Dokumen Laporan Akhir Final Project TSA-NP

“Automation Router Settings(Password, Banner motd, dan new interface with Ip) dan Broadcasting Webex Chat via Jenkins”

- Latar Belakang

  Diberikan permasalahan berikut :

  Terdapat 2 user yaitu A dan B. A adalah seorang programmer dan B adalah seorang user internet. Pada suatu hari B meminta bantuan A untuk membantunya membroadcastkan pesan kepada user WEBEX dengan daftar email yang diberikan. Keesokan harinya B meminta A lagi untuk membuatkan program yang dapat membuat interface baru(loopback) pada routernya dengan cepat, dan dapat mengganti password CLI router serta memberikan Warning Message di CLI routernya untuk memperingat user lain yang akan mengaksesnya.

- Rancangan Umum Project

  Karena A adalah seorang programmer, maka dia dapat langsung membuat kedua program tersebut. Kedua program ini akan diupload di GitHub yang dapet diakses oleh B. Untuk program tentang webexnya, B akan membuildnya di Jenkins karena dapat dibuild secara banyak langsung. Sedangkan untuk program config router maka akan dipull oleh B dan dijalankan langsung programnya.

- Penjelasan Project

  Part 1 : Automation Router Settings (Password, Banner MOTD, dan New Interface)

  Step 1 : Instal Virtual Machine dan CSR1kv VM serta pastikan keduanya terhubung menggunakan SSH.

  a. Setelah keduanya sudah terinstal, pastikan CSR1kv mendapatkan identitas Networknya atau IP. Bukak CSR1kv dan ketikkan **show ip interace brief** untuk mendapatkan IPv4 addressnya. Di sini yang digunakan adalah 192.168.56.101 , jika berbeda silahkan dicatat untuk digunakan nanti.
  b. Ping pada terminal dengan target IP dari CSR1kv yaitu **ping -c 5 102.168.56.101** yang akan mengirim paket sebanyak 5 kali ke tujuan IP. Pastikan tidak RTO(Request Time Out).
  c. Pada terminal di DEVASC VM, masukkan perintah berikut

```
devasc@labvm:~$ ssh  cisco@192.168.56.101
The authenticity of host '192.168.56.101 (192.168.56.101)' can't be
established.
RSA key fingerprint is SHA256:HYv9K5Biw7PFiXeoCDO/LTqs3EfZKBuJdiPo34VXDUY.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.56.101' (RSA) to the list of known hosts.
```

Apabila terdapat kendala seperti yang di bawah ini :

```
devasc@labvm:~$ ssh cisco@192.168.56.101
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
@    WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!     @
@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@@
IT IS POSSIBLE THAT SOMEONE IS DOING SOMETHING NASTY!
Someone could be eavesdropping on you right now (man-in-the-middle attack)!
It is also possible that a host key has just been changed.
The fingerprint for the RSA key sent by the remote host is
SHA256:ZimKx8r22FnWLVDXn79sGdJJhONU7xMNAlg2llbkT6I.
Please contact your system administrator.
Add correct host key in /home/devasc/.ssh/known_hosts to get rid of this message
.
Offending RSA key in /home/devasc/.ssh/known_hosts:2
  remove with:
  ssh-keygen -f "/home/devasc/.ssh/known_hosts" -R "192.168.56.101"
RSA host key for 192.168.56.101 has changed and you have requested strict checki
ng.
Host key verification failed.
```

Maka masukkan perintah **ssh-keygen -R "you server hostname or ip"** di Terminal. Lalu cobalah untuk mengkonek ulang secara ssh.

d. Masukkan password **cisco123!** dan akan otomatis masuk ke dalam privileged EXEC command pada CSR1kv.

Step 2 : Check NETCONF berjalan di CSR1kv dan akses NETCONF menggunakan SSH Terminal

a. Dari ssh yang sudah terhubung ke router masukkan perintah **show platform software yang-management process** dan pastikan bahwa **ncsshd** sudah running.

```
CSR1kv#show platform software yang-management process
confd              : Running
nesd               : Running
syncfd             : Running
ncsshd             : Running
dmiauthd           : Running
nginx              : Running
ndbmand            : Running
pubd               : Running
```

Jika tidak ncsshd tidak running seperti di atas, masuklah ke global config dan masukkan perintah **netconf-yang**.

```
CSR1kv# config t
CSR1kv (config)# netconf-yang
```

b. Masukkan perintah pada terminal seperti di bawah ini dengan password yang sama dengan ssh yaitu **cisco123!** .

```
devasc@labvm:~$ ssh cisco@192.168.56.101 -p 830 -s netconf
cisco@192.168.56.101's password:
```

Note : -p 830 adalah command unuk membuka port 830.

Maka akan muncul pesan client hello message yang diakhiri dengan]]>]]>.

c. Pindah ke CSR1kv VM dan masukkan **show netconf-yang sessions** untuk memverifikasi bahwa NETCONF session sudah running.

```
CSR1kv#show netconf-yang sessions
R: Global-lock on running datastore
C: Global-lock on candidate datastore
S: Global-lock on startup datastore

Number of sessions : 1

session-id  transport    username             source-host          global-lock
--------------------------------------------------------------------------------
22          netconf-ssh  cisco                192.168.56.1         None

CSR1kv#
```

d. Tekan **ctrl+C** pada terminal untuk menutup NETCONF session.

Step 3 : Gunakan ncclient untuk connect ke NETCONF

a. Pastikan bahwa ncclient sudah terinstal dan siap digunakan.

```
devasc@labvm:~$ pip3 list --format=columns | more
Package                Version
---------------------- -----------
ansible                2.9.6
apache-libcloud        2.8.0
appdirs                1.4.3
argcomplete            1.8.1
astroid                2.3.3
(output omitted)
ncclient               0.6.7
netaddr                0.7.19
netifaces              0.10.4
netmiko                3.1.0
ntlm-auth              1.1.0
oauthlib               3.1.0
(output omitted)
xmltodict              0.12.0
zipp                   1.0.0
devasc@labvm:~$
```

Step 4 : Buat script untuk ncclient connect ke NETCONF.

a. Bukak VS code. tentukan folder yang akan digunakan dan buat file **ncclient-netconf.py**

b. Pada script import class manager dar modul ncclient. Buat variable **m** untuk menjadi **connect()** method. Dalam method ini terdapat informasi yang dibutuhkan untuk connect ke NETCONF yang berjalan pada CSR1kv. Catat bahwa port yang digunakan adalah 830 unuk NETCONF.

ncclient-netconf.py

```
1   from ncclient import manager
2   import xml.dom.minidom
3
4   m = manager.connect(
5       host="192.168.56.101",
6       port=830,
7       username="cisco",
8       password="cisco123!",
9       hostkey_verify=False
10      )
```

Jika **hostkey_verify** disetting menjadi True, maka CSR1kv akan memintamu untuk memverifikasi SSH Fingerprint. Save dan run program.

c. Untuk memverifikasi bahwa CSR1kv menerima request NETCONFnya maka akan ada syslog pesan **%DMI-5-AUTH_PASSED** di CSR1kv VM.

Step 5 : konfigurasi ncclient untuk mendapatkan config

a. Tambahkan script dengan command m.get_config untuk mengambil config

```
netconf_reply = m.get_config(source="running")
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

Xml.dom.minidom adalah module untuk merapikan xml.

b. Running program untuk mendapatkan config router.
c. Copy output dan save untuk nanti membandingkan setelah hasil konfigurasi.

Step 6 : konfigurasi ncclient untuk settiing password router

a. Membuat fungsi untuk mengganti password router dengan nama **passw_router()**. Di dalammya terdapat syntax m.edit_config karena kita akan mengubah isi dari config dengan config yang sudah dibuat di dalam petik 3.

```
def passw_router():
    a=input("Masukkan password untuk router ! : ")
    netconf_passw_router= """
    <config>
        <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
            <enable>
                <password>
                    <secret> {} </secret>
                </password>
            </enable>
        </native>
    </config>
    """.format(a)
    netconf_reply = m.edit_config(target="running",config=netconf_passw_router)
    print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

Memberikan variable a untuk mengambil input dari user untuk menjadi password yang baru.

b. Call fungsi passw_router di atas akan menampilkan netconfnya sehingga akan muncul 2 output, yaitu output **ok** yang berarti password berhasil ditambahkan atau diubah dan output config terbaru yang apabila discroll ke arah format filternya maka akan tampil password barunya.

```
Masukkan password untuk router ! : gabriel
<?xml version="1.0" ?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc=
"urn:ietf:params:xml:ns:netconf:base:1.0" message-id="urn:uuid:9da52
9a0-cf35-4de8-898e-7f0d4c4efdfc">
        <ok/>
</rpc-reply>
```

```
<hostname>CSR1kv</hostname>
<enable>
        <password>
                <secret> gabriel </secret>
        </password>
</enable>
<username>
        <name>cisco</name>
        <privilege>15</privilege>
        <password>
                <encryption>0</encryption>
                <password>cisco123</password>
        </password>
</username>
```

Step 7 : konfigurasi ncclient untuk setting banner motd

a. Membuat fungsi **banner_motd()** untuk mengganti MOTD pada route. Di dalammya terdapat syntax m.edit_config karena kita akan mengubah isi dari config dengan config yang sudah dibuat di dalam petik 3.

```
def banner_motd():
    a=input("Masukkan pesan peringatan untuk user lain ! : ")
    netconf_banner = """
    <config>
        <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
            <banner>
                <motd>
                    <banner>{}</banner>
                </motd>
            </banner>
        </native>
    </config>
    """.format(a)
    netconf_reply = m.edit_config(target="running",config=netconf_banner)
    print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

b. Call fungsi banner_motd di atas akan menampilkan netconfnya sehingga akan muncul 2 output, yaitu output **ok** yang berarti password berhasil ditambahkan atau diubah dan output config terbaru yang apabila discroll ke arah format filternya maka akan tampil password barunya.

```
Masukkan pesan peringatan untuk user lain ! : Authorized Access Only!!!
<?xml version="1.0" ?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:para
ms:xml:ns:netconf:base:1.0" message-id="urn:uuid:ad5993e5-36db-4991-8da1-3ab176c9c
050">
        <ok/>
</rpc-reply>
```

```
<data>
       <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
              <version>16.9</version>
              <boot-start-marker/>
              <boot-end-marker/>
              <banner>
                     <motd>
                            <banner>Authorized Access Only!!!</banner>
                     </motd>
              </banner>
              <service>
                     <timestamps>
                            <debug>
                                   <datetime>
                                          <msec/>
                                   </datetime>
                            </debug>
                            <log>
                                   <datetime>
                                          <msec/>
                                   </datetime>
                            </log>
                     </timestamps>
              </service>
              <platform>
                     <console xmlns="http://cisco.com/ns/yang/Cisco-IOS
```

Step 8 : konfigurasi ncclient untuk menambahkan interface

a. Membuat fungsi **set_loopback()** untuk menambahkan interface loopback pada CSR1kv. Di dalammya terdapat syntax m.edit_config karena kita akan mengubah isi dari config dengan config yang sudah dibuat di dalam petik 3.

```
def set_loopback():
    a=int(input("Masukkan nomer Loopback interface yang ingin ditambahkan! : "))
    b=input("Masukkan deskripsi untuk loopback : ")
    c=input("Masukkan ip address! : ")
    d=input("Masukkan netmasknya! : ")
    netconf_interface ="""
    <config>
        <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
            <interface>
                <Loopback>
                    <name>{}</name>
                    <description>{}</description>
                    <ip>
                        <address>
                            <primary>
                                <address>{}</address>
                                <mask>{}</mask>
                            </primary>
                        </address>
                    </ip>
                </Loopback>
            </interface>
        </native>
    </config>
    """.format(a,b,c,d)
    netconf_reply = m.edit_config(target="running",config=netconf_interface)
    print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

b. Call fungsi **set_loopback()** di atas akan menampilkan netconfnya sehingga akan muncul 2 output, yaitu output **ok** yang berarti password berhasil ditambahkan atau diubah dan output config terbaru yang apabila discroll ke arah format filternya maka akan tampil password barunya.

```
Masukkan nomer Loopback interface yang ingin ditambahkan! : 1
Masukkan deskripsi untuk loopback : Loopback ke-1 Gabriel!
Masukkan ip address! : 10.1.1.1
Masukkan netmasknya! : 255.255.255.0
<?xml version="1.0" ?>
<rpc-reply xmlns="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns:nc="urn:ietf:para
ms:xml:ns:netconf:base:1.0" message-id="urn:uuid:0663d12a-6d5d-440e-ab25-e194ae833
8e2">
        <ok/>
</rpc-reply>
```

```
  <config>
          <name>Loopback1</name>
          <type xmlns:ianaift="urn:ietf:params:xml:ns:yang:iana-if-type">ianaift:so

          <description>Loopback ke-1 Gabriel!</description>
          <enabled>true</enabled>
  </config>
  <subinterfaces>
          <subinterface>
                  <index>0</index>
                  <config>
                          <index>0</index>
                          <description>Loopback ke-1 Gabriel!</description>
                          <enabled>true</enabled>
                  </config>
                  <ipv4 xmlns="http://openconfig.net/yang/interfaces/ip">
                          <addresses>
                                  <address>
                                          <ip>10.1.1.1</ip>
                                          <config>
                                                  <ip>10.1.1.1</ip>
                                                  <prefix-length>24</prefix-length>
                                          </config>
                                  </address>
                          </addresses>
                  </ipv4>
                  <ipv6 xmlns="http://openconfig.net/yang/interfaces/ip">
                          <config>
                                  <enabled>false</enabled>
                          </config>
                  </ipv6>
```

Step 9 : Kumpulkan dan Automation!

a. Buat kerangka menu untuk menjadi menu program. Syntax seperti berikut :

ncclient-netconf.py

```
while True:
    time.sleep(2)
    print("""
##################################
Menu konfigurasi :               #
0. Exit!                         #
1. Router Password               #
2. Message of the Day (BANNER MOTD) #
3. Loopback                      #
##################################
""")
    try:
        menu=int(input("PILIH PROGRAM YANG ANDA INGINKAN ! : "))
    except:
        print("Error! Wrong Input")
        continue
    if menu==0:
        break
    elif menu==1:
        passw_router()
        continue
    elif menu==2:
        banner_motd()
        continue
    elif menu==3:
        set_loopback()
        continue
```

Import module time untuk membuat output muncul dengan jeda 2 detik. Lalu untuk syntax sebelum dan sesudah gambar di atas, masukkan syntax untuk mendapatkan config (get config) seperti pada step 5.

b. Final syntax seperti berikut :

```
from ncclient import manager
import xml.dom.minidom
import time

m = manager.connect(
   host="192.168.56.101",
   port=830,
   username="cisco",
   password="cisco123!",
   hostkey_verify=False
   )

#Kumpulan filter
def passw_router():
   a=input("Masukkan password untuk router ! : ")
   netconf_passw_router= """
   <config>
      <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
         <enable>
            <password>
               <secret> {} </secret>
            </password>
         </enable>
      </native>
   </config>
```

```
    """.format(a)
    netconf_reply                                                    =
m.edit_config(target="running",config=netconf_passw_router)
    print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())

def banner_motd():
    a=input("Masukkan pesan peringatan untuk user lain ! : ")
    netconf_banner = """
    <config>
      <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
        <banner>
          <motd>
            <banner>{}</banner>
          </motd>
        </banner>
      </native>
    </config>
    """.format(a)
    netconf_reply = m.edit_config(target="running",config=netconf_banner)
    print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())

def set_loopback():
    a=int(input("Masukkan nomer Loopback interface yang ingin
ditambahkan! : "))
    b=input("Masukkan deskripsi untuk loopback : ")
    c=input("Masukkan ip address! : ")
    d=input("Masukkan netmasknya! : ")
    netconf_interface ="""
    <config>
      <native xmlns="http://cisco.com/ns/yang/Cisco-IOS-XE-native">
        <interface>
          <Loopback>
            <name>{}</name>
            <description>{}</description>
            <ip>
              <address>
                <primary>
                  <address>{}</address>
                  <mask>{}</mask>
                </primary>
              </address>
            </ip>
          </Loopback>
        </interface>
      </native>
    </config>
```

```
    """.format(a,b,c,d)
    netconf_reply                                                    =
m.edit_config(target="running",config=netconf_interface)
    print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())

netconf_reply = m.get_config(source="running")
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())

while True:
    time.sleep(2)
    print("""
#######################################
Menu konfigurasi :              #
0. Exit!                    #
1. Router Password              #
2. Message of the Day (BANNER MOTD) #
3. Loopback                 #
#######################################
""")
    try:
        menu=int(input("PILIH PROGRAM YANG ANDA INGINKAN ! :
"))
    except:
        print("Error! Wrong Input")
        continue
    if menu==0:
        break
    elif menu==1:
        passw_router()
        continue
    elif menu==2:
        banner_motd()
        continue
    elif menu==3:
        set_loopback()
        continue

netconf_reply = m.get_config(source="running")
print(xml.dom.minidom.parseString(netconf_reply.xml).toprettyxml())
```

Step 10 : Buat Git Repository yang terintegrasi dengan GitHub.

a. Buatlah directory untuk dijadikan Git Repository dan Initializing Git. Gunakan command **git init** dalam terminal dengan directory yang sudah dibuat. Contoh **/home/devasc/miniproject/router.**

```
devasc@labvm:~/miniproject/router$ pwd
/home/devasc/miniproject/router
devasc@labvm:~/miniproject/router$ git init
```

b. Buat GitHub account dan Buat Repository
   Setelah login account github maka akan ada **"New repository"**. Buat repository dengan **contoh** informasi,
   Repository name : **MiniProjectDevnet**
   Description : "Ini adalah mini project saya ..."
   Public / Private : **Public**
   Lalu pilih **Create Repository** dan salin link repositorynya.
c. Buat username git dengan emailnya.

```
devasc@labvm:~$ git config --global user.name "GitHub username"
devasc@labvm:~$ git config --global user.email GitHub-email-address
```

   Note : Username sensitive terhadap kapital!
d. Di directory folder yang sudah diinitialize Git masukkan command **git remote add origin *"link repository yang di GitHub"***. Command ini untuk menambahkan repository yang ada di GitHub akan menjadi import dan export dari git local (computer).
e. Masukkan command **git add *** untuk menambahkan semua file yang ada di directory ke dalam git. Lalu masukkan command **git commit -m "Program untuk mengkonfigurasi router"** untuk mengkomit dengan pesan atau deskripsi.
f. Gunakan command **git push origin master** untuk mengekspor file yang diadd git dan sudah dicommit. Lalu masukkan username akun github dengan passwordnya adalah token developer akun.
g. Cek GitHub account tadi dan di bawah tab "Repositories" pilih **username/MiniProjectDevnet** . Maka akan ada file yang tadi ditambahkan.
h. Untuk mengambil file dari repository tersebut ke dalam local (computer) maka gunakan command **git pull origin master** pada point 'f'.

Part 2 : Broadcasting Webex Chat via Jenkins

Step 1 : Membuat daftar email user webex yang akan dikirimkan Chat

a. Dari directory di part sebelumnya, yaitu directory mini project akan dibuatkan document dengan nama **email.txt**.
b. Edit file terseebut dengan menambahkan email per baris. Email yang digunakan adalah email yang terdaftar sebagai akun Webex.
c. Save email tersebut.
d. Bukak terminal dengan directory mini project dan tambahkan command **chmod a+x email.txt** supaya file tersebut dapat diakses dan execute dari semua user.

Step 2 : Mendapatkan Webex Team Access Tokens dan api untuk messages

a. Buka browser yang biasa Anda gunakan dan akses website Webex Developer yaitu https://developer.webex.com/
b. Login jika sudah memiliki akun, sign up apabila belum.
c. Setelah login, maka akan terlihat tab **Documentation**. Buka tab tersebut.
d. Scroll kebawah sampai menemukan tab **Getting Started**. Di sana terdapat PAT (Personal Access Token) dengan kode depan Bearer. Catat kode tersebut.

> **Note : PAT hanya berlaku setiap 12jam setelah login dan copy! Setelah itu kodenya akan ganti yang baru.**

e. Setelah itu scroll ke bawah sampai menemukan tab **Full API Reference**. Expand tab tersebut dan scroll lagi sampai mendapatkan tab **Messages** dan copy URI-nya yaitu https://webexapis.com/v1/messages .

Step 3 : Membuat bash dan script python untuk Broadcast Chat dari daftar email

a. Buka VS Code dan open folder directory **mini project**.
b. Pastikan di folder tersebut terlihat file **email.txt**
c. Buatlah file baru bernama **autochatwebex.py** dan **BroadcastingWebex.sh**
d. Untuk file .sh masukkan syntax berikut

```
$ BroadcastingWebex.sh
1   #!/bin/bash
2   python3 autochatwebex.py
```

e. Sedangkan untuk file .py isikan dengan syntax berikut :

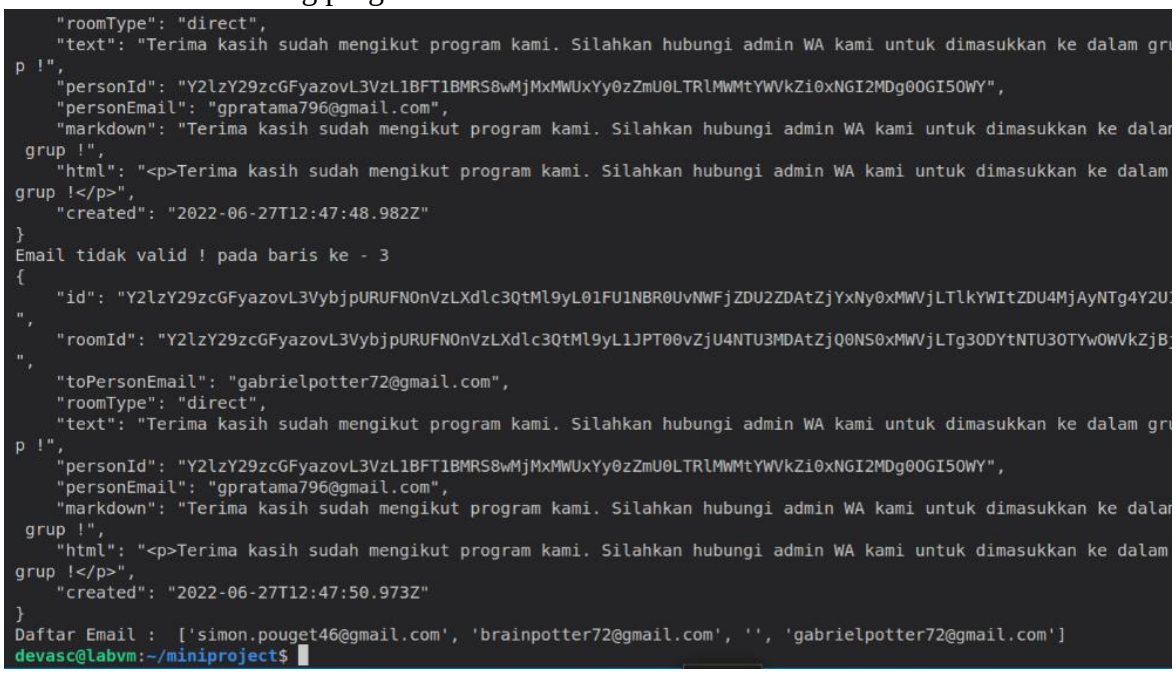
```
autochatwebex.py
1  #!/usr/bin/python3
2  import requests
3  import json
4  messages="Terima kasih sudah mengikut program kami. Silahkan hubungi admin WA kami untuk dimasukkan ke dalam grup !"
5  url = 'https://webexapis.com/v1/messages'
6  file=open("email.txt","r")
7  data=[]
8  for i in file:
9      try:
10         access_token = 'ZTA4OTQzYWItZDMzNS00ZTRkLTkyZDItMDgwNzVjNGIxM2JlMDhmM2UyYWMtNGM5_P0A1_296842f3-00c1-4128-b47d-4e1e1fd7d576'
11         i = i.replace("\n", "")
12         data.append(i)
13         headers={
14         'Authorization': 'Bearer {}'.format(access_token),
15         'Content-Type': 'application/json'
16         }
17         params = {'toPersonEmail': i, 'markdown':messages}
18         res = requests.post(url, headers=headers, json=params)
19         res.raise_for_status()
20         print(json.dumps(res.json(), indent=4))
21     except requests.exceptions.RequestException:
22         print("Email tidak valid ! pada baris ke -",data.index(i)+1)
23 print("Daftar Email : ",data)
```

Keterangan :

a. Line 1 pada code adalah enviroment yang akan dipake oleh Jenkins nanti. Supaya dapat dijalankan oleh Jenkins.
b. Line 2 dan 3 adalah module yang akan dipake, yaitu module requests(RESTAPI) dan json(untuk membuat output berformat .json)
c. Line 4 adalah pesan yang akan disampaikan
d. Line 5 adalah URL untuk API messages di WEBEX
e. Line 6 untuk open dan read file “email.txt”
f. Line 7 untuk membuat list / array email.
g. Line 10 adalah PAT.
h. Line 11 adalah syntax untuk mereplace string \n menjadi empty (delete).
i. Line 12 adalah synax untuk menambahkan email di baris ke-i masuk ke dalam array
j. Line 13 sampai 16 adalah header untuk authorization dan content type adalah format output yang akan dikeluarkan
k. Line 17 , Params adalah variable untuk menjadi parameter api.
l. Line 18 adalah syntax untuk mengakses api dengan module request dan action post. Dalam kurung di line ini adalah informasi yang dibutuhkan untuk melakukan post.
m. Line 19 adalah syntax untuk mendapatkan status dari mengakses API.
n. Except pada line 21 adalah syntax yang akan dijalankan apabila terdapat error yang ditangkap Line 19.
o. Line 23 untuk menampilkan email dalam array!

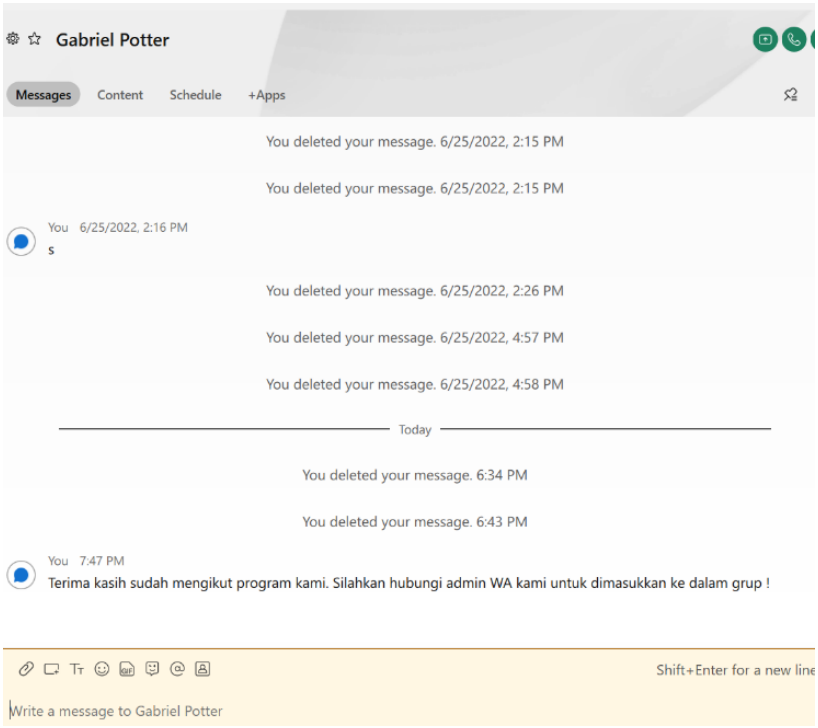
Step 4 : Tes program !

a. Running program



Except menangkap ketika i=2 atau ketika baris ke-3 dari email.txt diread ternyata isi pada baris ke-3 tersebut tidak valid (karena empty, bukan diisi email). Except juga akan dijalankan apabila email yang diberikan tidak terdafar dalam Webex.

b. Cek di Webex apakah pesan sudah terkirim ke berbagai user dengan daftar email tersebut !



Step 5 : Push dan pull program ke atau dari GitHub

a. Gunakan command **git add autochatwebex.py** dan **git commit -m "Update autochatwebex.py"**
b. Gunakan command **git add BroadcastingWebex.sh** dan **git commit -m "Update BroadcastingWebex.sh"**
c. Gunakan command **git add email.txt** dan **git commit -m "Script beserta email untuk dijalankan"**
d. Gunakan command **git push origin master** untuk mengekspor ke-3 file tersebut ke GitHub

Pada user B maka repositoy tersebut akan dipull untuk masuk ke computernya!

e. Gunakan command **git init** untuk folder yang akan menjadi tempat impor dari repository
f. Gunakan command "**git remote add origin *"link repository yang di GitHub"*** pada directory yang sudah dibuat
g. Gunakan command **git pull origin master** untuk mengambil semua file dari repository GitHub

Dari sini maka akan menjadi POV dari user B.

Step 6 : Instal Jenkins via Docker Container

a. Download Jenkins Docker Image

```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker pull
jenkins/jenkins:lts
lts: Pulling from jenkins/jenkins
3192219afd04: Pulling fs layer
17c160265e75: Pulling fs layer
cc4fe40d0e61: Pulling fs layer
9d647f502a07: Pulling fs layer
d108b8c498aa: Pulling fs layer
1bfe918b8aa5: Pull complete
dafa1a7c0751: Pull complete
650a236d0150: Pull complete
cba44e30780e: Pull complete
52e2f7d12a4d: Pull complete
d642af5920ea: Pull complete
e65796f9919e: Pull complete
9138dabbc5cc: Pull complete
f6289c08656c: Pull complete
73d6b450f95c: Pull complete
a8f96fbec6a5: Pull complete
9b49ca1b4e3f: Pull complete
d9c8f6503715: Pull complete
20fe25b7b8af: Pull complete
Digest: sha256:717dcbe5920753187a20ba43058ffd3d87647fa903d98cde64dda4f4c82c5c48
Status: Downloaded newer image for jenkins/jenkins:lts
docker.io/jenkins/jenkins:lts
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$
```

b. Start Jenkins dengan menggunakan docker container dengan command berikut

```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker run --rm -u root -p
8080:8080 -v jenkins-data:/var/jenkins_home -v $(which
docker):/usr/bin/docker -v /var/run/docker.sock:/var/run/docker.sock -v
"$HOME":/home --name jenkins_server jenkins/jenkins:lts
```

c. Masukkan command ini untuk mendapatkan password yang akan digunakan untuk membuka website jenkins

```
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$ docker exec -it
jenkins_server /bin/bash
root@19d2a847a54e:/# cat /var/jenkins_home/secrets/initialAdminPassword
77dc402e31324c1b917f230af7bfebf2
root@19d2a847a54e:/# exit
exit
devasc@labvm:~/labs/devnet-src/jenkins/sample-app$
```

Note: Your container ID (19d2a847a54e highlighted above) and password will be different.

d. Karena pada docker container yang baru belum terinstal PIP, Python3, dan module request (additional python content) maka akan diinstalkan PIP terlebih dahulu. Gunakan command **apt-get update** ketika sudah memasuki root dari container (menggunakan **docker exec -t jenkins_server /bin/bash)** untuk mengupdate semua yang berada di kontainer.

e. Masukkan command **apt-get -y install python3-pip** untuk menginstal pip dan python 3.

f. Masukkan command **pip install requests** untuk menambahkan module requests pada enviroment docker container sehingga ketika nanti jenkins menjalankan .py yang menggunakan module requests maka tidak akan ada kendala.

```
root@99c16990cb62:/bin# pip install requests
Collecting requests
  Downloading requests-2.28.0-py3-none-any.whl (62 kB)
     |████████████████████████████████| 62 kB 364 kB/s
Collecting charset-normalizer~=2.0.0
  Downloading charset_normalizer-2.0.12-py3-none-any.whl (39 kB)
Collecting certifi>=2017.4.17
  Downloading certifi-2022.6.15-py3-none-any.whl (160 kB)
     |████████████████████████████████| 160 kB 1.2 MB/s
Collecting urllib3<1.27,>=1.21.1
  Downloading urllib3-1.26.9-py2.py3-none-any.whl (138 kB)
     |████████████████████████████████| 138 kB 986 kB/s
Collecting idna<4,>=2.5
  Downloading idna-3.3-py3-none-any.whl (61 kB)
     |████████████████████████████████| 61 kB 545 kB/s
Installing collected packages: urllib3, idna, charset-normalizer, certifi, requ
sts
Successfully installed certifi-2022.6.15 charset-normalizer-2.0.12 idna-3.3 req
ests-2.28.0 urllib3-1.26.9
root@99c16990cb62:/bin#
```

Step 7 : Build App di Jenkins

a. Masuk ke Jenkins pada browser menggunakan http://localhost:8080/ . Login menggunakan user admin dengan password yang tadi sudah dicatat (Step 7C).

b. Click **Create a job** dan buat **New item.** Buat job dengan nama AutochatWebex dengan mode “Freestyle Project”

c. Pada tab Description isi dengan **“Autobroadcasting Chat Webex berdasarkan email.txt”**

d. Pada tab **Source Code Management** dari None pilih Git dengan repository urlnya adalah repository **MiniProjectDevnet** yang sudah dibuat sebelumnya.

e. Di bawahnya terdapat tab **Credentials,** pilih add dan configurasikan dengan akun github Anda.

f. Pada tab **Build** pilih execute shell dengan command berikut **bash ./BroadcastingWebex.sh**

g. Pilih Save dan click “Build Now”

Step 8 : Cek output!

a. Buka console output pada Build #1

Dashboard › AutochatWebex › #1


```
    "id": "Y2lzY29zcGFyazovL3VybjpURUFNOnVzLXdlc3QtMl9yL01FU1NBR0UvYmI5M2MxYjAtZjYxZC0xMWVjLThkOTktMzM2ZjMzYzcyOGVh",
    "roomId": "Y2lzY29zcGFyazovL3VybjpURUFNOnVzLXdlc3QtMl9yL1JPT00vZjM5YTRmZTAtZjQ0NS0xMWVjLWEzNWMtMGQ0MzJmOTNlOTRk",
    "toPersonEmail": "simon.pouget46@gmail.com",
    "roomType": "direct",
    "text": "Tes dari git! yang baru",
    "personId": "Y2lzY29zcGFyazovL3VzL1BFT1BMRS8wMjMxMWUxYy0zZmU0LTRlMWMtYWVkZi0xNGI2MDg0OGI5OWY",
    "personEmail": "gpratama796@gmail.com",
    "markdown": "Tes dari git! yang baru",
    "html": "<p>Tes dari git! yang baru</p>",
    "created": "2022-06-27T13:33:30.315Z"
}
{
    "id": "Y2lzY29zcGFyazovL3VybjpURUFNOnVzLXdlc3QtMl9yL01FU1NBR0UvYmM3NzFjZDAtZjYxZC0xMWVjLWI1YzAtMDU0MGEwZTI1NzRj",
    "roomId": "Y2lzY29zcGFyazovL3VybjpURUFNOnVzLXdlc3QtMl9yL1JPT00vZDQzOWM3YzAtZjQ0NS0xMWVjLTlmZmYtNTkwMWZlYWNhOGY0",
    "toPersonEmail": "brainpotter72@gmail.com",
    "roomType": "direct",
    "text": "Tes dari git! yang baru",
    "personId": "Y2lzY29zcGFyazovL3VzL1BFT1BMRS8wMjMxMWUxYy0zZmU0LTRlMWMtYWVkZi0xNGI2MDg0OGI5OWY",
    "personEmail": "gpratama796@gmail.com",
    "markdown": "Tes dari git! yang baru",
    "html": "<p>Tes dari git! yang baru</p>",
    "created": "2022-06-27T13:33:31.805Z"
}
Email tidak valid ! pada baris ke - 3
{
    "id": "Y2lzY29zcGFyazovL3VybjpURUFNOnVzLXdlc3QtMl9yL01FU1NBR0UvYmQ4OWM1NTAtZjYxZC0xMWVjLTg5NDEtN2I2NGViM2VlMTk5",
    "roomId": "Y2lzY29zcGFyazovL3VybjpURUFNOnVzLXdlc3QtMl9yL1JPT00vZjU4NTU3MDAtZjQ0NS0xMWVjLTg3ODYtNTU3OTYwOWVkZjBj",
    "toPersonEmail": "gabrielpotter72@gmail.com",
    "roomType": "direct",
    "text": "Tes dari git! yang baru",
    "personId": "Y2lzY29zcGFyazovL3VzL1BFT1BMRS8wMjMxMWUxYy0zZmU0LTRlMWMtYWVkZi0xNGI2MDg0OGI5OWY",
    "personEmail": "gpratama796@gmail.com",
    "markdown": "Tes dari git! yang baru",
    "html": "<p>Tes dari git! yang baru</p>",
    "created": "2022-06-27T13:33:33.605Z"
}
Daftar Email :  ['simon.pouget46@gmail.com', 'brainpotter72@gmail.com', '', 'gabrielpotter72@gmail.com']
Finished: SUCCESS
```

b. Cek Webex untuk memastikan bahwa chat terkirim