ROUTEROS SECURITY AUTOMATION WITH ANSIBLE VERSION 2.0

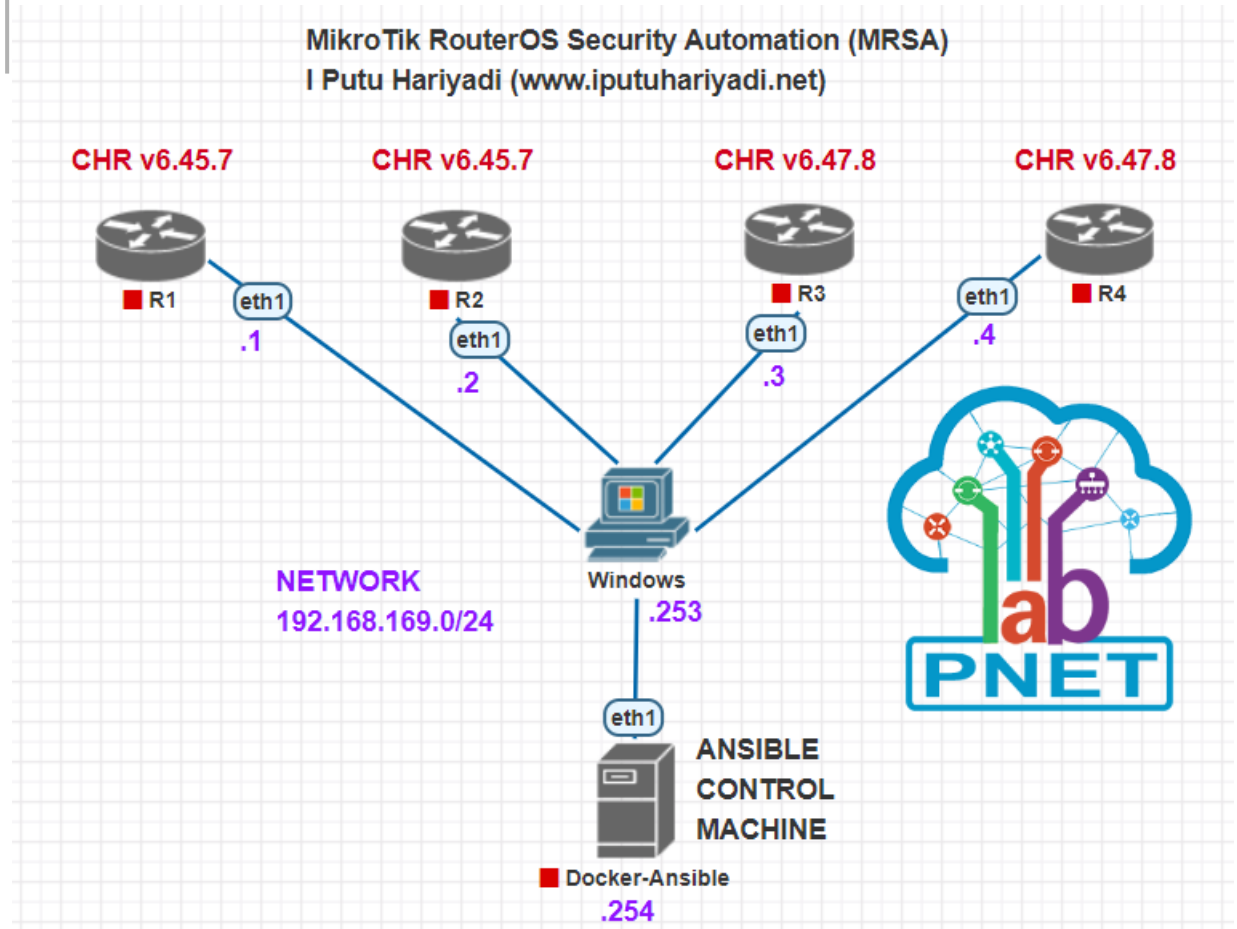
I PUTU HARIYADI
putu.hariyadi@universitasbumigora.ac.id
www.iputuhariyadi.net

MATERI

1. Rancangan Jaringan Ujicoba.
2. Rancangan Pengalamatan IP.
3. Konfigurasi Dasar pada setiap perangkat atau node.
4. Pengamanan Router MikroTik.
5. Manajemen Konfigurasi dan Otomatisasi dengan Ansible.
6. Studi Kasus Ansible Playbook untuk mengotomatisasi keamanan MikroTik RouterOS.

RANCANGAN JARINGAN UJICOBA DAN PENGALAMATAN IP

RANCANGAN JARINGAN UJICOBA



Rancangan jaringan ujicoba disimulasikan menggunakan **PNETLAB** yang diinstalasi pada VMWare Workstation 16 Pro. Terdapat **6 (enam)** object utama yang dibuat pada lab dari **PNETLAB** yaitu:

- 4 (empat) object berjenis node dengan template MikroTik RouterOS CHR versi 6.45.7 untuk R1 dan R2. Sedangkan versi 6.47.8 untuk R3 dan R4. Keseluruhan node router tersebut memiliki 5 (lima) *interface* yaitu ether1, ether2, ether3, ether4 dan ether5.
- 1 (satu) object berjenis **network** dengan **name/prefix Windows** dan **type Cloud2** yang terhubung ke **vmnet1 (mode host only)** serta **icon "Desktop_Win.png"**.
- 1 (satu) object berjenis node dengan template Docker.io yang menggunakan image **ptthanh1511/ansible:latest** sebagai Ansible Control Machine.

RANCANGAN PENGALAMATAN IP

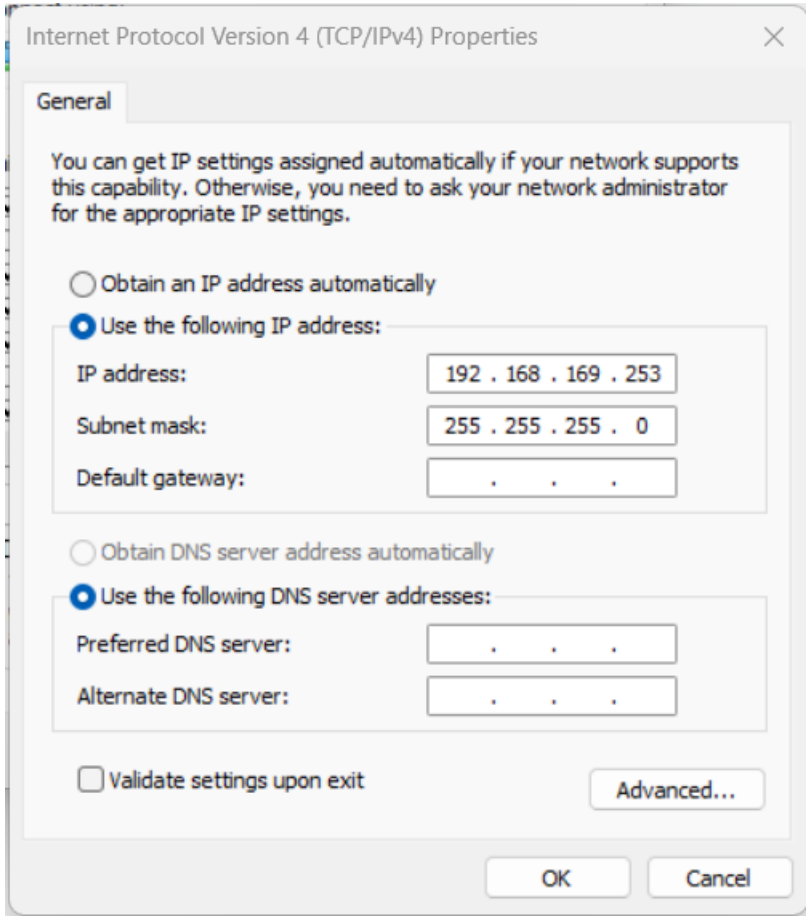
Alamat **network** menggunakan **192.168.169.0/24** dengan alokasi pengalamatan pada setiap **interface** dari perangkat adalah sebagai berikut:

No.	Perangkat	Interface	Alamat IP
1.	R1	ether1	192.168.169.1/24
2.	R2	ether1	192.168.169.2/24
3.	R3	Ether1	192.168.169.3/24
4.	R4	Ether1	192.168.169.4/24
5.	Windows	vmnet1	192.168.169.253/24
6.	Docker-Ansible	Eth1	192.168.169.254/24

KONFIGURASI DASAR PERANGKAT JARINGAN DAN VERIFIKASI KONEKSI

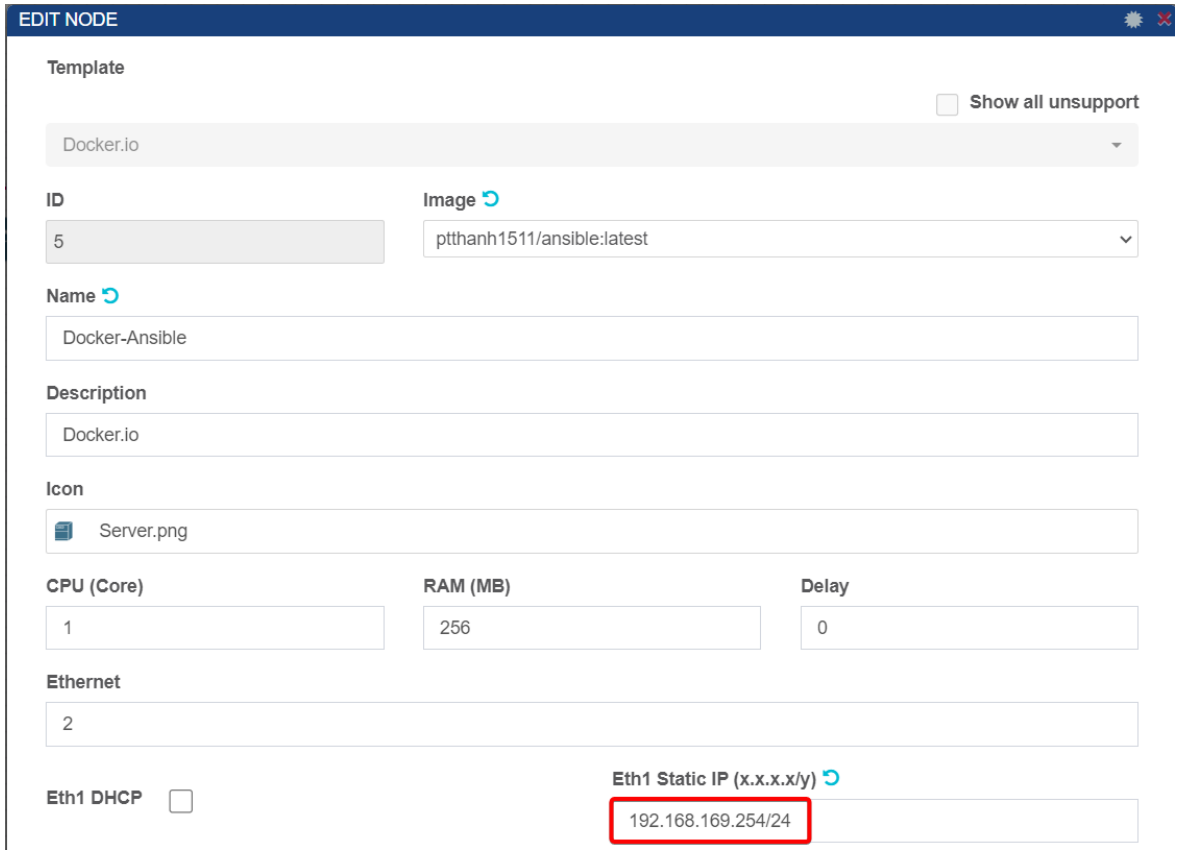
KONFIGURASI DASAR DI WINDOWS

Lakukan pengaturan pengalamatan **IP** pada **interface VMnet1** melalui **Network and Internet Setting** dari system operasi Windows agar menggunakan **192.168.169.253/24**.

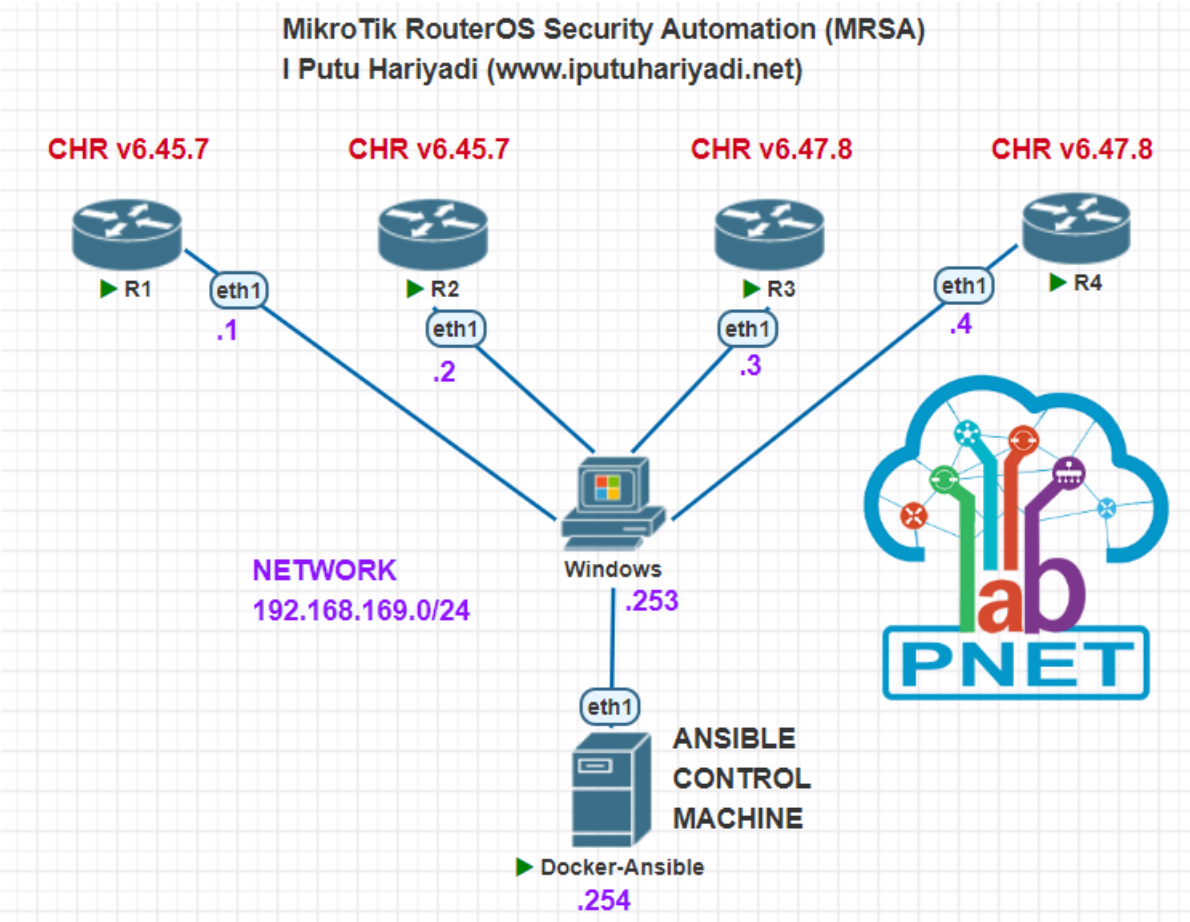


KONFIGURASI DASAR DOCKER-ANSIBLE PADA PNETLAB

Lakukan penyesuaian konfigurasi pada **node Docker-Ansible** agar **interface Eth1** menggunakan alamat **IP 192.168.169.254/24**.



MENJALANKAN SELURUH NODE DARI LAB MIKROTIK ROUTEROS SECURITY AUTOMATION (MRSA) DI PNETLAB



KONFIGURASI DASAR ROUTER R1 PADA PNETLAB

- Akses **terminal** dari **Router R1** dan lakukan **login** sebagai **admin** tanpa sandi **(blank password).**
- Lakukan konfigurasi **hostname (identity)** menggunakan **R1**, pengalamatan **IP** pada **interface ether1** menggunakan **192.168.169.1/24** dan memverifikasi pengaturan pengalamatan IP yang telah dilakukan. Selain itu menampilkan informasi **DHCP Client** dan menghapus pengaturan **DHCP Client** pada *interface* serta memverifikasi hasil penghapusannya.

R1

```
[admin@MikroTik] > system identity set name=R1
[admin@R1] > ip address add address=192.168.169.1/24 interface=ether1
[admin@R1] > ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         INTERFACE
 0   192.168.169.1/24   192.168.169.0   ether1
[admin@R1] > ip dhcp-client print
Flags: X - disabled, I - invalid, D - dynamic
 #   INTERFACE           USE ADD-DEFAULT-ROUTE STATUS        ADDRESS
 0   ether1              yes yes               searching...
[admin@R1] > ip dhcp-client remove 0
[admin@R1] > ip dhcp-client print
Flags: X - disabled, I - invalid, D - dynamic
 #   INTERFACE           USE ADD-DEFAULT-ROUTE STATUS        ADDRESS
[admin@R1] >
```

KONFIGURASI DASAR ROUTER R2 PADA PNETLAB

- Akses **terminal** dari **Router R2** dan lakukan login sebagai **admin** tanpa sandi **(blank password).**
- Lakukan konfigurasi **hostname (identity)** menggunakan **R2**, pengalamatan **IP** pada **interface ether1** menggunakan **192.168.169.2/24** dan memverifikasi pengaturan pengalamatan IP yang telah dilakukan. Selain itu menampilkan informasi **DHCP Client** dan menghapus pengaturan **DHCP Client** pada *interface* serta memverifikasi hasil penghapusannya.

R2

```
[admin@MikroTik] > system identity set name=R2
[admin@R2] > ip address add address=192.168.169.2/24 interface=ether1
[admin@R2] > ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         INTERFACE
 0   192.168.169.2/24   192.168.169.0   ether1
[admin@R2] > ip dhcp-client print
Flags: X - disabled, I - invalid, D - dynamic
 #   INTERFACE           USE ADD-DEFAULT-ROUTE STATUS        ADDRESS
 0   ether2              yes yes               searching...
[admin@R2] > ip dhcp-client remove 0
[admin@R2] > ip dhcp-client print
Flags: X - disabled, I - invalid, D - dynamic
 #   INTERFACE           USE ADD-DEFAULT-ROUTE STATUS        ADDRESS
[admin@R2] >
```

KONFIGURASI DASAR ROUTER R3 PADA PNETLAB

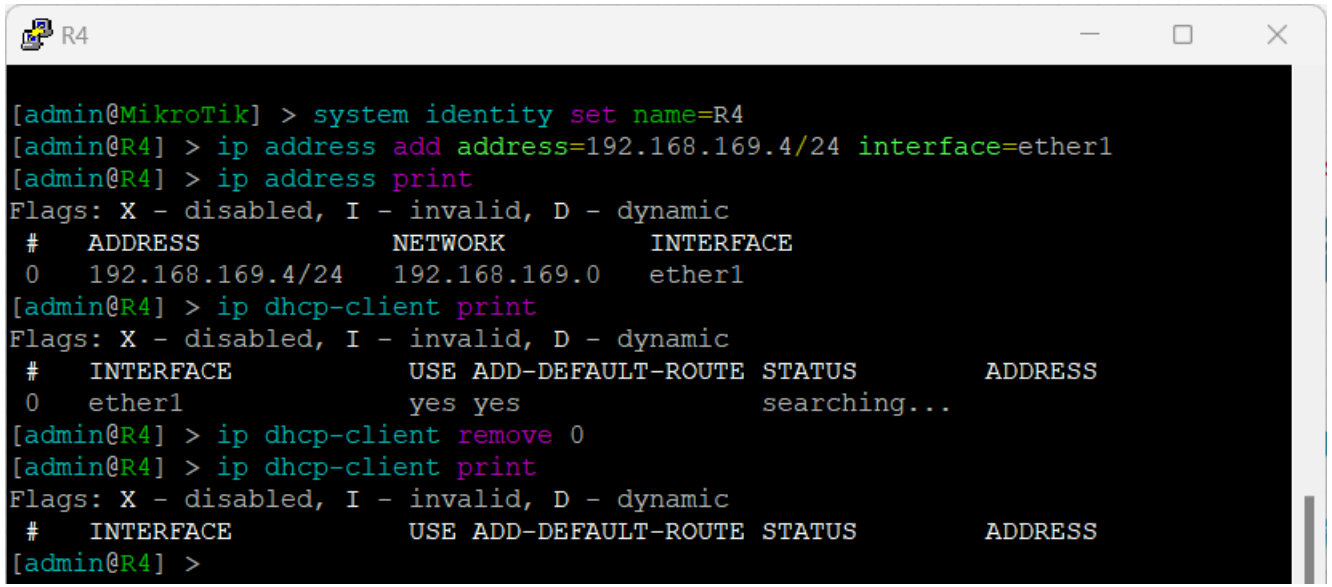
- Akses **terminal** dari **Router R3** dan lakukan login sebagai **admin** tanpa sandi **(blank password).**
- Lakukan konfigurasi **hostname (identity)** menggunakan **R3**, pengalamatan **IP** pada **interface ether1** menggunakan **192.168.169.3/24** dan memverifikasi pengaturan pengalamatan IP yang telah dilakukan. Selain itu menampilkan informasi **DHCP Client** dan menghapus pengaturan **DHCP Client** pada *interface* serta memverifikasi hasil penghapusannya.

R3

```
[admin@MikroTik] > system identity set name=R3
[admin@R3] > ip address add address=192.168.169.3/24 interface=ether1
[admin@R3] > ip address print
Flags: X - disabled, I - invalid, D - dynamic
 #   ADDRESS            NETWORK         INTERFACE
 0   192.168.169.3/24   192.168.169.0   ether1
[admin@R3] > ip dhcp-client print
Flags: X - disabled, I - invalid, D - dynamic
 #   INTERFACE          USE ADD-DEFAULT-ROUTE STATUS        ADDRESS
 0   ether1             yes yes               searching...
[admin@R3] > ip dhcp-client remove 0
[admin@R3] > ip dhcp-client print
Flags: X - disabled, I - invalid, D - dynamic
 #   INTERFACE          USE ADD-DEFAULT-ROUTE STATUS        ADDRESS
[admin@R3] >
```

KONFIGURASI DASAR ROUTER R4 PADA PNETLAB

- Akses **terminal** dari **Router R4** dan lakukan login sebagai **admin** tanpa sandi **(blank password).**
- Lakukan konfigurasi **hostname (identity)** menggunakan **R4**, pengalamatan **IP** pada **interface ether1** menggunakan **192.168.169.4/24** dan memverifikasi pengaturan pengalamatan IP yang telah dilakukan. Selain itu menampilkan informasi **DHCP Client** dan menghapus pengaturan **DHCP Client** pada *interface* serta memverifikasi hasil penghapusannya.



VERIFIKASI KONEKSI DARI DOCKER-ANSIBLE KE ALAMAT IP DARI SETIAP ROUTER MENGGUNAKAN PING

```
root@Docker-Ansible: /Docker-images
root@Docker-Ansible:/Docker-images# ping 192.168.169.1 -c 2
PING 192.168.169.1 (192.168.169.1) 56(84) bytes of data.
64 bytes from 192.168.169.1: icmp_seq=1 ttl=64 time=3.19 ms
64 bytes from 192.168.169.1: icmp_seq=2 ttl=64 time=2.36 ms

--- 192.168.169.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 2.357/2.774/3.192/0.417 ms
root@Docker-Ansible:/Docker-images# ping 192.168.169.2 -c 2
PING 192.168.169.2 (192.168.169.2) 56(84) bytes of data.
64 bytes from 192.168.169.2: icmp_seq=1 ttl=64 time=3.03 ms
64 bytes from 192.168.169.2: icmp_seq=2 ttl=64 time=1.16 ms

--- 192.168.169.2 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1001ms
rtt min/avg/max/mdev = 1.164/2.097/3.031/0.933 ms
```

```
root@Docker-Ansible: /Docker-images
root@Docker-Ansible:/Docker-images# ping 192.168.169.3 -c 2
PING 192.168.169.3 (192.168.169.3) 56(84) bytes of data.
64 bytes from 192.168.169.3: icmp_seq=1 ttl=64 time=2.83 ms
64 bytes from 192.168.169.3: icmp_seq=2 ttl=64 time=2.66 ms

--- 192.168.169.3 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 2.663/2.746/2.830/0.083 ms
root@Docker-Ansible:/Docker-images# ping 192.168.169.4 -c 2
PING 192.168.169.4 (192.168.169.4) 56(84) bytes of data.
64 bytes from 192.168.169.4: icmp_seq=1 ttl=64 time=3.10 ms
64 bytes from 192.168.169.4: icmp_seq=2 ttl=64 time=3.63 ms

--- 192.168.169.4 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1002ms
rtt min/avg/max/mdev = 3.097/3.364/3.631/0.267 ms
```

VERIFIKASI KONEKSI DARI COMMAND PROMPT WINDOWS KE ALAMAT IP DARI SETIAP ROUTER MENGGUNAKAN PING

```
Command Prompt

Microsoft Windows [Version 10.0.22621.1848]
(c) Microsoft Corporation. All rights reserved.

C:\Users\PUTU>ping 192.168.169.1 -n 2

Pinging 192.168.169.1 with 32 bytes of data:
Reply from 192.168.169.1: bytes=32 time=2ms TTL=64
Reply from 192.168.169.1: bytes=32 time=3ms TTL=64

Ping statistics for 192.168.169.1:
    Packets: Sent = 2, Received = 2, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 3ms, Average = 2ms

C:\Users\PUTU>ping 192.168.169.2 -n 2

Pinging 192.168.169.2 with 32 bytes of data:
Reply from 192.168.169.2: bytes=32 time=2ms TTL=64
Reply from 192.168.169.2: bytes=32 time=4ms TTL=64

Ping statistics for 192.168.169.2:
    Packets: Sent = 2, Received = 2, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 2ms, Maximum = 4ms, Average = 3ms

C:\Users\PUTU>
```

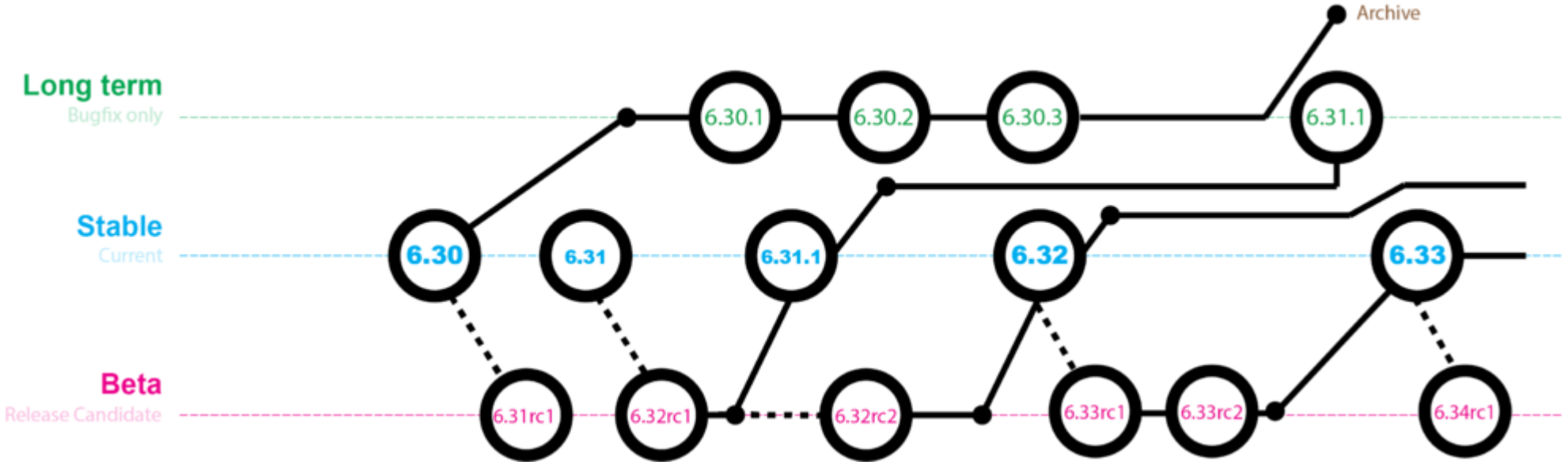
```
Command Prompt

C:\Users\PUTU>ping 192.168.169.3 -n 2

Pinging 192.168.169.3 with 32 bytes of data:
Reply from 192.168.169.3: bytes=32 time=6ms TTL=64
Reply from 192.168.169.3: bytes=32 time=4ms TTL=64

Ping statistics for 192.168.169.3:
    Packets: Sent = 2, Received = 2, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 6ms, Average = 5ms

C:\Users\PUTU>ping 192.168.169.4 -n 2

Pinging 192.168.169.4 with 32 bytes of data:
Reply from 192.168.169.4: bytes=32 time=5ms TTL=64
Reply from 192.168.169.4: bytes=32 time=4ms TTL=64

Ping statistics for 192.168.169.4:
    Packets: Sent = 2, Received = 2, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 4ms, Maximum = 5ms, Average = 4ms

C:\Users\PUTU>
```

PENGAMANAN ROUTER MIKROTIK

VERSI ROUTEROS

- Disarankan untuk menggunakan versi *Router*OS terkini.
- Beberapa rilis *Router*OS versi lama memiliki kelemahan atau kerentanan yang telah diperbaiki.
- MikroTik secara berkala menambahkan fungsionalitas baru dan meningkatkan unjuk kerja serta stabilitas dari *Router*OS dengan merilis pembaharuan.
- Lakukan pembaharuan versi *Router*OS dari perangkat yang digunakan untuk memastikan keamanannya.

PENOMORAN VERSI ROUTEROS



(**Image Source:** https://wiki.mikrotik.com/wiki/File:Metro-systemv2.png)

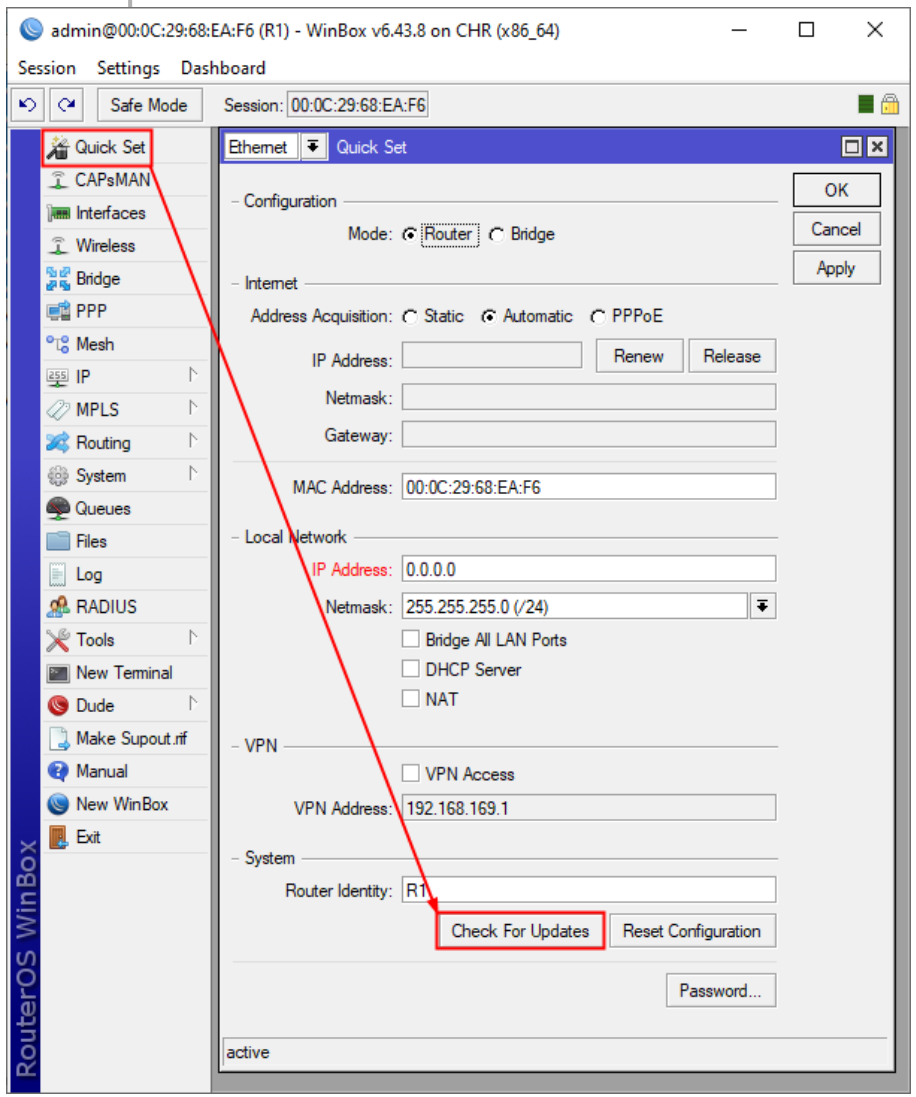
Versi RouterOS memiliki 3 (tiga) *release chains*:

- Beta: dirilis setiap beberapa hari dan hanya menjalani pengujian internal dasar.
- Stable: dirilis setiap beberapa minggu dan mencakup seluruh fitur serta perbaikan yang teruji.
- Long Term: jarang dirilis dan mencakup hanya perbaikan yang sangat penting serta peningkatan dalam satu nomor percabangan namun tidak menambah fitur baru.

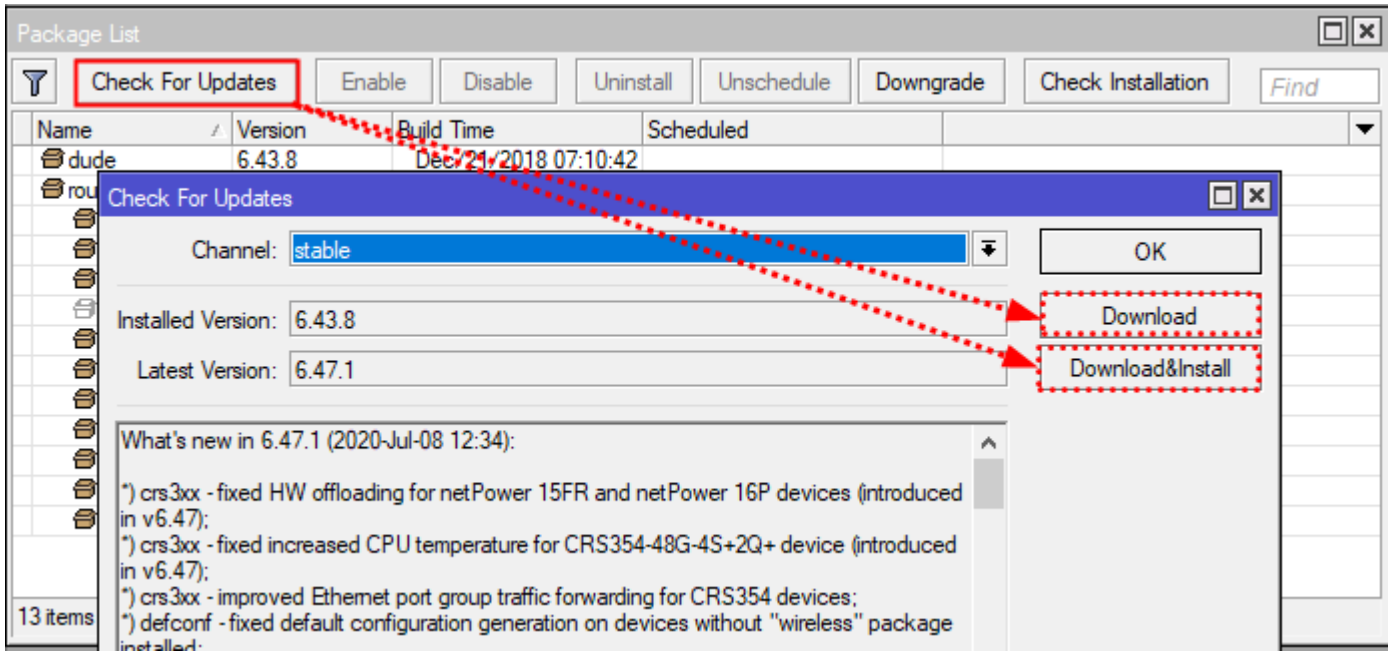
PEMBAHARUAN ROUTEROS

- Terdapat 2 (dua) metode **Upgrade** yaitu **Automatic Upgrade** dan **Manual Upgrade**.
- Fitur **Automatic Upgrade** akan menghubungkan ke **MikroTik download servers** dan mengecek apakah terdapat versi *RouterOS* yang lebih baru untuk perangkat yang digunakan saat ini.
- Terdapat beberapa cara yang dapat digunakan untuk melakukan pembaharuan *RouterOS* secara otomatis yaitu melalui:
  - Quickset
  - System Packages

METODE PEMBAHARUAN OTOMATIS



- Quickset
- System Packages



ROUTEROS AUTO-UPGRADE

- Tersedia mulai *RouterOS* versi 6.
- Otomatisasi proses pembaharuan dapat dilakukan dengan menjalankan skrip pada System Scheduler.
- Skrip untuk *RouterOS* setelah versi 6.31.

```
/system package update
check-for-updates once
:delay 3s;
:if ( [get status] = "New version is available") do={ install }
```

- Skrip untuk *RouterOS* sampai versi 6.31.

```
/system package update
check-for-updates
:delay 3s;
:if ( [get current-version] != [get latest-version]) do={ upgrade }
```

DEMO ROUTEROS AUTO-GRADE

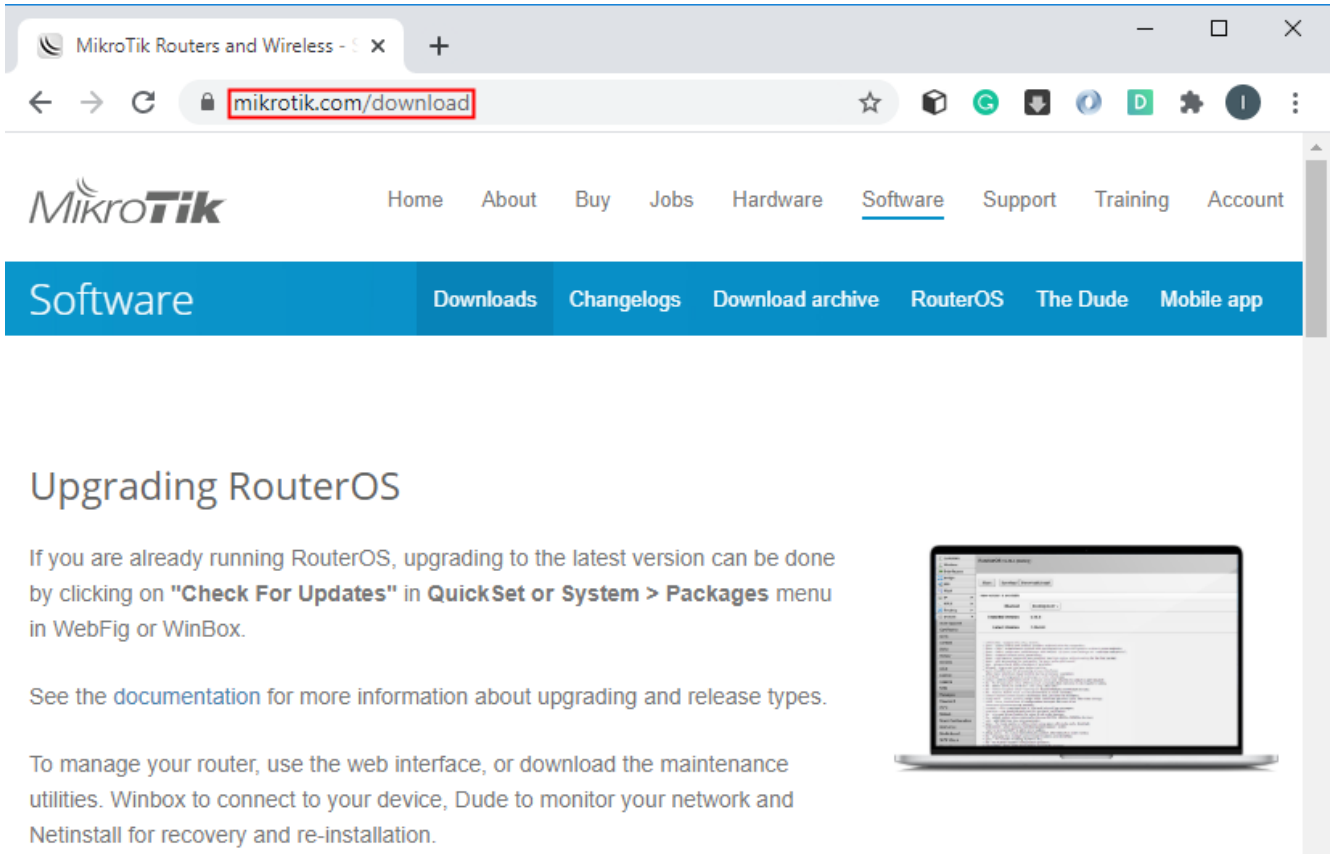
METODE PEMBAHARUAN MANUAL

Terdapat beberapa cara yang dapat digunakan untuk melakukan pembaharuan *RouterOS* secara manual yaitu melalui:

- Winbox, dengan melakukan *drag* & *drop file* routeros-*.npk ke menu **Files**.
- WebFig, dengan mengunggah *file* routeros-*.npk melalui menu **Files**.
- FTP, dengan mengunggah *file* routeros-*.npk ke *root directory*.
- The Dude

PROSES PEMBAHARUAN SECARA MANUAL

Awali dengan mengunduh *file upgrade package* routeros-*.npk sesuai dengan jenis sistem dari perangkat yang digunakan pada situs MikroTik https://www.mikrotik.com/download



DEMO METODE PEMBAHARUAN SECARA MANUAL

PENGAMANAN AKSES KE ROUTER

1. Mengubah *default username* admin menggunakan username berbeda, sebagai contoh menggunakan ansible.

```
/user add name=ansible password=ansiblesecret group=full
```

```
/user remove admin
```

Atau

```
/user set name=ansible admin
```

2. Mengatur sandi (*password*) dari *user* dan menggunakan sandi yang aman, sebagai contoh menggunakan ansiblesecret.

```
/user set password=ansiblesecret admin
```

Atau

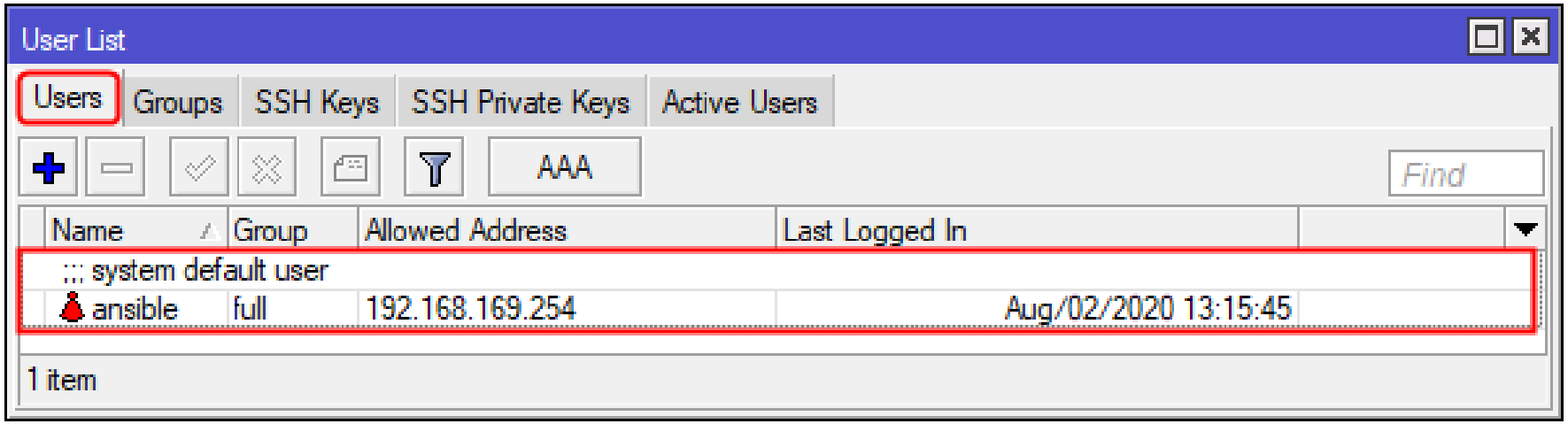
```
/password
```

PENGAMANAN AKSES KE ROUTER

3. Membatasi akses *username* agar hanya dapat digunakan login dari alamat IP sumber tertentu.

   Sebagai contoh *username* ansible hanya dapat digunakan untuk mengakses *RouterOS* dari IP 192.168.169.254.

```
/user set address=192.168.169.254 ansible
```



DEMO VERIFIKASI HASIL PENGAMANAN AKSES KE ROUTER

PENGAMANAN IP SERVICES DARI ROUTER

1. Menonaktifkan *IP Service* yang tidak aman, seperti ftp, telnet, www.

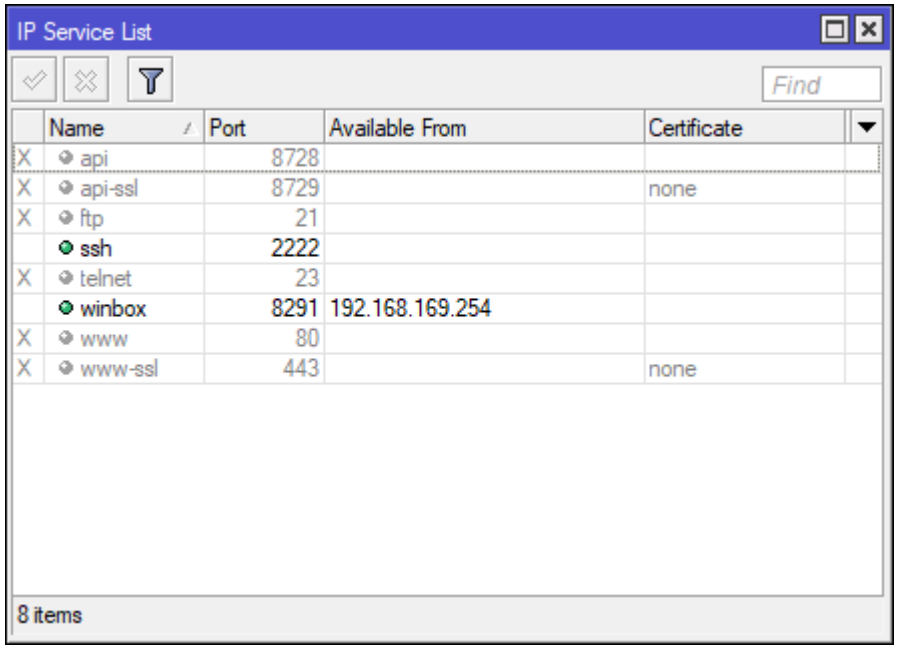
```
/ip service disable ftp,telnet,www
```

2. Menonaktifkan *IP Service* yang tidak digunakan, seperti api, api-ssl.

```
/ip service disable api,api-ssl
```

3. Mengubah nomor port dan membatasi akses pada *IP Service*.

```
/ip service set port=2222 ssh
/ip service set address=192.168.169.254 winbox
/ip service print
```



DEMO VERIFIKASI PENGAMANAN IP SERVICES DARI ROUTER

MENONAKTIFKAN ROUTEROS MAC-ACCESS

1. Menonaktifkan layanan MAC-Telnet.

```
[admin@R1] > /tool mac-server set allowed-interface-list=none
[admin@R1] > /tool mac-server print
  allowed-interface-list: none
```

2. Menonaktifkan layanan MAC-Winbox.

```
[admin@R1] > /tool mac-server mac-winbox set allowed-interface-list=none
[admin@R1] > /tool mac-server mac-winbox print
  allowed-interface-list: none
```

3. Menonaktifkan layanan MAC-Ping.

```
[admin@R1] > /tool mac-server ping set enabled=no
[admin@R1] > /tool mac-server ping print
  enabled: no
```

DEMO VERIFIKASI PENONAKTIFAN ROUTER MAC-ACCESS

MENONAKTIFKAN NEIGHBOR DISCOVERY

- **Mikrotik Neighbor Discovery Protocol (MNDP)** digunakan untuk menemukan dan mengetahui perangkat Mikrotik lainnya yang terdapat di jaringan.
- Menonaktifkan *neighbor discovery* pada semua *interface*.

```
[admin@R1] > /ip neighbor discovery-settings set discover-interface-list=none
[admin@R1] > /ip neighbor discovery-settings print
  discover-interface-list: none
```

DEMO VERIFIKASI PENONAKTIFAN MNDP

MENONAKTIFKAN BANDWIDTH SERVER

- Bandwidth Server digunakan untuk menguji throughput diantara 2 (dua) *router MikroTik*.
- Disarankan untuk menonaktifkan Bandwidth Test (Btest) Server pada lingkungan *production*.

```
[admin@R1] > /tool bandwidth-server set enabled=no
[admin@R1] > /tool bandwidth-server print
                  enabled: no
             authenticate: yes
  allocate-udp-ports-from: 2000
             max-sessions: 100
```

DEMO VERIFIKASI PENONAKTIFAN BTEST SERVER

MENONAKTIFKAN DNS CACHE

- Domain Name System (DNS) Cache digunakan untuk meminimalkan permintaan DNS ke server DNS eksternal sehingga meminimalkan waktu resolusi atau pemetaan DNS.
- Apabila tidak diperlukan maka DNS Cache dapat dinonaktifkan.

```
[admin@R1] > /ip dns set allow-remote-requests=no
[admin@R1] > /ip dns print
                    servers: 8.8.8.8,8.8.4.4
            dynamic-servers:
      allow-remote-requests: no
        max-udp-packet-size: 4096
       query-server-timeout: 2s
        query-total-timeout: 10s
    max-concurrent-queries: 100
max-concurrent-tcp-sessions: 20
                cache-size: 2048KiB
             cache-max-ttl: 1w
                cache-used: 17KiB
```

MENONAKTIFKAN CLIENT SERVICES

- Secara *default service* MikroTik Caching Proxy, MikroTik Socks Proxy, MikroTik (UPNP) Service dan MikroTik Dynamic Name Service tidak aktif. Namun apabila service tersebut telah dikonfigurasi dan tidak diperlukan atau tidak digunakan maka sangat disarankan untuk dinonaktifkan.

- Menonaktifkan MikroTik Caching Proxy.

```
[admin@R1] > /ip proxy set enabled=no
[admin@R1] > /ip proxy print
                    enabled: no
                src-address: ::
                       port: 8080
                  anonymous: no
               parent-proxy: ::
          parent-proxy-port: 0
      cache-administrator: webmaster
           max-cache-size: unlimited
    max-cache-object-size: 2048KiB
            cache-on-disk: no
  max-client-connections: 600
  max-server-connections: 600
           max-fresh-time: 3d
   serialize-connections: no
       always-from-cache: no
          cache-hit-dscp: 4
              cache-path: web-proxy
```

MENONAKTIFKAN CLIENT SERVICES

- Menonaktifkan MikroTik Socks Proxy.

```
[admin@R1] > /ip socks set enabled=no
[admin@R1] > /ip socks print
                   enabled: no
                      port: 1080
  connection-idle-timeout: 2m
           max-connections: 200
```

- Menonaktifkan MikroTik Universal Plug and Play (UPNP) Service.

```
[admin@R1] > /ip upnp print
                             enabled: no
  allow-disable-external-interface: no
                 show-dummy-rule: yes
```

MENONAKTIFKAN CLIENT SERVICES

- Menonaktifkan MikroTik Dynamic Name Service atau IP Cloud.

```
[admin@R1] > /ip cloud set ddns-enabled=no update-time=no
[admin@R1] > /ip cloud print
  ddns-enabled: no
   update-time: no
```

MENGAKTIFKAN STRONG CRYPTO SSH

- *RouterOS* mendukung penggunaan crypto yang lebih tangguh untuk SSH namun secara *default* belum aktif.

`strong-crypto` (*yes* | *no*; Default: **no**)	Use stronger encryption, HMAC algorithms, use bigger DH primes and disallow weaker ones: • prefer 256 and 192 bit encryption instead of 128 bits; • disable null encryption; • prefer sha256 for hashing instead of sha1; • disable md5; • use 2048bit prime for Diffie Hellman exchange instead of 1024bit.

- Mengaktifkan Strong Crypto SSH:

```
[admin@R1] > /ip ssh print
            forwarding-enabled: no
  always-allow-password-login: no
                strong-crypto: yes
                host-key-size: 2048
```

MENONAKTIFKAN INTERFACE ROUTER

- Disarankan untuk menonaktifkan seluruh *interface* yang tidak digunakan pada *router* sehingga mengurangi akses yang tidak sah atau tidak terotorisasi.
- Sebagai contoh untuk menonaktifkan *interface* ether2, ether3, ether4 dan ether5 pada *router* maka dapat mengeksekusi perintah berikut:

```
[admin@R1] > /interface set disabled=yes ether2,ether3,ether4,ether5
[admin@R1] > /interface print
Flags: D - dynamic, X - disabled, R - running, S - slave
 #     NAME                                TYPE       ACTUAL-MTU L2MTU  MAX-L2MTU
 0  R  ether1                              ether            1500
 1  X  ether2                              ether            1500
 2  X  ether3                              ether            1500
 3  X  ether4                              ether            1500
 4  X  ether5                              ether            1500
```

MANAJEMEN KONFIGURASI DAN OTOMATISASI DENGAN ANSIBLE

APA ITU ANSIBLE?

- Menurut situs Ansible, **Ansible** merupakan mesin otomatisasi Teknologi Informasi (TI) sederhana yang dapat mengotomatisasi *cloud provisioning*, manajemen konfigurasi, penerapan aplikasi, *intra-service orchestration* dan kebutuhan TI lainnya.
- Menurut situs Edureka, keuntungan menggunakan *Ansible* antara lain:

1. **Simple**, menggunakan sintak penulisan yang ditulis menggunakan **YAML** yang disebut dengan **playbook**.
2. **Agentless**, tidak diperlukan **agent** atau *software* khusus untuk diinstalasi pada host yang diautomasi.
3. **Powerful & Flexible**, memiliki banyak modul untuk manajemen infrastruktur, jaringan, sistem operasi dan layanan.
4. **Efficient**, tidak ada *software* yang diperlukan untuk instalasi pada server sehingga lebih banyak sumber daya yang dapat digunakan oleh aplikasi.

APA YANG DAPAT DILAKUKAN ANSIBLE?

1. **Provisioning**
   Ansible memastikan paket-paket yang dibutuhkan akan diunduh dan diinstalasi sehingga aplikasi dapat diterapkan.

2. **Configuration Management**
   Menetapkan dan mempertahankan konsistensi dari kinerja produk dengan mencatat dan memperbaharui informasi lengkap yang menjelaskan perangkat keras dan lunak perusahaan. Seperti versi dan pembaharuan yang telah diterapkan pada paket *software* yang terinstal dan lokasi serta alamat jaringan dari perangkat keras.

3. **Application Deployment**
   Ansible dapat memanajemen secara efektif keseluruhan *life cycle* dari aplikasi, mulai dari *development* sampai produksi.

4. **Security and Compliance**
   Kebijakan keamanan dapat didefinisikan pada *Ansible* sehingga proses pemindaian dan pemulihan kebijakan ke lokasi dapat diintegrasikan ke dalam proses secara otomatis.

APA YANG DAPAT DILAKUKAN ANSIBLE?

5. **Orchestration**
   Ansible menyediakan *orchestration* dalam arti menyelaraskan permintaan bisnis dengan aplikasi, data, dan infrastruktur. Ini mendefinisikan kebijakan dan tingkat layanan melalui alur kerja otomatis, penyediaan, dan manajemen perubahan sehingga menciptakan aplikasi selaras dengan infrastruktur yang dapat ditingkatkan dari atas ke bawah berdasarkan kebutuhan setiap aplikasi.

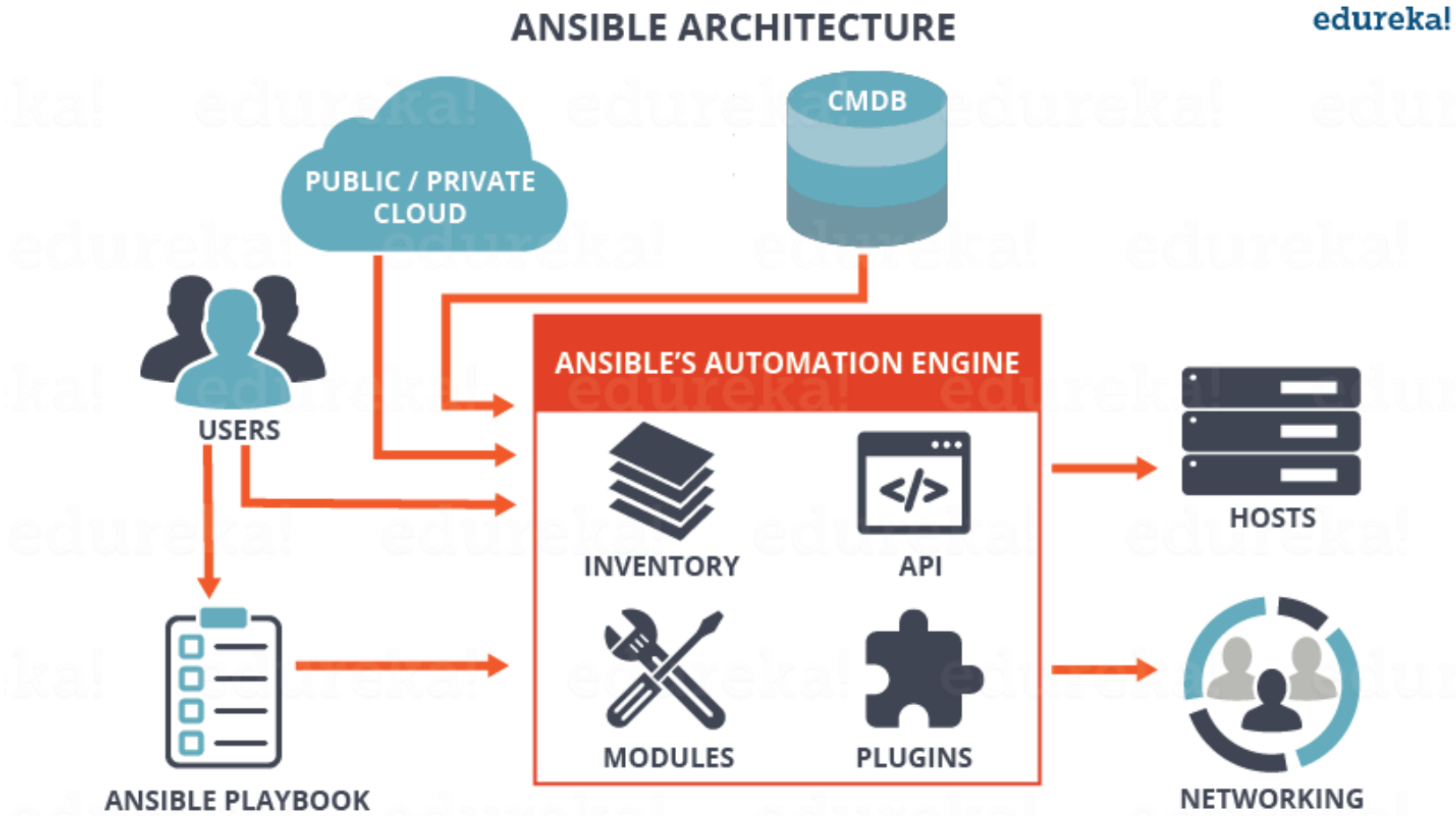
TERMINOLOGI PADA ANSIBLE

- **Controller Machine:** mesin dimana *ansible* diinstalasi dan bertanggungjawab untuk menjalankan *provisioning* pada *server* yang dikelola.
- **Inventory:** file inisialisasi yang memuat informasi tentang *server* yang dikelola.
- **Playbook:** Titik masuk untuk *Ansible provisioning*, dimana otomatisasi didefinisikan melalui tugas (*tasks*) menggunakan format YAML..
- **Task:** Blok yang mendefinisikan satu prosedur untuk dieksekusi, sebagai contoh instalasi *package* tertentu.
- **Module:** Modul merupakan abstraksi dari tugas sistem, seperti berkaitan dengan *package* atau membuat dan mengubah file. *Ansible* memiliki banyak modul *built-in* (bawaan), namun dapat juga dibuat modul khusus.

TERMINOLOGI PADA ANSIBLE

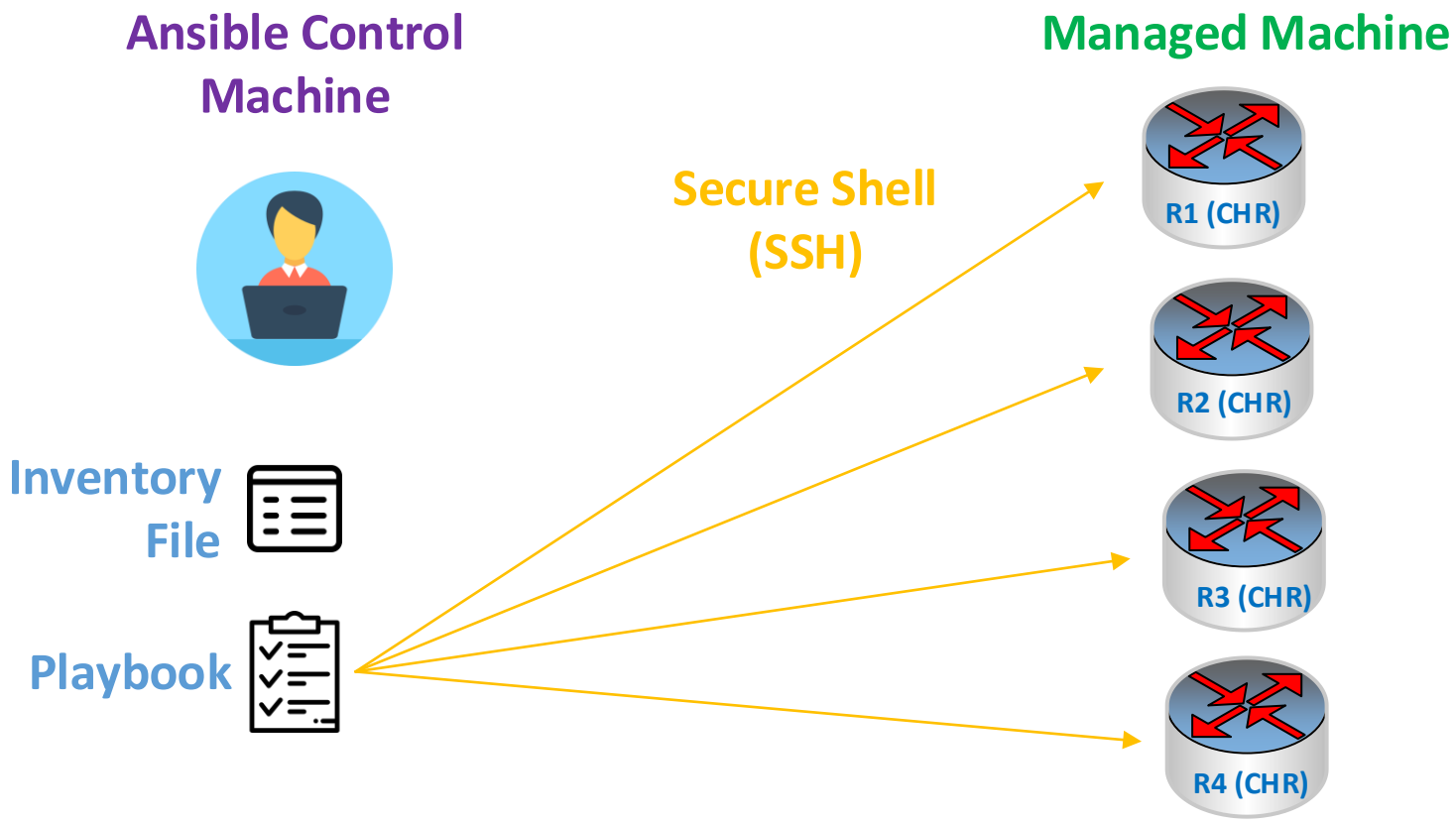
- **Role:** Cara yang telah ditentukan sebelumnya untuk mengatur *playbook* dan file lainnya untuk memfasilitasi berbagi pakai dan menggunakan kembali bagian dari *provisioning*.
- **Play:** *provisioning* yang dieksekusi mulai dari awal sampai akhir disebut dengan *play*. Dengan kata lain, eksekusi dari *playbook* disebut dengan *play*.
- **Facts:** *variable* global yang memuat informasi tentang sistem, seperti *interface* jaringan atau sistem operasi.
- **Handlers:** Digunakan untuk memicu perubahan status dari *service*, seperti me-*restart* atau menghentikan *service*.

ARSITEKTUR ANSIBLE



(**Image source:** https://d1jnx9ba8s6j9r.cloudfront.net/blog/wp-content/uploads/2016/11/Ansible-Architechture-What-Is-Ansible-Edureka-768x449.png)

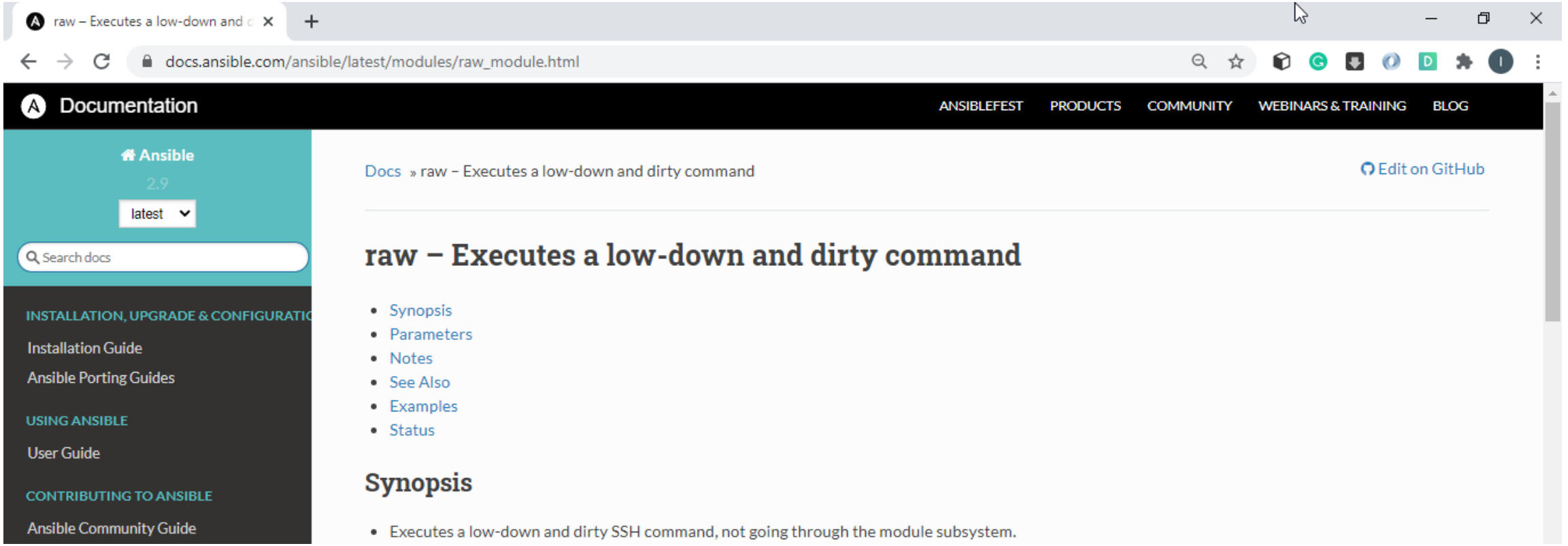
MIKROTIK SECURITY AUTOMATION WITH ANSIBLE



ANSIBLE MODULES (1)

- **raw**

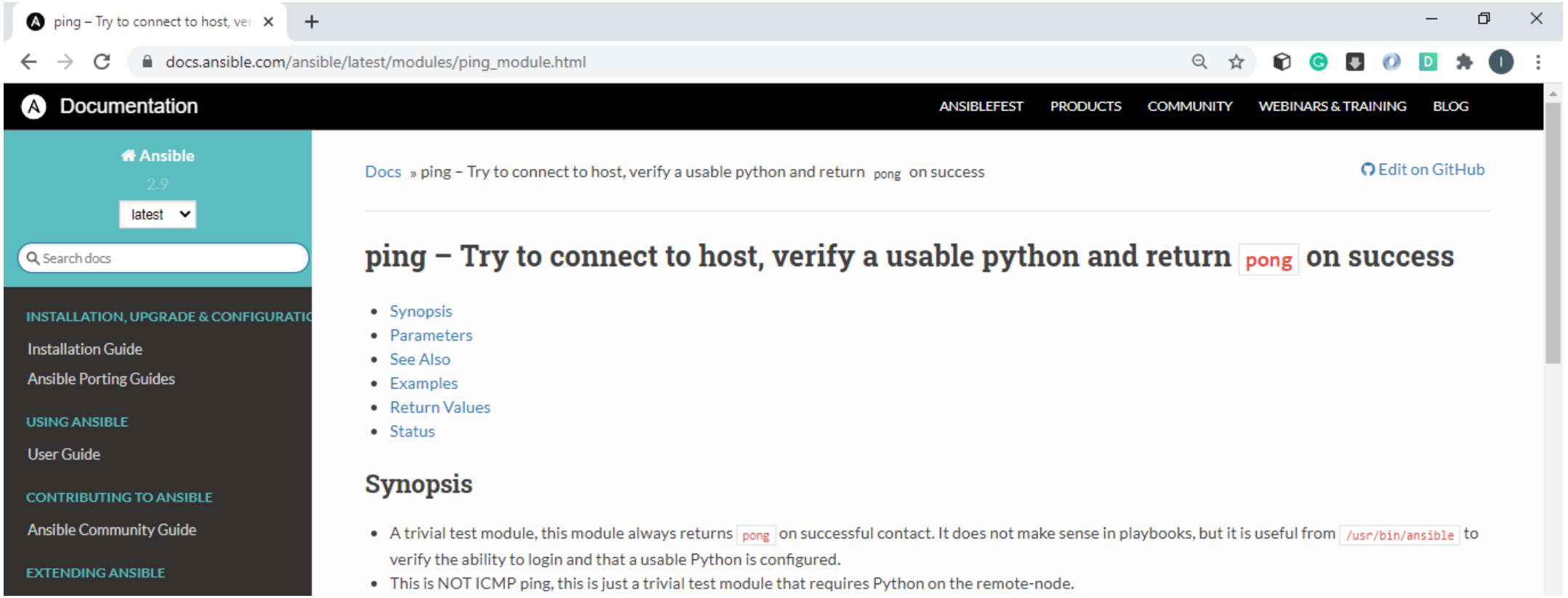
Executes a low-down and dirty SSH command, not going through the module subsystem. (https://docs.ansible.com/ansible/latest/modules/raw_module.html)



ANSIBLE MODULES (2)

- **ping**

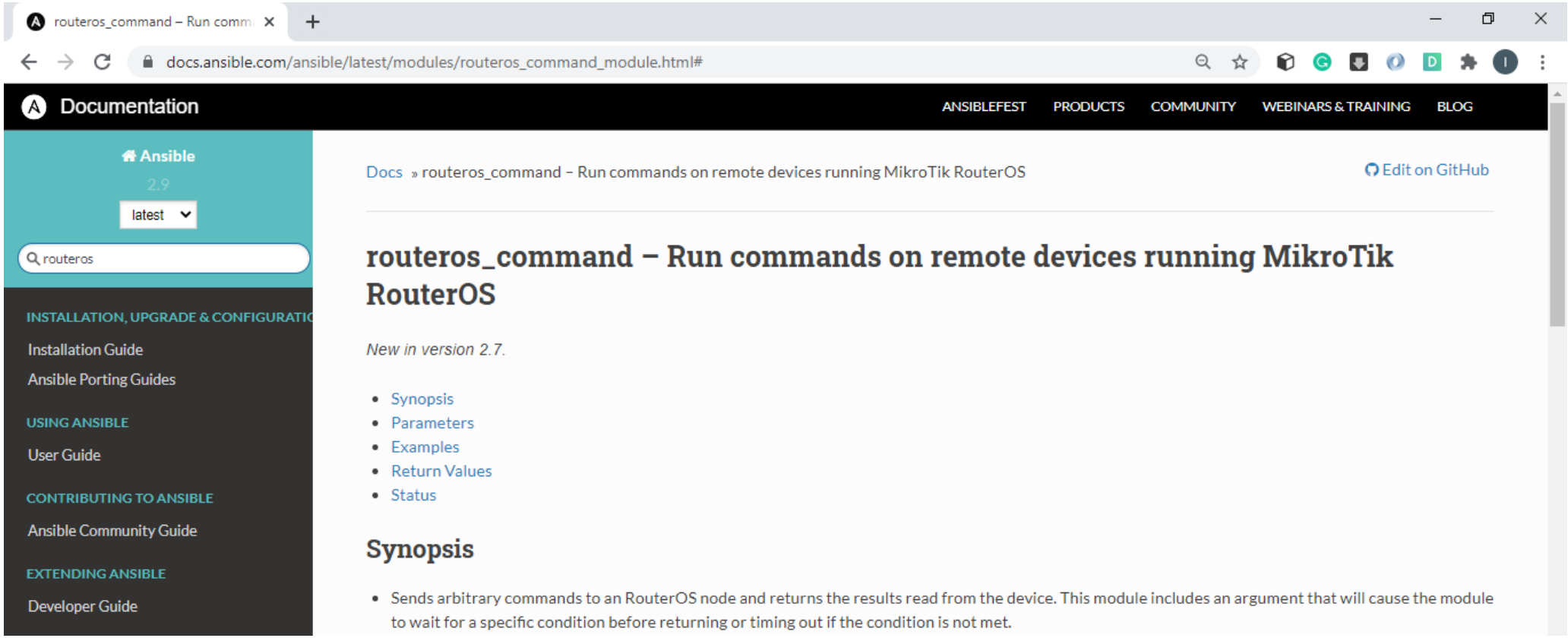
Try to connect to host, verify a usable python and return pong on success. (https://docs.ansible.com/ansible/latest/modules/ping_module.html)



ANSIBLE MODULES (3)

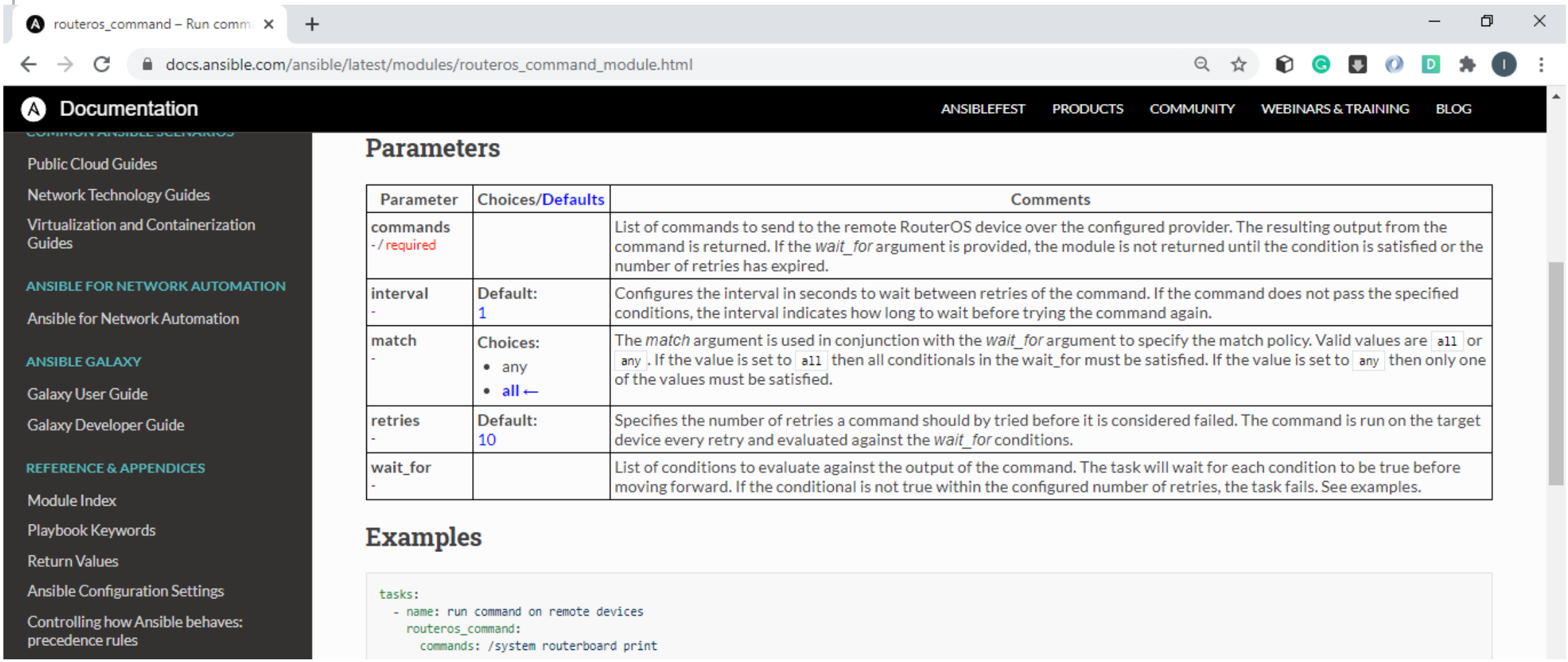
- **routeros_command**

Run commands on remote devices running MikroTik RouterOS.
(https://docs.ansible.com/ansible/latest/modules/routeros_command_module.html)



ANSIBLE MODULES (4)

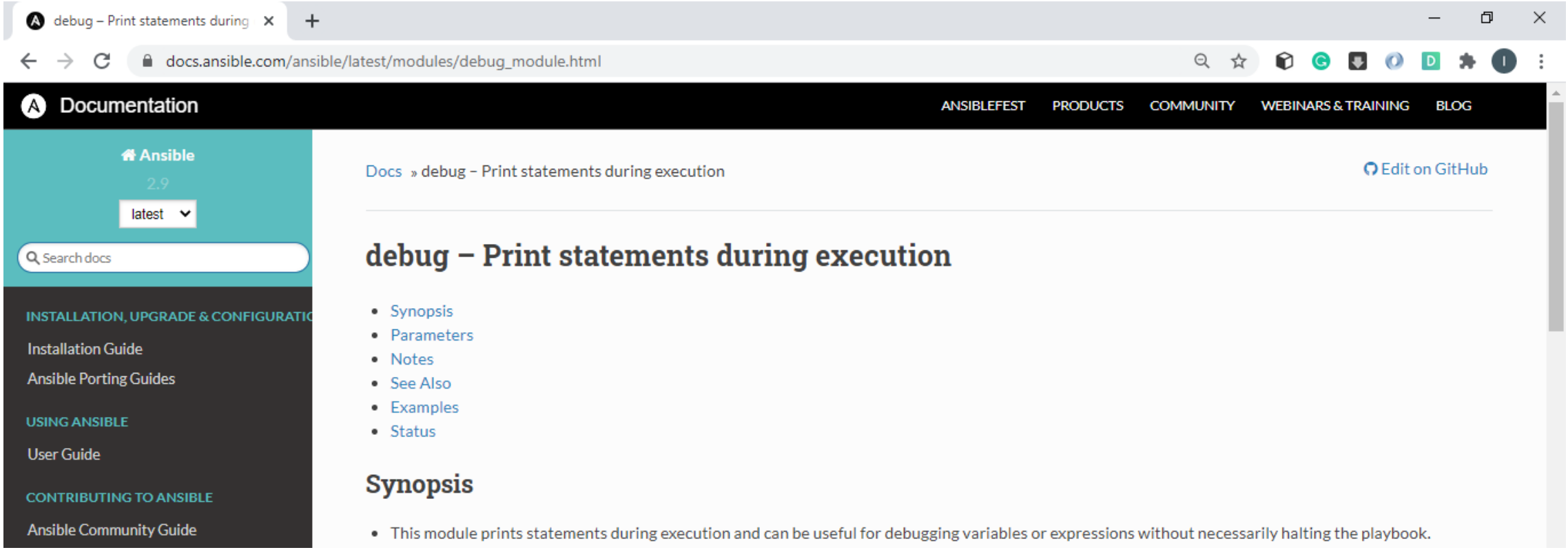
- **routeros_command Parameters.**



ANSIBLE MODULES (5)

- **debug**

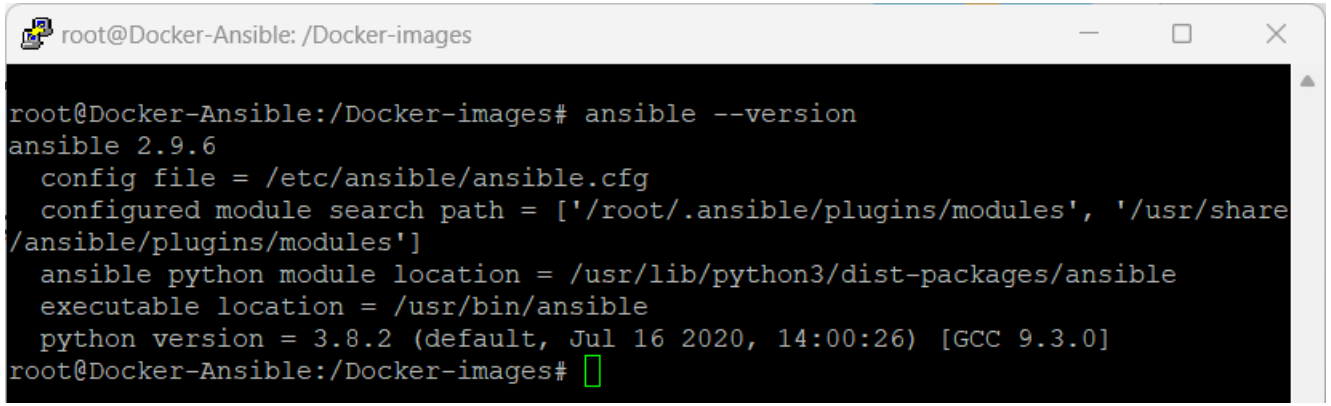
Print statements during execution.
(https://docs.ansible.com/ansible/latest/modules/debug_module.html)



MENGAKSES TERMINAL DARI DOCKER-ANSIBLE

- Pastikan **node Docker-Ansible** telah berjalan (start) dan akses terminal dari *node* tersebut.
- Menampilkan informasi versi **Ansible** yang telah terinstalasi pada *node* tersebut dengan mengeksekusi perintah berikut:

```
# ansible –version
```



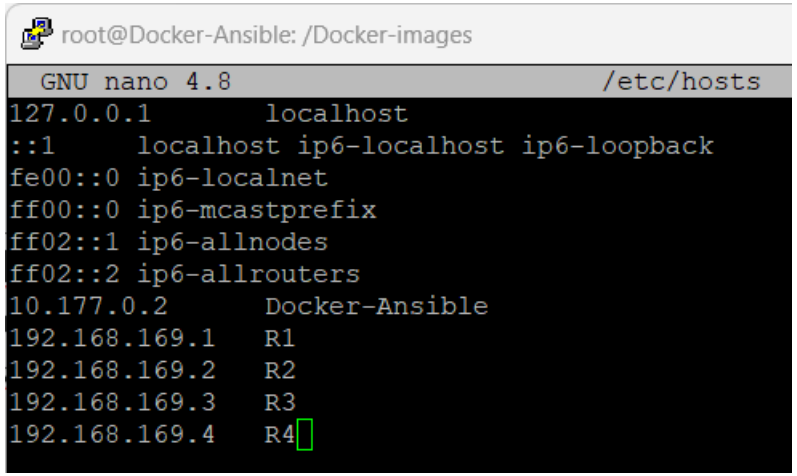
Terlihat versi Ansible yang digunakan adalah **2.9.6**.

MENGATUR PEMETAAN ALAMAT IP KE HOSTNAME PADA FILE /ETC/HOSTS

- Menambahkan pemetaan alamat **IP ke hostname** dari setiap **router** pada *file* **/etc/hosts** menggunakan editor **nano**. Cantumkan pada baris paling akhir dari file **hosts** tersebut dengan format penulisan setiap barisnya adalah **IP alias**.

```
# nano /etc/hosts
```

```
192.168.169.1   R1
192.168.169.2   R2
192.168.169.3   R3
192.168.169.4   R4
```



Simpan perubahan dengan menekan **CTRL+O** dan tekan **Enter.**

Keluar dari editor *nano* dengan menekan **CTRL+X.**

MENGATUR ANSIBLE INVENTORY

- **Inventory** merupakan file inisialisasi yang digunakan oleh *Ansible* untuk mendaftarkan dan mengelompokkan mesin atau *host* yang akan dikelola. Lokasi dari file **inventory** secara *default* adalah "**/etc/ansible/hosts**".
- Menambahkan **managed machines** ke file *inventory* menggunakan editor **nano**.
  # nano /etc/ansible/hosts

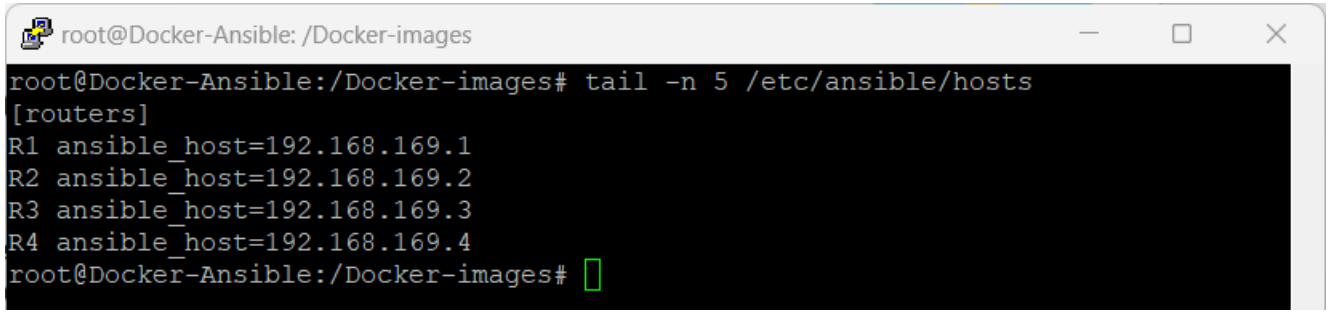
```
[routers]
R1 ansible_host=192.168.169.1
R2 ansible_host=192.168.169.2
R3 ansible_host=192.168.169.3
R4 ansible_host=192.168.169.4
```

Variable **ansible_host** digunakan untuk mengatur alamat IP dari target atau *managed machine* yaitu dalam hal ini adalah alamat IP dari setiap *router*. Sebagai contoh **R1** merupakan *inventory_hostname* atau alias bagi *router* dengan alamat **192.168.169.1**.

Simpan perubahan dengan menekan **CTRL+O** dan tekan **Enter**. Keluar dari editor *nano* dengan menekan **CTRL+X**.

VERIFIKASI ANSIBLE INVENTORY

- Memverifikasi hasil penambahan *inventory*.
  `# tail -n 5 /etc/ansible/hosts`



MENGATUR PUBLIC KEY AUTHENTICATION UNTUK SSH DENGAN MEMBUAT KEY PAIR

Membuat **key pair** menggunakan utilitas **ssh-keygen** pada **Ansible Control Machine.**

```
# ssh-keygen -m PEM
```

```
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
```

Tekan **Enter** pada inputan **Enter file in which to save the key (/root/.ssh/id_rsa):** untuk menggunakan lokasi dan nama file *default* penyimpanan *key* yaitu **/root/.ssh/id_rsa.**

```
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```

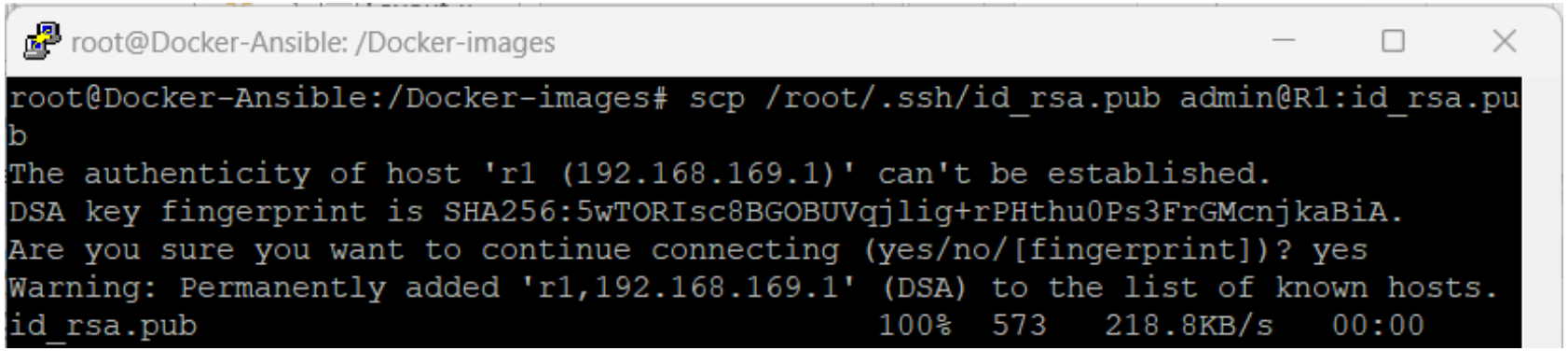
Tekan **Enter** pada inputan "**Enter passphrase (empty for no passphrase):**" dan "**Enter same passphrase again:**" yang tampil untuk apabila ingin mengosongkan **passphrase.** Tampil proses pembuatan *key*. Tunggu hingga proses selesai dilakukan.

MENYALINKAN PUBLIC KEY DARI ANSIBLE CONTROL MACHINE KE SELURUH ROUTER

- Penyalinan Public Key dari Docker-Ansible Control Machine ke *router* **R1** dapat dilakukan dengan menggunakan utilitas scp (*secure copy*).

```
# scp /root/.ssh/id_rsa.pub admin@R1:id_rsa.pub
```

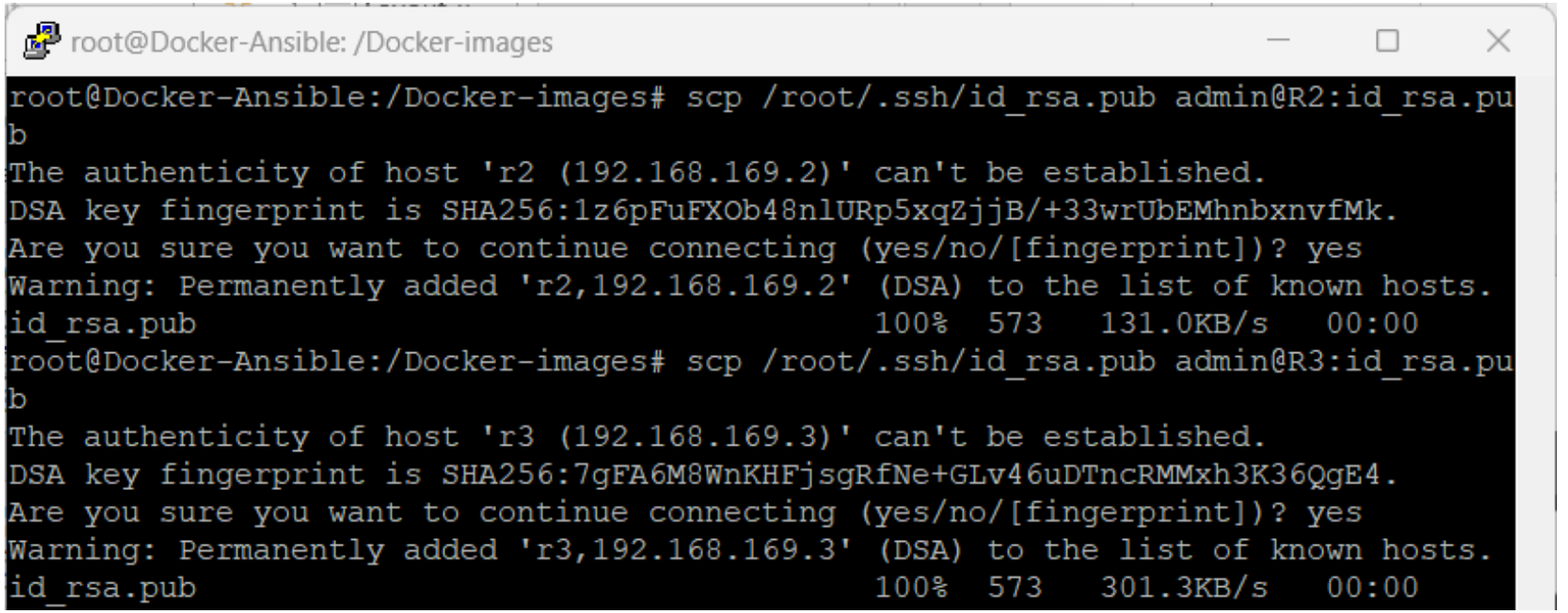
Ketik **yes** dan tekan **Enter** pada pesan konfirmasi **Are you sure you want to continue connecting (yes/no)?.**



Selanjutnya tampil inputan **password** untuk *user* **admin** dari **managed machine R1** dengan alamat IP **192.168.169.1**. Masukkan sandi dari *user* **admin** dan tekan **Enter.**

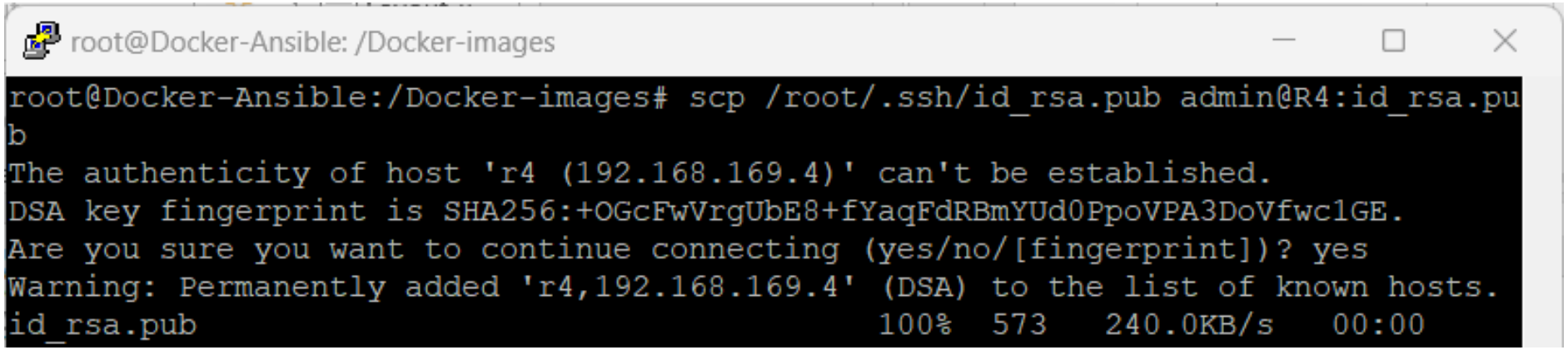
MENYALINKAN PUBLIC KEY DARI ANSIBLE CONTROL MACHINE KE SELURUH ROUTER

- Dengan cara yang sama, lakukan untuk **managed machine** **R2, R3** dan **R4**.
- Proses penyalinan Public Key ke **R2** dan **R3**.



MENYALINKAN PUBLIC KEY DARI ANSIBLE CONTROL MACHINE KE SELURUH ROUTER

- Proses penyalinan Public Key ke **R4**.



3 (TIGA) CARA MENJALANKAN ANSIBLE

1. **Ad-Hoc**
   `ansible <inventory> -m`
   Sebagai contoh eksekusi *ansible* untuk *group inventory* dengan nama "**routers**" dan modul **ping**.
   `# ansible routers –m ping`

2. **Playbooks**
   `ansible-playbook filename.yml`
   Sebagai contoh eksekusi *ansible playbook* dengan nama file **security.yml**.
   `# ansible-playbook security.yml`

3. **Automation Framework**

   Menurut situs Ansible, **Ansible Tower** merupakan solusi berbasis web yang membuat *Ansible* lebih mudah digunakan untuk tim TI dan dirancang untuk menjadi pusat semua tugas otomasi serta memungkinkan untuk mengontrol akses pengguna. **Inventory** dapat dikelola secara grafis atau disinkronisasikan dengan beragam sumber **Cloud**. **Tower** juga mencatat (*log*) semua pekerjaan, terintegrasi dengan **LDAP** dan memiliki **API REST**. Tersedia pula tool **Command Line** untuk memudahkan integrasi dengan **Jenkins** juga.

ANSIBLE AD-HOC MODUL RAW (1)

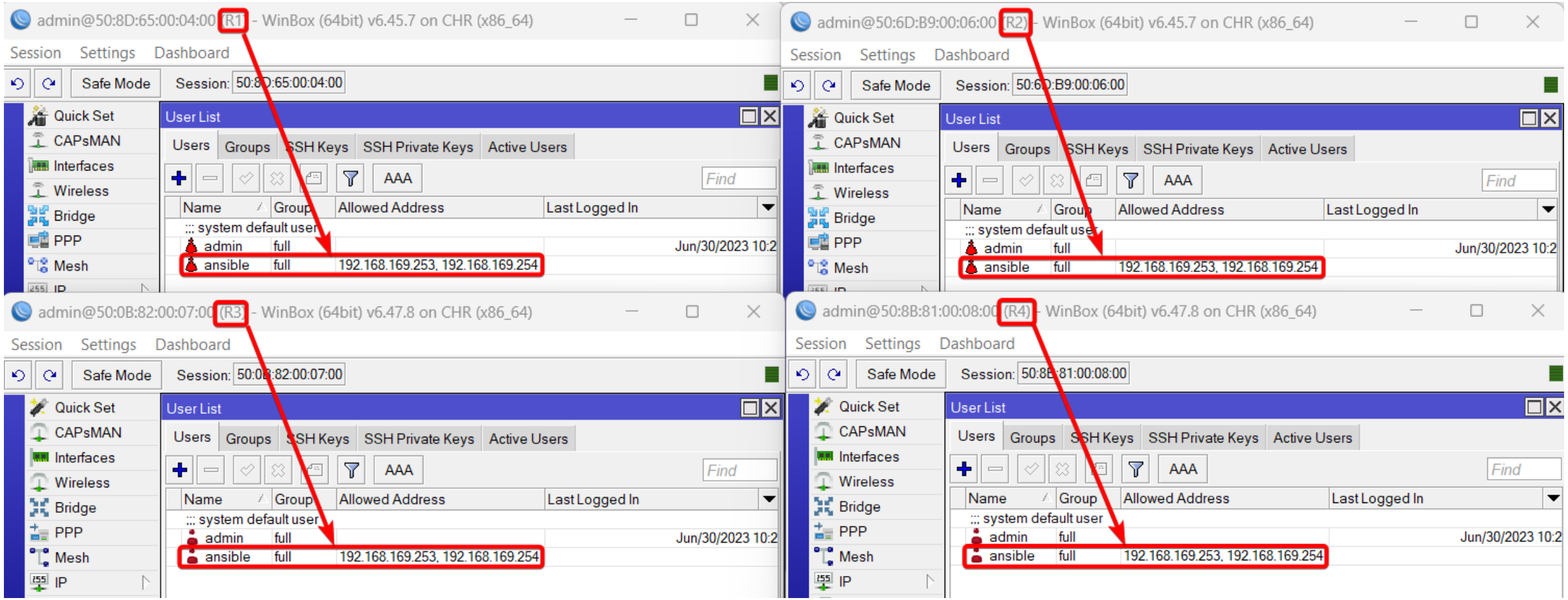
- Membuat *user* baru pada seluruh *router* dengan ketentuan:
  a. Username: ansible
  b. Password: ansiblesecret
  c. Group: full
  d. Allowed Address: 192.168.169.254 dan 192.168.169.253

```
# ansible routers -m raw -a "/user add name=ansible group=full
  password=ansiblesecret address=192.168.169.253,192.168.169.254;quit;"
  -u admin -k
SSH password:
R3 | CHANGED | rc=0 >>
interrupted
Shared connection to 192.168.169.3 closed.


R4 | CHANGED | rc=0 >>
interrupted
Shared connection to 192.168.169.4 closed.


R2 | CHANGED | rc=0 >>
interrupted
Shared connection to 192.168.169.2 closed.


R1 | CHANGED | rc=0 >>
interrupted
Shared connection to 192.168.169.1 closed.
```

Pada inputan **SSH password:** yang tampil, masukkan sandi dari user **admin** dan tekan **Enter.**

ANSIBLE AD-HOC MODULE RAW (2)

- Hasil verifikasi melalui Winbox menunjukkan bahwa *username* ansible berhasil dibuat pada setiap *router*.



ANSIBLE AD-HOC MODULE RAW (3)

- Meng-*import file* **SSH key id_rsa.pub** yang telah disalin ke setiap *router* pada tahap sebelumnya untuk *user* ansible.

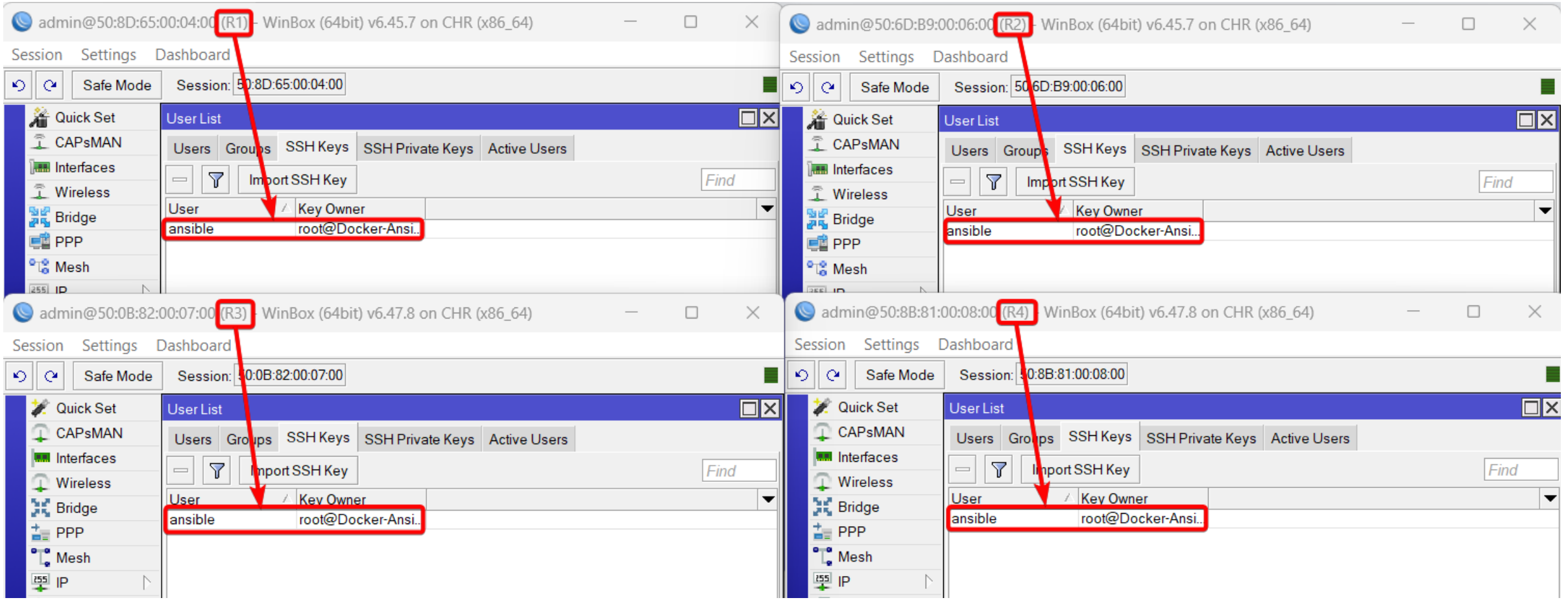
```
# ansible routers -m raw -a "/user ssh-keys import
  public-key-file=id_rsa.pub user=ansible;quit;" -u admin -k
```

```
SSH password:
R2 | CHANGED | rc=0 >>
interrupted
Shared connection to 192.168.169.2 closed.


R4 | CHANGED | rc=0 >>
interrupted
Shared connection to 192.168.169.4 closed.


R3 | CHANGED | rc=0 >>
interrupted
Shared connection to 192.168.169.3 closed.


R1 | CHANGED | rc=0 >>
interrupted
Shared connection to 192.168.169.1 closed.
```

Pada inputan **SSH password:** yang tampil, masukkan sandi dari user **admin** dan tekan **Enter.**

ANSIBLE AD-HOC MODULE RAW (4)

- Hasil verifikasi melalui Winbox menunjukkan bahwa **SSH key** untuk *user* ansible berhasil di *import* pada setiap *router*.



MENGATUR GROUP VARIABLES PADA FILE INVENTORY /ETC/ANSIBLE/HOSTS

 # nano /etc/ansible/hosts

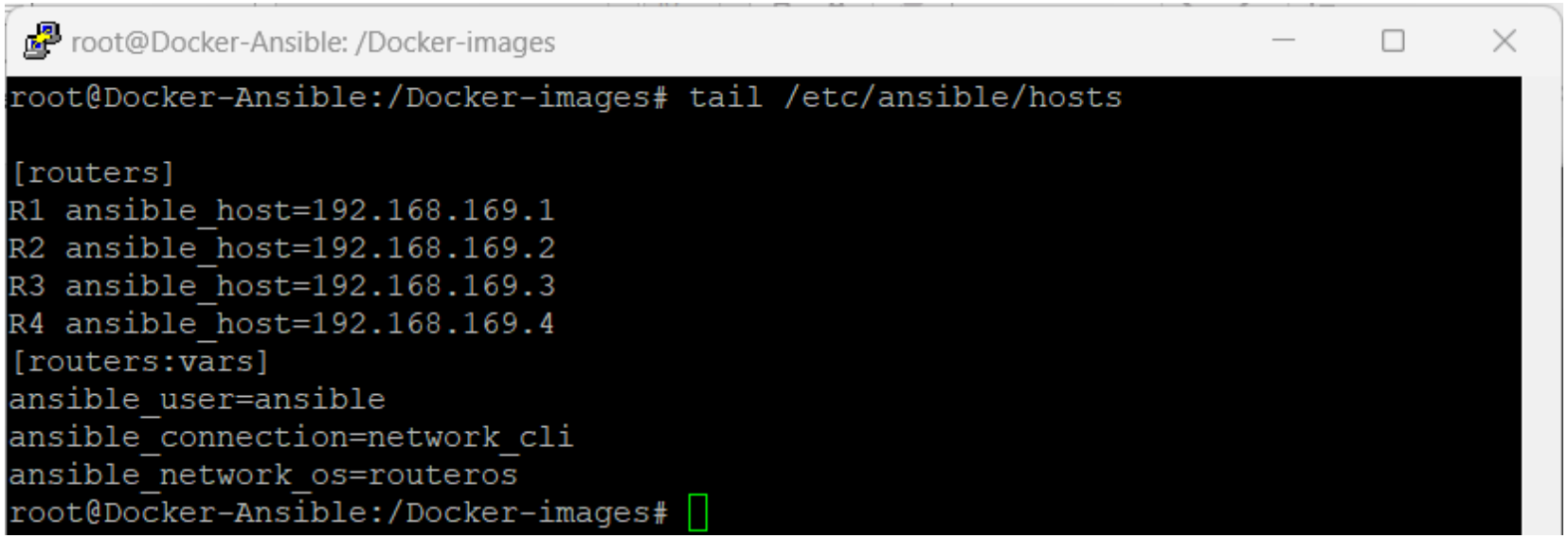
```
[routers]
R1 ansible_host=192.168.169.1
R2 ansible_host=192.168.169.2
R3 ansible_host=192.168.169.3
R4 ansible_host=192.168.169.4
[routers:vars]
ansible_user=ansible
ansible_connection=network_cli
ansible_network_os=routeros
```

- *Variable* ansible_user menentukan user yang digunakan untuk terkoneksi ke *router* yaitu ansible.
- *Variable* ansible_connection digunakan untuk mengatur agar *Ansible* memperlakukan *managed machine* sebagai perangkat jaringan dengan lingkungan eksekusi yang terbatas yaitu network_cli.
- *Variable* ansible_network_os digunakan untuk menginformasikan platform jaringan dari *managed machine* yaitu routeros.

Simpan perubahan dengan menekan **CTRL+O** dan tekan **Enter**. Keluar dari editor *nano* dengan menekan **CTRL+X**.

ANSIBLE AD-HOC MODULE RAW (6)

- Memverifikasi hasil penambahan *group variables* pada *file inventory*.
  `# tail /etc/ansible/hosts`



ANSIBLE AD-HOC MODULE PING

- Memverifikasi koneksi ke seluruh **managed machines** untuk *group* "**routers**" menggunakan modul **ping**.

root@Docker-Ansible: /Docker-images

```
root@Docker-Ansible:/Docker-images# ansible routers -m ping
R3 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
R2 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
R4 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
R1 | SUCCESS => {
    "ansible_facts": {
        "discovered_interpreter_python": "/usr/bin/python3"
    },
    "changed": false,
    "ping": "pong"
}
root@Docker-Ansible:/Docker-images#
```

ANSIBLE PLAYBOOK (1)

- Membuat file *playbook* menggunakan editor **nano** dengan ekstensi "**yml**" untuk menghapus *username* admin di seluruh *router*.

```
# nano remove-admin.yml
```

```
1  ---
2  - name: Manajemen User RouterOS
3    hosts: routers
4    tasks:
5    - name: Menghapus user admin
6      routeros_command:
7        commands: /user remove admin
```

Playbook | Play | Tasks

- Playbooks memuat Plays
  - Plays memuat Tasks
    - Tasks memanggil **Modules**

Simpan perubahan dengan menekan **CTRL+O** dan **Enter**.

Tekan **CTRL+X** untuk keluar dari editor *nano*.

ANSIBLE PLAYBOOK (2)

- Mengecek sintak dari *playbook* sebelum dieksekusi sehingga memastikan tidak terdapat permasalahan terkait sintak dengan mengeksekusi perintah **ansible-playbook** dengan flag **--syntax-check**.

```
# ansible-playbook remove-admin.yml --syntax-check
```

```
root@Docker-Ansible: /Docker-images
root@Docker-Ansible:/Docker-images# ansible-playbook remove-admin.yml --syntax-c
heck

playbook: remove-admin.yml
root@Docker-Ansible:/Docker-images#
```

- Menampilkan informasi *host* yang terdampak oleh *playbook* sebelum dieksekusi dapat menggunakan perintah:

```
root@Docker-Ansible: /Docker-images
root@Docker-Ansible:/Docker-images# ansible-playbook remove-admin.yml --list-hos
ts

playbook: remove-admin.yml

  play #1 (routers): Manajemen User RouterOS    TAGS: []
    pattern: ['routers']
    hosts (4):
      R1
      R3
      R2
      R4
root@Docker-Ansible:/Docker-images#
```

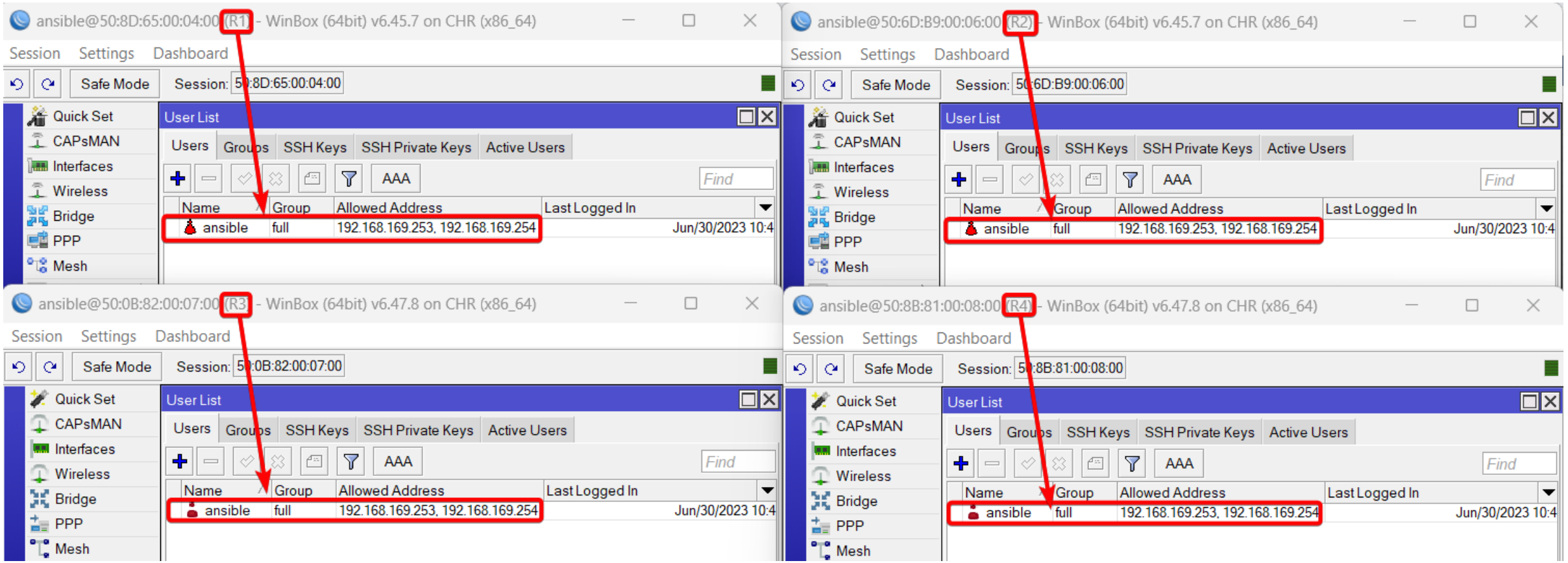
ANSIBLE PLAYBOOK (3)

- Mengeksekusi *file playbook* **remove-admin.yml** menggunakan **ansible-playbook.**

```
root@Docker-Ansible: /Docker-images
root@Docker-Ansible:/Docker-images# ansible-playbook remove-admin.yml

PLAY [Manajemen User RouterOS] ************************************************

TASK [Gathering Facts] ********************************************************
ok: [R4]
ok: [R3]
ok: [R2]
ok: [R1]

TASK [Menghapus user admin] ***************************************************
ok: [R2]
ok: [R1]
ok: [R4]
ok: [R3]

PLAY RECAP ********************************************************************
R1                         : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
R2                         : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
R3                         : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
R4                         : ok=2    changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

root@Docker-Ansible:/Docker-images#
```

ANSIBLE PLAYBOOK (4)

- Hasil verifikasi melalui Winbox dengan otentikasi **login** sebagai **user ansible** dan **password ansiblesecret** menunjukkan bahwa *user* admin telah berhasil dihapus pada setiap *router*.



STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (1)

Kebijakan Keamanan *Mikrotik RouterOS* yang diotomatisasi menggunakan *Ansible Playbook* pada setiap *Router* adalah sebagai berikut:

1. Menonaktifkan **RouterOS MAC-Access.**
2. Menonaktifkan **Neighbor Discovery.**
3. Menonaktifkan **BTest Server.**
4. Menonaktifkan **DNS Cache.**
5. Menonaktifkan **Client Services.**
6. Mengaktifkan **Strong Crypto SSH.**

STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (2)

7. Menonaktifkan **Interface Router** untuk ether4 dan ether5.
8. Mengaktifkan **RouterOS Auto-Upgrade** menggunakan *system scheduler* agar terjadwal untuk beroperasi setiap hari pada jam 01:00:00.
9. Menonaktifkan **IP Service** api, api-ssl, ftp, telnet, www dan www-ssl.
10. Membatasi akses pada **IP Service** Winbox dan SSH agar hanya dapat diakses dari 192.168.169.253 dan 192.168.169.254.

STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (3)

- Membuat file *playbook* menggunakan editor **nano** dengan ekstensi "**yml**" untuk mengotomatisasi pengamanan RouterOS berdasarkan kebijakan yang telah ditentukan pada seluruh *router*. *File* ini juga dapat diunduh dari GitHub Repository.

```
# nano security.yml
```

```yaml
---
- name: Mikrotik RouterOS Security Automation
  hosts: routers
  vars:
    available_from: "192.168.169.253,192.168.169.254"
    interfaces:
    - ether4
    - ether5
  tasks:
  - name: Menonaktifkan RouterOS Mac-Access (MAC-Telnet, MAC-Winbox dan MAC-Ping)
    routeros_command:
        commands:
        - /tool mac-server set allowed-interface-list=none
        - /tool mac-server mac-winbox set allowed-interface-list=none
        - /tool mac-server ping set enabled=no
```

STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (4)

```
17  - name: Menonaktifkan Neighbor Discovery pada seluruh interfaces
18    routeros_command:
19      commands: /ip neighbor discovery-settings set discover-interface-list=none
20
21  - name: Menonaktifkan Bandwidth Test (Btest) Server
22    routeros_command:
23      commands: /tool bandwidth-server set enabled=no
24
25  - name: Menonaktifkan DNS Cache
26    routeros_command:
27      commands: /ip dns set allow-remote-requests=no
28
29  - name: Menonaktifkan Client Services
30    routeros_command:
31      commands:
32      - /ip proxy set enabled=no
33      - /ip socks set enabled=no
34      - /ip cloud set ddns-enabled=no update-time=no
35
36  - name: Mengaktifkan Strong Crypto SSH
37    routeros_command:
38      commands: /ip ssh set strong-crypto=yes
```

STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (5)

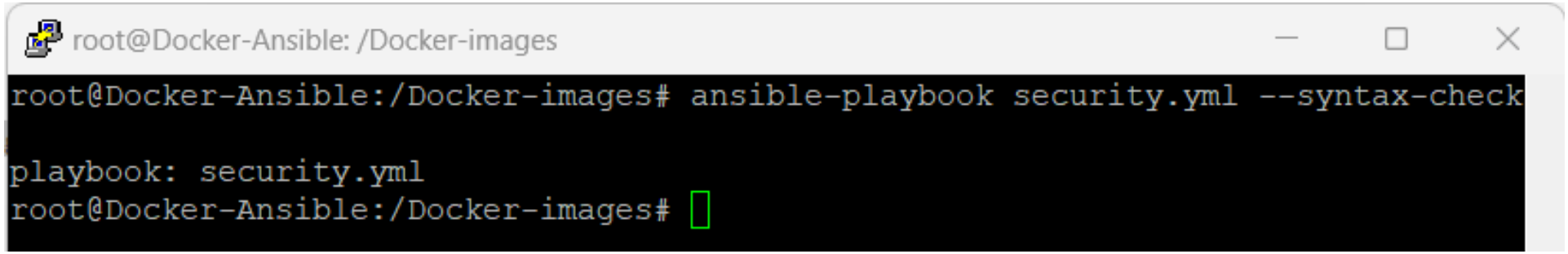
```
39
40  - name: Menonaktifkan Interface dari Router
41    routeros_command:
42      commands: /interface set [ find name={{ item }} ] disabled=yes
43    with_items: "{{ interfaces }}"
44
45  - name: Menambahkan Schedule RouterOS Auto-Upgrade
46    routeros_command:
47      commands: /system scheduler add name=UPGRADE_ROUTEROS start-date=Aug/01/2020 on-event="/system package update check-for-updates once;:delay 3s;:if
      ([/system package update get status] = \"New version is available\") do={ /system package update install }" interval=1d start-time=01:00:00
48
49  - name: Menonaktifkan IP Service
50    routeros_command:
51      commands: /ip service disable {{ item }}
52    with_items:
53    - api
54    - api-ssl
55    - ftp
56    - telnet
57    - www
58    - www-ssl
59
60  - name: Mengatur available from untuk service SSH dan Winbox
61    routeros_command:
62      commands:
63      - /ip service set [ find name={{ item }} ] address="{{ available_from }}"
64    with_items:
65    - winbox
66    - ssh
```

Simpan perubahan dengan menekan **CTRL+O** dan **Enter**. Tekan **CTRL+X** untuk keluar dari editor *nano*.

STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (6)

- Mengecek sintak dari *playbook* sebelum dieksekusi sehingga memastikan tidak terdapat permasalahan terkait sintak dengan mengeksekusi perintah **ansible-playbook** dengan flag **--syntax-check**.

```
# ansible-playbook security.yml --syntax-check
```



STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (7)

- Mengeksekusi *file playbook* **security.yml** menggunakan **ansible-playbook.**

```
root@Docker-Ansible: /Docker-images
root@Docker-Ansible:/Docker-images# ansible-playbook security.yml

PLAY [Mikrotik RouterOS Security Automation] ************************************

TASK [Gathering Facts] **********************************************************
ok: [R1]
ok: [R4]
ok: [R3]
ok: [R2]

TASK [Menonaktifkan RouterOS Mac-Access (MAC-Telnet, MAC-Winbox dan MAC-Ping)] *
**
ok: [R4]
ok: [R1]
ok: [R3]
ok: [R2]

TASK [Menonaktifkan Neighbor Discovery pada seluruh interfaces] ****************
ok: [R3]
ok: [R1]
ok: [R2]
ok: [R4]
```

STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (8)

root@Docker-Ansible: /Docker-images

```
TASK [Menonaktifkan Bandwidth Test (Btest) Server] ********************************
ok: [R1]
ok: [R4]
ok: [R2]
ok: [R3]

TASK [Menonaktifkan DNS Cache] ****************************************************
ok: [R1]
ok: [R2]
ok: [R3]
ok: [R4]

TASK [Menonaktifkan Client Services] **********************************************
ok: [R1]
ok: [R4]
ok: [R2]
ok: [R3]

TASK [Mengaktifkan Strong Crypto SSH] *********************************************
ok: [R1]
ok: [R4]
ok: [R2]
ok: [R3]
```

STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (9)

```
root@Docker-Ansible: /Docker-images
TASK [Menonaktifkan Interface dari Router] ****************************************
ok: [R3] => (item=ether4)
ok: [R4] => (item=ether4)
ok: [R1] => (item=ether4)
ok: [R2] => (item=ether4)
ok: [R2] => (item=ether5)
ok: [R4] => (item=ether5)
ok: [R1] => (item=ether5)
ok: [R3] => (item=ether5)

TASK [Menambahkan Schedule RouterOS Auto-Upgrade] *********************************
ok: [R1]
ok: [R2]
ok: [R4]
ok: [R3]

TASK [Menonaktifkan IP Service] ***************************************************
ok: [R2] => (item=api)
ok: [R3] => (item=api)
ok: [R1] => (item=api)
ok: [R4] => (item=api)
ok: [R2] => (item=api-ssl)
ok: [R3] => (item=api-ssl)
ok: [R4] => (item=api-ssl)
ok: [R1] => (item=api-ssl)
ok: [R2] => (item=ftp)
ok: [R3] => (item=ftp)
ok: [R4] => (item=ftp)
ok: [R1] => (item=ftp)
ok: [R2] => (item=telnet)
```

STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (10)

```
root@Docker-Ansible: /Docker-images
ok: [R4] => (item=telnet)
ok: [R3] => (item=telnet)
ok: [R1] => (item=telnet)
ok: [R2] => (item=www)
ok: [R4] => (item=www)
ok: [R3] => (item=www)
ok: [R1] => (item=www)
ok: [R2] => (item=www-ssl)
ok: [R4] => (item=www-ssl)
ok: [R3] => (item=www-ssl)
ok: [R1] => (item=www-ssl)

TASK [Mengatur available from untuk service SSH dan Winbox] **********************
ok: [R3] => (item=winbox)
ok: [R4] => (item=winbox)
ok: [R1] => (item=winbox)
ok: [R2] => (item=winbox)
ok: [R4] => (item=ssh)
ok: [R3] => (item=ssh)
ok: [R1] => (item=ssh)
ok: [R2] => (item=ssh)
```

STUDI KASUS: ANSIBLE PLAYBOOK UNTUK MENGOTOMATISASI KEAMANAN ROUTEROS (11)

root@Docker-Ansible: /Docker-images

```
PLAY RECAP *********************************************************************
R1                         : ok=11   changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
R2                         : ok=11   changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
R3                         : ok=11   changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0
R4                         : ok=11   changed=0    unreachable=0    failed=0    s
kipped=0    rescued=0    ignored=0

root@Docker-Ansible:/Docker-images#
```

DEMO VERIFIKASI HASIL OTOMATISASI MIKROTIK ROUTEROS SECURITY PADA SELURUH ROUTER

REFERENSI

- MikroTik, Manual: Security Your Router, 2019, https://wiki.mikrotik.com/wiki/Manual:Securing_Your_Router
- MikroTik, Manual: Upgrading RouterOS, 2021, https://wiki.mikrotik.com/wiki/Manual:Upgrading_RouterOS
- MikroTik, Manual:IP/DNS, 2022, https://wiki.mikrotik.com/wiki/Manual:IP/DNS
- Ansible Documentation, https://docs.ansible.com/
- Edureka, What is Ansible? – Configuration Management And Automation With Ansible, 2021, https://www.edureka.co/blog/what-is-ansible/
- I Putu Hariyadi, Modul One Day Workshop "Proxmox Automation With Ansible", Universitas Bumigora, 2019, https://iputuhariyadi.net/2019/11/04/modul-one-day-workshop-proxmox-automation-with-ansible/

REFERENSI

- I Putu Hariyadi, GitHub Repository RouterOS Security Automation Playbook, 2020, https://github.com/iputuhariyadi/routeros-security-automation-playbook

ADA PERTANYAAN?

TERIMAKASIH