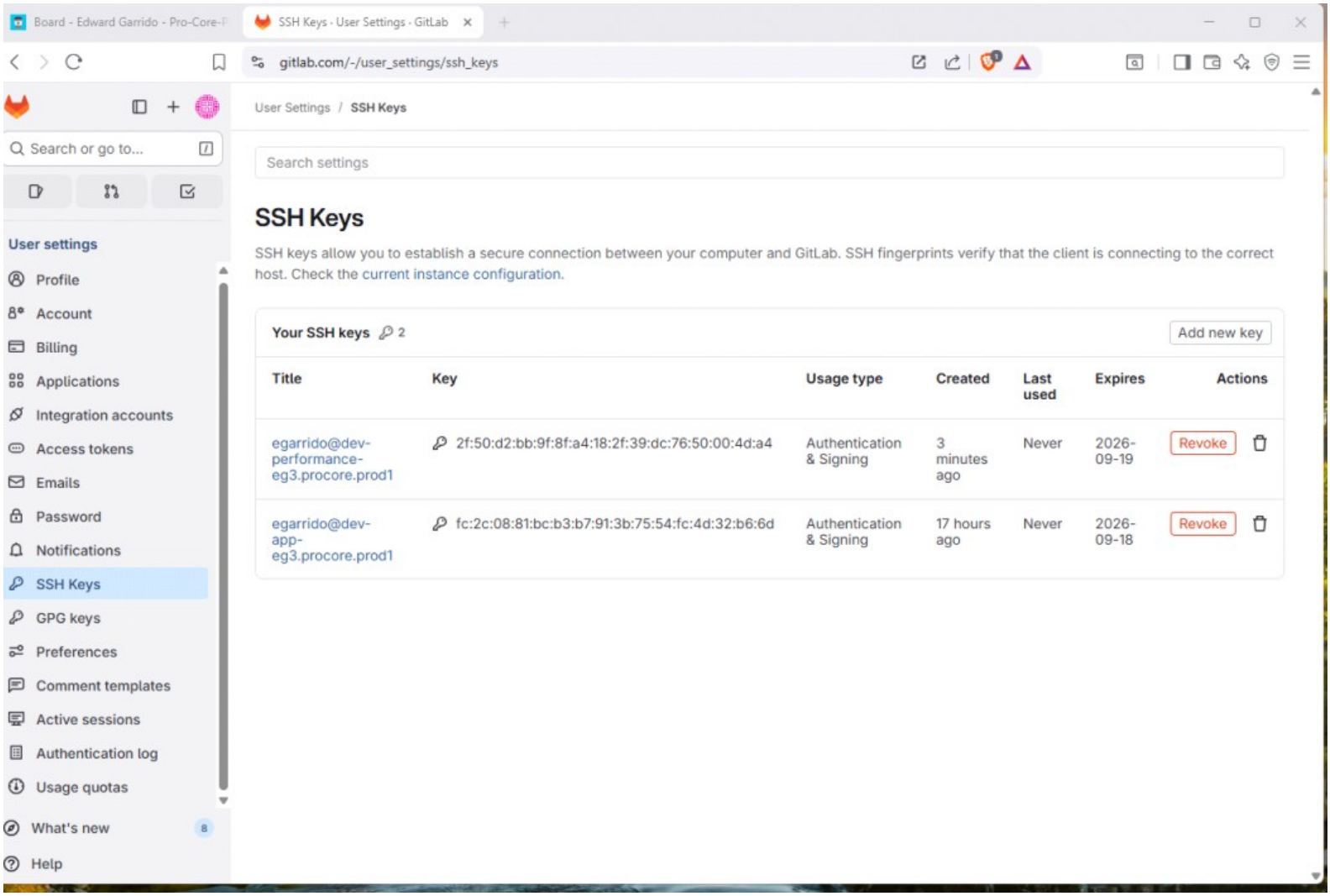


Two SSH public keys are registered in GitLab for separate hosts (dev-app-eg3 and dev-performance-eg3).

Both keys are enabled for Authentication & Signing with defined expiration dates.

Fingerprints confirm key integrity and correct association per host.

GitLab is now configured for secure, passwordless Git access from multiple environments.



Summary (last 15 screenshots):

SSH key pairs were generated on multiple systems to enable secure, passwordless authentication.

Public keys were verified locally and deployed to remote hosts using ssh-copy-id.

Successful SSH logins confirmed key-based access between application, performance, and Ansible control nodes.

Ansible inventory files were edited to include development, stage, and performance hosts with proper user and SSH settings.

/etc/hosts files were updated to provide consistent hostname resolution across environments.

Network configurations were validated using ip a, confirming correct interfaces and static IP assignments.

SSH keys were added to GitLab for both application and performance hosts with authentication and signing enabled.

GitLab SSH connectivity was tested and verified using ssh -T git@gitlab.com.

Multiple SSH keys are now managed centrally in GitLab, each associated with a specific host and expiration date.

End-to-end connectivity is confirmed across systems, ensuring readiness for Ansible automation and secure Git operations.