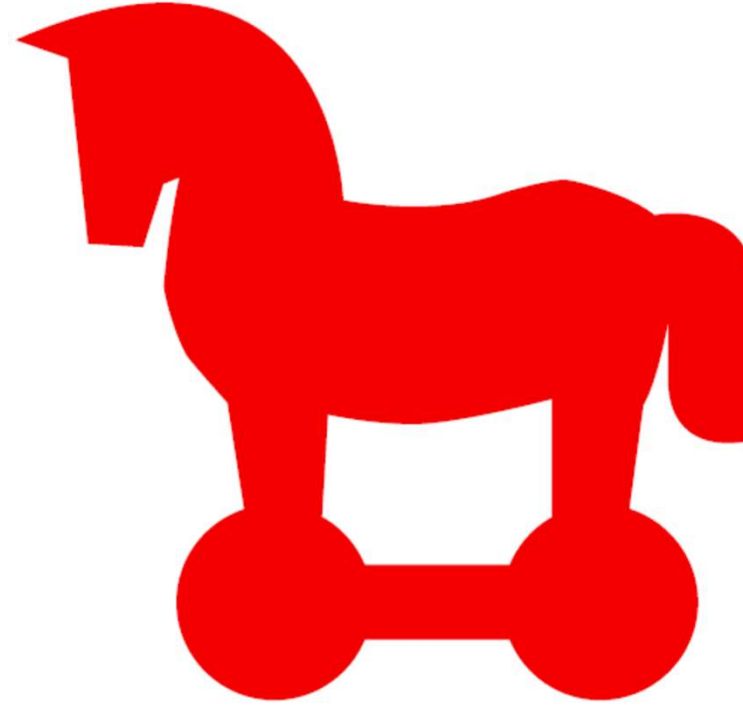


**GANTEK**  
INTEGRATING FUTURE



**GANTEK**  
**ACADEMY**

**40 YILLIK TECRÜBE İLE**

**ANSIBLE**

# Ansible İçerik

- Kurulum ve konfigürasyon,
- Envanter yönetimi,
- «Ad Hoc» komutları,
- «Play» ve «Playbook» yazılması,
- «Modules»
- Değişkenler
- «Secrets» ve «facts» kavramları
- Rol tanımlamaları
- Sektörün en iyi pratiklerini içeren örnekler



# Ansible nedir?

- Ansible, bir topluluk tarafından geliştirilen açık kaynak kodlu bir otomasyon aracıdır.
- Python'da yazılmıştır.
- Komut satırından birçok işlem yapılabilir:
  - Sistemleri yapılandırılabilir,
  - Yazılımları kurabilir ve devreye alabilir,
  - Uygulama geliştirme süreçlerinde kullanılan karmaşık iş akışlarını çalıştırabilir,
  - Sistem güncellemelerini yapabilir,
  - Ve bir çok benzer iş yapabilir.
- İki sürümü vardır:
  - Toplum Ansible: Python çalışan her bilgisayarda kullanabilir.
  - Red Hat Ansible Otomasyon platformu: ücretli ve kurumsal kullanıma yönelik geliştirilmiştir.

# Ansible nasıl çalışır?

- Ansible basit bir sistemdir ve gücü de basitliğinden gelir.
- Ansible'da yapılacak işler, «**playbook**» adı altında YAML dosyaları şeklinde oluşturulur. Her «**playbook**» içinde birden fazla «**play**» bulunabilir.
- **playbook**'lar Ansible sisteminde işlenerek, istenen otomasyon işleri gerçekleştirilir.
- Ansible'ın kullanımı için, yönetilecek bilgisayarlara aracı bir yazılım yüklenmesine gerek yoktur. Bütün iş SSH üzerinden yapılır.
- Ansible birden fazla bilgisayar üzerinde otomasyon işlerini yapabilir. Kontrol makinesinin çok kapsamlı bir makine olmasına gerek yoktur.
- Bulut yapılarında ve hibrit bulut sistemlerinde çalıştırılabilir.

# Ansible Özellikleri

- Ansible, yönetilen sistemlerde aynı işleri tekrardan yapmamak için kaynakların durum bilgilerini takip eder. Eğer bir paket kurulduysa, o paket yeniden kurulmaz.
- Eğer istenen bir kurulum veya değişiklik talebi, sistemde bir değişiklik yapmayacaksa o işlem yapılmaz. Buna «**idempotent**» davranış denir.
- Ansible'da otomasyon betikleri yazıldığında, değişkenler, koşullar ve döngüler tanımlanabilir. Bu şekilde otomasyon daha verimli şekilde yapılır.
- Ansible, idare ettiği konaklarla ilgili her türlü yapılandırma bilgisini (ağ kartları, grafik sistemi, IP'ler, vb) toplar. Bu bilgilere gerçekler (facts) adı verilir.
- Ansible, yüzlerce modülle birlikte gelir. «**apt**», «**npm**», «**gem**», «**composer**» gibi modüller çeşitli işlemleri yapabilir. Ayrıca istenirse, «**Ansible Galaxy**» portalı kullanılarak bu modüllere yenileri eklenebilir.

# ANSIBLE KURULUMU



# Ansible nasıl kurulur?

- Ubuntu'da kontrol makinesinde kurulum için aşağıdaki komutların girilmesi yeterlidir.  
\$ sudo apt-add-repository ppa:ansible/ansible  
\$ sudo apt update  
\$ sudo apt install ansible -y  
\$ ansible --version  
\$ ssh-keygen
- Ansible ile kontrol edilecek uzak makinelerde, OpenSSH kurulması gereklidir. Bu makinelerde «**sudo**» yetkisi olan bir kullanıcı bulunmalıdır.  
\$ sudo apt install openssh-server  
\$ sudo systemctl enable ssh  
\$ sudo useradd -s /bin/bash -d /home/ansibleuser -m -G sudo ansibleuser  
\$ sudo passwd ansibleuser  
\$ su - ansibleuser
- Kontrol makinesinde «**ssh-copy-id ansible@uzakmakineIP**» komutuyla, uzak makinelere kontrol makinesindeki anahtarın aktarılması sağlanabilir.

# Ansible'a konakların tanıtılması

- Ansible'ın idare edeceği sistemler, **/etc/ansible/hosts** dosyasında tanımlanmalıdır. Bu amaçla «**sudo nano /etc/ansible/hosts**» ile dosyaya aşağıdaki satırlar eklenebilir. Burada IP numaraları arka arkaya eklenebilir:

```
[sunucular]
192.168.211.139
```

- Envanterin görülmesi için (YAML formatında)

```
$ ansible-inventory --list -y
all:
  children:
    sunucular:
      hosts:
        192.168.211.139: {}
    ungrouped: {}
```

## Bağlantının test edilmesi

- **/etc/ansible/hosts** dosyasına eklediğimiz ve kontrol etmek istediğimiz sistemlere yapılacak bağlantının test edilmesi için aşağıdaki komut kullanılabilir:

```
$ ansible all -m ping -u ansibleuser
192.168.211.139 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
```

# ÇEŞİTLİ AD HOC KOMUTLAR

## Ad Hoc komutlar

- Ansible, bütün konaklarda istenen tekil komutları çalıştırabilir. Bunlara Ansible terminolojisinde «**Ad Hoc**» komutlar denir.
- Genel yapısı şöyledir:  
`ansible <konaklar> -m <modül> -a "modül parametreler"`
- Ad hoc komutların çalıştığını görebilmek için ilk kullanılması gereken modül «**ping**» modülür.

```
$ ansible all -m ping
192.168.211.153 | SUCCESS => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "ping": "pong"
}
```

## Ad Hoc komutlar

- Ansible, paralel olarak 5 konakta istenen komutları çalıştırabilir.
- Bu 5 konakta işlem bittikten sonra, bir 5 konakta daha komut çalıştırılır.
- Paralel çalıştırma için, istenirse, «**-f**» argümanı ile daha fazla paralel çalıştırma yani «**fork**» imkanı sağlanabilir.
- Genel yapısı şöyledir:  

```
ansible all -m ping -f 10
```
- Eğer istenirse, **ansible.cfg**'de varsayılan fork miktarı değiştirilebilir.

# Ansible ad hoc komutları

- Uzaktaki konakların disk kullanımını görebilmek için «df -h» komutu bütün konaklarda çalıştırılabilir.

```
$ ansible all -a "df -h" -u ansibleuser
192.168.211.139 | CHANGED | rc=0 >>
Filesystem      Size  Used Avail Use% Mounted on
udev            5.7G   0    5.7G   0% /dev
tmpfs           1.2G  2.3M   1.2G   1% /run
/dev/sda5       98G   63G   30G   68% /
tmpfs           5.7G  1.1M   5.7G   1% /dev/shm
tmpfs           5.0M  4.0K   5.0M   1% /run/lock
tmpfs           5.7G   0    5.7G   0% /sys/fs/cgroup
tmpfs           5.7G   0    5.7G   0% /run/qemu
/dev/loop0      128K  128K     0 100% /snap/bare/5
/dev/loop3      347M  347M     0 100% /snap/gnome-3-38-2004/115
/dev/loop6       46M   46M     0 100% /snap/snap-store/599
/dev/loop5       82M   82M     0 100% /snap/gtk-common-themes/1534
/dev/loop9       50M   50M     0 100% /snap/snapd/17950
/dev/loop7       46M   46M     0 100% /snap/snap-store/638
/dev/loop4      347M  347M     0 100% /snap/gnome-3-38-2004/119
/dev/loop10      92M   92M     0 100% /snap/gtk-common-themes/1535
/dev/sda1       511M  4.0K   511M   1% /boot/efi
tmpfs           1.2G  48K   1.2G   1% /run/user/1000
/dev/loop11      64M   64M     0 100% /snap/core20/1822
/dev/loop1       50M   50M     0 100% /snap/snapd/18596
/dev/loop8       64M   64M     0 100% /snap/core20/1852
tmpfs           1.2G  8.0K   1.2G   1% /run/user/1002
```

## Gerçeklerin (Facts) toplanması

- «**setup**» modülü çalıştırılarak, uzaktaki düğümle ilgili gerçekler (facts) toplanabilir. Çalıştırılacak komutlardan önce bu tür bir işlem yapılmaktadır.

```
$ ansible all -m setup -u ansibleuser
192.168.211.139 | SUCCESS => {
  "ansible_facts": {
    "ansible_all_ipv4_addresses": [
      "192.168.211.139"
    ],
    "ansible_all_ipv6_addresses": [
      "fe80::5042:c3ff:fee6:99c1",
      "fe80::8c20:dd5a:c76d:79d8"
    ],
    "ansible_apparmor": {
      "status": "enabled"
    },
    "ansible_architecture": "x86_64",
    "ansible_bios_date": "02/27/2020",
    "ansible_bios_vendor": "Phoenix Tech",
    "ansible_bios_version": "6.00",
```

```
    "ansible_board_asset_tag": "NA",
    "ansible_board_name": "440BX Desktop Platform",
    "ansible_board_serial": "NA",
    "ansible_board_vendor": "Intel Corporation",
    "ansible_board_version": "None",
    "ansible_br_0459dcfffcbl": {
      "active": false,
      "device": "br-0459dcfffcbl",
      "features": {
        "esp_hw_offload": "off [fixed]",
        "esp_tx_csum_hw_offload": "off [fixed]",
        "fcoe_mtu": "off [fixed]",
        "generic_receive_offload": "on",
        "generic_segmentation_offload": "on",
        "highdma": "on",
        "hsr_dup_offload": "off [fixed]",
        "hsr_fwd_offload": "off [fixed]",
```



# ad hoc komutlarla yazılım kurulumu

- Ansible'in modülleri ad hoc komutlar olarak çalıştırılabilir.
- Örneğin, karşıdaki hostta **htop**'un en son sürümünü kurabilmek için şu komut verilebilir (**apt** modülüyle)

```
$ ansible all -m apt -a "name=htop state=latest" -u ansibleuser  
192.168.211.139 | SUCCESS => {  
  "ansible_facts": {  
    "discovered_interpreter_python": "/usr/bin/python3"  
  },  
  "cache_update_time": 1680546632,  
  "cache_updated": false,  
  "changed": false  
}
```

«apt» modülü çalıştırılıyor.

# Kontrol düğümünden diğerlerine dosya kopyalama

- Kontrol düğümü ve yönetilen düğümler arasında dosya kopyalanması işlemi şöyle yapılabilir:

```
$ ansible all -m copy -a "src=./deneme.txt dest=~/.deneme.txt" -u ansibleuser
192.168.211.139 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": true,
  "checksum": "4dfae1a598767458694dffffda15560f7519a5f56",
  "dest": "/home/ansibleuser/deneme.txt",
  "gid": 1002,
  "group": "ansibleuser",
  "md5sum": "dbeb5936bdc66ffe06bc0717ddecba1a3",
  "mode": "0664",
  "owner": "ansibleuser",
  "size": 18,
  "src": "/home/ansibleuser/.ansible/tmp/ansible-tmp-1680550731.449603-12045-241565159649839/source",
  "state": "file",
  "uid": 1002
}
```

«copy» modülü çalıştırılıyor.

# Dosya izinlerinin değiştirilmesi

- Yönetilen düğümdeki bir dosyanın izinleri değiştirilebilir:

```
$ ansible all -m file -a "dest=/home/ansibleuser/deneme.txt mode=600 owner=root  
group=root" --become Y -u ansibleuser  
BECOME password:  
192.168.211.139 | CHANGED => {  
  "ansible_facts": {  
    "discovered_interpreter_python": "/usr/bin/python3",  
  },  
  "changed": true,  
  "gid": 0,  
  "group": "root",  
  "mode": "0600",  
  "owner": "root",  
  "path": "/home/ansibleuser/deneme.txt",  
  "size": 18,  
  "state": "file",  
  "uid": 0  
}
```

«file» modülü çalıştırılıyor.

# Servislerin çalıştırılması

- Servislerin çalıştırılması sağlanabilir

```
$ ansible all -m service -a "name=ssh state=restarted" --become -K -u ansibleuser
BECOME password:
192.168.211.139 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": true,
  "name": "ssh",
  "state": "started",
  "status": {
    "ActiveEnterTimestamp": "Mon 2023-04-03 21:18:41 +03",
    "ActiveEnterTimestampMonotonic": "22613926008",
    "ActiveExitTimestampMonotonic": "0",
    "ActiveState": "active",
    "After": "system.slice systemd-journald.socket auditd.service network.target
basic.target sysinit.target -.mount",
    "AllowIsolate": "no",
    "AllowedCPUs": "",
    "AllowedMemoryNodes": "",
    "AmbientCapabilities": "",
```

«service» modülü çalıştırılıyor.

# Sistemlerin yeniden başlatılması

- Sistemin **reboot** edilmesi sağlanabilir

```
$ ansible all -a "/sbin/reboot" -b -K
```

- Verilecek komutta sistemde değişiklik yapılması istenmiyorsa ve komutun testi isteniyorsa, **--check** opsiyonu kullanılabilir.

```
$ ansible all -a "df -h" --check -u ansibleuser  
192.168.211.139 | SKIPPED
```

## Ad hoc komut örnekleri

- Sistemlerin ne zamandan beri çalıştığını getirir

```
$ ansible all -m command -a uptime
192.168.211.153 | CHANGED | rc=0 >>
  13:58:04 up 3 min,  1 user,  load average: 0.07, 0.16, 0.08
$ ansible all -a uptime
192.168.211.153 | CHANGED | rc=0 >>
  13:59:40 up 4 min,  1 user,  load average: 0.01, 0.12, 0.07
```

- Uzak sistemlerde bellek miktarının izlenmesi

```
$ ansible all -a "free -m"
192.168.211.153 | CHANGED | rc=0 >>
```

	total	used	free	shared	buff/cache	available
Mem:	7725	606	6555	7	563	6874
Swap:	2047	0	2047			

# Ad hoc komut örnekleri

- Fiziksel bellek miktarının izlenmesi

```
$ ansible all -m shell -a "cat /proc/meminfo|head -2"  
192.168.211.153 | CHANGED | rc=0 >>  
MemTotal:          7911264 kB  
MemFree:           6719304 kB
```

«-b» root kullanıcısı olunacağını «-K»  
ise SUDO şifresinin sorulmasını ister.

- sudo** yetkisiyle bir programın çalıştırılması

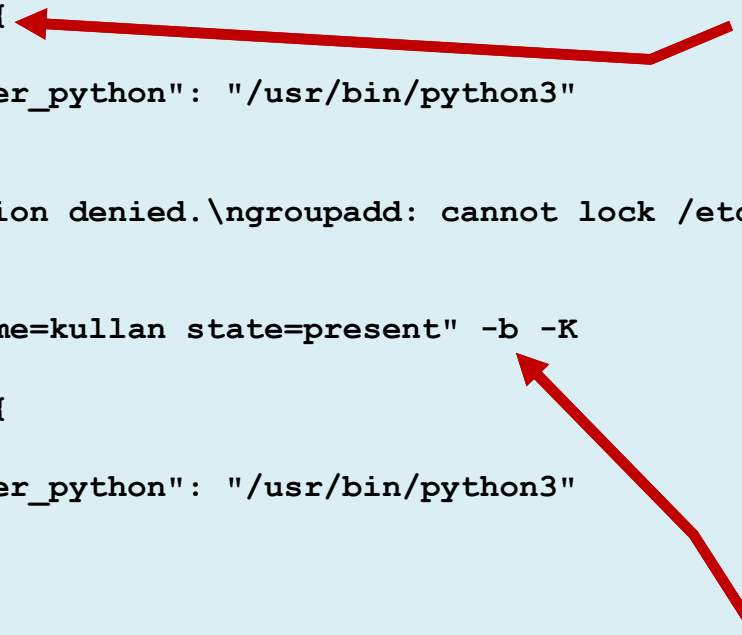
```
$ ansible all -m shell -a "cat /etc/shadow | grep -i ansibleuser" -b -K  
BECOME password:  
192.168.211.153 | CHANGED | rc=0 >>  
ansibleuser:$6$j/.7.znIwFVF6Ok1$Av/adQoymECJMbsPMhpn1YWFtguertXXShBNZmk490cARWI  
LwJ2hBuSkEylyt8NKBLlL73OUjckp6OHpqz2C31:19506:0:99999:7:::
```

«sudo» şifresini girmelisiniz.

## Ad hoc komut örnekleri

- Grup modülüyle «**kullan**» isimli kullanıcı grubunu oluşturalım. Burada «**state**» olarak «**present**» vereceğiz ve kullanıcı grubunun orada oluşturulmasını sağlayacağız.

```
$ ansible all -m group -a "name=kullan state=present"
192.168.211.153 | FAILED! => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": false,
  "msg": "groupadd: Permission denied.\ngroupadd: cannot lock /etc/group; try again later.\n",
  "name": "kullan"
}
$ ansible all -m group -a "name=kullan state=present" -b -K
BECOME password:
192.168.211.153 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": true,
  "gid": 1002,
  "name": "kullan",
  "state": "present",
  "system": false
}
```



Neden hata verdi?  
Verdiğimiz komut SUDO  
yetkisi istiyor.

-b ile «root» kullanıcısı  
olarak -K ile şifre sorarak  
komutu çalıştıracğız.



## Ad hoc komut örnekleri

- Grup modülüyle «**kullan**» isimli kullanıcı grubunu oluşturmuştuk. Bunu ortadan kaldıralım.

```
$ ansible all -m group -a "name=kullan state=absent" -b -K
BECOME password:
192.168.211.153 | CHANGED => {
  "ansible_facts": {
    "discovered_interpreter_python": "/usr/bin/python3"
  },
  "changed": true,
  "name": "kullan",
  "state": "absent"
}
```

Önceki komutta **state** olarak **present** vermiştik. Şimdi **absent** vererek kullanıcıyı ortadan kaldıracamız.

## Ad hoc komut örnekleri

- «**cron**» modülüyle, periyodik işleri çalıştıracak mekanizmayı oluşturabiliriz.

```
# Run the job every 15 minutes
$ ansible all -m cron -a "name='daily-cron-all-servers' minute=*/15
job='/path/to/minute-script.sh'"
# Run the job every four hours
$ ansible all -m cron -a "name='daily-cron-all-servers' hour=4
job='/path/to/hour-script.sh'"
# Enabling a Job to run at system reboot
$ ansible all -m cron -a "name='daily-cron-all-servers' special_time=reboot
job='/path/to/startup-script.sh'"
# Scheduling a Daily job
$ ansible all -m cron -a "name='daily-cron-all-servers' special_time=daily
job='/path/to/daily-script.sh'"
# Scheduling a Weekly job
$ ansible all -m cron -a "name='daily-cron-all-servers' special_time=weekly
job='/path/to/daily-script.sh'"
```

# ANSIBLE SÖZLÜĞÜ VE İLGİLİ KAVRAMLAR

[https://docs.ansible.com/ansible/latest/reference\\_appendices/glossary.html](https://docs.ansible.com/ansible/latest/reference_appendices/glossary.html)

# Ansible Mimarisi

- Ansible'in mimarisi basittir ve bu durum onu karmaşık olmayan bir çok amaçla kullanılabilmesini sağlamaktadır.
- Ansible'in nasıl çalıştığını anlamak için çeşitli kavramları öğrenmek ve Ansible içindeki yerini anlamak gereklidir.
- Temel kavramlar şunlardır:
  - Modüller
  - Modül yardımcı programları
  - Eklentiler
  - Envanter
  - Oyun (play)
  - Görev (task)
  - Rol (Role)
  - Derleme (collection)
  - Oyun kitabı (playbook)
  - Ansible Galaxy
  - Ve diğer kavramlar

# Ansible temel kavramlar

- Ansible'da kullanılan terminolojinin bir kısmı tiyatro sahnelerinden alınmıştır.
  - **Oyun** (play): uzaktaki düğümde yapılan minimal iş
  - **Oyun kitabı** (playbook): birden çok oyundan oluşan yapı
  - **Görev** (task): Bir oyun içinde çalıştırılan tek bir komut
  - **Rol**: Geniş kapsamlı bir kavram olarak bazı bilgilerin organize şekilde tutulmasını sağlar. Karmaşık oyun kitaplarının birden fazla dosyada tutulmasını sağlayan ve bu şekilde bunların tekrar kullanımını kolaylaştıran bir mekanizmadır. Bir oyun kitabı, rol dizinine dönüştürülebilir.
  - **Orkestrasyon**: bir oyunda yer alan bütün düğümlerin ve oyunların eşgüdümlü halde yürütülmesi

# Kavramlar

- **Kontrol düğümü** (Control node): Ansible'ın yüklü olduğu sistemdir. Birden fazla kontrol düğümü olabilir. Çok fazla kaynağa ihtiyaç yoktur. Windows makine kontrol düğümü olamaz (WSL2 kullanılabilir). Ancak, Windows makineler sadece yönetilebilir.
- **Yönetilen düğümler** (managed nodes): Ansible'ın yönettiği makinelerdir. Bunlara SSH üzerinde erişilir. En az Python 2.6 veya Python 2.5 veya yukarısının yüklü olması gereklidir.
- **Envanter** (Inventory): Ansible ile yönetilen bütün konakların listesidir. Ansible yüklendiğinde, varsayılan bir envanter dosyası yüklenir. İstenirse, kullanıcı kendi dizininde dinamik olarak envanter oluşturabilir. Veritabanlarından veya dışarıdaki bir kaynaktan envanter oluşturan betiklere envanter betiği adı verilir.
- **Oyun** (play): Birden fazla aşamadan yani görevden (task) oluşan otomasyon işidir.
- **Oyun-kitabı** (playbook): birbiri arkasına çalıştırılacak oyunları içeren YAML dosyasıdır.
- **Görev** (task): Oyun (play) içinde yer alan tek bir aşamada yapılan işe denir.
- **Rol (role)**: çeşitli amaçlarla kullanılan bir dizi paylaşılabilir çalıştırma kitaplarıdır. Örneğin, roller sayesinde bir veritabanı sunucusu kurulabilir, web sunucu yapılandırılabilir, PHP ortamı güncellenebilir.

# Oyun kitabı, oyun, görev ve roller

```
---
- hosts: all
  vars:
    backport: "stretch-backports"
    packages:
      - git
      - tree
      - curl
  pre_tasks:
    - name: Pre task 1
      debug: msg="Pretask"
    - name: Pre task 2
      debug: msg="Pretask"

  tasks:
    - name: Add the backport
      become: yes
      apt_repository:
        repo: deb http://ftp.debian.org/debian {{backport}} main
        filename: "{{backport}}"
        state: present

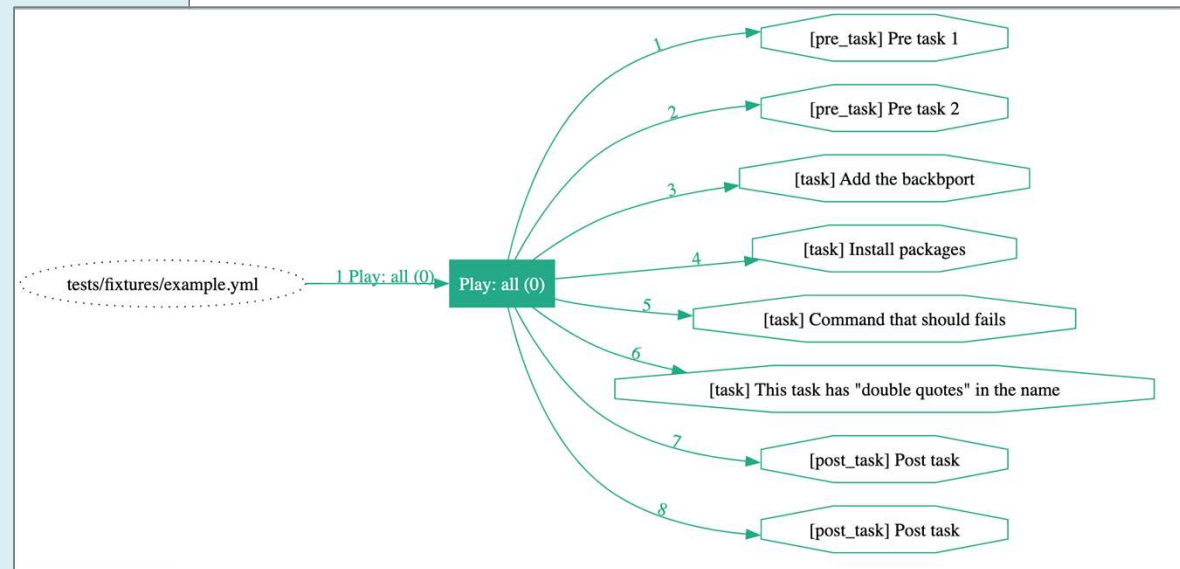
    - name: Install packages
      become: yes
      apt:
        name: "{{packages}}"
        state: present
        default_release: "{{backport}}"

    - name: Command that should fails
      command: /bin/false
      ignore_errors: true

    - name: This task has "double quotes" in the name
      command: /bin/true

  post_tasks:
    - name: Post task
      debug: msg="Post task 1"

    # We use the same name here just to test if we have two distinct nodes in the
    graph.
    - name: Post task
      debug: msg="Post task"
```



# Kavramlar

- **Ad Hoc:** Ansible'in yönetilen düğümlerde hızlı bir şekilde komut çalıştırmasına verilen addır. Uzakta çalışacak komutlar `usr/bin/ansible` programıyla çalıştırılır. İstenirse, envantere kayıtlı 50 sistem aynı anda yeniden başlatılabilir.
- **Orkestrasyon:** Orkestralarda bir parça icra edilirken, orkestradaki bütün enstrümanların notları uygun sırada çalıştırmasına benzer şekilde, otomasyon uygulamasının bütün görevleri uygun sırada yapmasına denir. Ansible'da orkestrasyon, **`usr/bin/ansible-playbook`** programıyla gerçekleştirilir.
- **ansible-core:** v2.11'den itibaren **`github.com/ansible/ansible`** reposundaki paketler ve komut satırı programlarına verilen isimdir.
- **Kontrol modu** (check mode): Ansible'in **`--check`** opsiyonuyla çalıştırılmasına verilen addır. Bu argümanla, Ansible, komutun uzaktaki sistemde değişiklik yapmasına izin vermez. Verilen komutun veya görevlerin verecekleri hataların görülmesi sağlanabilir.



# Kavramlar

- **Handlers:** bir servis üzerinde gerekli işlemi yapmak için kullanılırlar. Örneğin servis başlatılır veya durdurulur.
- **Modüller:** Bir görevi (task) gerçekleştirmek üzere yönetilen bir düğüme kopyalanacak kod veya bu düğümde çalıştırılacak programlardır.
  - Her modül belirli bir amaca yöneliktir. Örneğin, belirli bir veritabanında kullanıcıların yönetimi, sistemdeki kullanıcı yetkilerini tanımlamak veya bir ağ cihazında VLAN arayüzlerini tanımlamak gibi.
- **Derleme (collection):** eklentiler, roller, modüller ve diğer Ansible içeriğini bir araya getiren paketleme formatıdır. **ansible-core**'dan bağımsız olarak üretilen içeriklerdir.  
«**/usr/bin/ansible-galaxy collection install isimalanı.derleme**» komutuyla Ansible'a yüklenir.
  - Derlemelerin isimleri isim alanı ile başlar. cisco.ios, cisco.aci, cisco.nxos, vb

# Kavramlar

- **Paralel çalıştırma (forks)** : Ansible uzaktaki düğümlerde çalıştıracağı işlerin kaç tanesinin paralel çalıştırılacağını **--forks** argümanı ile değiştirebilir. Varsayılan olarak maksimum 5 görev paralel çalıştırılmaktadır.
- **Gerçekler (facts)**: uzaktaki düğümde, Ansible tarafından otomatik olarak keşfedilen gerçeklerdir.
  - Ansible tarafından uzaktaki düğümlerle ilgili keşfedilen gerçekler «**ansible <host> -m setup**» komutuyla görülebilir.
  - İstenirse toplanmayabilir.
- **Döngüler (loops)**: Ansible bir programlama dili olmamasına rağmen, **loop** adı verilen yapılara sahiptir. Bu yapı, listedeki görevlerin birden fazla sayıda tekrar edilmesini sağlar.
- **Koşullar (conditions)**: belirli bir görevi bir sistemde çalıştırma kararını vermek üzere Doğru veya Yanlış değerini veren bir ifadedir.

# Kavramlar

- **isimalanı**(namespace) : derlemelerin ilk isimleridir ve derlemenin ana konusunu belirler.
  - Örneğin, **cisco.ios.ios\_config** kategorisinde, **cisco** isim alanıdır ve bütün Cisco cihazların yapılandırılmasına yönelik ortak tanımdır.
- «**push**» ve «**pull**»: Ansible «**push**» mantığıyla çalışır ve iş akışı bu mantığa göre kurulmuştur. Kontrol düğümü, uzak sistemlere SSH üzerinden bağlantı kurar ve scp/sftp gibi yöntemlerle Python modüllerini o sisteme aktararak (yani **push** ederek), çalıştırır ve ardından modülleri siler. Normal olarak Ansible «**pull**» işlemini yapmaz. Ancak «**ansible-pull**» programı «**pull**» mantığıyla Ansible modüllerini uzak sisteme yükleyebilir.

## «**become**» kavramı

- Uzaktaki sistemlerde çalıştırılacak işlerin hangi kullanıcıyla yapılacağı ve hangi program üzerinden yetki artırımı yapılacağı konuları, «**become**» kavramıyla ilişkilidir.
- «**become**» kavramı, Ansible'da planlanması gereken konulardan biridir. Bu amaçla, **ansible.cfg** gibi yapılandırma dosyalarında kullanılacak yetki artırımı yöntemi (**sudo**, **su**, **machinectl**, vb) tanımlanmalıdır.
- Ayrıca envanter tanımlarının içinde hangi konakta hangi kullanıcının görevleri çalıştıracağı belirlenmelidir.
- «**root**» yetkisiyle otomasyon görevlerini gerçekleştirmek risk oluşturmaktadır.
- Bu nedenle, belirli işleri yapacak kadar yetkiye sahip kısıtlı yetkilere sahip kullanıcılar kullanmak daha uygun bir çözüm olur. Ancak, bunların kullanımı da bazı durumlarda sorun yaratabilir.

# Modüller

- Ansible'ın temel çalışma ilkesi, SSH üzerinden kontrol edilecek düğümlere bağlanarak «Ansible modülleri» denilen betik programları yüklemektir.
- Modüllerin büyük çoğunluğu sistemin ulaşılması istenen durumunu tanımlayacak parametreleri kabul ederek çalışır.
- Ansible, bu modülleri varsayılan olarak SSH üzerinden bağlandığı düğümlerde çalıştırır ve iş tamamlandığında siler.
- Modül kütüphaneniz herhangi bir makinede bulunabilir. Ansible, ana düğüm, sunucu, daemon/agent veya veritabanı olmaksızın çalışır.
- Kendi modüllerinizi yazabilirsiniz ama neden yazmanız gerektiğini düşünmeniz gerekir. Herhangi bir dilde yazılan modüller, sonuçlarını JSON olarak döndürmelidir.

# Modüller

- Ansible’da 450’den fazla modül bulunmaktadır.  
Modül listesine şu web sayfasından ulaşılabilir:  
`https://docs.ansible.com/ansible/2.9/modules/list\_of\_all\_modules.html`
- Çok sık kullanılan modüller şunlardır:
  - apt: APT paketleri
  - apt\_key: APT anahtarları
  - archive: arşiv dosyaları (tar, gz, vb)
  - at: belirli bir komutu belirli bir zamanda çalıştırma
  - aws\_\*: AWS bulut hizmetleri
  - azure\_\*: Azure bulut hizmetleri
  - capabilities: Linux özellikleri
  - copy: uzak konaklara dosya iletimi
  - cron: Linux’teki cron komutu
  - digital\_ocean\_\*: Digital Ocean bulut hizmetleri
  - docker\_\*: Docker
  - ec2\_\*: AWS
  - file: dosya özelliklerini değiştirme
  - filesystem: dosya sistemi oluşturma
  - find: kriterlere göre dosya sisteminde dosya arama
  - gcp\_\*: Google Cloud
  - group: Linux’te kullanıcı grupları
  - hostname: host ismi
  - iptables: Linux/Unix firewall
  - K8s\_\*: Kubernetes
  - mail: e-posta gönderme
  - os\_\*: OpenStack
  - reboot: makinenin yeniden başlatılması
  - stat: Dosya sisteminin durumu
  - systemd: Systemd servisleri
  - tempfile: geçici dosya oluşturma
  - timezone: zaman dilimi
  - user: kullanıcılar
  - win\_\*: Windows yönetimi

# Modüller

- «**ansible-doc**» komutu istenen modülle ilgili bilgi sunmaktadır.
- Ancak, 450 adet modülün olması, istenen özelliklere sahip modülün bulunmasını zorlaştırır.
- Örneğin eğer ping ile ilgili bir çalışma yapıyorsanız, «**ansible-doc -l**» diyerek bütün modülleri listeleyebilirsiniz.
- «**ping**» komutuyla ilgili modüllerin bulunması isteniyorsa «**ansible-doc -l | grep -i ping**» komutu verilebilir.
- Görüleceği üzere çok sayıda modül burada listelenecektir.

# Komut (command) modülleri

- **command** – hedef konaklarda komut çalıştırma
- **expect** – komut çalıştırır ve komut cevap verir.
- **psexec** – Windows konakta PsExec modeliyle komut çalıştırma
- **raw** – en düşük seviyede komut çalıştırma
- **script** – uzak düğümde lokal bir betik programını karşı tarafa aktardıktan sonra çalıştırma
- **shell** – hedefte kabuk komutları çalıştırma
- **telnet** – en düşük seviyeli telnet komutunu çalıştırma



# Modül yardımcı programları

- Modül yardımcı programları, birden fazla eklentide kullanılan ortak kodlardır.
- Bu kodlar kullanılarak yeni modüller yazabilirsiniz.
- Python koduna aşağıdaki satır eklenerek yeni bir modül hazırlanabilir.

```
from ansible.module_utils.basic import AnsibleModule
```

# Eklentiler (plugins)

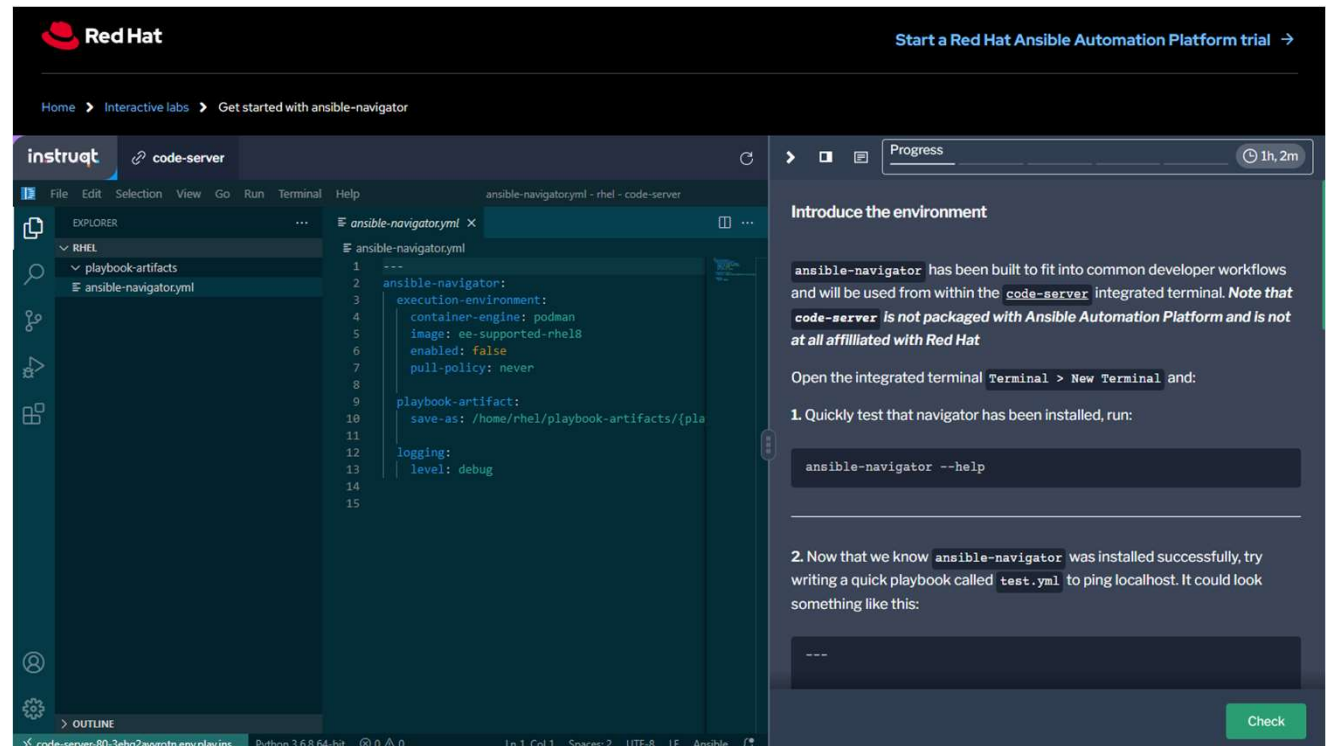
- Ansible'in var olan yeteneklerini geliştiren kodlardır.
- Ansible çok sayıda eklenti ile birlikte gelir.
  - Bağlantı eklentileri (connection plugin) yönetilen bir sisteme bağlanmak için kullanılır.
  - Filtre eklentileri (filter plugins) veri üzerinde değişiklik yaparlar.
  - Geriçağırma eklentileri (callback plugins), konsolda gösterilenleri kontrol ederler.
  - Aksiyon eklentileri (action plugins), modüller çalışmadan önce geri planda öncül işlemleri yaparlar.
  - Olma eklentileri (become plugins), hedef sistemde belirli bir işlemi yapabilmek için gerekli yetki artırımını yaparlar.
  - httpapi eklentileri, HTTP temelli API'lerle iletişim kurmak için ve cihazda görevleri çalıştırmak için kullanılır. Farklı tür httpapi eklentileri bulunmaktadır.
  - Cliconf, Docs, Inventory, Lookup, Modules, Module utilities, Netconf, Shell, Strategy, Terminal, Test, Vars eklentileri.

# Ansible Tower

- Ansible Tower, Ansible'in kurumsal sürümüdür.
- Sistem yöneticilerinin Ansible'in tüm özelliklerini kullanmasını sağlar.
- Web temelli grafik arayüzü bulunmaktadır.
- Çok «playbook»'dan oluşan iş akışlarını destekler.
- Rol temelli erişim kontrolü sağlar (açık kaynak kodlu Ansible'da bu özellik bulunmaz. Tower LDAP veya AD ortamlarına bağlanabilir)
- İşlerin belirli zamanlarda gerçekleşmesini sağlar.
- RESTful API'ye sahiptir ve gerekirse kullanıcılar kendi programlarını yazabilirler.
- Amazon EC2, Azure, vb bulut ortamlarına doğrudan bağlantı kurulabilir.
- Ek loglama özellikleri vardır.
- Gerçekleştirilen işle ilgili sonuç bilgisi değil canlı durum bilgisi alınmasını sağlar.

# Ansible Tower

- Red Hat Enterprise Linux (RHEL) ve Centos gibi sistemlerin üzerine kurulabilir.
- Kurulumdan önce PostgreSQL kurulması gereklidir.
- Deneme laboratuvarlarına aşağıdaki linkten erişilebilir:
  - <https://www.redhat.com/en/engage/redhat-ansible-automation-202108061218>



# ANSIBLE KOMUT SATIRI PROGRAMLARI

# Komut satırı programları

- Ansible komut satırından aşağıdaki programlarla kullanılabilir ve yönetilebilir:
  - **ansible**: tek bir komutun envanterde yer alan sistemlerde tanımlanması ve çalıştırılmasını sağlar. «**ad hoc**» komutlar bu programla çalıştırılabilir.
  - **ansible-config**: Ansible'in yapılandırabilmesi için kullanılan dosyaların, çevre değişkenlerinin, komut satırı opsiyonlarının ve değişkenlerin değiştirilerek yapılandırma yapan komut satırı programıdır.
  - **ansible-doc**: modül dokümanlarını gösterir.
  - **ansible-galaxy**: yeni rollerin, derlemelerin sisteme eklenmesi
  - **ansible-inventory**: uzak düğüm envanterini yönetir.
  - **ansible-playbook**: oyun kitabını çalıştırır.
  - **ansible-pull**: uzaktaki sistemde, «pull» mantığıyla Ansible'in indirilerek çalıştırılmasını sağlar.
  - **ansible-vault**: gizli bilgileri şifrelemek, getirmek ve kullanmak

# ansible programı

- Tek bir görevin, uzaktaki sistemlerde çalıştırılması sağlanabilir:

```
usage: ansible [-h] [--version] [-v] [-b] [--become-method BECOME_METHOD]
               [--become-user BECOME_USER]
               [-K | --become-password-file BECOME_PASSWORD_FILE]
               [-i INVENTORY] [--list-hosts] [-l SUBSET] [-P POLL_INTERVAL]
               [-B SECONDS] [-o] [-t TREE] [--private-key PRIVATE_KEY_FILE]
               [-u REMOTE_USER] [-c CONNECTION] [-T TIMEOUT]
               [--ssh-common-args SSH_COMMON_ARGS]
               [--sftp-extra-args SFTP_EXTRA_ARGS]
               [--scp-extra-args SCP_EXTRA_ARGS]
               [--ssh-extra-args SSH_EXTRA_ARGS]
               [-k | --connection-password-file CONNECTION_PASSWORD_FILE] [-C]
               [--syntax-check] [-D] [-e EXTRA_VARS] [--vault-id VAULT_IDS]
               [--ask-vault-password | --vault-password-file VAULT_PASSWORD_FILES]
               [-f FORKS] [-M MODULE_PATH] [--playbook-dir BASEDIR]
               [--task-timeout TASK_TIMEOUT] [-a MODULE_ARGS] [-m MODULE_NAME]
pattern
```

# ansible-config

- **ansible-config** komut satırı programı INI formatındaki **ansible.cfg** dosyasında, çevre değişkenlerinde, komut satırı opsiyonlarında, «**playbook**» anahtar kelimeleri ve değişkenler üzerinde değişiklik yaparak, Ansible'ın davranışını değiştirir.
- Program, kullanıcıların o andaki bütün yapılandırma parametrelerini görmelerini, varsayılan değerleri görüntülemelerini sağlar.
- Yapılandırma üzerinde yapılan değişiklikler aşağıda öncelik sırasında göre listelenmiş dosyalarda ilk bulunan dosyanın üzerinde gerçekleştirilir:



Azalan  
öncelik

**ANSIBLE\_CONFIG** (eğer çevre değişkeni tanımlandıysa)  
**ansible.cfg** (içinde bulunduğumuz dizinde)  
**~/ .ansible.cfg** (kullanıcının ev dizininde)  
**/etc/ansible/ansible.cfg**



## Varsayılan `ansible.cfg`

- Eğer istenirse, bütün parametreleri yorum satırı haline getirilmiş (`#` ve `;` ile işaretlenmiş) bir kopyasını oluşturulabilir.

```
$ ansible-config init --disabled  
[defaults]  
# (boolean) By default Ansible will issue a warning when a module or plugin  
# These warnings can be silenced by adjusting the following configurations  
;action_warnings=True  
  
# (list) Accept list of cowsay templates that are 'safe' to use, set to empty if  
;cowsay_enabled_stencils=bud-frogs, bunny, cheese, daemon, default, dragon, ele>  
  
# (string) Specify a custom cowsay path or swap in your cowsay implementation o>  
;cowpath=  
  
# (string) This allows you to chose a specific cowsay stencil for the banners o>  
;cow_selection=default  
  
# (boolean) This option forces color mode even when running without a TTY or th>  
;force_color=False
```

«`ansible-config init --disabled > ansible.cfg`» komutuyla, `ansible.cfg` dosyası oluşturulabilir.

# Varsayılan `ansible.cfg`

- Güncel yapılandırmada yer alan eklentileri de gösteren bir yapılandırma dosyası şöyle oluşturulabilir.

```
$ ansible-config init --disabled -t all
[defaults]
# (boolean) By default Ansible will issue a warning when a plugin is not found
# These warnings can be silenced by adjusting the following configurations
;action_warnings=True

# (list) Accept list of cowsay templates that are safe to use, set to empty list by default
;cowsay_enabled_stencils=bud-frogs, bunny, cheese, daemon, default, dragon, ele>

# (string) Specify a custom cowsay path or swap in your cowsay implementation o>
;cowpath=

# (string) This allows you to chose a specific cowsay stencil for the banners o>
;cow_selection=default

# (boolean) This option forces color mode even when running without a TTY or th>
;force_color=False
```

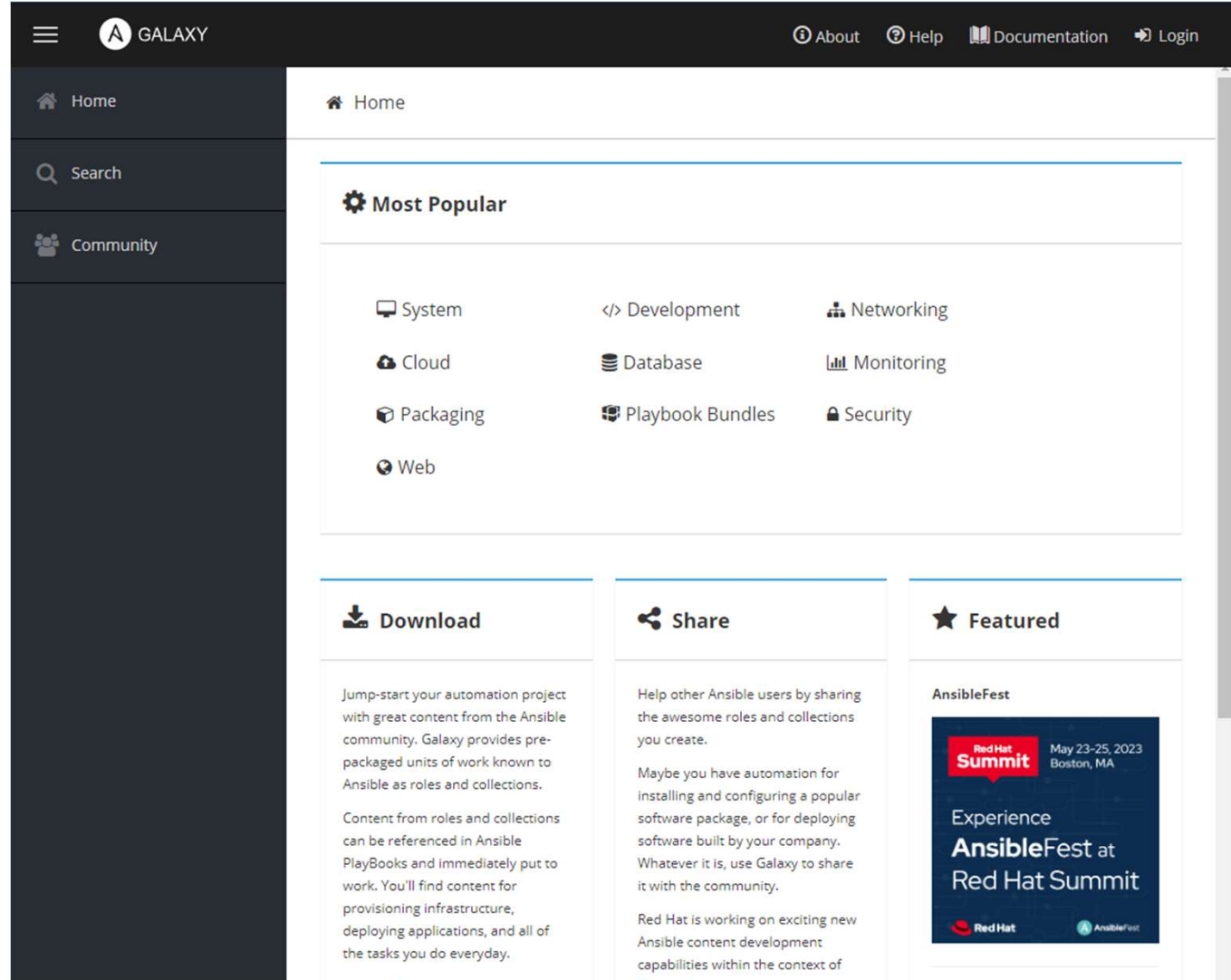
«`ansible-config init --disabled -t all`» komutuyla, bütün parametrelerle birlikte `ansible.cfg` dosyası oluşturulabilir.

# ansible-galaxy

- Ansible Galaxy, Ansible için rol ve derleme dağıtım yöneticisidir.
- **ansible-galaxy** komut satırı programı üzerinden, Ansible Galaxy'de yer alan yeni özellikleri indirerek kullanmaya başlayabilirsiniz.
- Ansible Galaxy'nin web arayüzü **<https://galaxy.ansible.com/>** sitesinde yer almaktadır.
- Web arayüzünde, ne tür bir özellik aranıyorsa, kategoriler içinde bulabilirsiniz ve ilgili dokümantasyonu görebilirsiniz.

# ansible-galaxy

- Herhangi bir otomasyon projesine başlamadan önce (tekerleği yeniden icat etmeden önce), Galaxy web sitesine bakmanız size zaman kazandırabilir.
- Projenizde işinize yarayacak bileşenleri bulabilirsiniz.



# Ansible Galaxy

- **community.postgresql** derlemesinin indirilmesi ve kurulması için «**ansible-galaxy collection install community.postgresql**» komutunun verilmesi gereklidir.

```
$ ansible-galaxy collection install community.postgresql
Starting galaxy collection install process
Process install dependency map
Starting collection install process
Downloading https://galaxy.ansible.com/download/community-postgresql-2.4.1.tar.gz to
/home/adminpc/.ansible/tmp/ansible-local-17333us_1fc57/tmptqwc8lou/community-postgresql-
2.4.1-m1c86igw
Installing 'community.postgresql:2.4.1' to
'/home/adminpc/.ansible/collections/ansible_collections/community/postgresql'
community.postgresql:2.4.1 was installed successfully
$
```

# ansible-inventory

- **ansible-inventory** komut satırı programı, Ansible'ın gördüğü envanteri görüntülemek için kullanılır.
- Faydalı opsiyonlar şunlardır:
  - y veya --yaml : varsayılan format olan JSON yerine YAML biçiminde çıktı verir.
  - i veya --inventory : dosya olarak verilen envanteri kullanır.
  - graph : metin grafiği olarak gösterir.
  - toml : TOML formatında çıktı verir.

```
$ ansible-inventory --list
{
  "_meta": {
    "hostvars": {}
  },
  "all": {
    "children": [
      "sunucular",
      "ungrouped"
    ]
  },
  "sunucular": {
    "hosts": [
      "192.168.211.139"
    ]
  }
}
$ ansible-inventory --list -y
all:
  children:
    sunucular:
      hosts:
        192.168.211.139: {}
    ungrouped: {}
```

Opsiyon verilmezse JSON formatı

-y opsiyonu YAML formatında çıktı verir.

# ansible-inventory

```
$ ansible-inventory -i /etc/ansible/hosts --graph
@all:
  |--@sunucular:
  |   |--192.168.211.139
  |--@ungrouped:
$ ansible-inventory --graph -vvv
ansible-inventory [core 2.12.10]
  config file = /etc/ansible/ansible.cfg
  configured module search path = ['/home/adminpc/.ansible/plugins/modules',
'/usr/share/ansible/plugins/modules']
  ansible python module location = /usr/lib/python3/dist-packages/ansible
  ansible collection location = /home/adminpc/.ansible/collections:/usr/share/ansible/collections
  executable location = /usr/bin/ansible-inventory
  python version = 3.8.10 (default, Mar 13 2023, 10:26:41) [GCC 9.4.0]
  jinja version = 2.10.1
  libyaml = True
Using /etc/ansible/ansible.cfg as config file
host_list declined parsing /etc/ansible/hosts as it did not pass its verify_file() method
script declined parsing /etc/ansible/hosts as it did not pass its verify_file() method
auto declined parsing /etc/ansible/hosts as it did not pass its verify_file() method
Parsed /etc/ansible/hosts inventory source with ini plugin
@all:
  |--@sunucular:
  |   |--192.168.211.139
  |--@ungrouped:
```

**-i** opsiyonuyla  
kullanılacak envanter  
dosyası belirtilebilir.

**-vvv** ayrıntılı çıktı üreterek  
hangi eklentilerin  
kullanıldığını gösterir.

# ansible-playbook

- Oyun kitapları (playbook) içinde birden fazla oyunun (play), her oyunun (play) içinde birden fazla görevin (task) tanımlı bulunduğu dosyalardır.
- Oyun kitapları belirli bir formatta yazılması gereken dosyalardır.
- Yandaki, playbook'u ana dizinimizde «**cat > deneme.yml**» **deneme.yml** adıyla oluşturuyoruz..
- Kontrol düğümünde «**mkdir files**» diyerek bir dizin oluşturuyoruz. Ardından «**touch /files/src.txt**» yazarak boş bir dosyanın oluşturulmasını sağlıyoruz.

```
$ cat > deneme.yml
# copy_file.yml
- name: copy files to destination
  hosts: localhost
  connection: local
  tasks:
    - name: copy src.txt as dest.txt in the same dir
      copy:
        src: files/src.txt
        dest: files/dest.txt
      tags:
        - simple_copy
$ mkdir files
$ touch files/src.txt
$ ansible-playbook deneme.yml
```

```
PLAY [copy files to destination]
*****
```

```
TASK [Gathering Facts]
*****
ok: [localhost]
```

```
TASK [copy src.txt as dest.txt in the same dir]
*****
changed: [localhost]
```

```
PLAY RECAP
*****
localhost                : ok=2    changed=1    unreachable=0
failed=0    skipped=0    rescued=0    ignored=0
```

«localhost» yani içinde bulunduğumuz bilgisayarda bunu gerçekleştirdik.

«localhost»a bağlandığımız için bağlantı yöntemi olarak «local» kullandık. Normalde **ssh** kullanılır.



# ENVANTER OLUŐTURMA

# Envanter

- Ansible kullanmaya başlamadan önce, hangi düğümlerin yönetileceğini belirlemek için bir envanter oluşturmak gerekir.
- Ansible, envanterde yer alan konaklar için otomasyon görevlerini gerçekleştirir.
- Bu nedenle, Ansible kullanırken, otomasyon görevlerini doğru sistemlerde çalıştırabilmek için envanter oluşturmak gerekir.
- Ansible'da otomasyon araçlarını kullanabilmek için, öncelikli olarak yöneteceğiniz sistemlerin bir envanterini oluşturmanız gereklidir.
- Envanter varsayılan olarak **/etc/ansible/hosts** dosyasında tanımlanır. Ancak istenirse, farklı dosyalar kullanılabildiği gibi «**-i dosyayolu**» opsiyonuyla hangi envanter dosyasının kullanılacağı belirlenebilir.

# Envanter

- Envanter dosyası farklı şekillerde oluşturulabilir.
- Dosya yapısı **INI** formunda olabildiği gibi **YAML** dosya türü de kullanılabilir. Ancak en basit tanım **INI** formatında yapılabilir.
- Örneğin, en basit şekilde **/etc/ansible/hosts** dosyası **INI** formatında şu şekilde oluşturulabilir:

```
mail.example.com

[webservers]
foo.example.com
bar.example.com

[dbservers]
one.example.com
two.example.com
three.example.com
```

«**mail.example.com**»,  
herhangi bir gruba bağlı  
değildir.

«**webservers**» ve  
«**dbservers**» grup isimleridir.  
Grup isimleri kullanılarak,  
grupta yer alan bütün  
sistemlere erişilebilir

# Envanter

- Bir envanter dosyası oluşturulduğunda, tanımlanan gruplar yanında «**ungrouped**» ve «**all**» isimli iki grup da oluşturulur.
- «**all**» grubu, dosya içinde yer alan bütün konakları içerir.
- «**ungrouped**» grubu, belirli bir gruba dahil olmayanları içerir.

```
mail.example.com  
  
[webservers]  
foo.example.com  
bar.example.com  
  
[dbservers]  
one.example.com  
two.example.com  
three.example.com
```

«mail.example.com»  
hem «ungrouped» hem de  
«all» gruplarındadır.

# Envanter

- Bir konak (host) envanterde yer alan grupların birden fazlası içinde tanımlanabilir.
- Örneğin, bir sunucu, «**prod**», «**istanbul**» ve «**websunucuları**» grupları içinde yer alabilir.
- Grup isimleri oluştururken, ne için kullanıldığı, nerede bulunduğu ve yazılım geliştirme işi içinde hangi süreçte kullanılacağı belirtilebilir.

```
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:
  east:
    hosts:
      foo.example.com:
      one.example.com:
      two.example.com:
  west:
    hosts:
      bar.example.com:
      three.example.com:
  prod:
    hosts:
      foo.example.com:
      one.example.com:
      two.example.com:
  test:
    hosts:
      bar.example.com:
      three.example.com:
```

# ANSIBLE PLAYBOOK

# Ansible Oyun Kitapları

- Ansible’da oyun kitapları (playbook), birden fazla oyunu içeren YAML formatındaki dosyalardır. Her oyun (play) içinde de bir veya daha fazla görev (tasks) bulunur.
- Her oyun kitabı farklı bölümlerden oluşur.
- Zorunlu bölümler şunlardır:
  - Hedef bölümü: hangi konaklara uygulanacağı (envanter kullanılır)
  - Değişken bölümü: oyun kitabı tarafından kullanılan değişkenler
  - Görevler bölümü: konaklar üzerinde yapılacak işler tanımlanır
- Tercihe bağlı bölümler şunlardır:
  - İşleyici (**handler**) bölümü
  - Döngüler (**loops**)
  - Koşullar (**conditionals**)
  - Döngüden çıkış koşulu (**until**)
  - Bildirim (**notify**)

# Örnek Oyun Kitabı

Envanterde «**webserver**s» ve «**databases**» grupları tanımlanmış.

İlk «**play**», web sunucularının güncellenmesi. Bu amaçla kullanılan APT aracı, **ansible.builtin.apt** modülüyle yapılabilir.

**state** olarak son sürümün(**latest**) yüklü olmasını istiyoruz.

**ansible.builtin.service**, herhangi bir servisin başlatılmış olduğuna emin olacak.

```
---
- name: Update web servers
  hosts: webserver
  remote_user: root

  tasks:
    - name: Ensure apache is at the latest version
      ansible.builtin.apt:
        name: httpd
        state: latest
    - name: Write the apache config file
      ansible.builtin.template:
        src: /srv/httpd.j2
        dest: /etc/httpd.conf

- name: Update db servers
  hosts: database
  remote_user: root

  tasks:
    - name: Ensure postgresql is at the latest version
      ansible.builtin.apt:
        name: postgresql
        state: latest
    - name: Ensure that postgresql is started
      ansible.builtin.service:
        name: postgresql
        state: started
```



# Ubuntu'da root kullanıcısı

- Uzaktaki makine: Ubuntu'da güvenlik gerekçesiyle root kullanıcısıyla login yapılmasına izin verilmemektedir. Farklı bir kullanıcı oluşturulmakta ve «**sudo**» komutuyla yetki artırımı yapılmaktadır.
  - Giriş yaptığınız kullanıcıda «**sudo su - root**» diyerek «**root**» kullanıcısı ile kabuk açın. «**passwd**» komutu ile, yeni şifre girin.
- Uzaktaki makine: Ayrıca OpenSSH'da «**root**» olarak bağlantıya da izin verilmemektedir.
  - «**nano /etc/ssh/sshd\_config**» komutuyla dosyasını açıp, ilgili satırı şu hale getirin: «**PermitRootLogin yes**». Sonra «**sudo systemctl restart sshd**» diyerek servisi yeniden başlatıp, değişikliğin uygulanmasını sağlayın.
- Kontrol düğümü: «**ssh-copy-id root@UZAKIPNOSUNUYAZ**» yazarak ve «**root**» şifresini de girerek, açık anahtarımızı uzak sisteme gönderiyoruz.
- Artık Ansible için her şey tamamlanmıştır. Ansible **root** şifresiyle uzaktaki sisteme giriş yapabilecektir.

# PostgreSQL yükleme - 1

- Aşağıda yer alan ve içinde iki görev bulunan oyun kitabını, «**deneme.yml**» olarak kaydedelim.

```
- name: Update db servers
  hosts: databases
  remote_user: root

tasks:
- name: Ensure postgresql is at the latest version
  ansible.builtin.apt:
    name: postgresql
    state: latest
- name: Ensure that postgresql is started
  ansible.builtin.service:
    name: postgresql
    state: started
```

## PostgreSQL yükleme - 2

- Ardından «**ansible-playbook deneme.yml**» komutuyla çalıştıralım

```
$ ansible-playbook deneme.yml

PLAY [Update db servers] *****

TASK [Gathering Facts] *****
ok: [192.168.211.139]

TASK [Ensure postgresql is at the latest version] *****
changed: [192.168.211.139]

TASK [Ensure that postgresql is started] *****
ok: [192.168.211.139]

PLAY RECAP *****
192.168.211.139      : ok=3    changed=1    unreachable=0    failed=0
skipped=0    rescued=0    ignored=0
```

# Oyun kitabı çalıştırma

- Ansible her bir görevi sırayla, istenen konaklarda çalıştırır.
- Her görev, bir modülün karşı konağa yüklenmesini ve belirli argümanlarla çalıştırılmasını sağlar. Çalıştırmayı paralel yapabilir.
- Bir görev, bütün konaklarda çalıştırıldıktan sonra, diğer göreve geçilir.
- Görevleri çalıştırırken bu sırayı değiştirmek mümkündür. Buna oyun-kitabı çalıştırma stratejileri adı verilir. Mesela, sadece belirli bir konakta çalıştırılacak görevler olabilir. Bir görev paralel çalıştırılmak istenmiyorsa, bu strateji olarak o görevde belirtilmelidir.

# OYUN KİTABI ÇALIŞTIRMA STRATEJİLERİ

# Oyun kitabı çalıştırma stratejileri

- Ansible, YML dosyasında «tasks» başlığında tanımlı her bir görevi, konak isimleri uyan makinelerde sırasıyla çalıştırır.
- Bir görev bütün uzak makinelerde çalıştıktan sonra, Ansible diğer görevi çalıştırmaya başlar.
- Bu çalışma düzeni, «**strategy**» amacıyla kullanılan eklentiler diğer bir eklenti ile değiştirilebilir.
- Çeşitli stratejiler (**run\_once**, **free**, **serial**, **throttle**, vb) kullanılabilir.

```
- hosts: all
  strategy: free
  tasks:
    # ...
```

# Oyun kitabı çalıştırma stratejileri

- Varsayılan çalıştırma stratejisi, doğrusal stratejidir ve ansible-core içinde tanımlıdır. «linear» ifadesini kullanarak «collections:» içinde belirtmeye gerek duyulmaz. Sırayla bütün konaklarda çalışır.
- «free» stratejisi, her konağın oyunları sonuna kadar en hızlı şekilde tamamlamasını sağlar.

```
- hosts: all
  strategy: free
  tasks:
  # ...
```

- İstenirse, **ansible.cfg** dosyası içinde varsayılan strateji olarak tanımlanabilir.

```
[defaults]
strategy = free
```

# Oyun kitabı çalıştırma stratejileri

- Eğer yeterince işlem gücünüz varsa, paralel çalıştırdığınız görevlerin sayısını artırabilirsiniz. Örneğin, 30 paralel görevi aynı anda gerçekleştirebilmek için **ansible.cfg** içine tanım yapılması gerekir.

```
[defaults]
forks = 30
```

- Stratejiler yanında, birkaç anahtar kelime, oyunların çalıştırılmasında etkili olur. Kaç tane konakta aynı anda oyun çalıştırılacağı, serial parametresiyle belirlenir. Örneğin aynı anda 3 konakta oyun çalıştırılması için oyun-kitabı aşağıdaki ek yapılabilir.

```
- name: test play
  hosts: webservers
  serial: 3
  gather_facts: False

  tasks:
    - name: first task
      command: hostname
    - name: second task
      command: hostname
```

```
- name: test play
  hosts: webservers
  serial: "30%"
  gather_facts: False

  tasks:
    - name: first task
      command: hostname
    - name: second task
      command: hostname
```



# Oyun kitabı çalıştırma stratejileri

- Eğer istenirse, çalıştırma paket uzunları da liste olarak verilebilir.

```
- name: test play
  hosts: webservers
  serial:
    - 1
    - 5
    - 10
```

```
- name: test play
  hosts: webservers
  serial:
    - "10%"
    - "20%"
    - "100%"
```

- Ya da paket büyüklüklerini hem sayı hem de yüzde olarak karışık şekilde kullanabilirsiniz.

```
- name: test play
  hosts: webservers
  serial:
    - 1
    - 5
    - "20%"
```

# Oyun kitabı çalıştırma stratejileri

- Bazen, oyun-kitaplarının çalıştırılmasının kısıtlanması gerekebilir. «**throttle**» anahtar kelimesi, belirli bir görevin (task) çalıştırılmasında kullanılan paralel çalıştırma imkanını kısıtlama imkanı sağlar. Bu şekilde, çok fazla işlem gücü gerektiren uygulamaların sistemi yavaşlatması engellenebilir.

```
tasks:  
- command: /yoğun/CPU/kullanan/program  
  throttle: 1
```

- Eğer «**fork**» sayısını veya paralel şekilde görevlerin çalıştırıldığı konak sayısını düşürdüyseniz, «**throttle**» ile daha da düşürebilirsiniz. Örneğin, «**fork**» veya «**serial**» olarak «**3**» değerini verdiyseniz, «**throttle**» ile «**1**» veya «**2**»ye indirebilirsiniz. Ama değeri yükseltmezsiniz.

## Koşullu çalıştırma stratejisi

- Envanterinizde yer alan bir grup makineden (mesela **websunuculari** grubu), bir görevi, sadece listenin başında yer alan makinede çalıştırmak istiyorsanız «**run\_once**» anahtarını «**true**» değeri verebilirsiniz.

```
# ...
tasks:
  # ...
  - command: /opt/application/upgrade_db.py
    run_once: true

# ...
```

- Benzer bir amaçla kullanılabilecek bir yaklaşım da koşullu çalıştırmadır:

```
- command: /opt/application/upgrade_db.py
  when: inventory_hostname == webservers[0]
```

## Belirli bir konakta görev çalıştırma

- «**run\_once**» konak grubundaki ilk makinede görevi çalıştırır. Peki, sadece belirli bir sunucuda çalıştırılmak üzere, bir görev nasıl tanımlanır?
  - Buna delegasyon denir ve «**delegate\_to**» anahtarıyla «**run\_once**» ile sadece ismi verilen konakta o görevin çalıştırılması sağlanır.

```
- command: /opt/application/upgrade_db.py  
  run_once: true  
  delegate_to: web01.example.org
```

- Delegasyon sırasında, görev ilgili sisteme yönlendirilir ama o anda başka bir konağın görevi işletiliyor olabilir.

# Delegasyon

- Bir görevi belirli bir konak üzerinde gerçekleştirmek istiyorsanız, «**delegate\_to**» anahtar kelimesini kullanmanız gereklidir. Çünkü, bazı işlemler sadece belirli konaklarda gerçekleştirilir.

```
---
- hosts: webservers
  serial: 5

  tasks:
    - name: Take out of load balancer pool
      ansible.builtin.command: /usr/bin/take_out_of_pool {{ inventory_hostname }}
      delegate_to: 127.0.0.1

    - name: Actual steps would go here
      ansible.builtin.yum:
        name: acme-web-stack
        state: latest

    - name: Add back to load balancer pool
      ansible.builtin.command: /usr/bin/add_back_to_pool {{ inventory_hostname }}
      delegate_to: 127.0.0.1
```

# «local\_action»

- «delegate\_to: 127.0.0.1» yerine kullanılabilecek daha kısa yöntem, «local\_action» anahtar kelimesi ile gerçekleştirilir.

```
---  
# ...  
  
tasks:  
  - name: Take out of load balancer pool  
    local_action: ansible.builtin.command /usr/bin/take_out_of_pool {{ inventory_hostname }}  
  
# ...  
  
  - name: Add back to load balancer pool  
    local_action: ansible.builtin.command /usr/bin/add_back_to_pool {{ inventory_hostname }}
```

# ANSIBLE İLE WINDOWS SİSTEMLERİN YÖNETİMİ

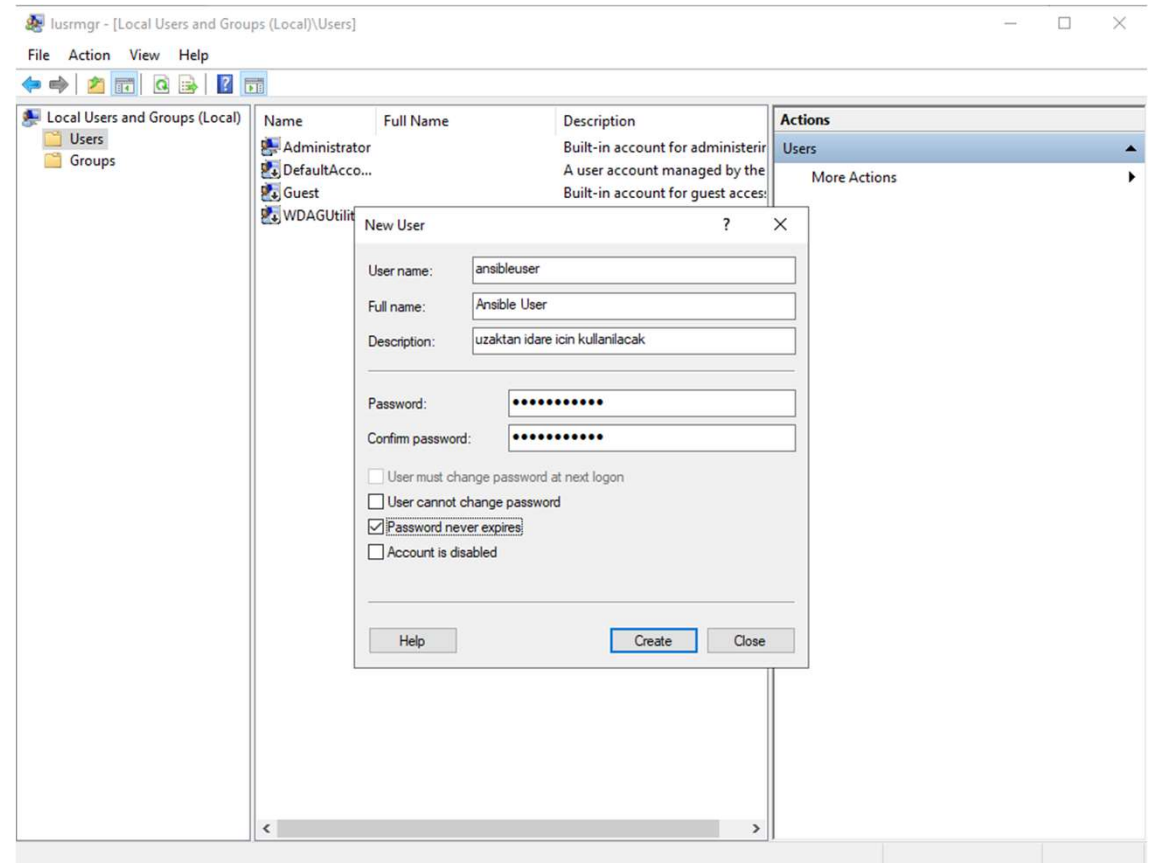
# Windows Uzaktan Yönetim (WRM)

- Linux/Unix konaklarına SSH ile bağlanılmaktadır.
- Windows konakları ise WinRM ile yapılandırılabilir. WinRM, Windows sunucuların diğerleriyle haberleşebilmesi için kullanılan protokoldür.
- WinRM SOAP temelli bir protokoldür ve Windows 2012'den beri varsayılan şekilde sistemlerde yüklü gelmektedir.



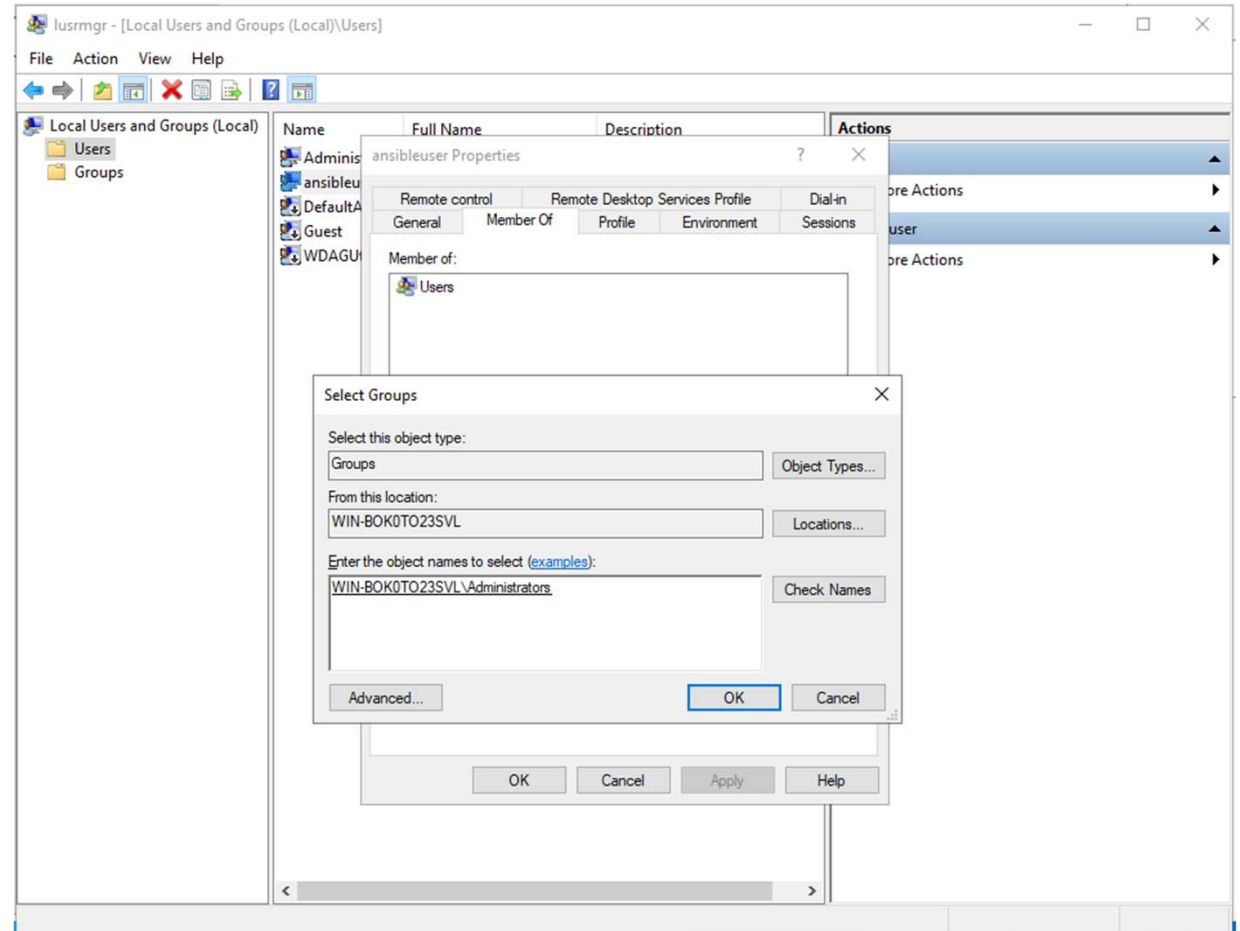
# Windows Uzaktan Yönetim (WRM)

- Windows 2022 kuracağız ve **ansibleuser** isminde bir yönetici/poweruser hesabı açacağız.



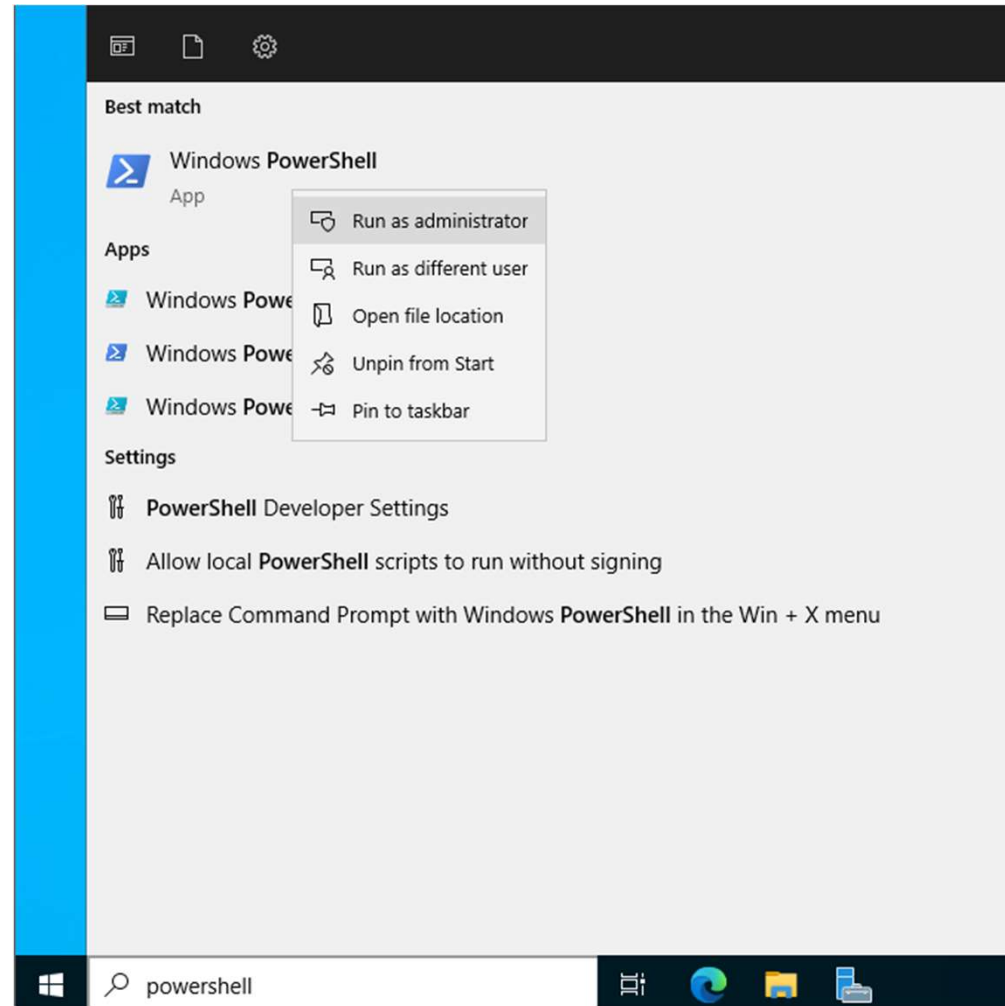
# Windows Uzaktan Yönetim (WRM)

- **ansibleuser** kullanıcısını Lokal Yönetici olarak ekliyoruz.



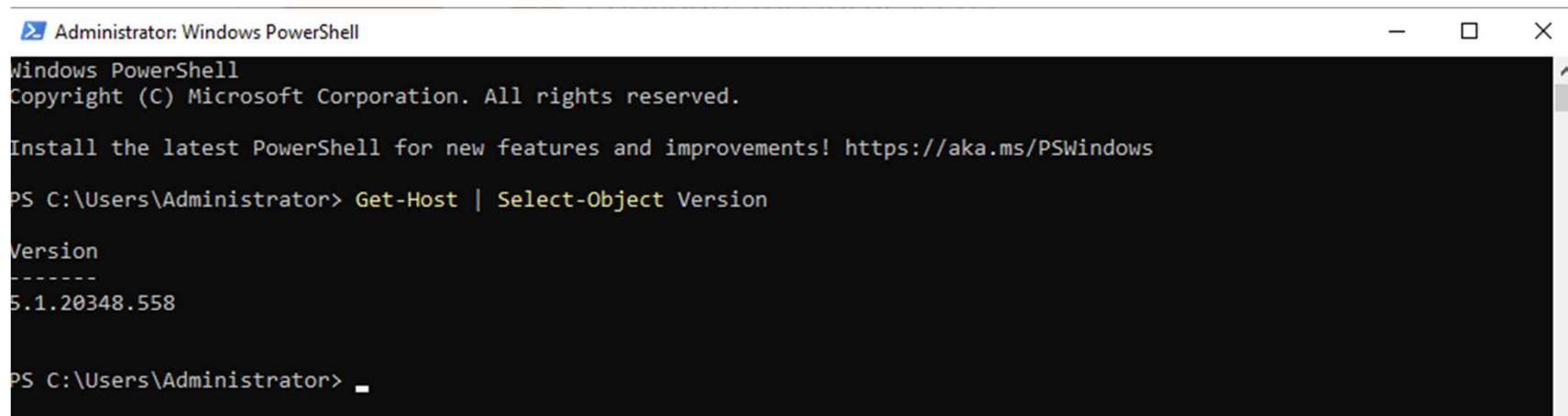
# Windows Uzaktan Yönetim (WRM)

- «**powershell**»  
programını  
Administrator olarak  
çalıştırıyoruz.



# Windows Uzaktan Yönetim (WRM)

- «powershell» içinde «**Get-Host | Select-Object Version**» diyerek **.net** versiyonunun ne olduğuna bakıyoruz.
- Ansible için v4.1'in üzerinde olmalı.



```
Administrator: Windows PowerShell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\Administrator> Get-Host | Select-Object Version

Version
-----
5.1.20348.558

PS C:\Users\Administrator> _
```

# Windows Uzaktan Yönetim (WRM)

- «powershell» içinde «winrm get winrm/config/Service» diyerek WinRM durumuna bakacağız.

```
PS C:\Users\Administrator> winrm get winrm/config/Service
Service
RootSDDL = O:NSG:BAD:P(A;;GA;;;BA)(A;;GR;;;IU)S:P(AU;FA;GA;;;WD)(AU;SA;GXGW;;;WD)
MaxConcurrentOperations = 4294967295
MaxConcurrentOperationsPerUser = 1500
EnumerationTimeoutms = 240000
MaxConnections = 300
MaxPacketRetrievalTimeSeconds = 120
AllowUnencrypted = false
Auth
  Basic = false
  Kerberos = true
  Negotiate = true
  Certificate = false
  CredSSP = false
  CbtHardeningLevel = Relaxed
DefaultPorts
  HTTP = 5985
  HTTPS = 5986
IPv4Filter = *
IPv6Filter = *
EnableCompatibilityHttpListener = false
EnableCompatibilityHttpsListener = false
CertificateThumbprint
AllowRemoteAccess = true

PS C:\Users\Administrator>
```

«Basic» kimlik doğrulama (kullanıcı adı ve şifreyle) kullanılamaz durumda.

5986 nolu porttan HTTPS REST API kullanacağız .

# Windows Uzaktan Yönetim (WRM)

- «powershell» içinde «winrm enumerate winrm/config/Listener» diyerek WinRM Listener durumuna bakacağız.

```
PS C:\Users\Administrator> winrm enumerate winrm/config/Listener
Listener
  Address = *
  Transport = HTTP
  Port = 5985
  Hostname
  Enabled = true
  URLPrefix = wsman
  CertificateThumbprint
  ListeningOn = 127.0.0.1, 192.168.211.156, ::1, fe80::148d:a7c7:7191:bf15%2
```

Windows 10 gibi kullanıcı bilgisayarlarında, Listener özelliğinin çalışmadığını görebilirsiniz.



# Windows Uzaktan Yönetim (WRM)

- «**listener**» bölümünün çalışmadığını gördük. Ne yapmalıyız?
- Şu adrese giriş yapalım ve «**WinRM Setup**» bölümüne gidelim:
  - [https://docs.ansible.com/ansible/latest/os\\_guide/windows\\_setup.html](https://docs.ansible.com/ansible/latest/os_guide/windows_setup.html)

```
PS> $url="https://raw.githubusercontent.com/ansible/ansible-documentation/devel/examples/scripts/ConfigureRemotingForAnsible.ps1"
PS> $file = "$env:temp\ConfigureRemotingForAnsible.ps1"
PS> (New-Object -TypeName System.Net.WebClient).DownloadFile($url, $file)
PS> powershell.exe -ExecutionPolicy Bypass -File $file
Self-signed SSL certificate generated; thumbprint: C265D6DCCA555A19F2856F78AAEE535EA04742FF

wxf      : http://schemas.xmlsoap.org/ws/2004/09/transfer
a        : http://schemas.xmlsoap.org/ws/2004/08/addressing
w        : http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd
lang     : en-US
Address  : http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous
ReferenceParameters : ReferenceParameters

Ok.
```

[https://docs.ansible.com/ansible/latest/os\\_guide/windows\\_winrm.html](https://docs.ansible.com/ansible/latest/os_guide/windows_winrm.html)

# Windows Uzaktan Yönetim (WRM)

- «**listener**» bölümünün çalışıp çalışmadığını aşağıdaki komutla kontrol edelim

```
PS C:\Users\Administrator> winrm enumerate winrm/config/Listener
Listener
  Address = *
  Transport = HTTP
  Port = 5985
  Hostname
  Enabled = true
  URLPrefix = wsman
  CertificateThumbprint
  ListeningOn = 127.0.0.1, 192.168.211.156, ::1, fe80::148d:a7c7:7191:bf15%2

Listener
  Address = *
  Transport = HTTPS
  Port = 5986
  Hostname = WIN-BOK0TO23SVL
  Enabled = true
  URLPrefix = wsman
  CertificateThumbprint = C265D6DCCA555A19F2856F78AAEE535EA04742FF
  ListeningOn = 127.0.0.1, 192.168.211.156, ::1, fe80::148d:a7c7:7191:bf15%2
```



# Windows Uzaktan Yönetim (WRM)

- «**Service**» bölümünün çalışıp çalışmadığını kontrol edelim

```
PS C:\Users\Administrator> winrm get winrm/config/Service
Service
RootSDDL = O:NSG:BAD:P(A;;GA;;;BA)(A;;GR;;;IU)S:P(AU;FA;GA;;;WD)(AU;SA;GXGW;;;WD)
MaxConcurrentOperations = 4294967295
MaxConcurrentOperationsPerUser = 1500
EnumerationTimeoutms = 240000
MaxConnections = 300
MaxPacketRetrievalTimeSeconds = 120
AllowUnencrypted = false
Auth
    Basic = true
    Kerberos = true
    Negotiate = true
    Certificate = false
    CredSSP = false
    CbtHardeningLevel = Relaxed
DefaultPorts
    HTTP = 5985
    HTTPS = 5986
IPv4Filter = *
IPv6Filter = *
EnableCompatibilityHttpListener = false
EnableCompatibilityHttpsListener = false
CertificateThumbprint
AllowRemoteAccess = true
```

«Basic» kimlik doğrulama yöntemi  
«true» değerini aldı.

# Windows Uzaktan Yönetim (WRM)

- «**Winrs**» bölümünün çalışıp çalışmadığını kontrol edelim

```
PS C:\Users\Administrator> winrm get winrm/config/Winrs
Winrs
  AllowRemoteShellAccess = true
  IdleTimeout = 7200000
  MaxConcurrentUsers = 2147483647
  MaxShellRunTime = 2147483647
  MaxProcessesPerShell = 2147483647
  MaxMemoryPerShellMB = 2147483647
  MaxShellsPerUser = 2147483647
```

# Ansible Windows bağlantısı

- **Ansible** tarafından `/etc/ansible/hosts` dosyasına Windows grubu ekleyeceğiz ve bağlanmak için kullanacağımız değişkenleri tanımlayacağız.

```
[websunucular]  
192.168.211.153
```

```
[diger]  
192.168.211.134
```

```
[windows]  
win2022 ansible_host=192.168.211.156
```

```
[windows:vars]  
ansible_user=ansibleuser  
ansible_password=Selamlar123  
ansible_port=5986  
ansible_connection=winrm  
ansible_winrm_transport=basic  
ansible_winrm_server_cert_validation=ignore
```

«Basic» kullanıcı doğrulama işlemi sadece lokal kullanıcılar için kullanılır.  
Eğer «basic» kimlik doğrulama kullanılamıyorsa, şu komutla geçerli hale getirilebilir:  
`Set-Item -Path WSMan:\localhost\Service\Auth\Basic -Value $true`

Buraya «ntlm» yazılarak doğrulama yöntemi Domain kullanıcıları için kullanılabilir.

## Kimlik doğrulama Yöntemleri

- Windows'ta 5 tür doğrulama yöntemi bulunuyor: Temel düzey, Sertifika, Kerberos, NTLM, CredSSP.
- «Domain»e bağlı kullanıcılar için en uygun yöntem Kerberos ve NTLM'dir. NTLM, biraz eski ve riskli olabilecek bir yöntemdir.

Option	Local Accounts	Active Directory Accounts	Credential Delegation	HTTP Encryption
Basic	Yes	No	No	No
Certificate	Yes	No	No	No
Kerberos	No	Yes	Yes	Yes
NTLM	Yes	Yes	No	Yes
CredSSP	Yes	Yes	Yes	Yes

# Ansible Windows bağlantısı

- Bir oyun kitabı yazarak gerçekten Windows makineye bağlanıyor muyuz diye bakacağız.

```
$ cat > deneme.yml
- name: Windows ping modulu
  hosts: windows
  become: false
  gather_facts: false
  tasks:
    - name: test
      ansible.windows.win_ping:

$ ansible-playbook deneme.yml
$ ansible-playbook deneme.yml

PLAY [Windows ping modulu] *****

TASK [test] *****
ok: [win2022]

PLAY RECAP *****
win2022                : ok=1    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

# Ansible Windows modülleri

- Windows'un yönetimi için kullanılabilecek modüllerin tüm listesi şöyle:
  - [https://docs.ansible.com/ansible/latest/modules/list\\_of\\_windows\\_modules.html](https://docs.ansible.com/ansible/latest/modules/list_of_windows_modules.html)

# win\_whoami modülü

- Giriş yapan kullanıcının kim olduğunu belirtir:

```
$ cat > deneme3.yml
name: Whoami
hosts: windows
become: false
gather_facts: false

tasks:
  - name: find whoami based on credentials supplied
    win_whoami:
      register: whoami_base

  - name: output base user
    debug:
      msg: "{{ whoami_base.account.account_name }}"

$ ansible-playbook deneme3.yml

PLAY [Whoami] *****

TASK [find whoami based on credentials supplied] *****
ok: [win2022]

TASK [output base user] *****
ok: [win2022] => {
  "msg": "ansibleuser"
}

PLAY RECAP *****
win2022                : ok=2    changed=0    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

# win\_user ile Windows'a yeni kullanıcı ekleme

- Kullanıcı eklemek için **ansible.windows.win\_user** modülü kullanılır.

```
$ cat > deneme4.yml
---
- name: windowsa kullanıcı ekleme
  hosts: windows
  become: false
  gather_facts:

  vars:
    usr_name: 'deneme'
    usr_password: 'Selamlar456'
    usr_groups: "Users"
  tasks:
    - name: lokal kullanıcı oluşturun
      ansible.windows.win_user:
        name: "{{ usr_name }}"
        password: "{{ usr_password }}"
        groups: "{{ usr_groups }}"
        update_password: on_create
        password_expired: true

$ ansible-playbook deneme4.yml

PLAY [windows user add] *****

TASK [Gathering Facts] *****
ok: [win2022]

TASK [create local user] *****
changed: [win2022]

PLAY RECAP *****
win2022                : ok=2    changed=1    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```



# win\_acl

- Dosya, kullanıcı ve «**registry**» üzerinde izin işlemleri yapılabilir.

```
$ cat > deneme5.yml
---
- name: Remove FullControl AccessRule for IIS_IUSRS
  win_acl:
    path: C:\inetpub\wwwroot\MySite
    user: IIS_IUSRS
    rights: FullControl
    type: allow
    state: absent
    inherit: ContainerInherit, ObjectInherit
    propagation: 'None'

- name: Set registry key right
  win_acl:
    path: HKCU:\Bovine\Key
    user: BUILTIN\Users
    rights: EnumerateSubKeys
    type: allow
    state: present
    inherit: ContainerInherit, ObjectInherit
    propagation: 'None'
```

# win\_updates

- **win\_updates**, Linux'teki **yum** ve **apt** gibi bir sistem güncelleme yazılımıdır. Bu modülle kritik ve güvenlik yamalarını yapabilir ve istenen bir programı kurabilirsiniz.
- **win\_reboot** modülü Windows sistemini reboot etmek için kullanılır.

```
---
- name: updates
  hosts: Windows
  become: false
  gather_facts: false

  tasks:
    - name: critical and security update installation
      win_updates:
        category_names:
          - CriticalUpdates
          - SecurityUpdates
        state: installed
        register: update_result
    - name: if required reboot host if required
      win_reboot:
        when: update_result.reboot_required
```

# win\_chocolatey

- **chocolatey** Linux'teki **yum** ve **apt** gibi bir paket yönetim yazılımıdır.
- **win\_get\_url** modülüyle URL belirterek programı indirebilirsiniz:

```
---  
- name: program install  
  hosts: windows  
  become: false  
  gather_facts: false  
  
tasks:  
  - name: install adobeReader  
    win_chocolatey:  
      name: adobereader  
      state: present  
  - name: install GoogleChrome  
    win_chocolatey:  
      name: googlechrome  
      state: present  
  - name: Download the 7-Zip package  
    win_get_url:  
      url: https://www.7-zip.org/a/7z2300-x64.msi  
      dest: C:\temp\7z.msi
```

**chocolatey** paket isimleri için:  
<https://community.chocolatey.org/packages>

**7-zip** programının son sürümünün MSI dosyasını [www.7-zip.org](http://www.7-zip.org) sitesine giderek **Downloads** bölümünde bulabilirsiniz. Bağlantıyı kopyalayıp buraya eklemeniz gerekiyor.

## win\_user

- **ansible.windows.win\_user** modülü, Windows'ta **lokal** kullanıcılarla ilgili işlemleri yapmak üzere kullanılır. (**name**: kullanıcı ismi, **state**: present/absent, **password**: şifre, **update\_password**: always/on\_create.

```
---
- name: windows change password
  hosts: windows
  become: false
  gather_facts: false

vars:
  user_name: 'deneme'
  user_passwd: 'Selamlar456'
tasks:
  - name: change password
    ansible.windows.win_user:
      name: "{{ user_name }}"
      password: "{{ user_passwd }}"
```

# win\_user

- **ansible.windows.win\_user** modülü, Windows'ta lokal kullanıcıları silebilir.

```
---  
- name: windows change password  
  hosts: windows  
  become: false  
  gather_facts: false  
  
vars:  
  user_name: 'deneme'  
tasks:  
  - name: change password  
    ansible.windows.win_user:  
      name: "{{ user_name }}"  
      state: absent
```

# win\_domain\_user

- **community.windows.win\_domain\_user**, Windows AD'de kullanıcı hesaplarının idaresi için kullanılabilir.

```
---
- name: deneme olmayacak (absent)
  community.windows.win_domain_user:
    name: deneme
    state: absent

- name: Ensure user is created with delegates and spn's defined
  community.windows.win_domain_user:
    name: shmemmmmy
    password: The3rubberducki33!
    state: present
    groups:
      - Domain Admins
      - Enterprise Admins
    delegates:
      - CN=shenetworks,CN=Users,DC=ansible,DC=test
      - CN=mk.ai,CN=Users,DC=ansible,DC=test
      - CN=jessiedotjs,CN=Users,DC=ansible,DC=test
    spn:
      - MSSQLSvc/us99db-svr95:2433
```

# ansible.windows.win\_user\_right

- **ansible.windows.win\_user\_right** modülü  
Windows kullanıcılarının/gruplarının haklarını değiştirir.

```
---
- name: Replace the entries of Deny log on locally
  ansible.windows.win_user_right:
    name: SeDenyInteractiveLogonRight
    users:
      - Guest
      - Users
    action: set

- name: Add account to Log on as a service
  ansible.windows.win_user_right:
    name: SeServiceLogonRight
    users:
      - .\Administrator
      - '{{ansible_hostname}}\local-user'
    action: add
```

# ansible.windows.win\_features

- **ansible.windows.win\_features** modülü Windows'ta çeşitli eklentiler yükler.

```
---
- name: IIS kurulumu
  hosts: windows
  become: false
  gather_facts: false

  tasks:

    - name: Install IIS (Web-Server only)
      ansible.windows.win_feature:
        name: Web-Server
        state: present
    - name: Install IIS (Web-Server and Web-Common-Http)
      ansible.windows.win_feature:
        name:
          - Web-Server
          - Web-Common-Http
        state: present
```

```
- name: Install NET-Framework-Core from file
  ansible.windows.win_feature:
    name: NET-Framework-Core
    source: C:\Temp\iso\sources\sxs
    state: present

- name: Install IIS with sub features and mgnt tools
  ansible.windows.win_feature:
    name: Web-Server
    state: present
    include_sub_features: true
    include_management_tools: true
    register: win_feature

- name: Reboot if required
  ansible.windows.win_reboot:
    when: win_feature.reboot_required
```

Yüklü olmayan Windows özelliklerinin isimlerini bulabilmek için powershell komutu:

```
Get-WindowsFeature|where {$_.Installed -eq $false} | select -expand name
```



# ansible.windows.win\_features

- **ansible.windows.win\_features** modülü Windows'ta yüklü eklentiler hakkında bilgi de verebilir.

```
---
- name: bilgi alalim
  hosts: windows
  become: false
  gather_facts: false

  tasks:

    - name: Bilgi alalim
      community.windows.win_feature_info:
        register: feature_info

    - name: ozellikleri goster
      debug:
        msg: "{{ feature_info }}"
```

# win\_service

- Windows servislerini durdurup başlatmak için kullanılır:

```
---  
- name: servisi kapat  
  hosts: windows  
  become: false  
  gather_facts: false  
  
  tasks:  
  
    - name: Stop service Tomcat  
      win_service:  
        name: "??????"  
        state: stopped
```

# win\_disk\_facts

- Window'ta tanımlı disklerle ilgili bilgi toplanabilir

```
---
- name: disk bilgileri
  hosts: windows
  become: false
  gather_facts: false

  tasks:

    - name: disklerle ilgili bilgi alalım
      win_disk_facts:

    - name: disk boyutunu gosterelim
      debug:
        msg: "{{ ansible_facts.disks[0].size }}"

    - name: farkli gosterim
      debug:
        msg: '{{ ansible_facts.disks|selectattr("system_disk")|first }}'
```

# win\_command

- Window'taki kaynakları win\_command ile gösterebiliriz.

```
---
- name: disk bilgileri
  hosts: windows
  become: false
  gather_facts: false

  tasks:

- name: Get disk facts
  win_command: wmic cpu get caption, deviceid, name, numberofcores, maxclockspeed
  register: usage

- name: göster
  debug: msg="{{ usage.stdout }}"
```

# win\_environment

- Çevre değişkenlerini set edebiliriz..

```
---  
- name: cevre degiskenleri  
  hosts: windows  
  become: false  
  gather_facts: false  
  
  tasks:  
  
- name: butun kullanicilar için cevre degiskeni  
  win_environment:  
    state: present  
    name: POSTGRES_USER  
    value: webserver  
    level: machine
```