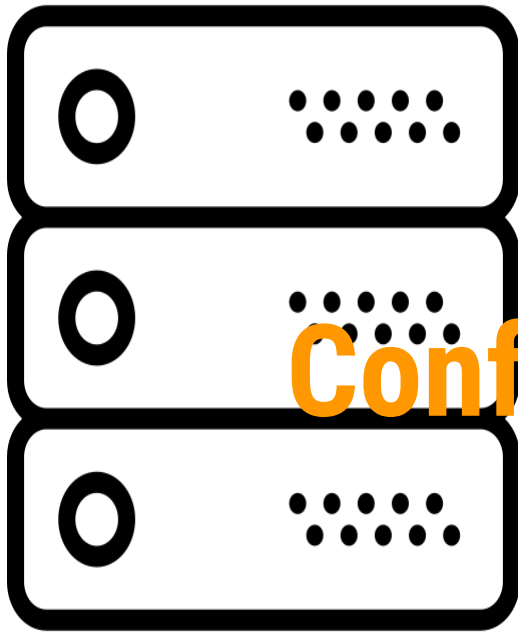


# Ansible Advanced



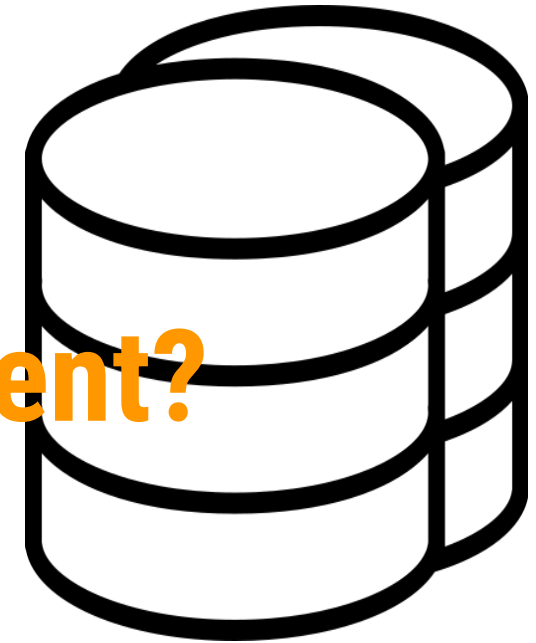
# Traditional Datacenter



Servers



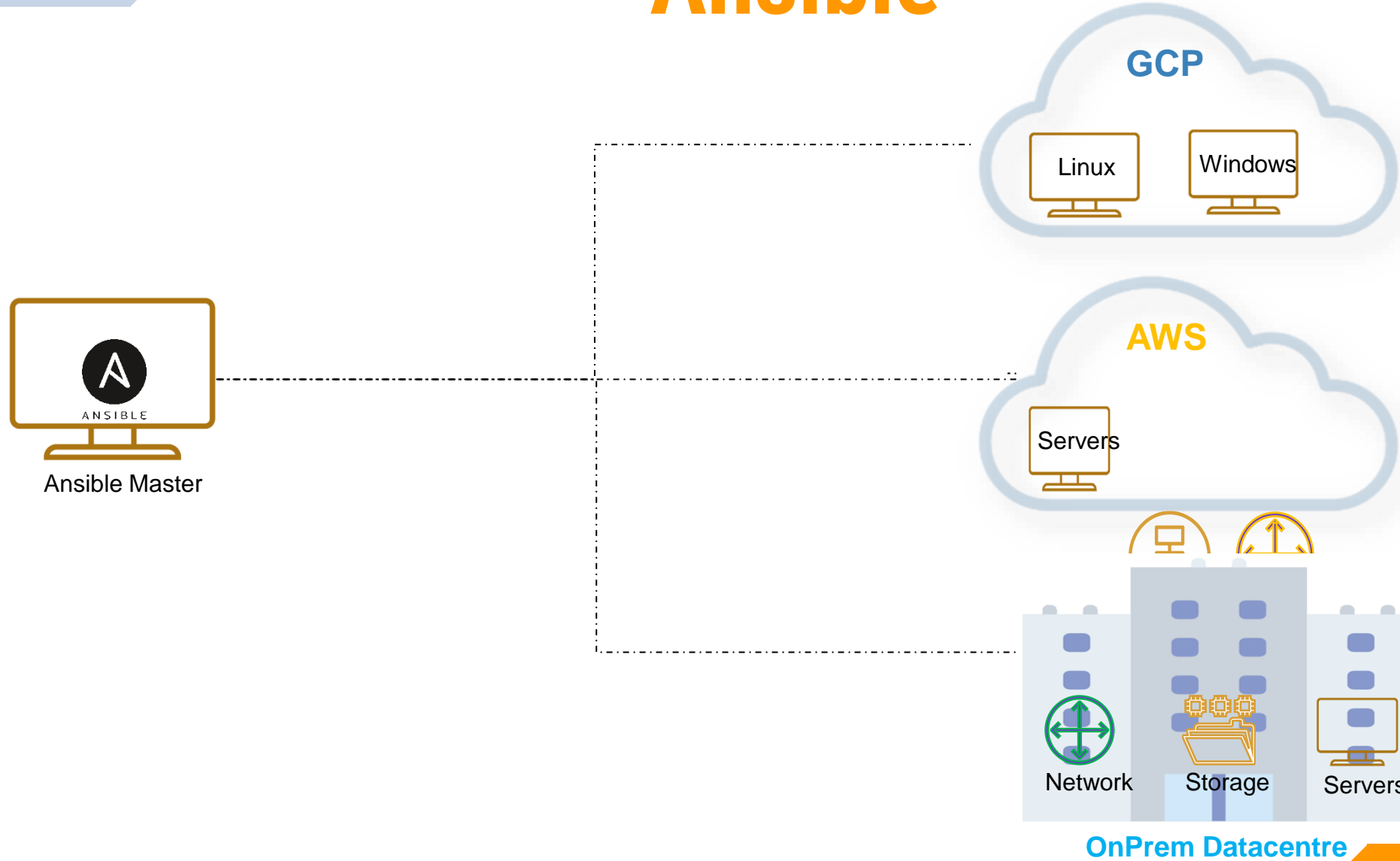
Network



Storage

What is  
Configuration Management?

# Ansible

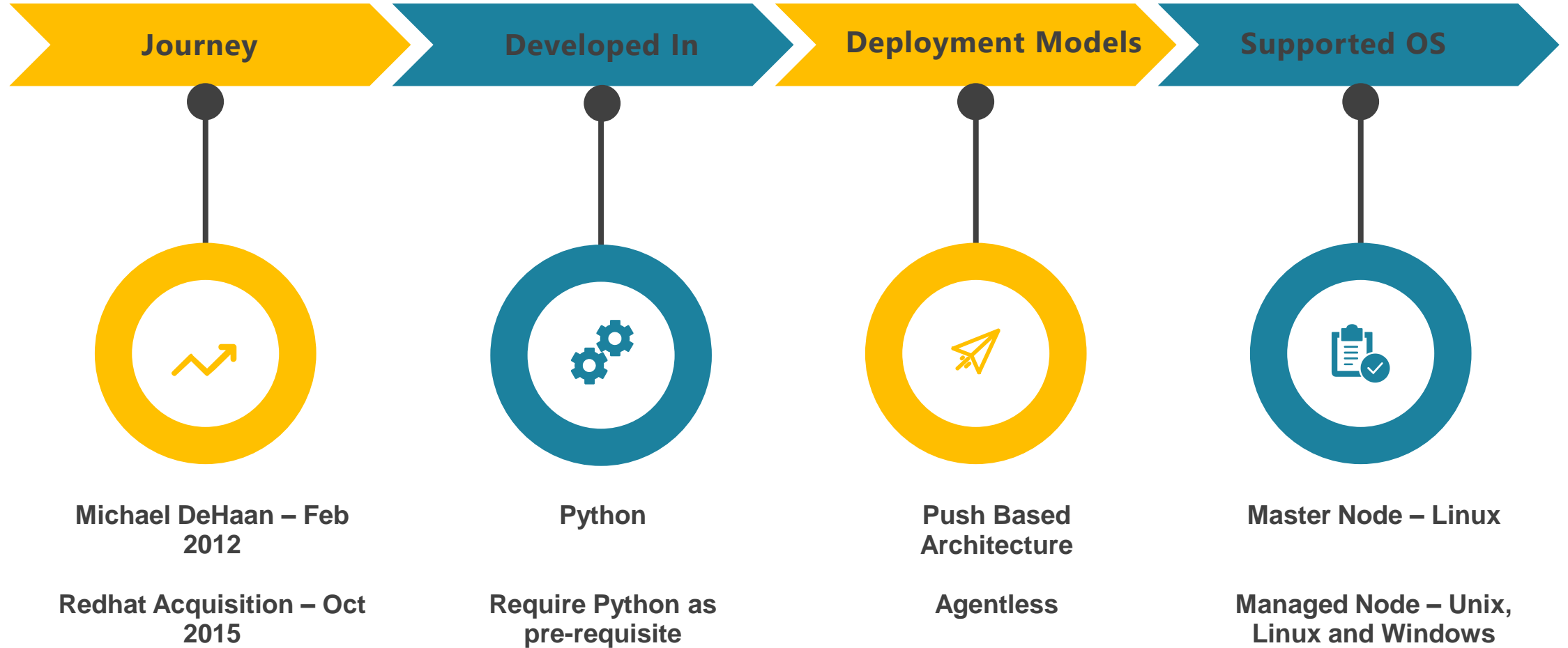


# Ansible

Ansible is an easy-to-use IT Automation, Configuration Management & Orchestration Software for System Administrators & DevOps Engineers.

- Founded in Feb, 2012
- First commercial product release in 2012
- Multiple in-built functional modules
- Multiple Community Members
- 40,000+ Users
- 50,000+ Nodes managed in the largest deployments
- Support for Red Hat, CentOS, Ubuntu, Oracle Linux, MAC, OS, Solaris 10/11, Windows.
- Ansible Controller node Supported on Linux variants only

# Ansible Introduction



# Why Ansible?



**Declarative**



**Increased  
Productivity**



**Agent Less**



**Simplicity**

# Current IT Automation State

**Manually Configure:** Literally logging into every node to configure it.

**Golden Images:** Creating a single copy of a node's software and replicating that across nodes.

**Custom One-off Scripts:** Custom code written to address a specific, tactical problem.

**Software Packages:** Typically all or nothing approach.

# Current IT Automation State

- **Manually Configure:**
  - Difficult to scale.
  - Impossible, for all intents and purposes, to maintain consistency from node-to-node.



# Current IT Automation State

- **Golden Images:**
  - Need separate images for different deployment environments, e.g. development, QA, production, or different geo locations.
  - As number of images multiply it becomes very difficult to keep track and keep consistent.
  - Since they're monolithic copies, golden images are rigid and thus difficult to update as the business needs change.

# Current IT Automation State

- **Custom One-off Scripts:**
  - No leverage – effort typically cannot be reused for different applications or deployments.
  - Brittle – as needs change, often the entire script must be re-written.
  - Difficult to maintain when the original author leaves the organization.

# Current IT Automation State

- **Software Packages:**
  - These packages typically require that all resources be placed under management – cannot selectively adopt and scale automation.
  - As a result, longer deployments times.
  - Dated technology developed before virtualization and cloud computing – lacks responsiveness to changing requirements.

# Why Configuration Management?

- To provide optimized level of automated way to configure Applications and Software's inside your system.
- Enable you to Discover, Provision, Configure and Manage the systems.
- Developers should be able to use a single command to build and test software in minutes or even in seconds.
- Maintaining configuration state of all systems simultaneously should be easy.
- Login into every client machine for CM tasks should not be mandate.
- It should be easy to maintain desired state as per policy

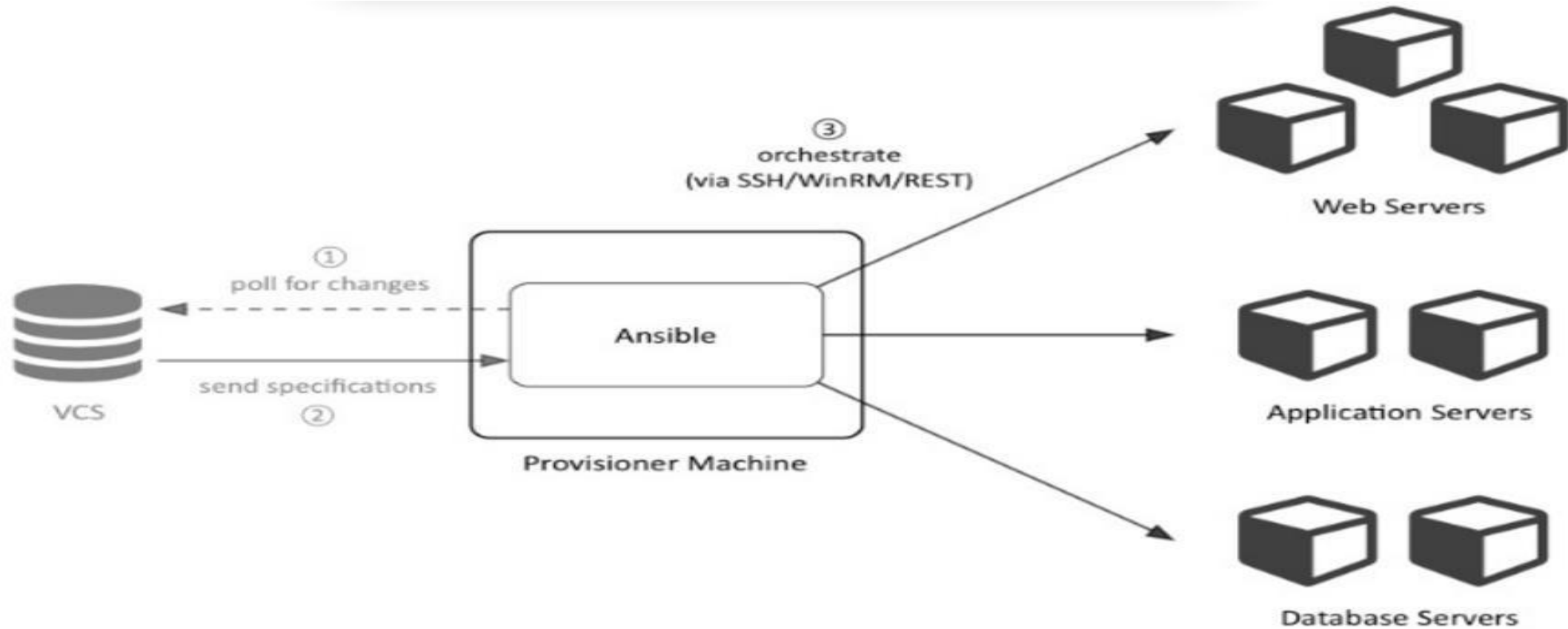
# Why Orchestration with Ansible?

- A single tool for deployment and Configuration management
- Easy to manage and use
- Compatible with all major cloud service providers
- Can Orchestrate Infrastructure and Software both

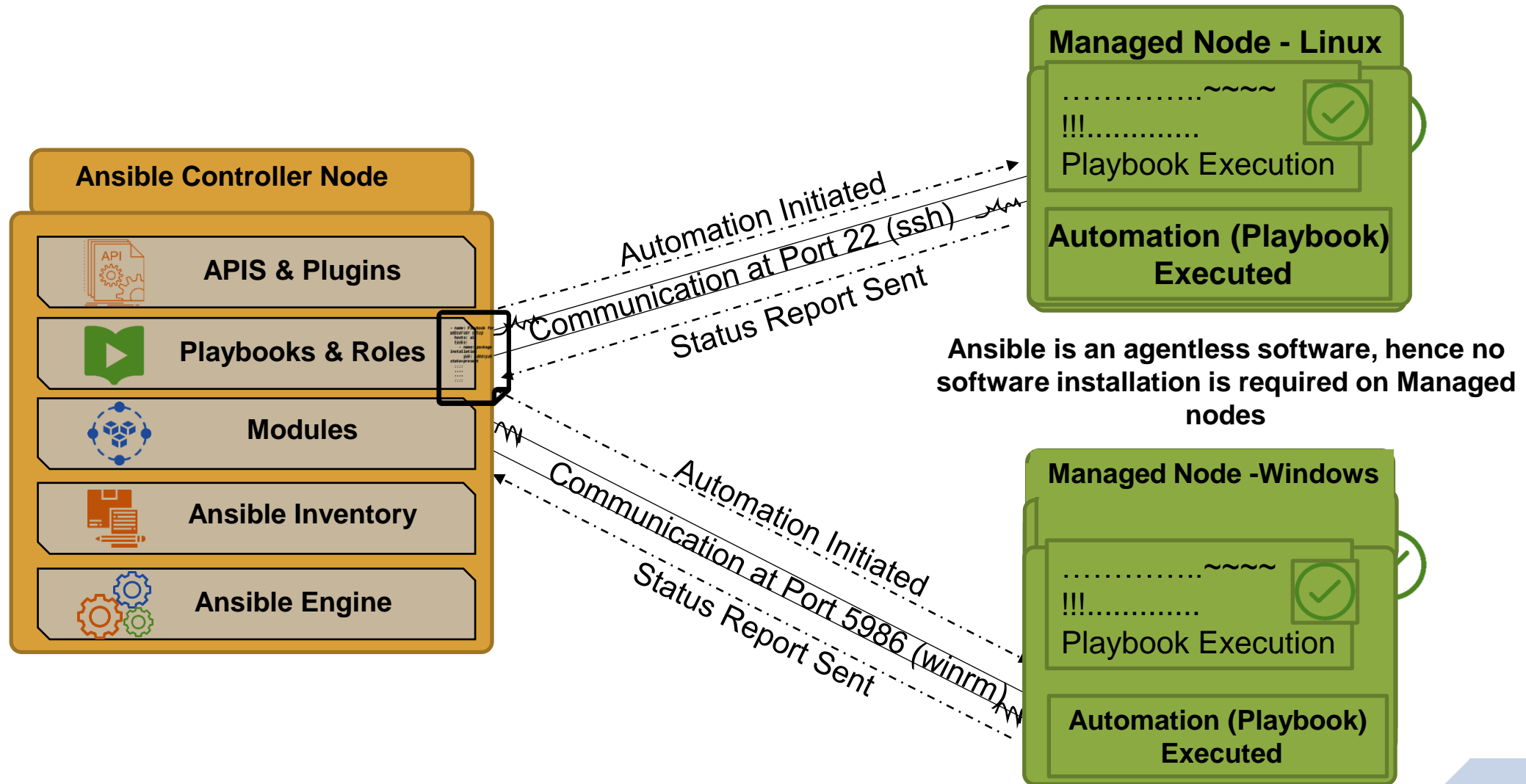
# Ansible Components

- Ansible consists of **Agentless Model** and majorly have two parts:
  - **Controller / Master:** The central configuration server where we will have our all configurations stored.
  - **Managed Nodes / Clients:** All clients getting configured from Ansible Master.
- **Note:**
- Ansible Master can be run from any Linux machine (Windows not supported) with Python 2 (version 2.7) or Python 3 (versions 3.5 and higher) installed.
- On the managed nodes, you need a way to communicate, which is normally SSH. By default this uses SFTP. If that's not available, you can switch to SCP in ansible.cfg. You also need Python 2 (version 2.6 or later) or Python 3 (version 3.5 or later).

# Dataflow



# Ansible Architecture





# Ansible and its Peers

Many tools available in Market. Few things to consider, before selecting any tool:

- Configuration Management vs Orchestration
- Mutable Infrastructure vs Immutable Infrastructure
- Procedural vs Declarative
- Client/Server Architecture vs Client-Only Architecture

# Ansible and its Peers

	Chef	Puppet	Ansible	SaltStack	CloudFormation	Terraform
<b>Code</b>	Open source	Open source	Open source	Open source	Closed source	Open source
<b>Cloud</b>	All	All	All	All	AWS only	All
<b>Type</b>	Config Mgmt	Config Mgmt	Config Mgmt	Config Mgmt	Orchestration	Orchestration
<b>Infrastructure</b>	Mutable	Mutable	Mutable	Mutable	Immutable	Immutable
<b>Language</b>	Procedural	Declarative	Declarative	Declarative	Declarative	Declarative
<b>Architecture</b>	Client/Server	Client/Server	Client-Only	Client/Server	Client-Only	Client-Only

# Knowledge Checks

- What is Configuration Management?
- List a few available configuration Management tools.
- What are the Advantages of Ansible?
- Explain Data flow of Ansible.

# Ansible Installation

# Installation of Ansible

- The Ansible **master** is the machine that controls the infrastructure and dictates policies for the servers it manages.
- Currently Ansible can be run from any machine with Python 2.6 or 2.7 installed (Windows isn't supported for the control machine).
- This includes Red Hat, Ubuntu, Debian, CentOS, OS X, any of the BSDs, and so on.