Hands-on Infrastructure Automation with Terraform on AWS

Alexander Savchuk

Working with Terraform as a team

Section 7

Locking state

Section overview

- Locking state
- Managing secrets securely
- Running Terraform in automation for CI/CD
- Wrap-up and next steps

Section overview

- **Locking state**
- Managing secrets securely
- Running Terraform in automation for CI/CD
- Wrap-up and next steps

Section overview

- Locking state
- **Managing secrets securely**
- Running Terraform in automation for CI/CD
- Wrap-up and next steps

Section overview

- Locking state
- Managing secrets securely
- **Running Terraform in automation for CI/CD**
- Wrap-up and next steps

Section overview

- Locking state
- Managing secrets securely
- Running Terraform in automation for CI/CD
- **Wrap-up and next steps**

Development best practices

- **Consistent style, naming, and file layout**

Development best practices

- Consistent style, naming, and file layout
- **Source control and code reviews for everything**

Development best practices

- Consistent style, naming, and file layout
- Source control and code reviews for everything
- **State file stored remotely using a backend**

State locking

Terraform can lock the state file to prevent concurrent writes, which can make resources orphaned or corrupt the state file.

State locking with S3 backend

Locking is done using a separate DynamoDB table.

```
terraform {
  backend "s3" {
    dynamodb_table = "terraform-lock-table"
  }
}
```

Removing the lock manually

```
terraform force-unlock LOCK_ID
```

Managing secrets securely

Next video