Introduction to Automation and Ansible

A structured enterprise training session for IT professionals — covering IT automation fundamentals, Infrastructure as Code, and Ansible from core concepts to platform-level deployment.

ENTERPRISE TRAINING

1.5 – 2 HOURS

What is IT Automation?

IT Automation is the use of software tools and scripts to automatically perform repetitive IT tasks — replacing manual, error-prone processes with consistent, reliable execution.

Server Provisioning

Spin up servers on demand without manual setup

App Deployment

Push releases consistently across all environments

Config Management

Keep system configurations in sync at scale

Cloud & Security

Automate cloud resources and security patching

Why IT Automation Matters

Manual IT

- Error-prone and inconsistent
- Slow, costly deployments
- Hard to scale beyond small teams
- Relies on tribal knowledge

Automated IT

- Consistent and repeatable
- Faster releases, lower cost
- Scales effortlessly
- Documented and version-controlled

In multi-cloud, AKS/EKS, and microservices environments — automation is **non-negotiable**.

70%
Faster Time to Market

↓Cost
Reduced Ops Spend

99.9%
Improved Reliability

DevOps
CI/CD Enablement

Introduction to Infrastructure as Code (IaC)

Infrastructure as Code (IaC) means managing your servers, networks, and cloud resources using code — not clicking through dashboards or running manual commands. It brings engineering discipline to infrastructure management.

❌ Traditional Approach

"Create 3 Ubuntu VMs, install Nginx, configure the firewall" — typed manually into a terminal, undocumented, unrepeatable.

✅ IaC Approach

```
- name: Install Nginx
 hosts: webservers
 tasks:
 - name: Install package
 apt:
 name: nginx
 state: present
```

Benefits of Infrastructure as Code

Version Control

Store infrastructure definitions in Git — track every change, roll back when needed, and review with pull requests.

Repeatable Environments

Run the same code to create identical Dev, Test, and Prod environments — eliminating "works on my machine" issues.

Disaster Recovery

Rebuild entire infrastructure from code in minutes rather than days — critical for enterprise business continuity.

Audit-Friendly

Every change is tracked and peer-reviewed — making compliance audits and governance straightforward.

Types of IaC Tools

The IaC ecosystem covers the full infrastructure lifecycle. Each tool category solves a distinct problem — and they are often used together in enterprise environments.

Provisioning

Create and destroy cloud resources — VMs, networks, storage. Examples: **Terraform**, **CloudFormation**

Configuration Management

Install packages, manage files, configure services. Examples: **Ansible**, **Chef**, **Puppet**

Image-Based

Build machine images for consistent deployments. Example: **Packer**

Container Orchestration

Manage containerized workloads at scale. Example: **Kubernetes**

Evolution: Manual Configuration to Automation

Phase 1 — Manual Configuration

The starting point for most IT teams: SSH into each server, install packages by hand, edit config files directly. Fast to start, but impossible to scale.

No Documentation

Changes exist only in someone's memory or a wiki that's already outdated.

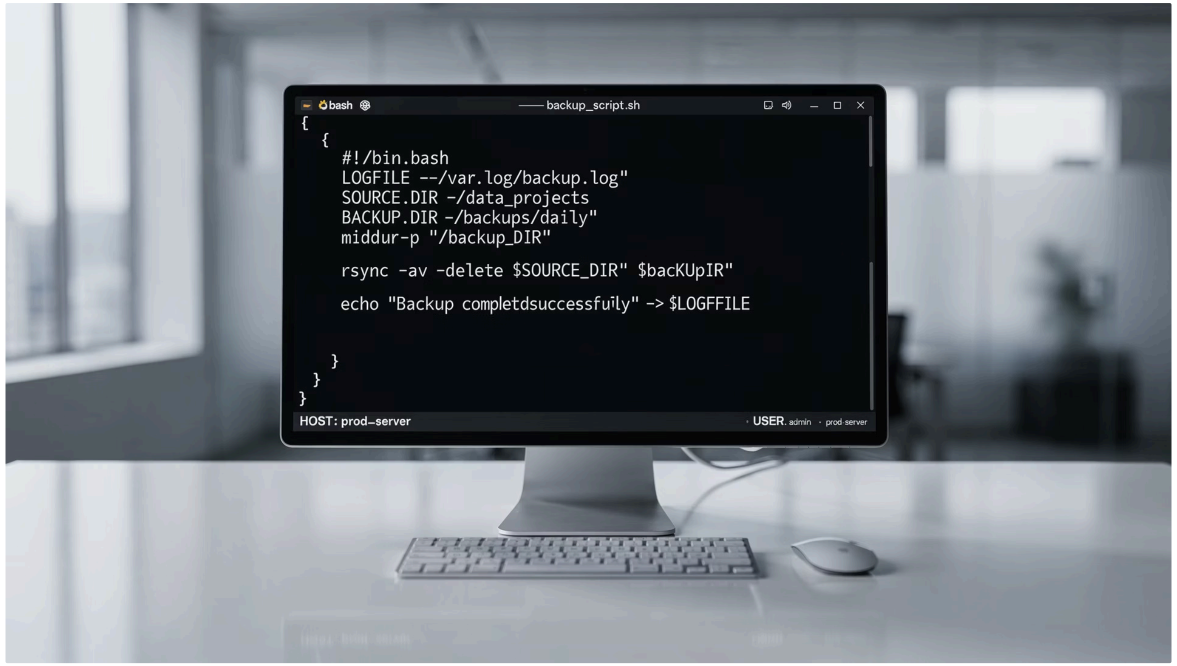
Configuration Drift

Servers diverge over time — each one slightly different, each difference a potential incident.

Environment Mismatch

Dev, QA, and Prod no longer match — bugs appear only in production.

Phase 2 — Shell Scripts



```
{
  {
    #!/bin.bash
    LOGFILE --/var.log/backup.log"
    SOURCE.DIR -/data_projects
    BACKUP.DIR -/backups/daily"
    middur-p "/backup_DIR"

    rsync -av -delete $SOURCE_DIR" $bacKUpIR"

    echo "Backup completdsuccessfuïly" -> $LOGFFILE


   }
  }
}
HOST: prod–server                              USER. admin · prod-server
```

An Improvement — But Not Enough

Bash and PowerShell scripts brought a degree of automation and repeatability. Teams could finally hand off tasks to a script instead of a person.

The limitations quickly emerged:

- **Hard to maintain** — scripts grow complex and fragile over time
- **Not idempotent** — running twice can break things
- **Platform-specific** — Bash won't run on Windows; PowerShell won't run on Linux without extra setup

Shell scripts solve individual problems. Configuration management tools solve systemic ones.

Phase 3 — Configuration Management Tools

The modern answer to the complexity and fragility of manual work and shell scripts. Configuration management tools bring structure, reliability, and scalability to infrastructure operations.

Declarative Approach

You describe the *desired state* — the tool figures out how to get there.

Idempotent Execution

Run the same playbook 10 times — the result is always the same, no side effects.

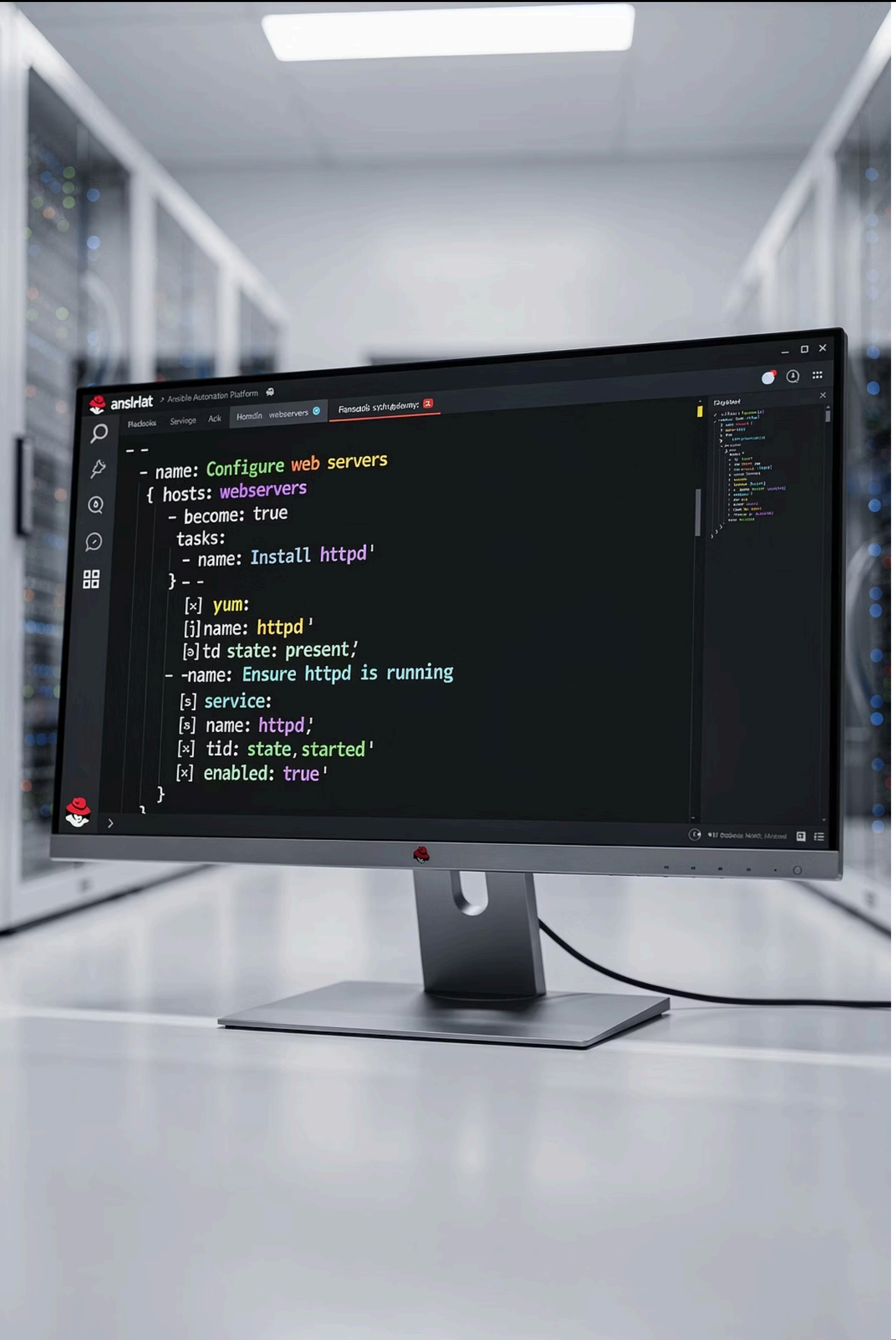
Centralized Orchestration

Manage hundreds of servers from a single control node.

Agentless Option

No software to install on target servers — **Ansible** uses standard SSH.

This is where **Ansible** enters the picture — the most accessible and widely adopted configuration management tool in the enterprise.



```
- -
 - name: Configure web servers
  { hosts: webservers
    - become: true
     tasks:
      - name: Install httpd'
    } - -
      [x] yum:
      [j] name: httpd '
      [a] td state: present,'
   - -name: Ensure httpd is running
     [s] service:
     [s] name: httpd,'
     [x] tid: state, started'
     [x] enabled: true'
    }
```

What is Ansible?

Ansible is an open-source IT automation engine that uses human-readable YAML playbooks to automate configuration, deployment, provisioning, and orchestration — without requiring any agents on target systems.

Agentless

Communicates over SSH (Linux) or WinRM (Windows) — nothing to install on managed nodes.

YAML-Based

Playbooks are written in plain YAML — readable by humans, maintainable by teams.

Idempotent

Re-running a playbook produces no unintended changes if the system is already in the desired state.

Easy to Learn

Low barrier to entry — most engineers are productive within hours of first contact.

How Ansible Works

Control Node

Runs Ansible and executes playbooks

Inventory

Grouped list of managed hosts and roles

Playbook

YAML script defining automation tasks

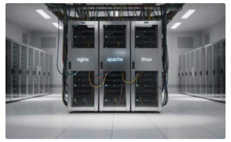
Modules

Reusable units like apt, copy, service

Ansible's architecture is intentionally simple: a single control node connects to managed nodes over standard protocols, reads from an inventory file, and executes tasks defined in YAML playbooks using reusable modules — no daemons, no databases, no agents.

Common Ansible Use Cases

Ansible covers the full DevOps lifecycle — from bare-metal provisioning to Kubernetes Helm deployments. In cloud-heavy environments (Azure, AWS, Kubernetes), it serves as a universal automation layer across providers.

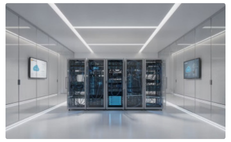


Server Provisioning

Install and configure Apache or Nginx across 100 servers in a single playbook run.

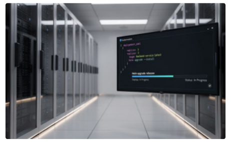
App Deployment

Deploy .NET or Java applications to Linux servers with zero manual steps.



Cloud Automation

Provision and configure Azure VMs, AWS EC2 instances, and cloud networking resources.



Kubernetes & Security

Deploy Helm charts, enforce firewall rules, and configure Cisco network devices.

Ansible Core vs. Ansible Automation Platform

Ansible comes in two flavors. Understanding which version fits your environment is essential for designing a scalable automation strategy.

Ansible Core

The open-source foundation. Everything starts here.

- Command-line interface (CLI)
- YAML Playbooks
- 500+ built-in modules
- Inventory management
- Roles and collections

Best for: developers, small teams, learning, and basic automation projects.

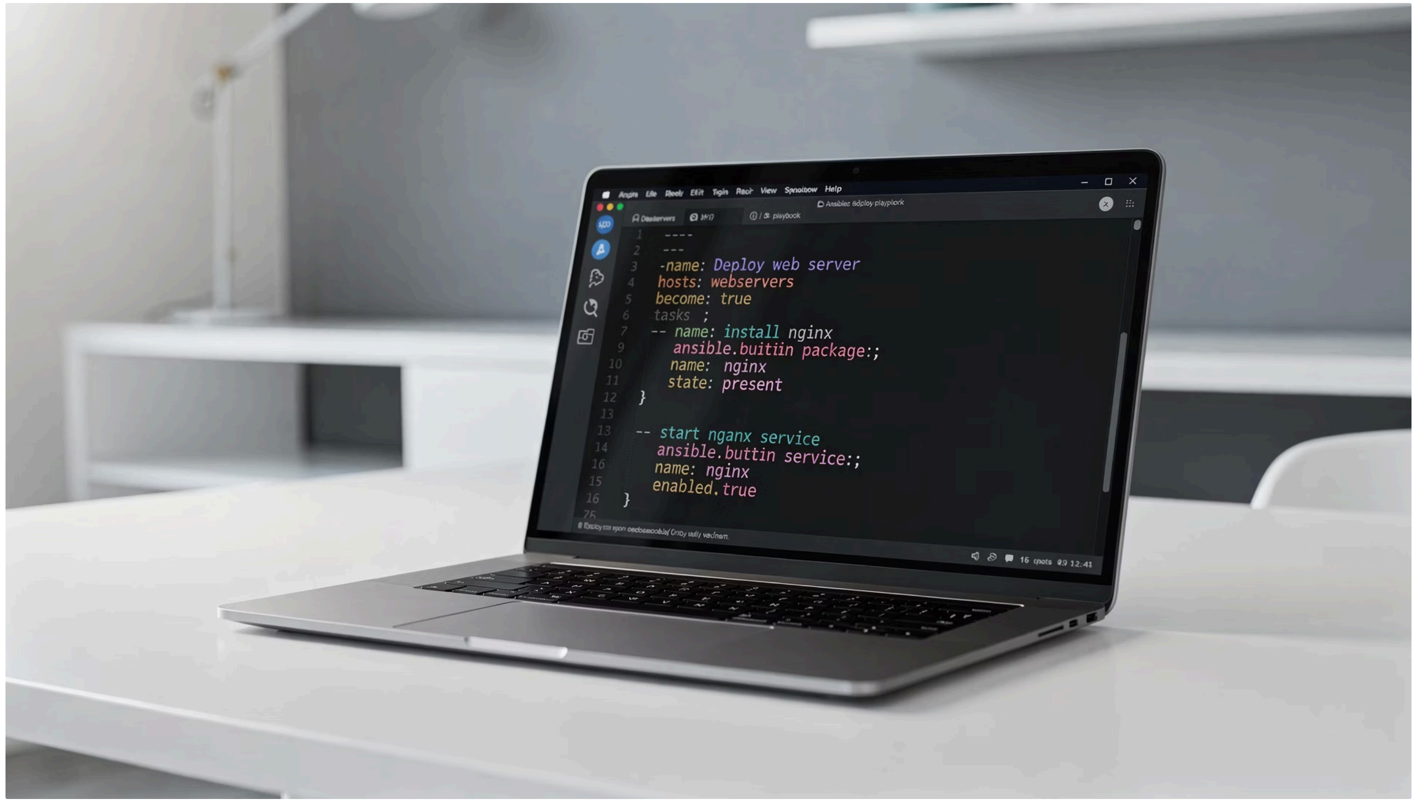
Ansible Automation Platform (AAP)

The enterprise edition from **Red Hat** — built for scale, governance, and compliance.

- Web UI (Automation Controller)
- Role-Based Access Control (RBAC)
- Workflow automation engine
- API integration and scheduling
- Central logging and Automation Hub

Best for: large enterprises, regulated industries, multi-team environments.

Ansible Core — Deep Dive



```
----
---
-name: Deploy web server
hosts: webservers
become: true
tasks  ;
-- name: install nginx
  ansible.buitiin package:;
  name:  nginx
  state: present
}

-- start nganx service
  ansible.buttin service:;
  name: nginx
  enabled. true
}
```

The Open-Source Foundation

Ansible Core is the engine that powers everything. It is free, community-supported, and includes everything you need to start automating immediately.

CLI-First

Run playbooks from any terminal with `ansible-playbook`

Rich Module Library

500+ modules covering cloud, OS, network, and security tasks

Galaxy & Collections

Share and reuse roles from Ansible Galaxy community

Feature Comparison

Choose the right tool for your scale and governance requirements.

Feature	Ansible Core	Automation Platform
Command-Line Interface	✅ Yes	✅ Yes
Web UI (Controller)	❌ No	✅ Yes
Role-Based Access Control	❌ No	✅ Yes
Workflow Automation Engine	❌ No	✅ Yes
Credential Vault	❌ No	✅ Yes
Audit & Compliance Logs	Limited	✅ Advanced
Enterprise Support (Red Hat)	❌ No	✅ Yes

When to Use Ansible Core

Ansible Core is the right choice when:

→ **Learning Automation**

Ideal for building foundational skills without infrastructure overhead.

→ **Small Team or Solo Use**

A single operator or small team sharing playbooks via Git is perfectly served.

→ **Simple, Scoped Projects**

Automating a single application stack or internal tooling doesn't need an enterprise platform.

→ **Budget Constraints**

Zero licensing cost — full power of Ansible at no charge.

When to Use Ansible Automation Platform

When your automation spans teams, clouds, and compliance boundaries, the CLI alone is no longer sufficient.

Large Enterprise

Hundreds of servers across multiple business units and environments

Multiple Teams

Separate Dev, QA, and Ops teams with different permissions and ownership

RBAC & Approvals

Change management gates require human approval before automation runs

Audit Requirements

Every execution is logged — who ran what, when, and with what result

CI/CD Integration

Trigger Ansible jobs via API from Jenkins, GitHub Actions, or Azure DevOps

Scheduling

Run automation jobs on defined schedules without human intervention

Real Enterprise Scenario

Banking or Healthcare Environment

Imagine managing **500+ servers** across Dev, QA, and Prod for multiple application teams — with strict audit trails, separation of duties, and no tolerance for unauthorized changes.

Credential Vault

No plaintext passwords. Secrets are stored securely and never exposed to operators directly.

Approval Workflows

Production changes require a lead engineer's sign-off before the job executes.

Full Audit Logging

Every job execution is recorded — timestamp, user, parameters, and output — for compliance review.

Role-Based Access

Developers can run dev playbooks; only Ops can trigger production jobs. Access is enforced, not trusted.

In regulated industries, **Ansible Automation Platform** isn't a luxury — it's the only viable path to compliant, auditable automation at scale.

Session Summary

Everything covered in this session — from first principles to platform selection — builds toward one goal: reliable, scalable, auditable automation in the enterprise.

IT Automation

Replaces manual, error-prone tasks with consistent, software-driven execution

Infrastructure as Code

Infrastructure defined, versioned, and deployed as code — enabling repeatability and auditability

Evolution of Automation

Manual → Shell Scripts → Configuration Management — each phase solving the limitations of the last

Ansible

Agentless, YAML-based, idempotent automation engine — the most accessible tool in the IaC ecosystem

Ansible Core vs. AAP

Core for learning and small teams; Automation Platform for enterprise governance, RBAC, and compliance

What's Next?

This session is the foundation. The real learning happens in the lab. Choose your next step based on your training objective:

🧪 Hands-On Lab

Install Ansible and automate your first Ubuntu VM — ideal for engineers new to the tool

☁️ Cloud Lab

Enterprise Use Case: automate across AWS, Azure, and Kubernetes in a single playbook workflow

🗓️ ILT Lab Plan

Full 3-hour instructor-led session plan with exercises, discussion points, and assessments

📊 Training Deck

PPT-ready slide content optimized for live training delivery and classroom presentation

Define your training objective and the next session will be tailored to your team's specific environment and goals. 🎯