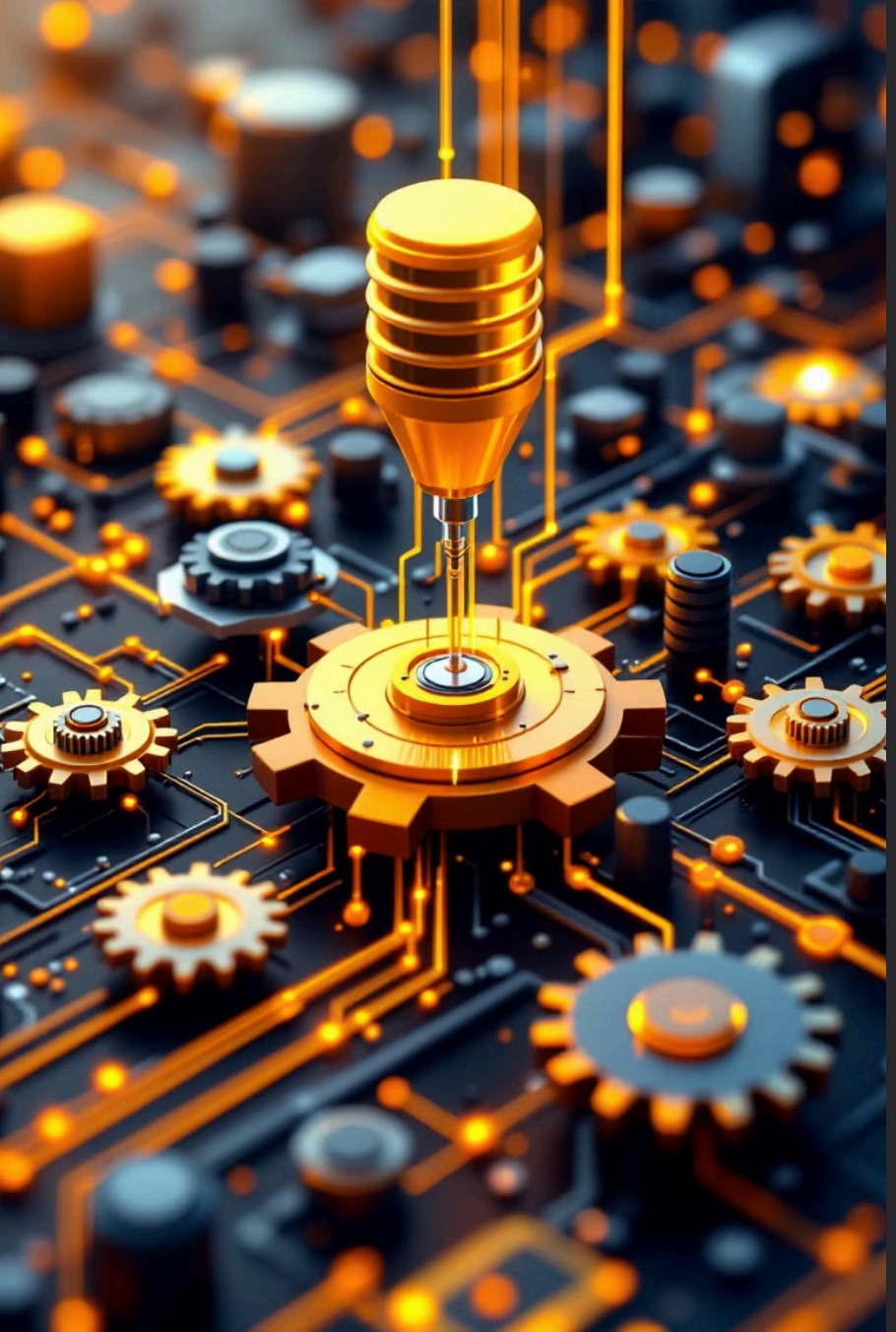


# Configuration Management: introduzione ad Ansible

## Hands On Workshop

by Evanghelos Papaioannu

# Welcome and pleased to meet you!

- Let's make it easy: Evan
- Worked in some big companies (VF, IBM, YNAP)
- Mostly in Operations and mission critical environments
- Linux System Administrator > DevOps > SRE
  - System Design
  - Observability
  - AI
- Love outdoor life with family
- Loved to hike
- Loved to bike
- Loved to travel

# What is Configuration Management?

Configuration management tools automate and standardize system setups across your infrastructure.

## Consistency

Identical environments across development, testing, and production reduce "works on my machine" issues.

## Version Control

Track and roll back changes with complete infrastructure history.

## Scalability

Deploy coherent configurations to thousands of servers with minimal effort.

Cattle vs Pet.

## Compliance

Enforce security policies and meet regulatory requirements consistently.

# Configuration Management in the IT Lifecycle

Configuration Management bridges development and operations, enabling infrastructure consistency throughout the entire IT lifecycle.

### Infrastructure as Code

Define infrastructure through version-controlled code rather than manual processes.

### Iteration & Evolution

Refine infrastructure based on operational insights and new requirements.

### Configuration Management

Maintain system state and enforce desired configurations across environments.

### Continuous Deployment

Automate testing and deployment of infrastructure changes.

### Monitoring & Feedback

Track performance and compliance to inform optimization efforts.

# Cognitive Load in IT and Automation Benefits

## The Cognitive Burden

Managing complex IT infrastructures requires constant mental effort and attention.

- Manual configurations demand deep technical knowledge
- Tracking changes across environments causes mental fatigue
- Context-switching between tools fragments attention

## Automation Relief

Configuration management tools like Ansible offload repetitive mental tasks.

- Codified processes require less working memory
- Templated solutions reduce decision fatigue
- Self-documenting infrastructure enhances knowledge sharing

## Operational Benefits

The cognitive savings translate to tangible IT lifecycle improvements.

- More time for innovation and strategic thinking
- Reduced errors in critical environments
- Faster response to changing requirements

# Configuration Management Alternatives

## Chef

Ruby-based tool with a focus on continuous automation.

- Uses "recipes" and "cookbooks"
- Pull architecture
- Steep learning curve

## Puppet

Model-driven approach with declarative language.

- Custom DSL syntax
- Strong reporting tools
- Mature enterprise support

## SaltStack

Python-based with event-driven automation.

- High-speed communication
- YAML for configurations
- Strong remote execution

## Ansible

Agentless with simple YAML syntax.

- No agents required
- Low learning curve
- SSH-based communication

While **IaC tools like Terraform (OpenTofu) and Pulumi** focus on provisioning infrastructure (creating servers, networks), **configuration management** tools maintain the state of existing resources. IaC builds the foundation; CM tools shape what runs on it.

# Key Ansible Features

## Agentless

Uses SSH (Linux) or WinRM (Windows) to execute tasks remotely without requiring agents on managed nodes.

## Idempotency

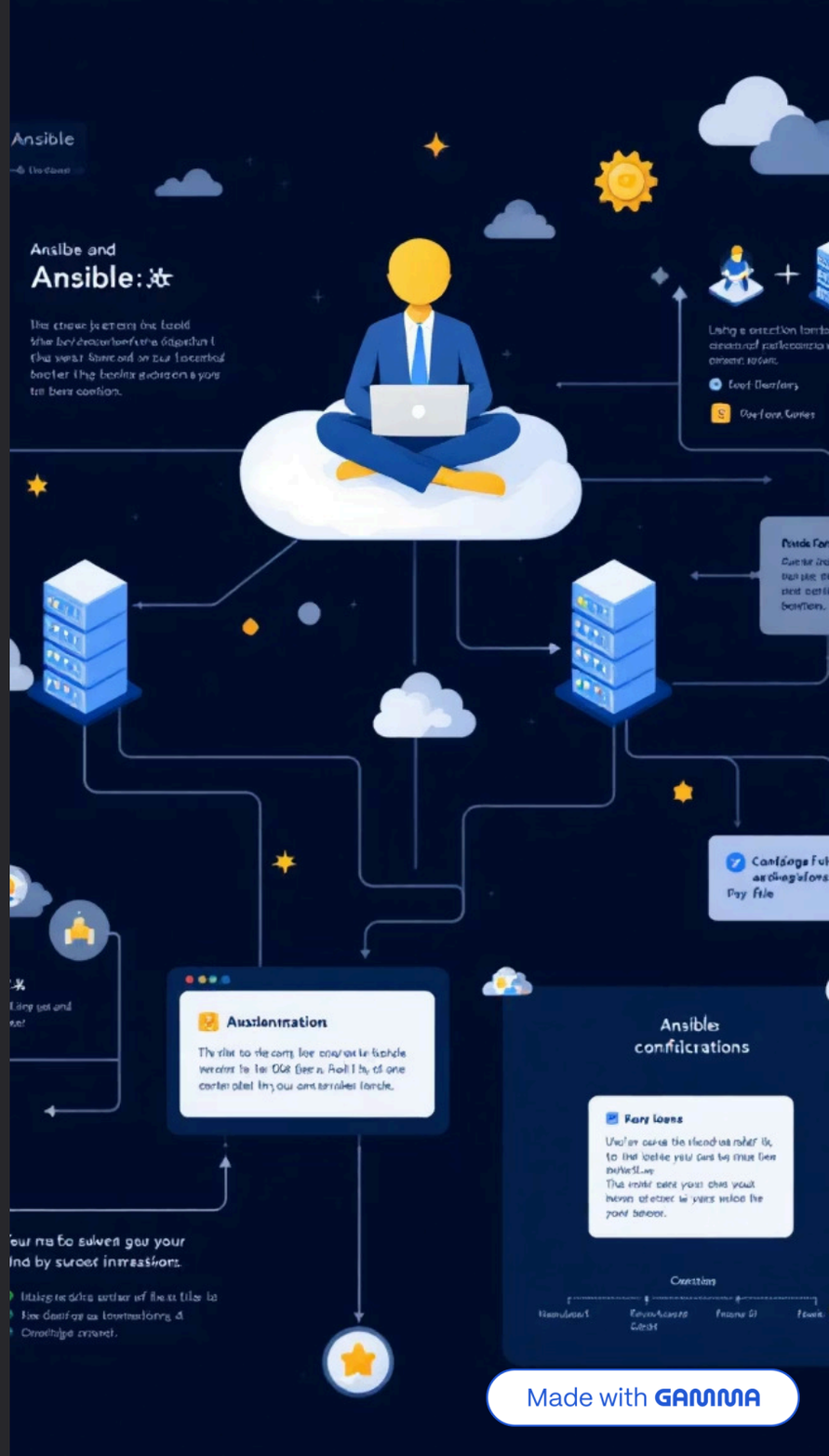
Running a playbook multiple times won't cause unintended side effects; changes are only applied if needed.

## Inventory Management

Manages dynamic and static inventories from various sources like cloud providers or databases.

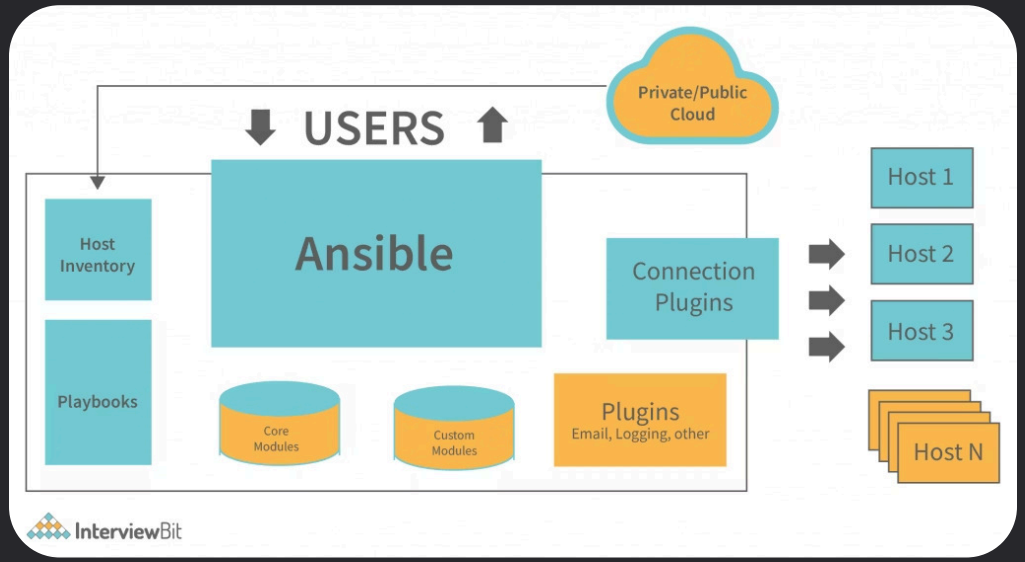
## Event-driven Automation

Responds to events and integrates with tools like AWX/Ansible Tower for automation workflows.



# Ansible Architecture Overview

- Inventory
- Tasks
- Playbook
- (Roles)
- (Modules)



# Hands on!

git clone https://github.com/ebah80/lab_ansible.git

# Understanding Ansible Inventory

## Inventory Formats

- INI format
- YAML format
- JSON format

Inventories help manage servers at scale - from tens to thousands of servers.

## Grouping Servers

Servers can be organized in groups like:

- [web] for web servers
- [db] for database servers
- [demo] for testing environments

# Ansible Facts

## System Properties

Automatically gathered system information including OS, network interfaces, CPU, and memory details.

## Gathering Facts

Use the setup module:

ansible -i hosts hostname -m setup

## Filtering Facts

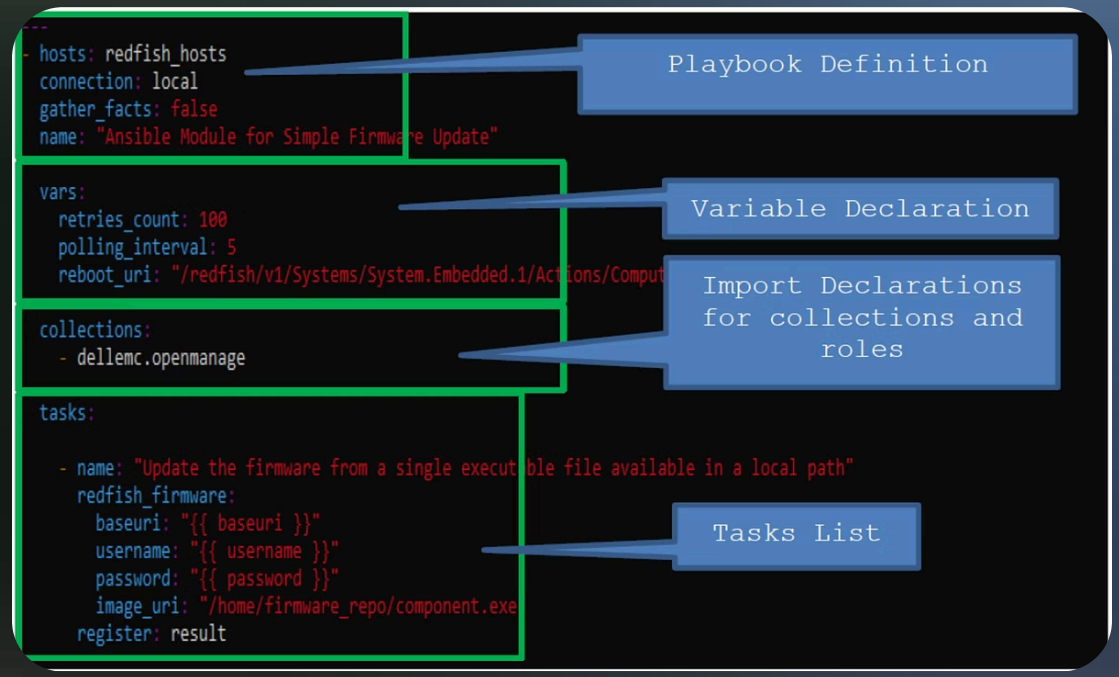
Filter specific information:
ansible -i hosts hostname -m setup -a "filter=ansible_distribution*"

# Ansible Galaxy

- **Namespaces**
  - ansible
  - community
  - amazon
  - azure
- **Collections**
  - Modules (idempotent units of work)
  - Plugins (callback, connection, lookup, filter)
  - Roles (reusable task/variable/file structures)
  - Playbooks (example/playbook YAML files bundled for demos or integration tests)

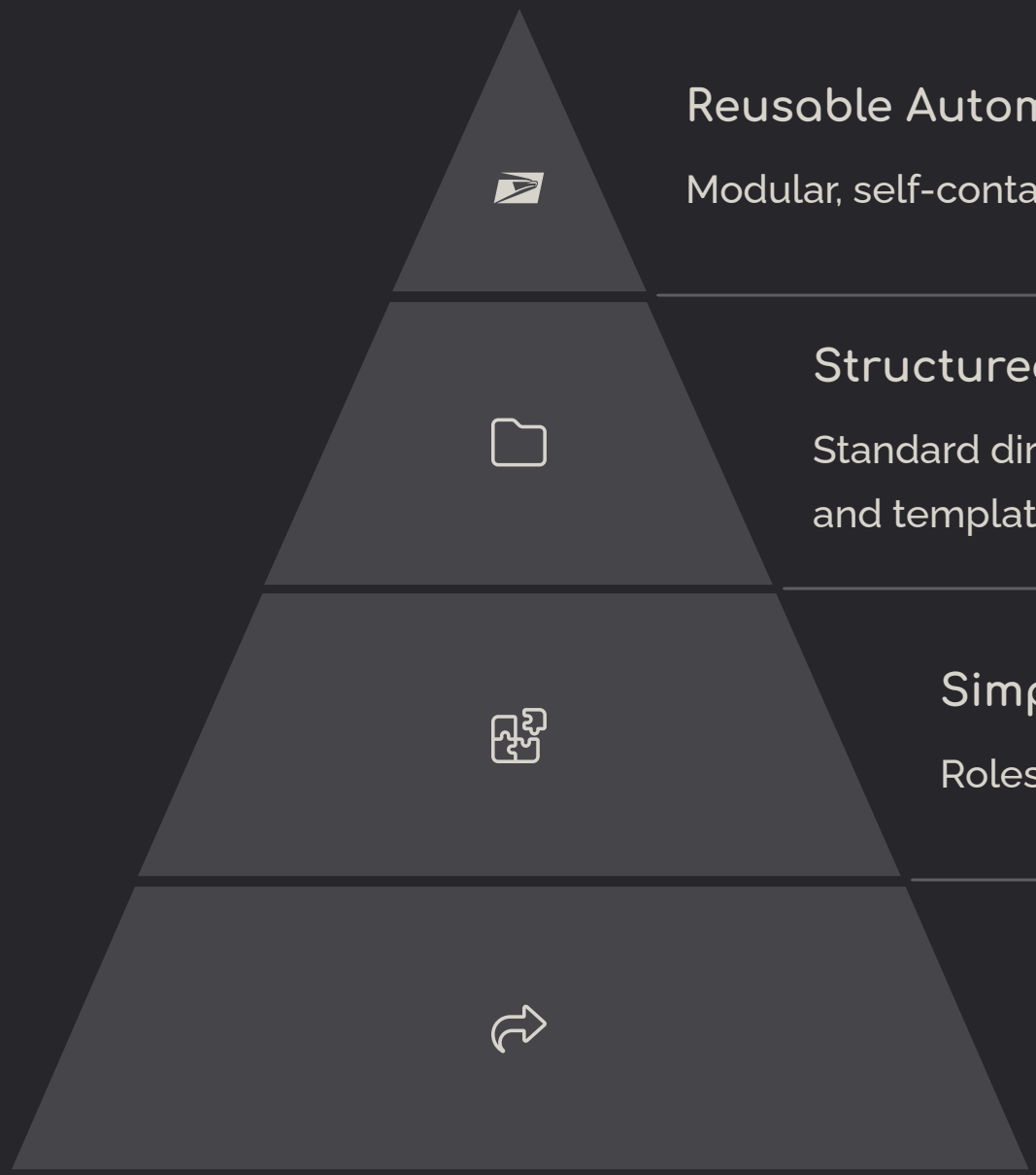
# Playbook

Ansible playbooks are YAML files that define a sequence of tasks to automate system configuration, deployment, and orchestration. They offer a human-readable blueprint describing automation processes in a structured and repeatable format.



# Roles

Ansible Roles are reusable bundles of automation that organize related tasks, variables, and files into a consistent structure. Think of them as pre-packaged recipes for configuration management.



### Reusable Automation

Modular, self-contained units that can be shared across projects.

### Structured File Organization

Standard directory layout separates tasks, handlers, variables, and templates.

### Simplified Playbooks

Roles make playbooks cleaner and more maintainable.

### Community Sharing

Easily import from Ansible Galaxy or create your own.

Roles eliminate repetitive code by packaging related automation tasks into reusable components. They follow a structured directory layout that keeps your automation organized and maintainable.

# Automation: Craft, Not Replacement

## The Automation Misconception

Many fear automation means job elimination and reduced quality.

This view misses how automation transforms rather than replaces human work.

- Changes skill requirements
- Shifts focus to higher-level thinking
- Eliminates repetitive tasks

## The IT Artisan's Approach

Modern IT engineers are digital craftspeople creating automation solutions.

They blend technical expertise with creative problem-solving.

- Designs resilient systems
- Creates elegant automation workflows
- Builds with precision and care

## Elevated Capabilities

Automation brings new competencies to the forefront.

Engineers who embrace these skills deliver superior results.

- Systems thinking
- Architecture design
- Continuous improvement

# Resources

ANSIBLE

docs.ansible.com

**Ansible Documentation**


SEMAPHORE UI

Modern UI for Ansible

Semaphore UI for Ansible,...

**Modern UI and powerful AP...**

If your project has outgrown manual terminal deployments,...

Cattle vs Pet

cloudscaling

Cloudscaling

**The History of Pets vs Cattl...**

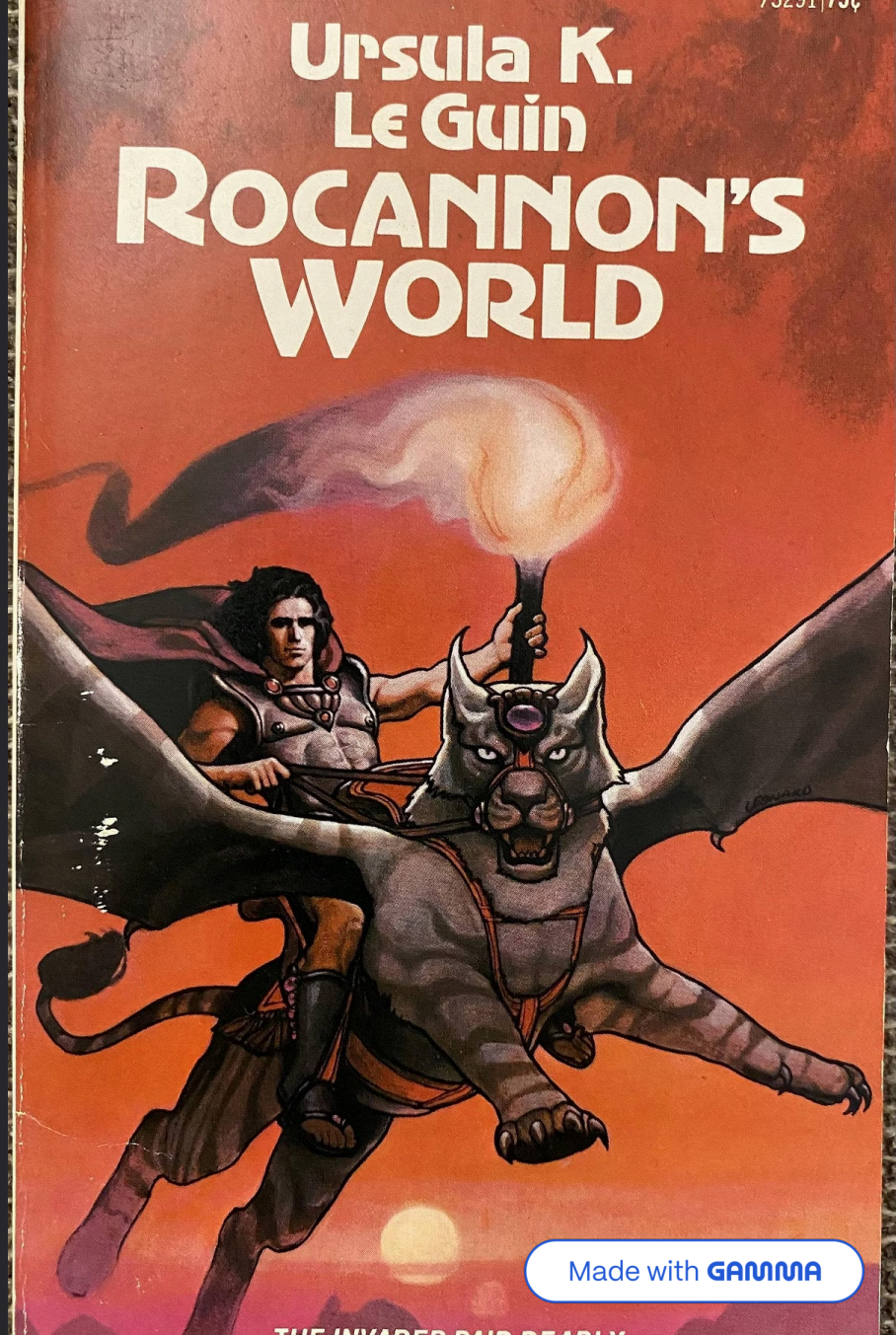
I have been meaning to write this post for a long time, but one...

# Ansible whaaaat?

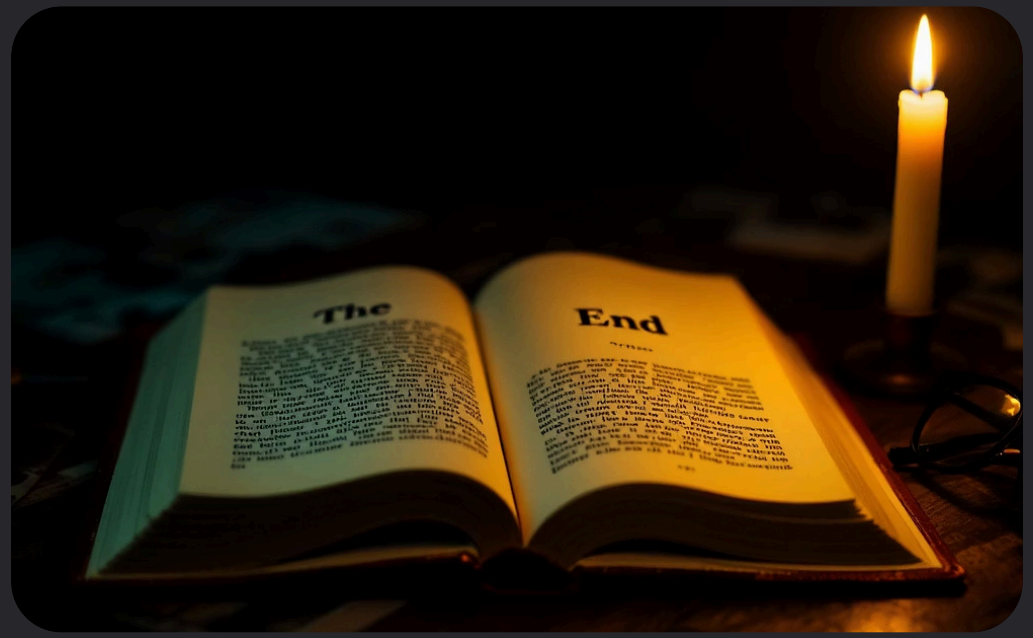
The name "Ansible" pays homage to science fiction literature, specifically Ursula K. Le Guin's novels. In her stories, the ansible was a fictional device enabling instantaneous communication across vast interstellar distances.

First appeared in Le Guin's 1966 novel "Rocannon's World" as a faster-than-light communication device.

Like its namesake, Ansible technology bridges distances between systems, enabling seamless communication and coordination across entire IT environments.



# The End




Closing thoughts

Q&A

Thanks for your time!