

Linux-szerver automatikus telepítése on-premise környezetben

Összefoglalás: „Az automatizálás a banális és ismétlődő feladatok visszaszorulását eredményezi.” Rendszermérnökként gyakran találkozunk olyan feladattal, amelyet időről időre el kell végezni és a munkafolyamat előre meghatározott lépésekből áll. Azért, hogy az emberi hibát minimalizáljuk Linux operációs rendszer telepítést és konfigurálását is automatizálhatjuk. Jelen cikk azt mutatja be, hogy egy világszinten jelentős szolgáltató esetében milyen módon valósítható meg egy bérelt on-premise szerver telepítése és előkészítése további konfiguráció-menedzsment feladatok végrehajtására.
Kulcsszavak: Linux, szkript, automatikus telepítés.

Abstract: “Automation is driving the decline of banal and repetitive tasks.” As a systems engineer, we often encounter a task that needs to be performed from time to time, and the workflow consists of predefined steps. To minimize human error, we can automate the installation and configuration of Linux operating systems. This paper describes how to deploy and prepare a leased on-premise server for further configuration management tasks in case of a major global service provider.

Keywords: Linux, script, automatic installation.

* *Dunaiújvárosi Egyetem, Informatikai Intézet*
E-mail: hadarics@uniduna.hu

Bevezetés

Manapság sok olyan infrastruktúra-szolgáltató létezik, amely fizikai szerverek bérletét teszi lehetővé ügyfelek számára. Ezen szolgáltatás igénybevétele esetén a szolgáltató a hálózati infrastruktúrát és a hardver meghibásodások kezelését/cseréjét biztosítja az ügyfeleknek alapértelmezésként. A szerver számítógépre telepített szoftverek konfigurálása, üzemeltetése az ügyfél feladata.

[1] Hetzner Online GmbH – <https://www.hetzner.com>

[2] Installimage – <https://github.com/hetzneronline/installimage>

[3] Hetzner robot – <https://robot.hetzner.com>

[4] Ansible – <https://www.ansible.com>

Ezen publikáció esetében ismertetek egy saját megoldást, amely adott szerver-szolgáltató által biztosított lehetőségek figyelembevételével, automatizált Debian GNU/Linux-telepítés és -konfiguráció menedzsmentjének megoldására mutat be egy gyakorlati megoldást.

A szolgáltató sajátosságai

A Hetzner Online [1] Európa egyik legnagyobb adatközpont üzemeltetője. 1997 óta foglalkozik IT infrastruktúra-szolgáltatásokkal, szerverek elhelyezésével és bérbe adásával. Németországi adatközpontjai mellett Finnországban, USA-ban, Szingapúrban is rendelkezik telephelyekkel. Több százezer működő szervert szolgál ki az infrastruktúrája. Az ügyfeleket elsősorban az innovatív technológiákkal és a vonzó árakkal igyekszik magához csábítani, de kiemelten figyel a fenntarthatóságra is. Németországban 100%-ban a szükséges villamos energiát megújuló energiaforrás (vízenergia) segítségével állítja elő.

A Hetzner Online az ügyfelek részére egy speciális Linux netboot környezetet biztosít bérelt szerverek esetében. Az ún. Rescue-környezet segítségével az ügyfeleket többek között:

- operációs rendszereket telepíthetnek, az ún. Installimage [2] segítségével,
- hardver ellenőrzéseket futtathatnak,
- egy esetlegesen nem működő (nem bootoló) rendszer esetében adatmentést, helyreállítást végezhetnek Linux parancsok segítségével.

A szolgáltató emellett lehetővé teszi az ún. Robot Webservice API [3] használatával ügyféloldali műveletek automatikus elvégzését.

A kliensoldali környezet, és működési feltételek

A címbéli automatizált szervertelepítési megoldással kapcsolatban az alábbiak a célkitűzések:

- Input-adatként használja fel a szolgáltatótól kapott paramétereket.
- Végezze el előre meghatározott módon a szerver meghajtóinak particionálását és a szerver nevének beállítását.
- Kezelje a szerver újraindítását, és újraindítást követően folytatódjon a végrehajtása.
- Telepítse a választott konfigurációmenedzsment-megoldást (Ansible [4]), és tegye lehetővé az újonnan telepített szerver távoli menedzsmentjét.

- Az automatizált telepítést Linux asztali környezetből vagy egy korábban létrehozott szerverről lehet indítani, adott szkript végrehajtásával.

Az előzőek figyelembevételével, a működési feltételek az alábbiak:

- *openssh-client* – OpenSSH-kliens parancsok,
- *sshpas* – SSH-jelszóalapú hitelesítés használata.

A szolgáltató a sikeres bérletet követően az alábbi adatokat biztosítja az ügyfél részére (minta):

„Your server #server_id (192.168.108.1) has been activated and is now ready for use. Currently it is booted in the Rescue System. You can access the server via SSH2 using the following details:

IPv4 Address: 192.168.108.1

IPv6 Address: fe80::250:56ff:fec0:8

Username: root

Password: xDm8MieTXsU9ue

Host key: MYsjNcUoTgGEVHSUYfeJhRW9BOmh1m
HDFTLQaZ7bh/c (ECDSA 256)
k0pj5G4hiE7xBgf6325xfoicLUERmN8Kl5kLmVCNpno (ED25519 256)
Qvty3unE+4mAgTS2RFA7XcicBm/OPx88cXv/Fb3tHfQ (RSA 3072)

Az előző táblázatban alkalmazott IPv4 és IPv6 címek módosításra kerültek, és helyükre értelemszerűen a szolgáltatótól kapott publikus címeket kell elképzelnünk.

Az elkészített megoldás

A megoldás esetében egy input nevű fájlban rögzítjük a telepítendő szerverszolgáltatótól kapott releváns paramétereket és az új szerver nevét:

```
server="192.168.108.1"
```

```
hostname="debian12server"
```

```
sshuser="root"
```

```
export SSHPASS="xDm8MieTXsU9ue"
```

```
export TERM="xterm-256color"
sshtimeout=20
sshhostkeyalg=ECDSA
sshhostkeysize=256
sshhostkeytype="SHA${sshhostkeysize}"
sshhostkeyfingerprint="MYsjNcUoTgGEVHSUYfeJhRW9BOmh1mHDFTLQaZ7bh/c"
```

1. ábra. Forráskód első része

```
1  #!/usr/bin/env bash
2
3  . input
4
5  providedfingerprint=${sshhostkeysize} ${sshhostkeytype} ${sshhostkeyfingerprint} ${server} ${sshhostkeyalg})"
6  actualfingerprint=$(ssh-keyscan -4 -t ecdsa "${server}" 2>/dev/null | ssh-keygen -lf -)
7
8  if [ "$providedfingerprint" = "$actualfingerprint" ]; then
9
10 # Add new server to known hosts
11 ssh-keygen -F "${server}" &> /dev/null
12 ret=$?
13
14 if [ $ret -eq 1 ]; then
15     ssh-keyscan -4 -t ecdsa "${server}" >> ~/.ssh/known_hosts 2> /dev/null
16 fi
17
18 # Generate installimage configuration
19 installconfig="installimage.conf"
20 cat << EOT > "${installconfig}"
21 DRIVE1 /dev/nvme0n1
22 DRIVE2 /dev/nvme1n1
23 SWRAID 1
24 SWRAIDLEVEL 1
25 HOSTNAME ${hostname}
26 USE_KERNEL_MODE_SETTING yes
27 PART /boot/efi esp 250M
28 PART swap swap 64G
29 PART /boot ext3 1024M
30 PART / ext4 all
31 IMAGE /root/.oldroot/nfs/install/.../images/Debian-1207-bookworm-amd64-base.tar.gz
32 EOT
33
34 # Upload installer config
35 command="cat > /${installconfig}"
36 cat "${installconfig}" | sshpass -e ssh -o 'StrictHostKeyChecking=yes' -o 'ConnectTimeout=${sshtimeout}' ${sshuser}@${server} "${command}"
37
38 # Execute installer
39 installimage="/root/.oldroot/nfs/install/installimage"
40 command="bash ${installimage} -o < /${installconfig}"
41 result=$(sshpass -e ssh -t -o 'StrictHostKeyChecking=yes' -o 'ConnectTimeout=${sshtimeout}' ${sshuser}@${server} "${command}")
42 echo $result
43
44 # Reboot the server
45 command="reboot"
46 result=$(sshpass -e ssh -t -o 'StrictHostKeyChecking=yes' -o 'ConnectTimeout=${sshtimeout}' ${sshuser}@${server} "${command}")
47 echo $result
48
49 # Remove old server from known hosts
50 ssh-keygen -f ~/.ssh/known_hosts -R "${server}"
```

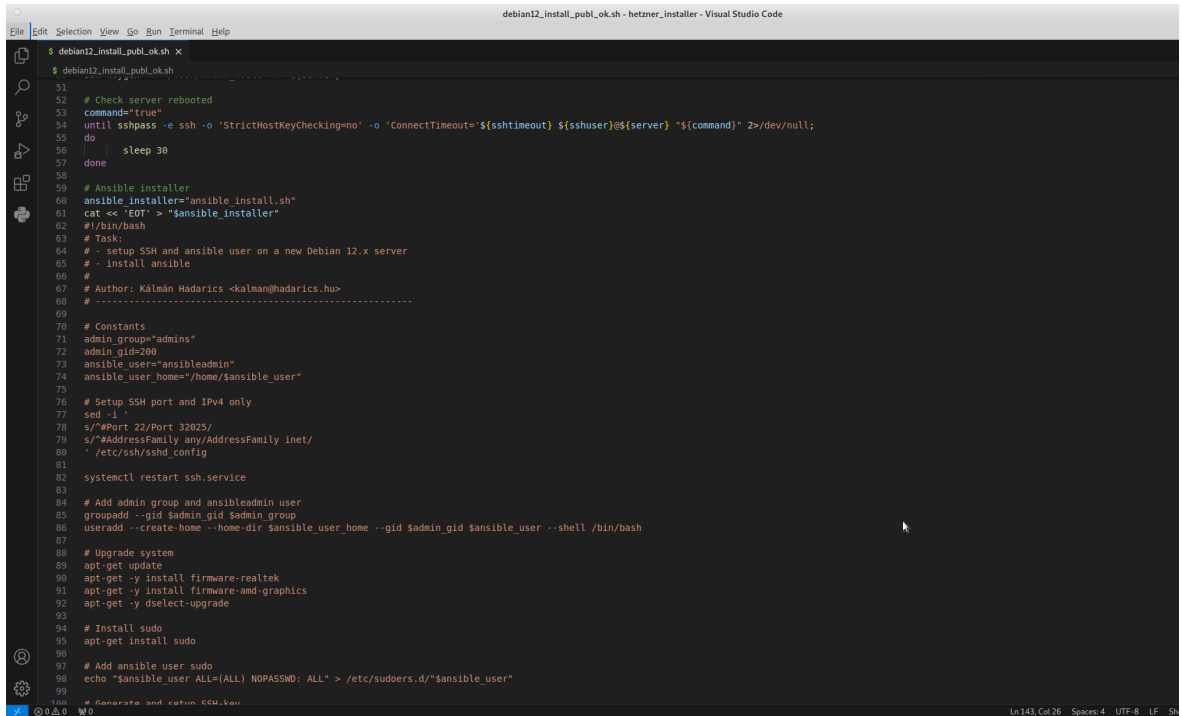
Forrás: Saját ábra.

A szkript az alábbiakat végzi el:

- Beolvassa az input fájlbeli paramétereket.

- Ellenőrzi, hogy a szolgáltató által küldött SSH ECDSA 256 kulcs lenyomata egyezik-e azzal, amit az aktuális kapcsolódáskor kapott a szkript. Amennyiben nem, akkor a szkript az ujjenyomatok értékeivel fog visszatérni.
- Amennyiben egyezik az ujjenyomat, akkor az alábbiak kerülnek végrehajtásra:

2. ábra. Forráskód második része



```
51
52 # Check server rebooted
53 command="true"
54 until sshpass -e ssh -o 'StrictHostKeyChecking=no' -o 'ConnectTimeout=${sshtimeout}' ${sshuser}@${server} "${command}" 2>/dev/null;
55 do
56     sleep 30
57 done
58
59 # Ansible installer
60 ansible_installer="ansible_install.sh"
61 cat << 'EOT' > "${ansible_installer}"
62 #!/bin/bash
63 # Task:
64 # - setup SSH and ansible user on a new Debian 12.x server
65 # - install ansible
66 #
67 # Author: Kálmán Hadarics <kalman@hadarics.hu>
68 # -----
69
70 # Constants
71 admin_group="admins"
72 admin_gid=200
73 ansible_user="ansibleadmin"
74 ansible_user_home="/home/${ansible_user}"
75
76 # Setup SSH port and IPv4 only
77 sed -i \
78 s/"#Port 22/Port 32025/ \
79 s/"#AddressFamily any/AddressFamily inet/ \
80 ' /etc/ssh/sshd_config
81
82 systemctl restart ssh.service
83
84 # Add admin group and ansibleadmin user
85 groupadd --gid $admin_gid $admin_group
86 useradd --create-home --home-dir $ansible_user_home --gid $admin_gid $ansible_user --shell /bin/bash
87
88 # Upgrade system
89 apt-get update
90 apt-get -y install firmware-realtek
91 apt-get -y install firmware-amd-graphics
92 apt-get -y dselect-upgrade
93
94 # Install sudo
95 apt-get install sudo
96
97 # Add ansible user sudo
98 echo "${ansible_user} ALL=(ALL) NOPASSWD: ALL" > /etc/sudoers.d/${ansible_user}
99
100 # Generate and caten CGM-key
```

Forrás: Saját ábra.

- Az új szerver hozzáadása a kliens `.ssh/known_hosts` fájljához.
- Az installimage [2] konfigurációjának (`installimage.conf`) dinamikus létrehozása.
- Az `installimage.conf` fájl feltöltése SSH-kliens segítségével.
- Az `installimage` parancs távoli végrehajtása SSH segítségével a szerveren. Ez elvégzi a konfiguráció alapján a telepítést és a hálózat alapbeállításait.

- A telepített szerver újraindítása.
- A korábbi tárolt ujjlenyomatok törlése a `.ssh/known_hosts` fájlból. A szerver újrategelítése új SSH-kulcsokat hoz létre, így értelemszerűen az ujjlenyomatok is változnak.
- Várákozás az szerver újraindulására, 30 másodpercenként csatlakozási kísérlet.
- Az `ansible_installer.sh` szkript dinamikus létrehozása. Ez a szkript az alábbiakat végzi:
 - admins csoport és rendszer felhasználó (`ansible_admin`) létrehozása (`$ansible_user` változó tartalma)
 - az alapértelmezett OpenSSH-port módosítása (biztonsági megfontolásból);
 - a feltelepített rendszer frissítése,
 - sudo tegelítése és `ansible_admin` sudo privilégiumának konfigurációja,
 - SSH-kulcspár létrehozása az `ansible_admin` felhasználó esetében.
 - az ansible csomag tegelítése, amely lehetővé teszi a távoli adminisztrációt SSH felhasználásával.
- Az `ansible_installer.sh` szkript távoli végrehajtása SSH segítségével a szerveren.
- Az `ansible_admin` számára generált kulcspár privát kulcsának letöltése a helyi rendszerbe, majd a távoli szerverről a privát kulcs törlése.
- Ansible inventory fájl létrehozása, amely felhasználásával a távoli szerver Ansible segítségével távolról vezérelhető lesz.

3. ábra. Forráskód harmadik része

```

100 # Generate and setup SSH-key
101 ssh_dir="${ansible_user_home}/ssh"
102 ssh_key_private="${ssh_dir}/${(hostname) id_rsa}"
103 ssh_key_public="${ssh_dir}/${(hostname) id_rsa.pub}"
104 ssh_key_comment="${(hostname) id_rsa}"
105 ssh_auth_keyfile="${ssh_dir}/authorized_keys"
106
107 sudo -u ${ansible_user} mkdir -m 700 $ssh_dir
108 sudo -u ${ansible_user} echo -e y\N | sudo -u ${ansible_user} ssh-keygen -t rsa -b 4096 -C "${ssh_key_comment}" -f "${ssh_key_private}" -N ""
109 sudo -u ${ansible_user} cp "${ssh_key_public}" "${ssh_auth_keyfile}"
110 sudo -u ${ansible_user} chmod 600 "${ssh_auth_keyfile}"
111
112 # Install ansible
113 apt-get -y install ansible
114 EOF
115
116 # Upload Ansible installer
117 command="cat > /root/${ansible_installer}"
118 cat "${ansible_installer}" | sshpass -e ssh -o 'StrictHostKeyChecking=no' -o 'ConnectTimeout=${ssh_timeout}' ${sshuser}@${server} "${command}"
119
120 # Execute Ansible installer
121 command="bash /root/${ansible_installer}"
122 result=$(sshpass -e ssh -t -o 'StrictHostKeyChecking=no' -o 'ConnectTimeout=${ssh_timeout}' ${sshuser}@${server} "${command}")
123 echo $result
124
125 # New ssh port
126 ssh_port=32025
127
128 # Get generated Ansible admin private key
129 ansible_user="ansibleadmin"
130 ansible_user_home="/home/${ansible_user}"
131 ssh_dir="${ansible_user_home}/ssh"
132 ssh_key_private_file="${(hostname) id_rsa}"
133 ssh_key_private_path="${ssh_dir}/${ssh_key_private_file}"
134
135 command="cat $ssh_key_private_path"
136 sshpass -e ssh -o "Port=${ssh_port}" -o 'StrictHostKeyChecking=no' -o 'ConnectTimeout=${ssh_timeout}' ${sshuser}@${server} "${command}" > "${ssh_key_private_file}"
137
138 # Delete remote Ansible admin private key
139 command="rm $ssh_key_private_path"
140 sshpass -e ssh -o "Port=${ssh_port}" -o 'StrictHostKeyChecking=no' -o 'ConnectTimeout=${ssh_timeout}' ${sshuser}@${server} "${command}"
141
142 # Generate Ansible inventory related to host
143 inventory="inventory ${hostname}"
144 cat << EOF > "${inventory}"
145
146     ${hostname}:
147         ansible_host: ${server}
148         ansible_ssh_private_key_file: /home/kami/ansible_project/keys/${ssh_key_private_file}
149
150 EOF

```

Forrás: Saját ábra.

Összegzés

Aholgyakorli üzemeltetési feladat Linux-szerverek telepítése, mindenképpen gondolkodni kell azon, hogy milyen módon valósítható ez meg automatizált módon. On-premise környezetben az automatizált telepítés megoldása nagyban függ a szolgáltató által biztosított feltételektől és lehetőségektől. Némi szkript-írási és integrálási tapasztalatokkal elkészíthető olyan megoldás, amely rugalmasan felhasználható automatikus telepítés kivitelezésére.

A bemutatott megoldás képes arra, hogy az újonnan telepített rendszer Ansible által vezérelhető legyen, a használt Ansible-inventory kibővítésével.