Lets get started

What are we going to see in this session?

- Introduction to Ansible
- What we do using Ansible ?
- Why Ansible ?
- Simple Scenario
- Pull vs Push configuration tool
- Common Ansible use cases
- Ansible Architecture

Introduction to Ansible

- Ansible was originally written by Michael DeHaan.
- Ansible is an open source configuration management and orchestration utility.
- Using this you can automate and standardize the configuration of remote hosts and virtual machines.
- Ansible, Inc. was the company setup to commercially support and sponsor Ansible. Later it was acquired by RedHat in October 2015.

What we do using Ansible ?

IT Automation:
- Instructions are written to automate the IT professional's work.
- Instructions can be executed in multiple remote hosts.

Configuration Management :
- Consistency of all systems in Infrastructure is maintained.
- Mainly used for patch management.

Automatic Deployments :

Applications are deployed automatically on variety of environments.

Why Ansible ?

SIMPLE

Human readable automation

No special coding skills needed

Tasks executed in order

Get productive quickly

POWERFUL

App deployment

Configuration management

Workflow orchestration

Orchestrate the app lifecycle

AGENTLESS

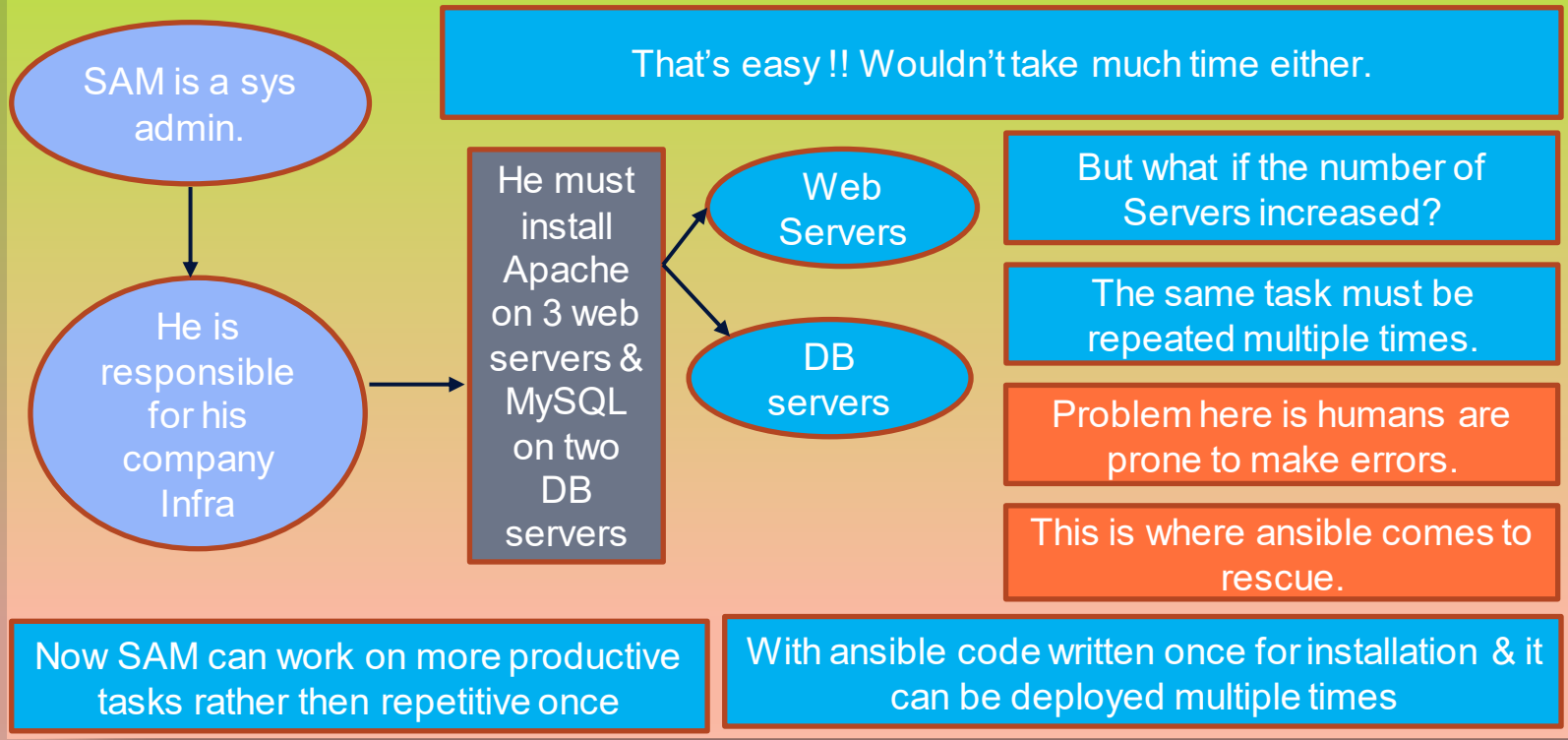
Agentless architecture

Uses OpenSSH & WinRM

No agents to exploit or update

More efficient & more secure

Simple Scenario



Pull Vs Push Configuration

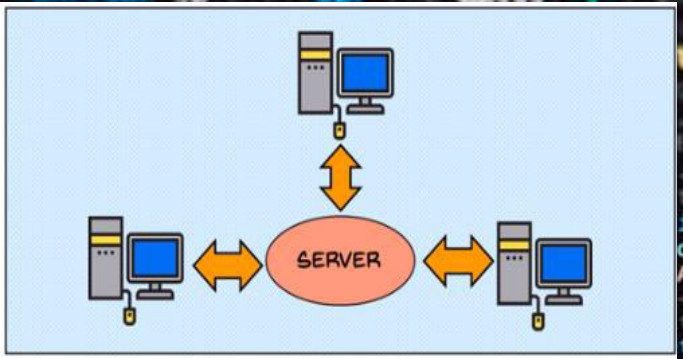
Pull Configuration Tools :

- ✓ In Pull based Tools, there will be a MASTER server where all the instructions are placed.
- ✓ Master machine also has the information's of client machines connected to it.
- ✓ Then piece of software which also called as agent installed on all other target machines which will enable the communication between MASTER and SLAVE machines.

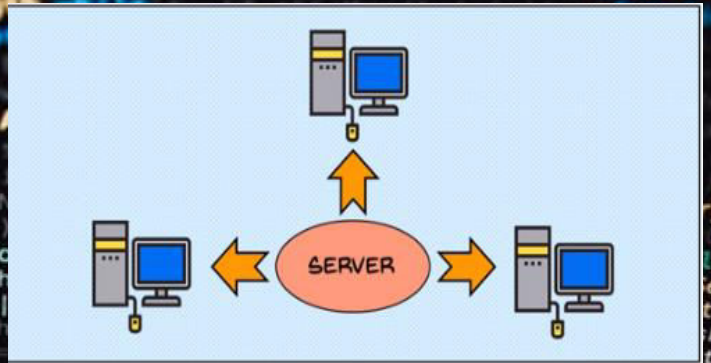
Push Configuration Tools :

- ✓ This also has MASTER server which passes set of instructions to clients but major difference here is its doesn't need any agent to be installed on SLAVES.
- ✓ Here you are just achieving your job by simply pushing the changes to SLAVE machine and forcing it to restructure.

Cont.



Pull configuration: Nodes check with the server periodically and fetch the configurations from it



Push configuration: Server pushes configuration to the nodes

Cont.

Any Guess what could be pull based mechanism tools?

- ✓ Chef
- ✓ Puppet

Disadvantage :

- ✓ Whenever new machine comes into Environment you need install the agent on it to establish its communication to Master server.
- ✓ When Master is upgraded with newer version all Client agents should be upgraded.

Ansible is push based mechanism Tools.

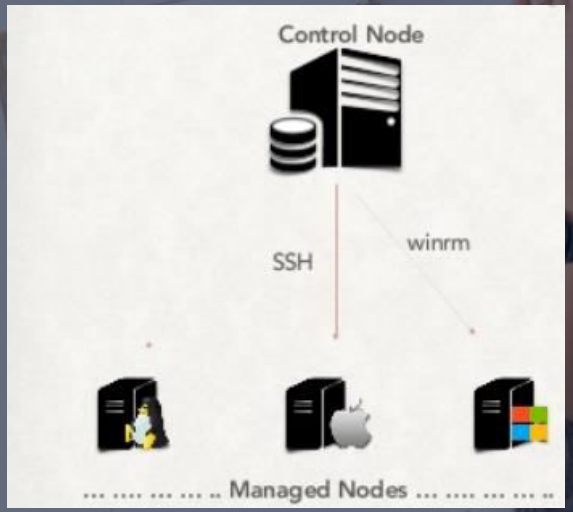
Common use case of Ansible

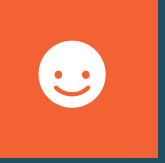
- ✓ Provisioning
- ✓ Configuration Management
- ✓ Application Deployment
- ✓ Continuous Delivery
- ✓ Security & Compliance
- ✓ Orchestration

Ansible Architecture

Prerequisites to have Ansible in your Environment.

- ✓ One control node to be installed with Ansible in any Linux flavor.
- ✓ Managed Nodes can be with any OS, it can be LINUX, APPLE and WINDOWS.
- ✓ Linux nodes and Mac nodes will be managed through SSH agent.
- ✓ WINDOWS nodes will be managed to WINRM agent.



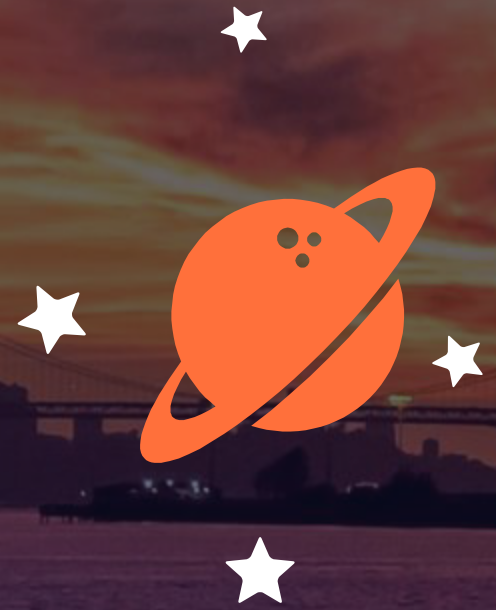


End of this topic !

Any questions?

What are we going to see in this session?

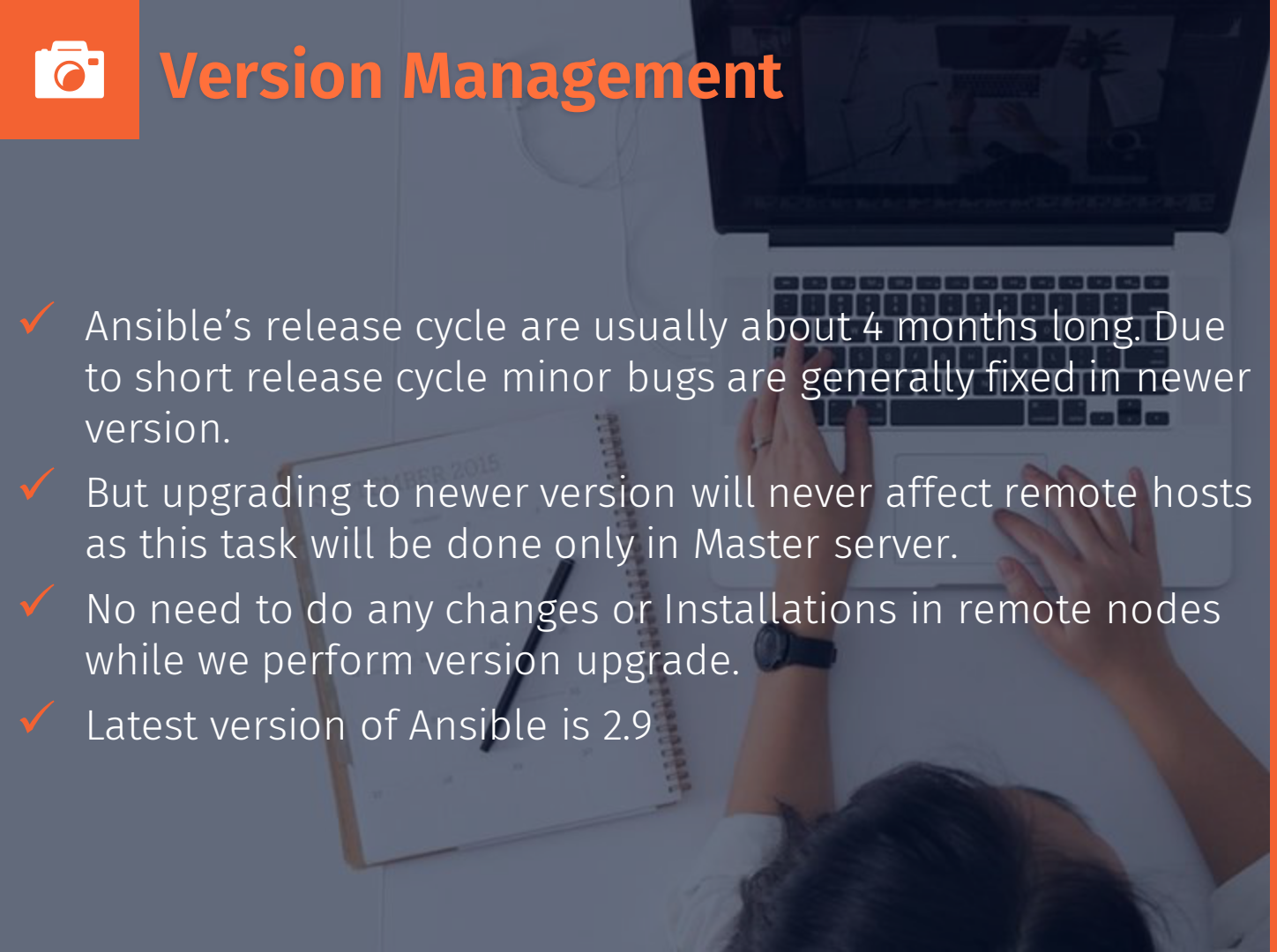
- Ansible Installation
  - What will be installed?
  - Version Management
  - Infrastructure Prerequisites
  - Prerequisites
  - Types of Installation
    - Through YUM
    - Through PIP
- Install SSH Pass
- Exploring Ansible Configuration file
- How, What and Where?



What will be Installed?

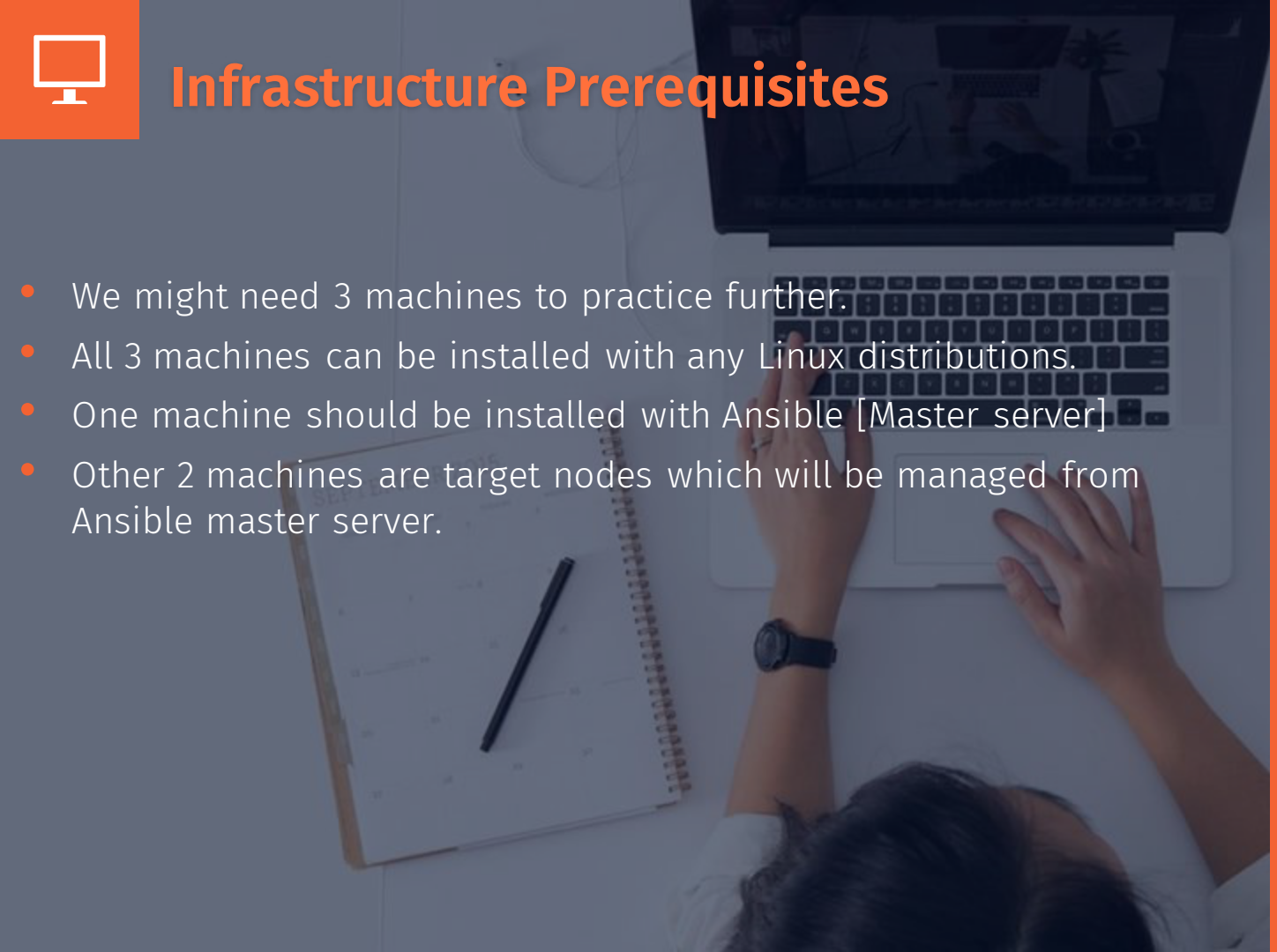
- ✓ Ansible by default manages machines over the SSH protocol.
- ✓ Once Ansible is installed, it will not add any database, and there will be no daemons to start or keep running.
- ✓ You only need to install it on one machine, and it can manage an entire other remote machines.
- ✓ It does not leave any software/agent installed or running on remote machines, so there's no real question about how to upgrade Ansible when moving to a new version.

Version Management



- ✓ Ansible's release cycle are usually about 4 months long. Due to short release cycle minor bugs are generally fixed in newer version.
- ✓ But upgrading to newer version will never affect remote hosts as this task will be done only in Master server.
- ✓ No need to do any changes or Installations in remote nodes while we perform version upgrade.
- ✓ Latest version of Ansible is 2.9

Infrastructure Prerequisites



- We might need 3 machines to practice further.
- All 3 machines can be installed with any Linux distributions.
- One machine should be installed with Ansible [Master server]
- Other 2 machines are target nodes which will be managed from Ansible master server.

Prerequisites

Control Node prerequisites :

- Python 2 (version 2.7) or Python 3 (versions 3.5 and higher).
- SSHPASS
- Windows isn't supported for the control node.

Target Node prerequisites :

- Python 2 (version 2.7) or Python 3 (versions 3.5 and higher).
- SSH should be up and running.
- If that's not available, you can switch to SCP in [ansible.cfg],

Types of Installation

Installation can be done in 2 ways

- ✓ Through Package Manager [Yum]
- ✓ Through Python PIP installer

Through YUM

Installing Ansible through package manager such as YUM :

- If incase you machine is RHEL [7&8], You need to enable repos :

  subscription-manager repos --enable ansible-2.8-for-rhel-8-x86_64-rpms

  subscription-manager repos --enable rhel-7-server-ansible-2.8-rpms

- Centos :

  Setup “epel” repo to install Ansible

  yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-8.noarch.rpm

  yum install https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm

- Install using root :

  yum install ansible

- Install from non-root user :

  sudo yum install ansible

Ubuntu distribution

Installing Ansible through package manager such as APT :

- ✓ Install using root :

```
apt update
apt install software-properties-common
apt-add-repository --yes --update ppa:ansible/ansible
apt install ansible
```

- ✓ Install from non-root user :

```
sudo apt update
sudo apt install software-properties-common
sudo apt-add-repository --yes --update ppa:ansible/ansible
sudo apt install ansible
```

Install Ansible through Python PIP

Using PIP module you can install Ansible.

- ✓ First Install pip package [if it is not available]

  yum install python3-pip

- ✓ Then install the ansible package using PIP

  pip3 install ansible

Note : Both yum & pip is going to install ansible under root directories, You can see that using commands

ansible-config –version (or) ansible --version

What if you want to install ansible under customized directory ?

For this you need to use python virtual environment.

Install Ansible under virtual environment

What is Virtual Environment ?

- ✓ Virtualenv is used to manage Python packages for different projects.
- ✓ Virtualenv allows you to avoid installing Python packages globally which could brake system tools or other projects.

Configure Virtual Environment in your custom directory.

- ✓ Install virtual env using pip [if its is not there]

```
pip3 install virtualenv
virtualenv ansible
source ansible/bin/activate
pip3 install ansible
ansible –version (or) ansible-config --version
```

Install SSH Pass

- ✓ yum install -y https://dl.fedoraproject.org/pub/epel/epel-release-latest-7.noarch.rpm
- ✓ yum-configuration-manager --enable epel
- ✓ yum install sshpass

Use of SSHPASS is to enable non-interactive SSH connections

Exploring Ansible Configuration file

Creating/Managing Ansible configuration file :

- ✓ If installing ansible from a package manager, the latest ansible.cfg file should be present in /etc/ansible.
- ✓ If you installed ansible from pip or from source, you may want to create this file in order to override default settings in ansible an example file available in Github
- ✓ You can get all this details by using command

  ansible-config –version

Major things to notice in configuration file :

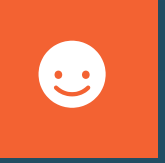
- ✓ Your default hosts directory if you install it through YUM :

  inventory = /etc/ansible/*
- ✓ It can be modified based on your requirement.

How, What & Where?

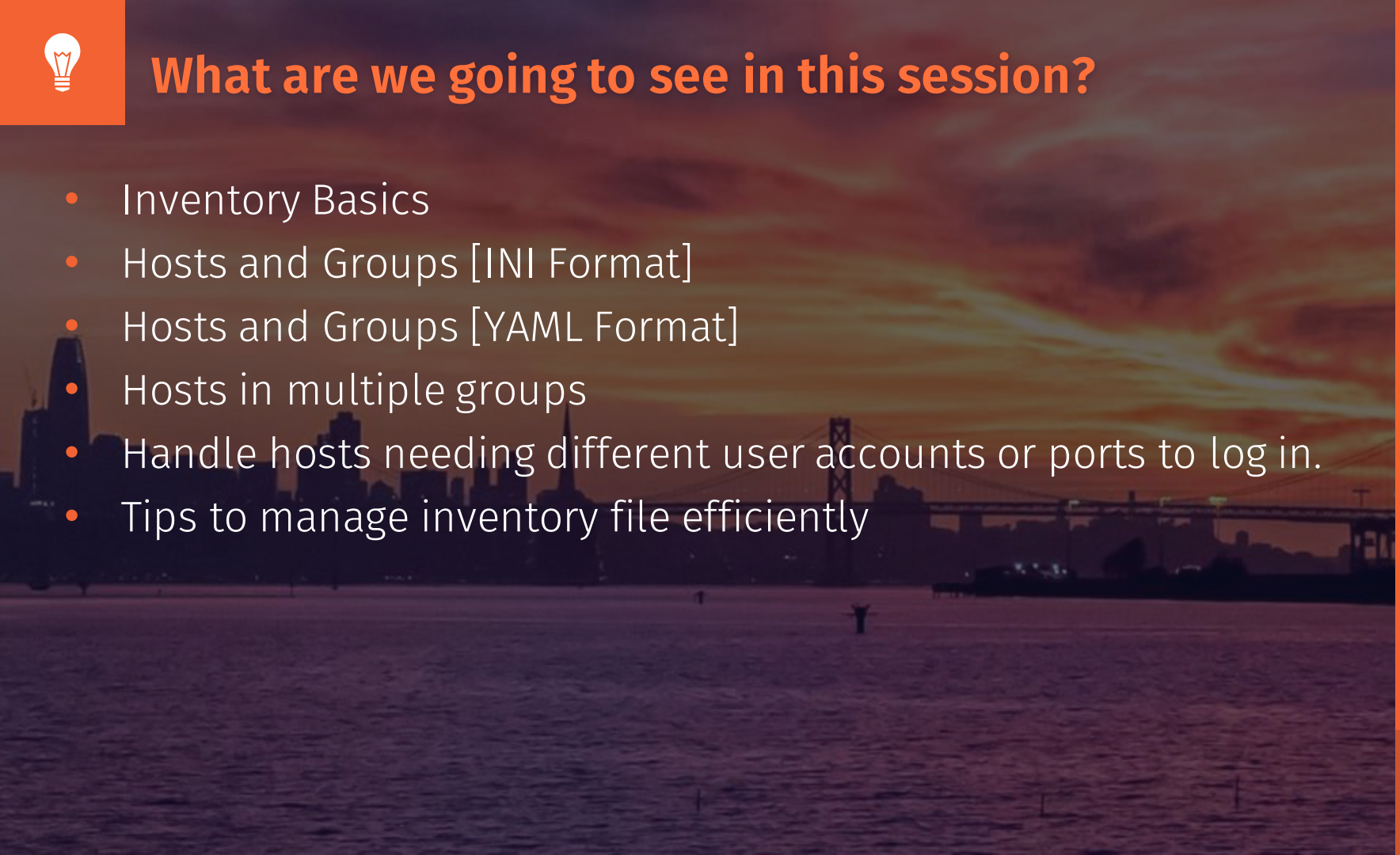
This image clearly shows how ansible works.

- ✓ ANSIBLE.CFG file is going to instruct ansible how to work.
- ✓ PLAYBOOK is to hold the instructions of what to do.
- ✓ INVENTORY has the information of where to do.



End of this topic !

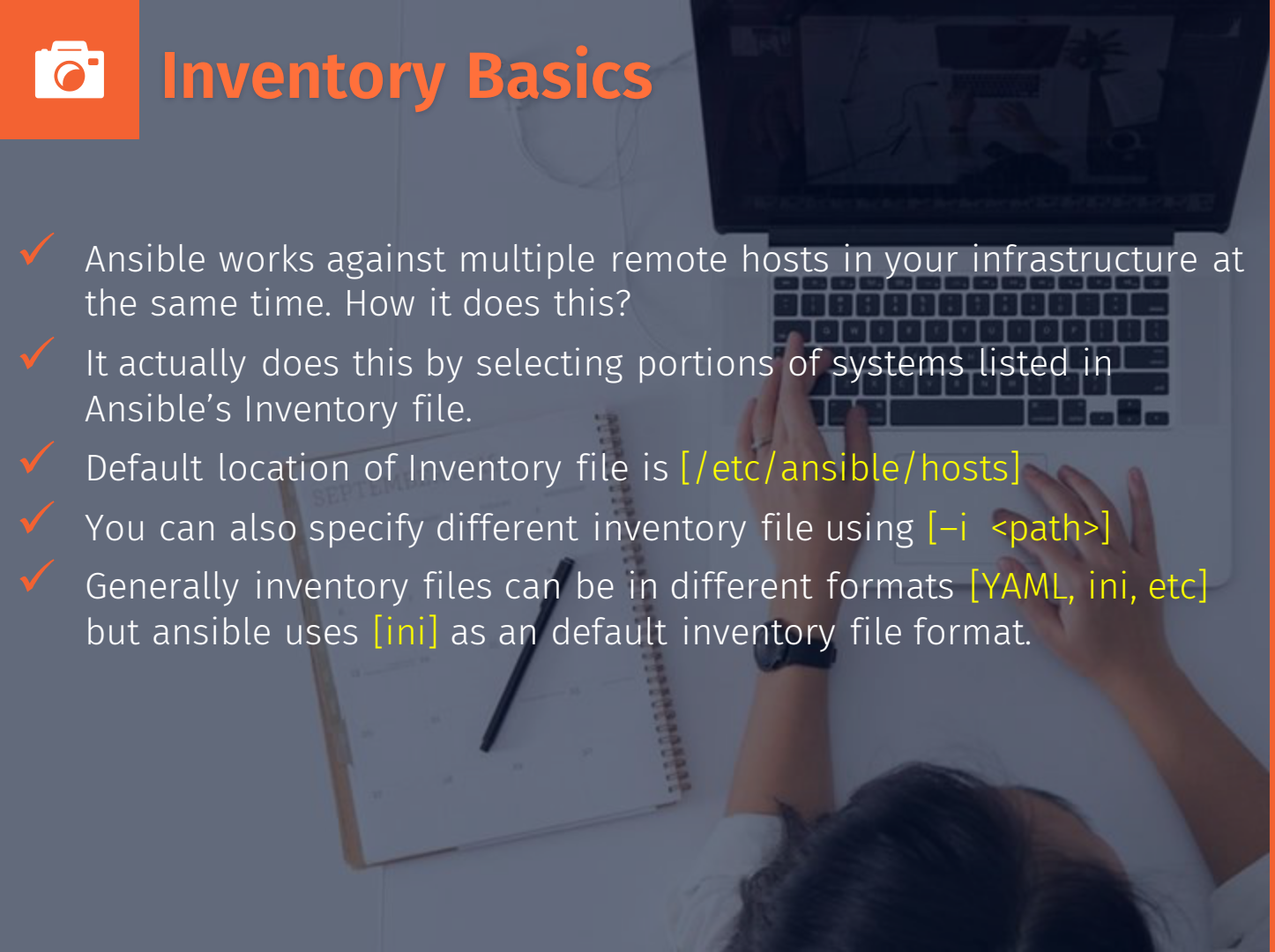
Any questions?



What are we going to see in this session?

- Inventory Basics
- Hosts and Groups [INI Format]
- Hosts and Groups [YAML Format]
- Hosts in multiple groups
- Handle hosts needing different user accounts or ports to log in.
- Tips to manage inventory file efficiently

Inventory Basics



- Ansible works against multiple remote hosts in your infrastructure at the same time. How it does this?
- It actually does this by selecting portions of systems listed in Ansible's Inventory file.
- Default location of Inventory file is [/etc/ansible/hosts]
- You can also specify different inventory file using [–i <path>]
- Generally inventory files can be in different formats [YAML, ini, etc] but ansible uses [ini] as an default inventory file format.

Hosts and Groups [INI-Format]

- Lets see some examples in INI format [This is the default format of Ansible]
- The headings in [brackets] are group names, which are basically used to classify the hosts and deciding what systems you are controlling at what times and for what purpose.

```
# Single host without any group
mail.example.com
# Hosts listed under Group # Below we have classified all the web servers.
[WEB servers]
one.example.com
two.example.com
# Below we have classified all the DB servers.
[DB servers]
three.example.com
four.example.com
five.example.com
```

Hosts and Groups [YAML-Format]

- Lets see some examples in YAML format.
- [Note : This is not and default format used by Ansible, Its just one of the options available for our comfort]
- Here no [brackets] for group names and groups are classified under children's category

```
all:
  hosts:
    mail.example.com:
  children:
    webservers:
      hosts:
        foo.example.com:
        bar.example.com:
    dbservers:
      hosts:
        one.example.com:
        two.example.com:
        three.example.com:
```

Hosts in Multiple groups

- ✓ You can also put systems in more than one group, for instance a server could be part of both webserver and specific datacenter.
- ✓ For Example you can create groups that track.
  - What – Whether that host is Webserver, Database or Performance testing etc..
  - Where – You can specify the datacenter region.
  - When – You can specify whether it is PROD, TEST or STAAGING.

Handle hosts needing different user accounts or ports to log in

- ✓ This is called inventory variables (or) host variables.
- ✓ Setting inventory variables in the inventory file is the easiest way.
- ✓ For instance, suppose these hosts have different usernames and ports

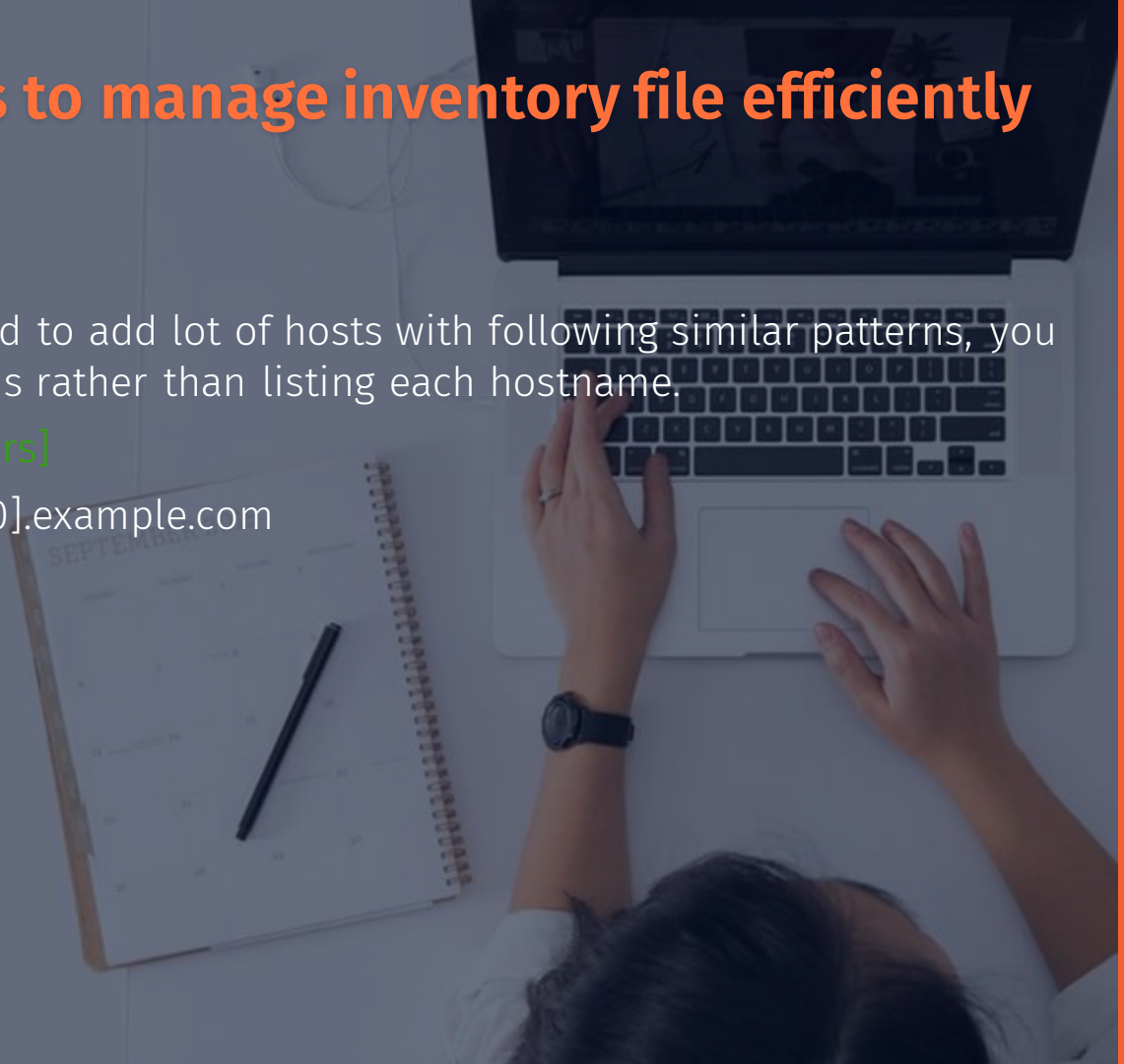
```
[webservers]
server1.example.com  ansible_port=5000  ansible_user=oracle
server2.example.com  ansible_port=5001  ansible_user=mysql
```

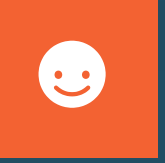
Tips to manage inventory file efficiently

- If you need to add lot of hosts with following similar patterns, you can do this rather than listing each hostname.

  [webservers]

  www[01:50].example.com





End of this topic !

Any questions?

What are we going to see in this session?

- What is Modules ?
- Sample Modules
- Where you can find all this modules for reference ?
- Finding Module information’s from Command line
- Run Commands

What is Modules ?

- Modules are programs that Ansible uses to perform operations on managed hosts.
- They are ready-to-use tools designed to perform specific operations.
- Modules can be executed from the Ansible command line or used in playbooks to execute the tasks.
- Three Type of Modules :
  - ✓ Core Modules : These modules are written and maintained by Ansible development team. Core modules are most important modules and are used for common administrative tasks.
  - ✓ Extra Modules : These modules are developed by the community.
  - ✓ Custom Modules : These modules are mostly developed by end users itself.

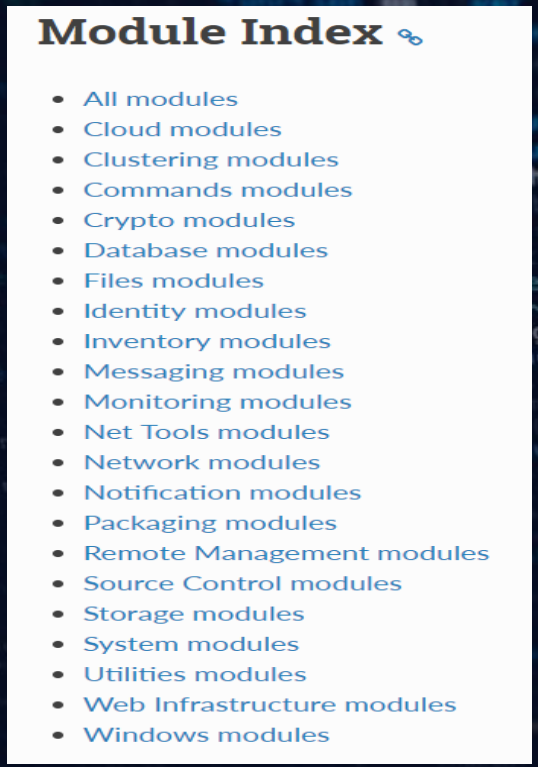
If a module not already exist for a task, an admin can create by its own which is called custom ones. [Modules are written in Python]

Sample Modules

- Ansible ships several hundreds of modules today, some samples are
  - ✓ apt/yum
  - ✓ copy
  - ✓ file
  - ✓ ping
  - ✓ service
  - ✓ git
  - ✓ get_url
  - ✓ shell

Where you can find all this modules for reference ?

Modules



Module Index

- All modules
- Cloud modules
- Clustering modules
- Commands modules
- Crypto modules
- Database modules
- Files modules
- Identity modules
- Inventory modules
- Messaging modules
- Monitoring modules
- Net Tools modules
- Network modules
- Notification modules
- Packaging modules
- Remote Management modules
- Source Control modules
- Storage modules
- System modules
- Utilities modules
- Web Infrastructure modules
- Windows modules

Finding Module information's from Command line

- You can retrieve all the modules and use case of modules from machine where Ansible is installed.

```
ansible-doc –l
ansible-doc -l | grep copy
ansible-doc copy
```

RUN Commands

- If Ansible doesn't have a module that suits your needs, there are some "run command" modules.
  - command: Takes the command and executes it on the host. The most secure and predictable.
  - shell: Executes through a shell like /bin/sh so you can use pipes etc. Be careful.
  - script: Runs a local script on a remote node after transferring it.

End of this topic !

Any questions?