

DevOps Project

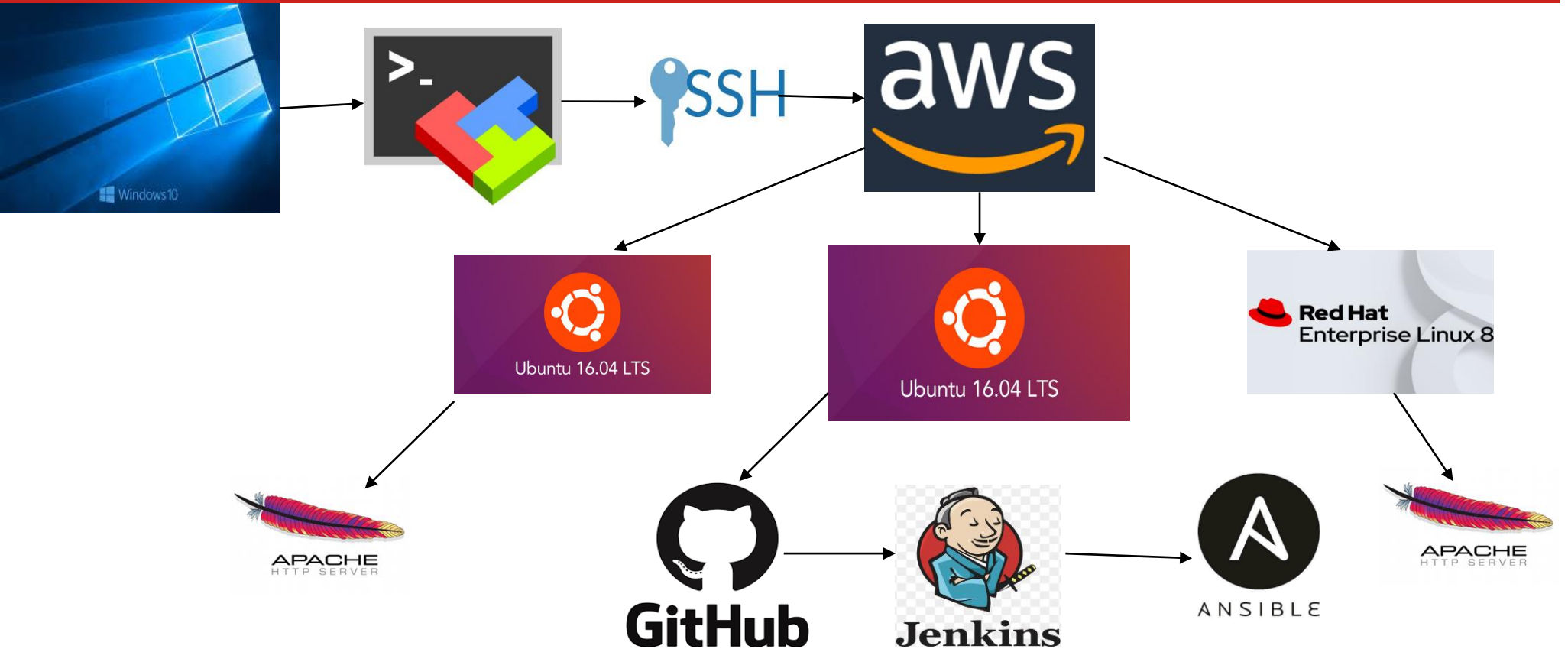
*Automating process of
deployment web sites
in different platform*

Igor Ivanov

Servers

EPAM DevOps 2020

Used tools



Steps on AWS

1)Raising 3 Instances

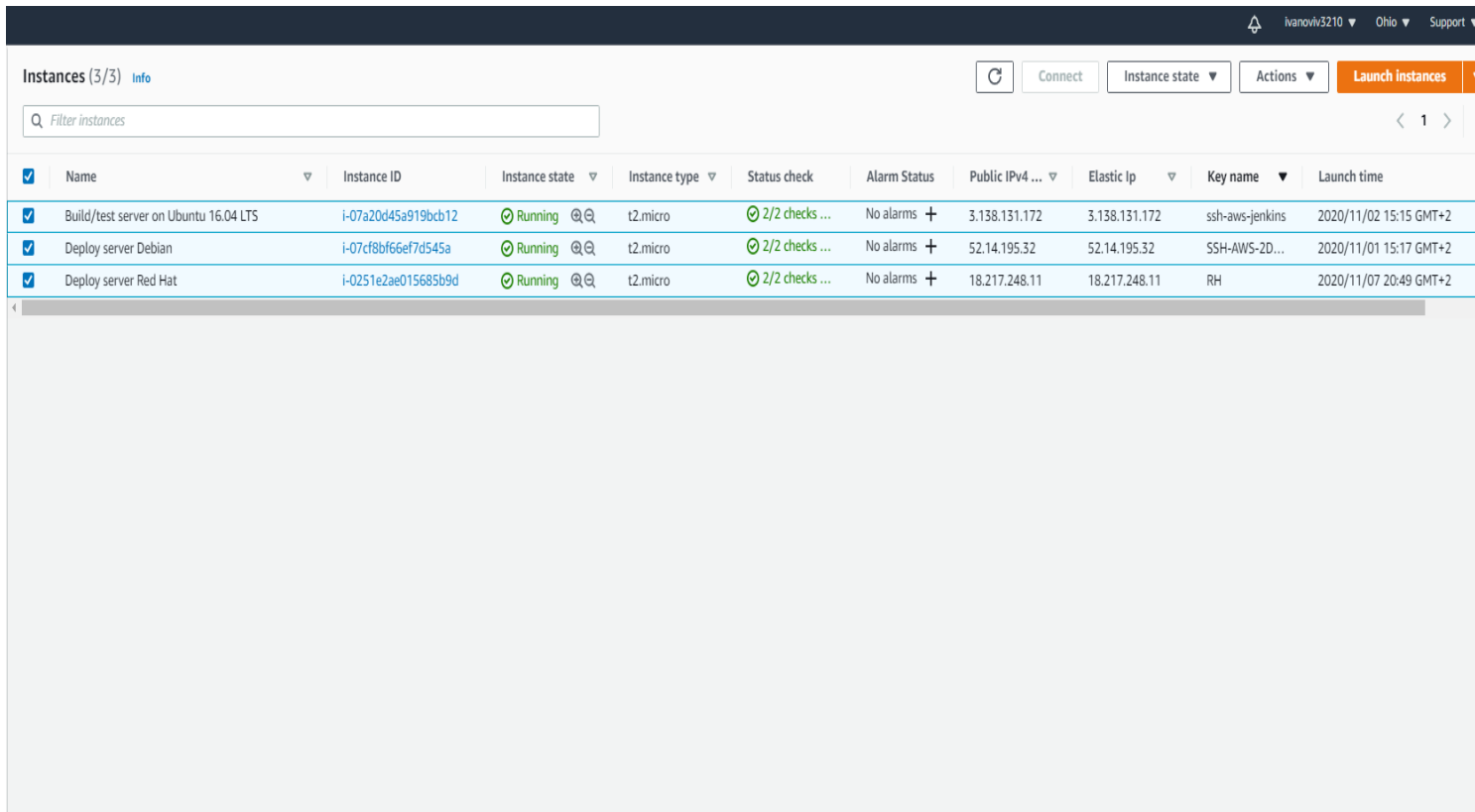
- *Development*

*server(Git, Jenkins
and Ansible).*

- *Deployment server
on Debian OS.*

- *Deployment server
on Red Hat OS.*

2)Making Elastic
Public IP Adresses for
instances.



Instances (3/3) Info											
<input type="text" value="Filter instances"/>											
<input checked="" type="checkbox"/>	Name	Instance ID	Instance state	Instance type	Status check	Alarm Status	Public IPv4 ...	Elastic Ip	Key name	Launch time	
<input checked="" type="checkbox"/>	Build/test server on Ubuntu 16.04 LTS	i-07a20d45a919bcb12	Running	t2.micro	2/2 checks ...	No alarms +	3.138.131.172	3.138.131.172	ssh-aws-jenkins	2020/11/02 15:15 GMT+2	
<input checked="" type="checkbox"/>	Deploy server Debian	i-07cf8bf6gef7d545a	Running	t2.micro	2/2 checks ...	No alarms +	52.14.195.32	52.14.195.32	SSH-AWS-2D...	2020/11/01 15:17 GMT+2	
<input checked="" type="checkbox"/>	Deploy server Red Hat	i-0251e2ae015685b9d	Running	t2.micro	2/2 checks ...	No alarms +	18.217.248.11	18.217.248.11	RH	2020/11/07 20:49 GMT+2	

Steps on AWS

3) Creating SSH Key pairs, Downloading and transferring to servers to connect.

4) Making Security Groups and open Ports For access to Jenkins,Docker,any TCP.

Welcome to the new instances experience!
We're redesigning the EC2 console to make it easier to use. To switch between the old console and the new console, use the New console link in the top right corner.

Key pairs (5)

Filter key pairs

<input type="checkbox"/>	Name	Fingerprint	ID
<input type="checkbox"/>	RH	eb:42:e6:a3:15:e1:37:51:a8:64:f4:e9:7...	key-0721c5c683299a9d1
<input type="checkbox"/>	snoop2docker	46:00:90:23:8c:11:3b:30:3f:a5:2d:67:6...	key-0e64725096b68322c
<input type="checkbox"/>	SSH-AWS-2Deploy	76:f3:ea:5c:59:6a:9b:65:34:65:f2:62:bd...	key-0d847a67896c63cbd
<input type="checkbox"/>	ssh-aws-jenkins	c5:10:33:34:fc:2d:0f:af:83:b1:11:d7:07...	key-02ca02538a5014d3a
<input type="checkbox"/>	sshaws2	8a:7c:84:39:11:c7:24:0d:df:52:ad:1e:9...	key-03a198a1496e11bf4

Security details

IAM Role: -

Owner ID: 515788519955

Security groups: sg-08b5b19f8149326c4 (ssh-jenkins)

Inbound rules

Filter rules

Port range	Protocol	Source	Security groups
80	TCP	0.0.0.0/0	ssh-jenkins
8888	TCP	0.0.0.0/0	ssh-jenkins
8080	TCP	0.0.0.0/0	ssh-jenkins
8000	TCP	0.0.0.0/0	ssh-jenkins
22	TCP	0.0.0.0/0	ssh-jenkins

Outbound rules

Filter rules

Port range	Protocol	Destination	Security groups
All	All	0.0.0.0/0	ssh-jenkins

Storing Delopers project on Gih Hub

1) Creating a repository where our site with a database and playbooks Will be stored

The screenshot shows a GitHub repository page for 'Final-Project' under the user 'EpamDevops'. The repository is private and was generated from 'EpamDevops/DevOps_online_Dnipro_2020Q3Q4'. The page displays a list of files and folders, including 'html', 'Final_Project_EPAM_Summer_2020_Iv...', 'RH.pem', 'SSH-AWS-2Deploy.pem', 'ansible.cfg', 'hostsjenans.txt', 'jenans.yml', and 'ssh-aws-jenkins.pem'. The commit history shows a recent commit '3210snoop3210 adding private keys AWS' by 'scfd35' 42 seconds ago. The right sidebar contains sections for 'About', 'Releases', 'Packages', and 'Languages'. The 'Languages' section shows a bar chart with the following data:

Language	Percentage
CSS	37.1%
HTML	28.8%
SCSS	25.9%
JavaScript	8.2%

The footer of the page includes the GitHub logo, copyright information '© 2020 GitHub, Inc.', and links for Terms, Privacy, Security, Status, Help, Contact GitHub, Pricing, API, Training, Blog, and About.

Steps on Git Hub

2) Updating code with Github Web Hook

The screenshot shows the GitHub interface for a repository named 'Final-Project' under the user 'EpmDevOps'. The page is titled 'Webhooks / Manage webhook' and shows the configuration for a specific webhook. A success message at the top states 'Okay, the hook was successfully updated.' The left sidebar contains a menu with options: Options, Manage access, Security & analysis, Branches, Webhooks (highlighted), Notifications, Integrations, Deploy keys, Secrets, and Actions. The main content area displays the following fields and options:

- Payload URL ***: `http://3.138.131.172:8080/github-webhook/`
- Content type**: `application/x-www-form-urlencoded`
- Secret**: (Empty field)
- Which events would you like to trigger this webhook?**:
 - ☒ Just the push event.
 - ☐ Send me everything.
 - ☐ Let me select individual events.
- Active**: ☒ (We will deliver event details when this hook is triggered.)

At the bottom, there are two buttons: 'Update webhook' (green) and 'Delete webhook' (red). Below the configuration section, a 'Recent Deliveries' table shows one entry:

Delivery ID	Status	Timestamp
463a4108-226b-11eb-988f-89aec567764a	redelivery	2020-11-09 11:28:05

Steps on Jenkins

1) Integrating our GitHub Repository to Jenkins Project

← → ↻ Не защищено | 3.138.131.172:8080/job/AWS-Jenkins-Pipe/configure

Jenkins > AWS-Jenkins-Pipe >

General Управление исходным кодом Триггеры сборки Среда сборки Сборка Постсборочные операции

Управление исходным кодом

☐ Her
☒ Git

Repositories

Repository URL:

Credentials:

Branches to build

Branch Specifier (blank for 'any'):

Просмотрщик репозитория: (Автоматический)

Additional Behaviours:

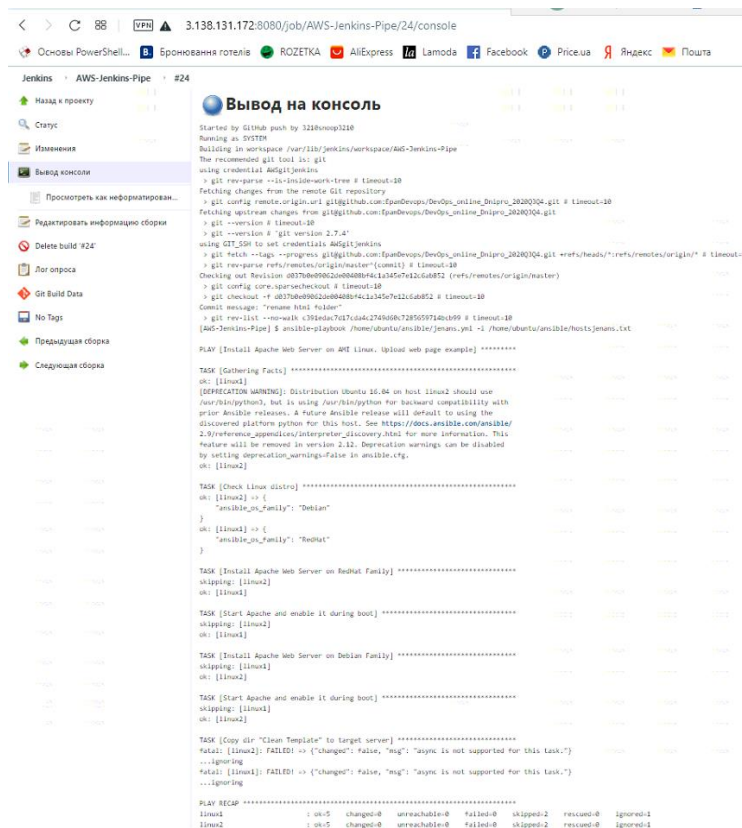
Триггеры сборки

☐ Trigger builds remotely (e.g., from scripts)
☐ Build after other projects are built
☐ Запускать периодически
☒ GitHub hook trigger for GITScm polling
☒ Опрашивать SCM об изменениях

Расписание:

Steps on Jenkins

2) CI/CD by Build In step –Ansible plugin

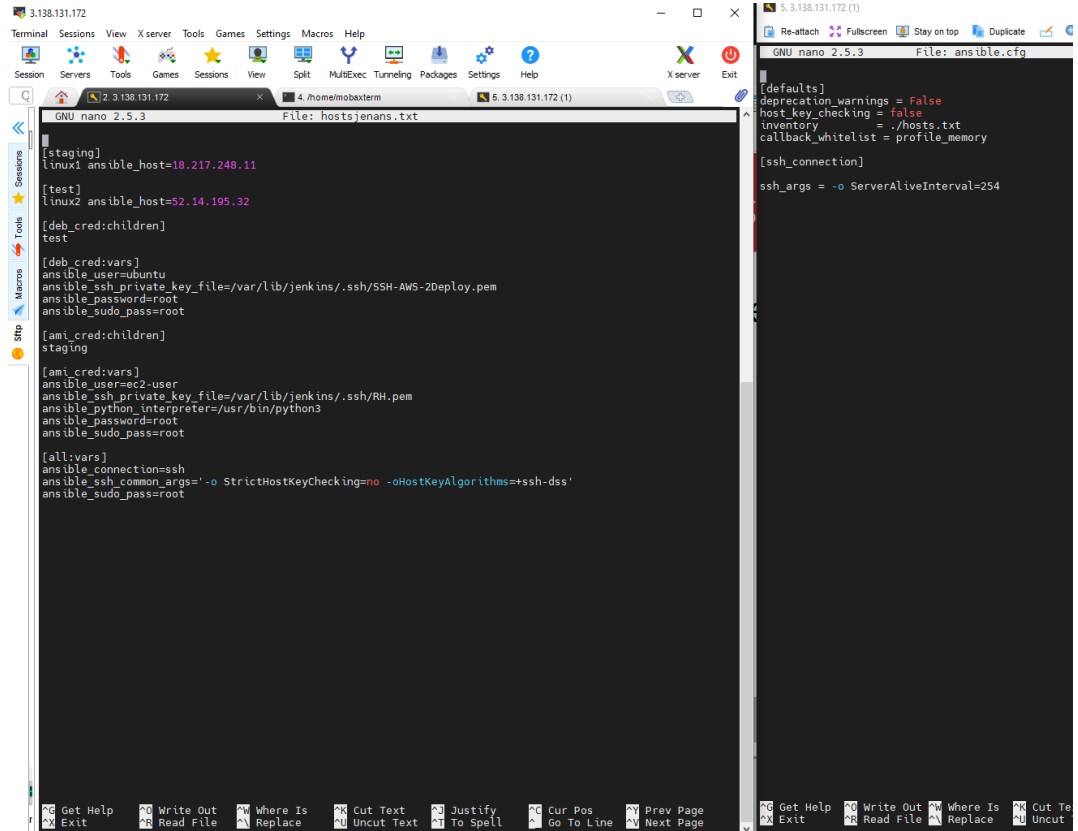


The screenshot shows the Jenkins console output for a build named 'AWS-Jenkins-Pipe' (ID #24). The output is titled 'Вывод на консоль' (Output to console). The build is started by a GitHub push to the 'main' branch. The output shows the following steps:

- Started by GitHub push to 328b0001210
- Running as SYSTEM
- Building in workspace /var/lib/jenkins/workspace/AWS-Jenkins-Pipe
- The recommended git tool is: git
- using credential AdgGitJenkins
- > git rev-parse --is-inside-work-tree & timeout 10
- Fetching changes from the remote Git repository
- > git config remote.origin.url git@github.com:spandevops/DevOps_online_Builds_202004.git & timeout 10
- Fetching upstream changes from git@github.com:spandevops/DevOps_online_Builds_202004.git
- > git --version & timeout 10
- > git --version & 'git version 2.7.4'
- using GIT_SSH to set credentials AdgGitJenkins
- > git fetch --tags --prune git@github.com:spandevops/DevOps_online_Builds_202004.git --refs/heads/*:refs/remotes/origin/* & timeout 10
- > git rev-parse refs/remotes/origin/master (commit) & timeout 10
- Checking out Revision 0870b09020b080bfc1a3a7e12c6d872 (refs/remotes/origin/master)
- > git config core.sshCommand & timeout 10
- > git checkout -f 0870b09020b080bfc1a3a7e12c6d872 & timeout 10
- Commit message: "rename html folder"
- > git rev-list --max-count=1 0870b09020b080bfc1a3a7e12c6d872 & timeout 10
- [AWS-Jenkins-Pipe] \$ ansible-playbook /home/ubuntu/ansible/hosts/jenkins.txt
- PLAY [Install Apache Web Server on AWS linux, upload web page example] *****
- TASK [Gathering Facts] *****
- ok: [linux1]
- (DEPRECATION WARNING): Distribution Ubuntu 16.04 on host linux2 should use /usr/bin/python, but is using /usr/bin/python for backward compatibility with prior Ansible releases. A future release will default to using the discovered platform python for this host. See https://docs.ansible.com/ansible/2.9/reference_appendices/interpreter_discovery.html for more information. This feature will be removed in version 2.12. Deprecation warnings can be disabled by setting deprecation_warnings=False in ansible.cfg.
- ok: [linux2]
- TASK [Check Linux distro] *****
- ok: [linux2] => {
- "ansible_os_family": "Debian"
- }
- ok: [linux1] => {
- "ansible_os_family": "RedHat"
- }
- TASK [Install Apache Web Server on RedHat Family] *****
- shipping: [linux2]
- ok: [linux1]
- TASK [Start Apache and enable it during boot] *****
- shipping: [linux2]
- ok: [linux1]
- TASK [Install Apache Web Server on Debian Family] *****
- shipping: [linux1]
- ok: [linux2]
- TASK [Start Apache and enable it during boot] *****
- shipping: [linux1]
- ok: [linux2]
- TASK [Copy the "Clear Template" to target server] *****
- fatal: [linux1]: FAILED! => ("changed": false, "msg": "async is not supported for this task.")
- ...ignoring
- fatal: [linux1]: FAILED! => ("changed": false, "msg": "async is not supported for this task.")
- ...ignoring
- PLAY RECAP *****
- linux1 : ok=5 changed=0 unreachable=0 failed=0 skipped=2 rescued=0 ignored=0
- linux2 : ok=5 changed=0 unreachable=0 failed=0 skipped=2 rescued=0 ignored=0

Steps on Ansible

- 1) Creating inventory file
“hostsjenans.txt”
- 2) Config file
“ansible.cfg”



The image shows two terminal windows side-by-side. The left window, titled '3.138.131.172', shows the creation of the inventory file 'hostsjenans.txt' using the nano text editor. The file content is as follows:

```
[staging]
linux1 ansible_host=18.217.248.11

[test]
linux2 ansible_host=52.14.195.32

[deb_cred:children]
test

[deb_cred:vars]
ansible_user=ubuntu
ansible_ssh_private_key_file=/var/lib/jenkins/.ssh/SSH-AWS-2Deploy.pem
ansible_password=root
ansible_sudo_pass=root

[ami_cred:children]
staging

[ami_cred:vars]
ansible_user=ec2-user
ansible_ssh_private_key_file=/var/lib/jenkins/.ssh/RH.pem
ansible_python_interpreter=/usr/bin/python3
ansible_password=root
ansible_sudo_pass=root

[all:vars]
ansible_connection=ssh
ansible_ssh_common_args='-o StrictHostKeyChecking=no -o HostKeyAlgorithms=+ssh-dss'
ansible_sudo_pass=root
```

The right window, titled '5.3.138.131.172 (1)', shows the creation of the configuration file 'ansible.cfg' using the nano text editor. The file content is as follows:

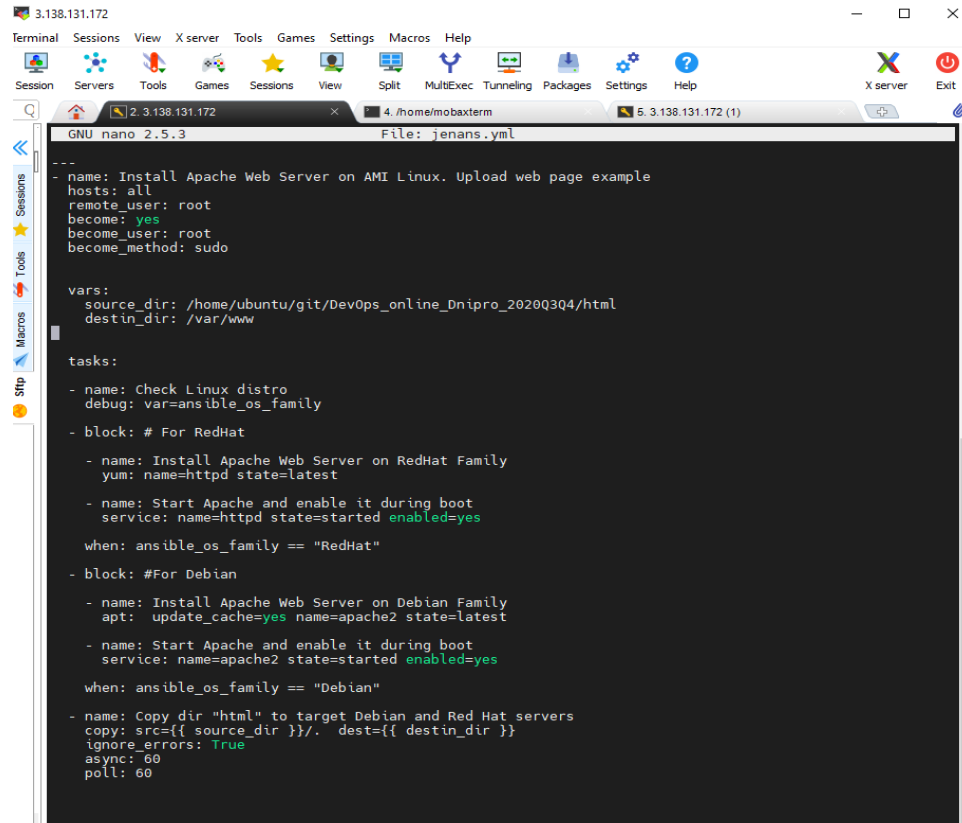
```
[defaults]
deprecation_warnings = False
host_key_checking = false
inventory = ./hosts.txt
callback_whitelist = profile_memory

[ssh_connection]
ssh_args = -o ServerAliveInterval=254
```

Steps on Ansible

*3) Creating playbook
Jenans.yml, which
Include some steps:*

- hosts parameters*
- vars*
- tasks*



The screenshot shows a terminal window with the title bar '3.138.131.172'. The window contains the content of a file named 'jenans.yml' being edited in 'GNU nano 2.5.3'. The playbook content is as follows:

```
---
- name: Install Apache Web Server on AMI Linux. Upload web page example
  hosts: all
  remote_user: root
  become: yes
  become_user: root
  become_method: sudo

  vars:
    source_dir: /home/ubuntu/git/DevOps_online_Dnipro_2020Q3Q4/html
    destin_dir: /var/www

  tasks:
    - name: Check Linux distro
      debug: var=ansible_os_family

    - block: # For RedHat
      - name: Install Apache Web Server on RedHat Family
        yum: name=httpd state=latest

      - name: Start Apache and enable it during boot
        service: name=httpd state=started enabled=yes

      when: ansible_os_family == "RedHat"

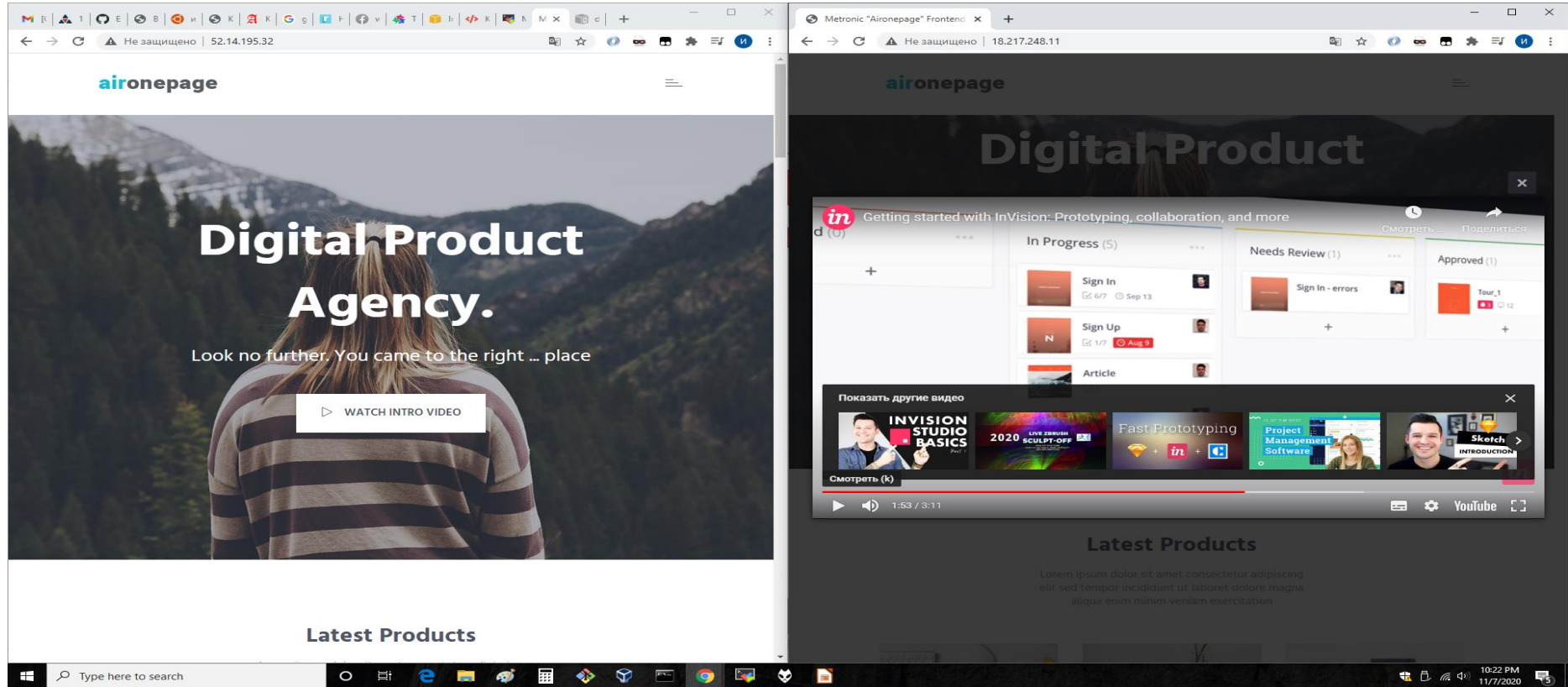
    - block: #For Debian
      - name: Install Apache Web Server on Debian Family
        apt: update_cache=yes name=apache2 state=latest

      - name: Start Apache and enable it during boot
        service: name=apache2 state=started enabled=yes

      when: ansible_os_family == "Debian"

    - name: Copy dir "html" to target Debian and Red Hat servers
      copy: src={{ source_dir }}/. dest={{ destin_dir }}
      ignore_errors: True
      async: 60
      poll: 60
```

Deployed Web sites on servers at work



Future steps

What we can made better in future :

- we could use wordpress site db to deploy by ansible;*
- we could use docker containers for building and testing artefact (saving in volume). After that we killing containers to saving machines power.*

End of presentation

*Thanks for watching my presentation,
waiting for your questions!*