

Membangun Backup Data Penting Secara Otomatis Menggunakan *Bash Script* Pada *Red Hat Enterprise Linux*

Yahdi Kusnadi^{1*)}, Ardi Widiyanto²⁾

¹⁾ Sistem Informasi, Universitas Bina Sarana Informatika

²⁾ Teknologi Informasi, Universitas Bina Sarana Informatika

*) Correspondence author: yahdi.ydk@bsi.ac.id, Jakarta, Indonesia

DOI: <https://doi.org/10.37012/jtik.v10i2.2313>

Abstrak

Data menjadi aset penting bagi individu maupun organisasi. Data penting, seperti dokumen, email, foto, dan video, perlu dilindungi dari kehilangan atau kerusakan. Salah satu cara untuk melindungi data adalah dengan membuat backup secara berkala. Penelitian ini menyajikan sebuah solusi inovatif untuk mengelola pencadangan data pada lingkungan *Red Hat Enterprise Linux* (RHEL) versi 9 melalui pengembangan *skrip bash* yang efisien dan handal. Penelitian ini bertujuan untuk mengembangkan sebuah skrip *bash* (*Bash Script*) sebagai alat cadangan untuk file-file kritis. Skrip ini dirancang untuk memberikan solusi yang dapat diandalkan dan mudah digunakan bagi pengguna, terutama dalam konteks pengelolaan data yang sensitif dan vital. Dengan memanfaatkan otomatisasi dalam proses pencadangan, penelitian ini berhasil mengatasi berbagai tantangan yang sering dihadapi oleh administrator sistem dalam menjaga keamanan, integritas, dan ketersediaan data di lingkungan *server*. *Skrip* yang dikembangkan tidak hanya mampu melakukan pencadangan data secara berkala dengan jadwal yang dapat disesuaikan, tetapi juga dilengkapi dengan fitur transfer data yang aman menggunakan protokol SFTP, sehingga menjamin bahwa data yang dicadangkan dapat dipindahkan ke lokasi lain tanpa khawatir akan risiko keamanan. Selain itu, mekanisme penamaan file yang terstruktur memudahkan pengelolaan dan penelusuran berkas pencadangan, sehingga pengguna dapat dengan mudah menemukan dan memulihkan data yang dibutuhkan. Hasil dari penelitian ini memberikan kontribusi yang signifikan dalam bidang manajemen data, terutama bagi pengguna RHEL 9 yang membutuhkan solusi pencadangan yang tidak hanya reliabel tetapi juga mudah diimplementasikan dan dikelola, sehingga dapat meningkatkan efisiensi operasional dalam pengelolaan data.

Kata Kunci: Keamanan data, Backup data, Bash script, Crontab, Red Hat Enterprise Linux (RHEL)

Abstract

Data is an important asset for individuals and organizations. Important data, such as documents, emails, photos and videos, needs to be protected from loss or damage. One way to protect data is to create regular backups. This research presents an innovative solution for managing data backup in a Red Hat Enterprise Linux (RHEL) version 9 environment through the development of efficient and reliable bash scripts. This research aims to develop a bash script (Bash Script) as a backup tool for critical files. This script is designed to provide a reliable and easy-to-use solution for users, especially in the context of managing sensitive and vital data. By utilizing automation in the backup process, this research succeeded in overcoming various challenges often faced by system administrators in maintaining security, integrity and data availability in server environments. The developed script is not only capable of performing regular data backups with an adjustable schedule, but is also equipped with a secure data transfer feature using the SFTP protocol, thus guaranteeing that the backed up data can be moved to another location without worrying about security risks. Additionally, a structured file naming mechanism makes it easier to manage and browse backup files, so users can easily find and restore the data they need. The results of this research provide a significant contribution in the field of data management, especially for RHEL 9 users who need a backup solution that is not only reliable but also easy to implement and manage, so as to increase operational efficiency in data management.

Keywords: Data security, data backup, Bash script, Crontab, Red Hat Enterprise Linux (RHEL)

<https://journal.thamrin.ac.id/index.php/jtik/article/view/2313>

PENDAHULUAN

Data menjadi aset penting bagi individu maupun organisasi. Data penting, seperti dokumen, email, foto, dan video, perlu dilindungi dari kehilangan atau kerusakan. Salah satu cara untuk melindungi data adalah dengan membuat backup secara berkala. Penelitian ini bertujuan untuk mengembangkan sebuah skrip *bash* (*Bash Script*) sebagai alat cadangan untuk file-file kritis. Skrip ini dirancang untuk memberikan solusi yang dapat diandalkan dan mudah digunakan bagi pengguna, terutama dalam konteks pengelolaan data yang sensitif dan vital.

Membuat *backup data* secara manual dapat memakan waktu dan rawan kesalahan. Oleh karena itu, diperlukan solusi yang lebih efisien dan terautomasi. Salah satu solusi yang dapat digunakan adalah dengan menggunakan *Bash Script* pada *Red Hat Enterprise Linux*.

“*Bash* adalah *shell*, atau penerjemah bahasa perintah, untuk sistem operasi gnu. Itu nama adalah akronim dari '*Bourne-Again SHell*', plesetan dari Stephen Bourne, penulisnya dari nenek moyang langsung dari Unix shell *sh* saat ini, yang muncul di Edisi Ketujuh Unix versi Penelitian *Bell Labs*” (Ramey & Fox, 2022). Dengan menggunakan *bash script*, *backup data* dapat dilakukan secara otomatis pada waktu yang ditentukan. Pengembangan *Bash Script* ini bertujuan untuk memberikan solusi backup yang tidak hanya efektif tetapi juga ekonomis, mengingat *Bash Script* merupakan bagian dari banyak distribusi *Linux* yang tidak memerlukan biaya tambahan untuk penggunaannya. Dengan demikian, *Bash Script* sebagai alat *backup* untuk file kritis dapat menjadi pilihan yang tepat bagi mereka yang membutuhkan solusi *backup* yang handal dan terjangkau.

Crontab adalah utilitas yang digunakan untuk menjadwalkan tugas-tugas di *Linux*. Dengan menggunakan *crontab*, *bash script* dapat dijalankan secara otomatis pada waktu yang diinginkan.

METODE

Penelitian ini menggunakan metode kualitatif, yang berfokus pada pemahaman mendalam tentang fenomena dan pengalaman manusia dalam konteks tertentu. Metode ini memungkinkan peneliti untuk menjelajahi kompleksitas dan makna di balik data yang dikumpulkan. Metode pengumpulan data yang digunakan dalam penelitian ini meliputi:

<https://journal.thamrin.ac.id/index.php/jtik/article/view/2313>

- a. Studi literatur Mengkaji literatur yang relevan dengan Bash Script dan Backup data pada *Red Hat Enterprise Linux*, seperti dokumentasi resmi *Red Hat Enterprise Linux*, artikel ilmiah, dan jurnal online.
- b. Wawancara Melakukan wawancara dengan administrator sistem yang berpengalaman dalam menggunakan *Bash Script* untuk *backup data* pada *Red Hat Enterprise Linux*.
- c. Observasi Mengamati penggunaan *Bash Script* dalam *backup data* di *Red Hat Enterprise Linux* secara langsung.
- d. Analisis dokumen Menganalisis dokumen terkait konfigurasi dan penggunaan *Bash Script* di sistem *Red Hat Enterprise Linux*.

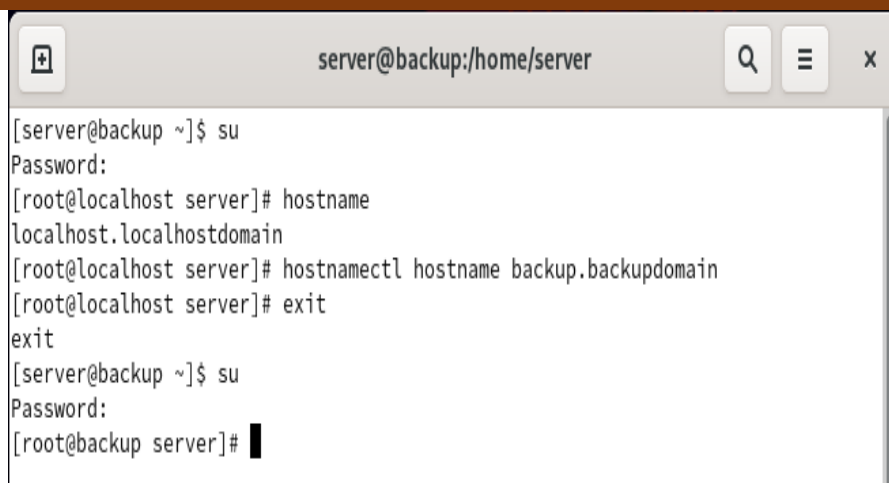
HASIL DAN PEMBAHASAN

Dalam pelaksanaan project "Membangun Backup Data Penting Secara Otomatis Menggunakan *Bash script* Pada *Red Hat Enterprise Linux*" beberapa hasil signifikan telah dicapai yang menunjukkan efisiensi dan efektivitas penggunaan *Backup Data* Penting Menggunakan *Bash script* pada *Red Hat Enterprise Linux*. Berikut adalah implementasi dan Hasil mengenai Membangun *Backup Data* Penting Secara Otomatis menggunakan *bash script*:

Bash script yang dibuat untuk mengotomatisasi *backup data* telah berhasil diimplementasikan pada *Red Hat Enterprise Linux* (RHEL) 9. Script ini dirancang untuk: Mencadangkan data secara berkala dari lokasi sumber yang telah ditentukan, Menggunakan protokol SFTP yang aman untuk mentransfer data terenkripsi ke lokasi penyimpanan backup, Membuat direktori backup terstruktur dengan menyertakan tanggal backup untuk memudahkan pengelolaan file. jadi berikut tata cara konfigurasinya:

Langkah per-tama yang harus kita lakukan adalah *login* sebagai *root* dan membuat *hostname* terlebih dahulu agar kita bisa membedakan yang mana *server* dan *target*.

Pada gambar 1 berikut, perintah tersebut berfungsi untuk memberi nama *hostname* pada *device server*, dan setelah kita mengganti *hostname* kita harus keluar lalu *login* lagi agar *hostname* dapat ter-*update*.



```
server@backup:/home/server
[server@backup ~]$ su
Password:
[root@localhost server]# hostname
localhost.localhostdomain
[root@localhost server]# hostnamectl hostname backup.backupdomain
[root@localhost server]# exit
exit
[server@backup ~]$ su
Password:
[root@backup server]#
```

Gambar 1. Hostname Server

Sumber : Penelitian 2024

Langkah ke-dua sama seperti langkah pertama tetapi kita ubah *hostname* pada *device target*.



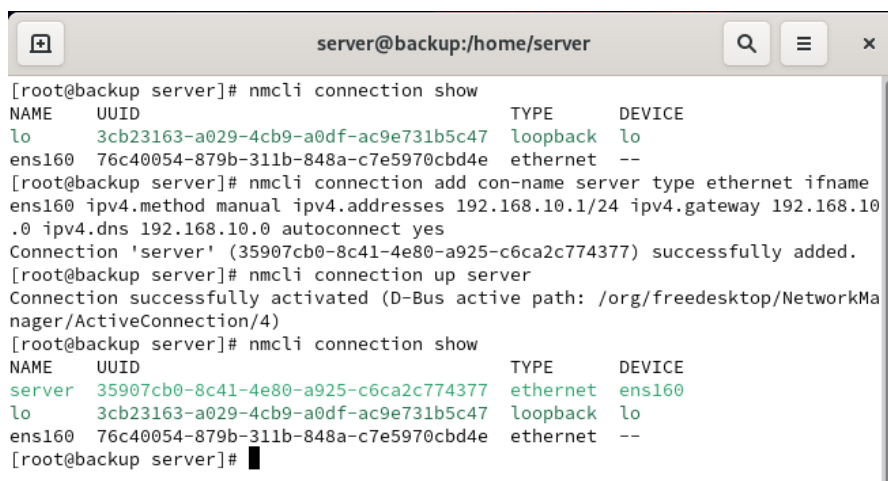
```
targetbackup@target:/home/targetbackup
[targetbackup@localhost ~]$ su
Password:
[root@localhost targetbackup]# hostnamectl hostname target.targetdomain
[root@localhost targetbackup]# exit
exit
[targetbackup@localhost ~]$ su
Password:
[root@target targetbackup]#
```

Gambar 2. Hostname Target

Sumber : Penelitian 2024

Pada gambar 2 perintah tersebut berfungsi untuk memberi nama *hostname* pada *device Target*, dan setelah kita mengganti hostname kita harus keluar lalu *login* lagi agar *hostname* dapat ter-*update*.

Langkah ke-tiga kita harus membuat koneksi antara *device* saling terhubung satu sama lain dengan cara membuat *network* baru bernama *server* pada *device server*.



```
server@backup:/home/server
[root@backup server]# nmcli connection show
NAME     UUID                                     TYPE     DEVICE
lo       3cb23163-a029-4cb9-a0df-ac9e731b5c47  loopback lo
ens160   76c40054-879b-311b-848a-c7e5970cbd4e  ethernet --
[root@backup server]# nmcli connection add con-name server type ethernet ifname
ens160 ipv4.method manual ipv4.addresses 192.168.10.1/24 ipv4.gateway 192.168.10
.0 ipv4.dns 192.168.10.0 autoconnect yes
Connection 'server' (35907cb0-8c41-4e80-a925-c6ca2c774377) successfully added.
[root@backup server]# nmcli connection up server
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkMa
nager/ActiveConnection/4)
[root@backup server]# nmcli connection show
NAME     UUID                                     TYPE     DEVICE
server   35907cb0-8c41-4e80-a925-c6ca2c774377  ethernet ens160
lo       3cb23163-a029-4cb9-a0df-ac9e731b5c47  loopback lo
ens160   76c40054-879b-311b-848a-c7e5970cbd4e  ethernet --
[root@backup server]#
```

Gambar 3. Membuat Network pada Server

Sumber : Penelitian 2024

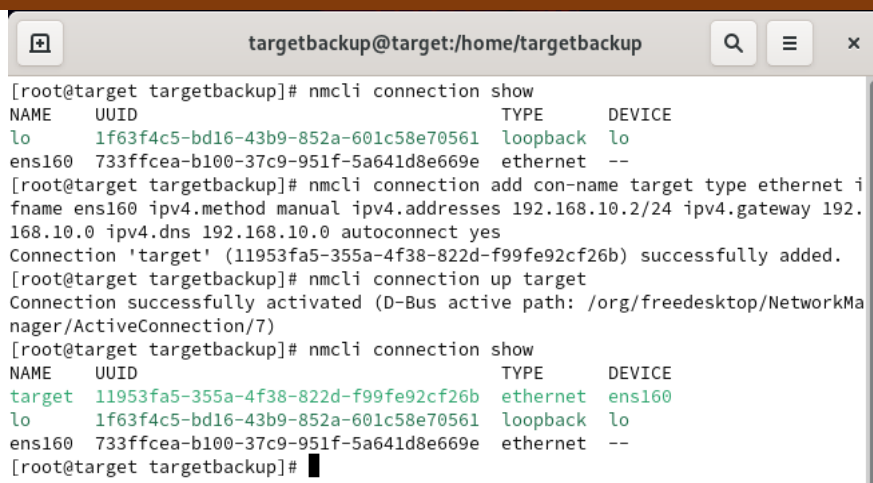
Pada gambar 3 kita membuat ipv4 secara manual agar ip nya selalu konsisten dan tidak berubah – ubah, berikut perintah yang di gunakan lalu kita buat *auto connect* agar saat di perangkat di *restart* nantinya akan *otomatis connect* ke *network* yang kita buat (*server*) :

nmcli connection add con-name server type ethernet ifname ens160 ipv4.method manual ipv4.addresses 192.168.10.1/24 ipv4.gateway 192.168.10.0 ipv4.dns 192.168.10.0 autoconnect yes

jangan lupa untuk menjalankan si *network* yang kita buat tadi dengan perintah:

nmcli connection up server

Langkah ke-empat kita harus membuat koneksi antara *device* saling terhubung satu sama lain dengan cara membuat *network* baru bernama *target* pada *device target*.



```
targetbackup@target:/home/targetbackup
[root@target targetbackup]# nmcli connection show
NAME      UUID                                  TYPE      DEVICE
lo        1f63f4c5-bd16-43b9-852a-601c58e70561  loopback  lo
ens160    733ffcea-b100-37c9-951f-5a641d8e669e  ethernet  --
[root@target targetbackup]# nmcli connection add con-name target type ethernet ifname ens160 ipv4.method manual ipv4.addresses 192.168.10.2/24 ipv4.gateway 192.168.10.0 ipv4.dns 192.168.10.0 autoconnect yes
Connection 'target' (11953fa5-355a-4f38-822d-f99fe92cf26b) successfully added.
[root@target targetbackup]# nmcli connection up target
Connection successfully activated (D-Bus active path: /org/freedesktop/NetworkManager/ActiveConnection/7)
[root@target targetbackup]# nmcli connection show
NAME      UUID                                  TYPE      DEVICE
target    11953fa5-355a-4f38-822d-f99fe92cf26b  ethernet  ens160
lo        1f63f4c5-bd16-43b9-852a-601c58e70561  loopback  lo
ens160    733ffcea-b100-37c9-951f-5a641d8e669e  ethernet  --
[root@target targetbackup]#
```

Gambar 4. Membuat Network pada *Target*

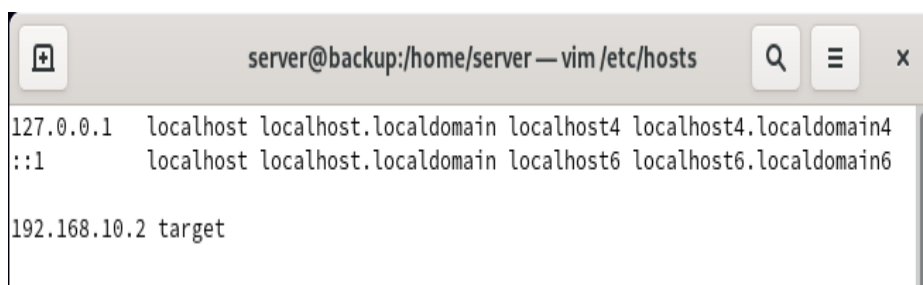
Sumber : Penelitian 2024

Pada gambar 4 kita membuat ipv4 secara manual agar ip nya selalu konsisten dan tidak berubah – ubah, berikut perintah yang di gunakan lalu kita buat *auto connect* agar saat di perangkat di *restart* nantinya akan otomatis *connect* ke *network* yang kita buat (*target*) :

nmcli connection add con-name target type ethernet ifname ens160 ipv4.method manual ipv4.addresses 192.168.10.2/24 ipv4.gateway 192.168.10.0 ipv4.dns 192.168.10.0 autoconnect yes

jangan lupa untuk menjalankan si *network* yang kita buat tadi dengan perintah **nmcli connection up target**

Langkah ke-lima kita inialisasikan *ip device target* pada *device server*.

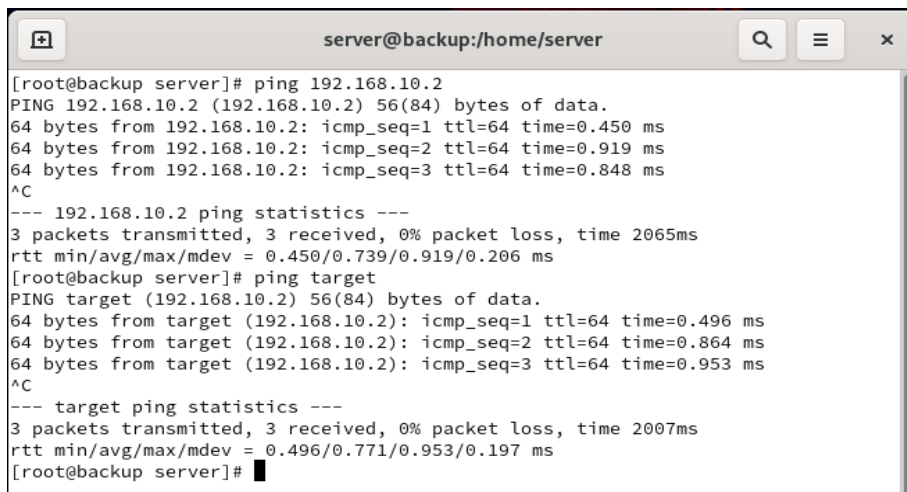


```
server@backup:/home/server — vim /etc/hosts
127.0.0.1 localhost localhost.localdomain localhost4 localhost4.localdomain4
::1 localhost localhost.localdomain localhost6 localhost6.localdomain6
192.168.10.2 target
```

Gambar 5. Hosts Server ke *Target*

Sumber : Penelitian 2024

Pada gambar 5 kita menambahkan *host ip* 192.168.10.2 bernama *target* pada *server* agar nanti saat menuliskan skrip pada *bash script* tidak menggunakan *ip* lagi tapi menggunakan *target*. Dengan perintah **vim /etc/hosts**.



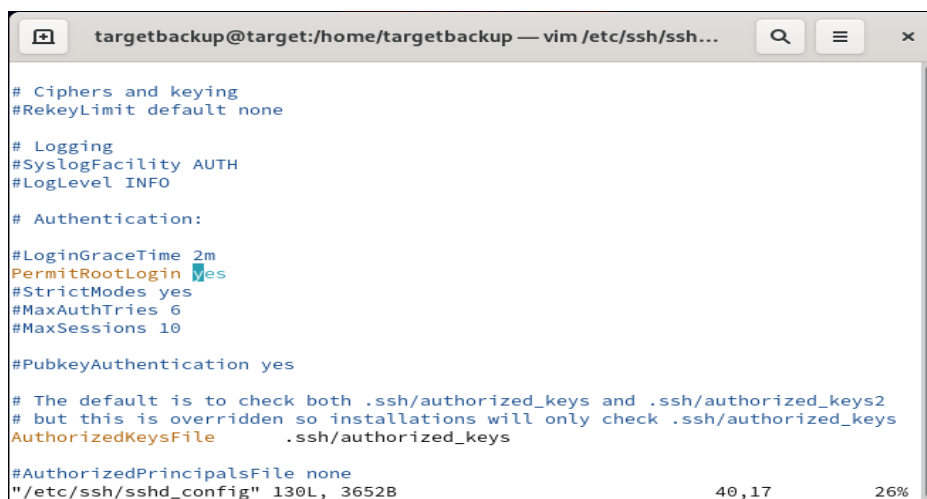
```
server@backup:/home/server
[root@backup server]# ping 192.168.10.2
PING 192.168.10.2 (192.168.10.2) 56(84) bytes of data.
64 bytes from 192.168.10.2: icmp_seq=1 ttl=64 time=0.450 ms
64 bytes from 192.168.10.2: icmp_seq=2 ttl=64 time=0.919 ms
64 bytes from 192.168.10.2: icmp_seq=3 ttl=64 time=0.848 ms
^C
--- 192.168.10.2 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2065ms
rtt min/avg/max/mdev = 0.450/0.739/0.919/0.206 ms
[root@backup server]# ping target
PING target (192.168.10.2) 56(84) bytes of data.
64 bytes from target (192.168.10.2): icmp_seq=1 ttl=64 time=0.496 ms
64 bytes from target (192.168.10.2): icmp_seq=2 ttl=64 time=0.864 ms
64 bytes from target (192.168.10.2): icmp_seq=3 ttl=64 time=0.953 ms
^C
--- target ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2007ms
rtt min/avg/max/mdev = 0.496/0.771/0.953/0.197 ms
[root@backup server]#
```

Gambar 6. Test Ping ke Target

Sumber : Penelitian 2024

Pada gambar 6 kita coba *ping* ke *ip* 192.168.10.2 dan *target* dari *server* jika *ping* berhasil maka mereka sudah saling terkoneksi antara *server* dan *target*.

Langkah ke-enam kita konfigurasi *sshd service* pada *device target* menjadi *yes* agar pada saat *backuptools.sh* di jalankan bisa mengakses *user root*



```
targetbackup@target:/home/targetbackup — vim /etc/ssh/sshd_...
# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
#StrictModes yes
#MaxAuthTries 6
#MaxSessions 10

#PubkeyAuthentication yes

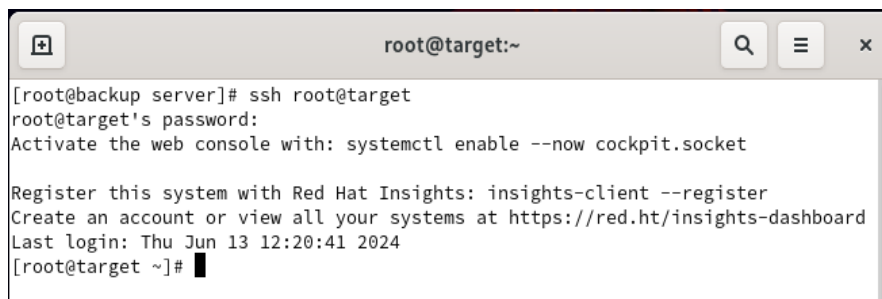
# The default is to check both .ssh/authorized_keys and .ssh/authorized_keys2
# but this is overridden so installations will only check .ssh/authorized_keys
AuthorizedKeysFile .ssh/authorized_keys

#AuthorizedPrincipalsFile none
"/etc/ssh/sshd_config" 130L, 3652B 40,17 26%
```

Gambar 7. /etc/ssh/sshd_config

Sumber : Penelitian 2024

Pada gambar 7 kita *setting sshd service* pada *device target* di `/etc/ssh/sshd_config` dengan perintah `vim /etc/ssh/sshd_config` ubat `permitrootlogin` nya menjadi `yes` agar dari *server* bisa mengakses ssh menggunakan *user root* ke *target*.



```
root@target:~
[root@backup server]# ssh root@target
root@target's password:
Activate the web console with: systemctl enable --now cockpit.socket

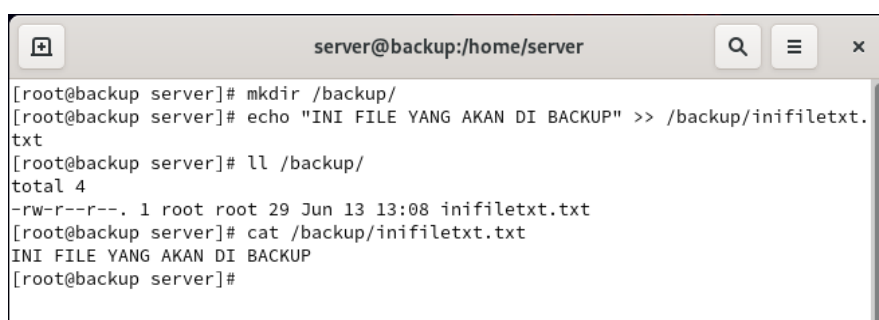
Register this system with Red Hat Insights: insights-client --register
Create an account or view all your systems at https://red.ht/insights-dashboard
Last login: Thu Jun 13 12:20:41 2024
[root@target ~]#
```

Gambar 8. Test SSH

Sumber : Penelitian 2024

Pada gambar 8 kita *test ssh* nya dari server ke target. Jika belum *bisa ssh* coba cek kembali pada pengaturan *ssh* di yang berada pada *device target*.

Langkah ke-tujuh kita buat direktori dan file yang akan di backup.



```
server@backup:/home/server
[root@backup server]# mkdir /backup/
[root@backup server]# echo "INI FILE YANG AKAN DI BACKUP" >> /backup/inifiletxt.txt
[root@backup server]# ll /backup/
total 4
-rw-r--r--. 1 root root 29 Jun 13 13:08 inifiletxt.txt
[root@backup server]# cat /backup/inifiletxt.txt
INI FILE YANG AKAN DI BACKUP
[root@backup server]#
```

Gambar 9. Membuat Direktori dan File

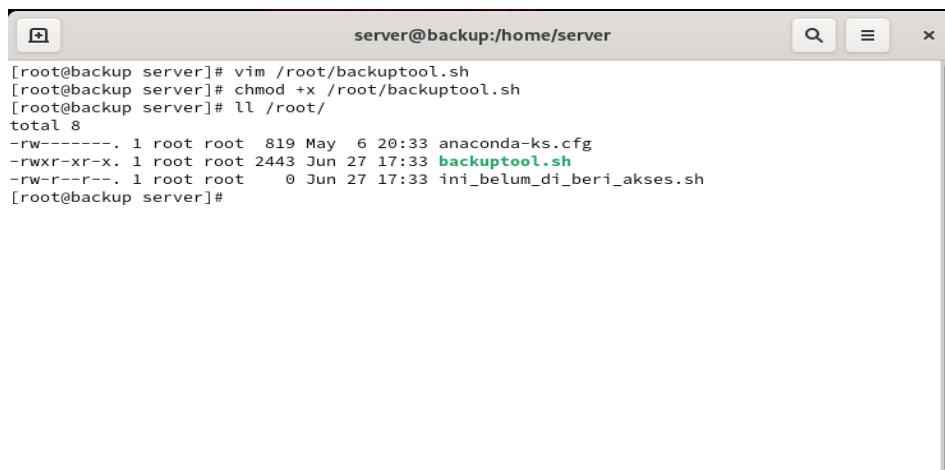
Sumber : Penelitian 2024

Pada gambar 9 kita buat direktori dan *file* dengan perintah pada gambar ini penulis membuat direktori yang bernama *backup* dan *file* yang bernama `inifiletxt.txt` yang berisikan “INI FILE YANG AKAN DI BACKUP” dengan perintah `mkdir /backup/` dan `vim inifiletxt.txt`.

Langkah ke-delapan kita buat *file script* nya dengan perintah `vim /root/backuptool.sh`.

Skrip *Bash* yang digunakan adalah skrip untuk melakukan pencadangan (*backup*) beberapa direktori yang ditentukan ke dalam satu arsip terkompresi menggunakan *tar* dan kemudian mentransfer arsip tersebut ke server jauh menggunakan *sftp*. Skrip *Bash* ini dirancang untuk mem-*backup* direktori ke *device target*.

Langkah ke-sembilan setelah selesai membuat script jangan lupa untuk memberi akses dan izin eksekusi agar bisa dijalankan dengan perintah **chmod +x /root/backuptool.sh**



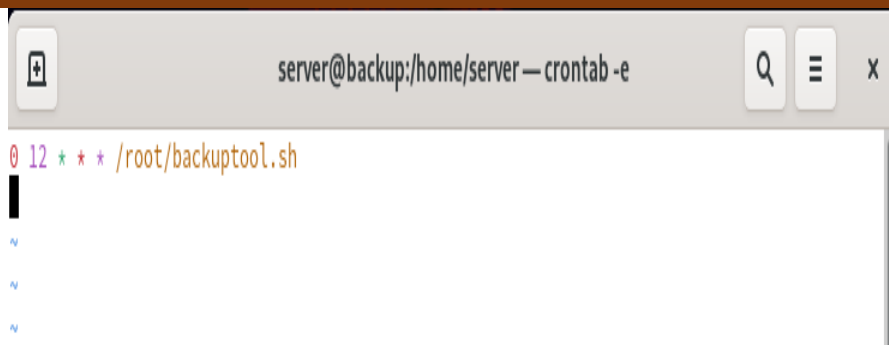
```
server@backup:/home/server
[root@backup server]# vim /root/backuptool.sh
[root@backup server]# chmod +x /root/backuptool.sh
[root@backup server]# ll /root/
total 8
-rw-----. 1 root root 819 May 6 20:33 anaconda-ks.cfg
-rwxr-xr-x. 1 root root 2443 Jun 27 17:33 backuptool.sh
-rw-r--r--. 1 root root 0 Jun 27 17:33 ini_belum_di_beri_akses.sh
[root@backup server]#
```

Gambar 10. *chmod +x backuptool.sh*

Sumber : Penelitian 2024

berikut penjelasan dari perintah tersebut "**chmod**" adalah perintah untuk mengubah izin *file*. "**+x**" menambahkan izin eksekusi kepada *file*. "**backuptool.sh**" adalah nama *file* yang akan diberikan izin eksekusi.

Langkah ke-sepuluh selanjutnya kita akan *Scheduling Shell Script* menggunakan *Crontab Scheduler* agar berjalan secara otomatis *Crontab* adalah application *daemon* (berjalan di belakang layar) yang digunakan untuk menjalankan tugas terjadwal pada suatu waktu di sistem operasi *Linux*. Setiap pengguna pada sistem yang memiliki *file Crontab*, mengizinkan *file* tersebut melakukan suatu tindakan dan sesuai waktu yang ditentukan. Pada project ini, Penjadwal *Crontab* digunakan untuk secara otomatis mengambil cadangan folder tertentu menggunakan skrip shell **backuptool.sh** setiap hari pada jam 12 siang. Penjadwal *Crontab* melakukan tugasnya pada jadwal yang ditentukan mengingat mesin dalam keadaan menyala pada waktu itu. Buka *Crontab* untuk pengguna root menggunakan perintah **crontab -e** dimana tag **-e** digunakan untuk membuka *crontab* dalam *mode edit*.

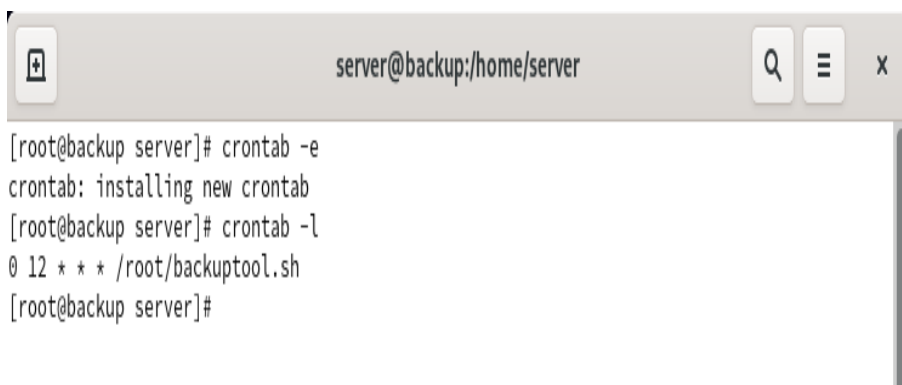


Gambar 11. Konfigurasi crontab -e
Sumber : Penelitian 2024

Penjelasan pada gambar 11 adalah sebagai berikut :

0	12	*	*	*	/root/backuptool.sh
Menit	Jam	Tanggal	Bulan	Hari	Perintah

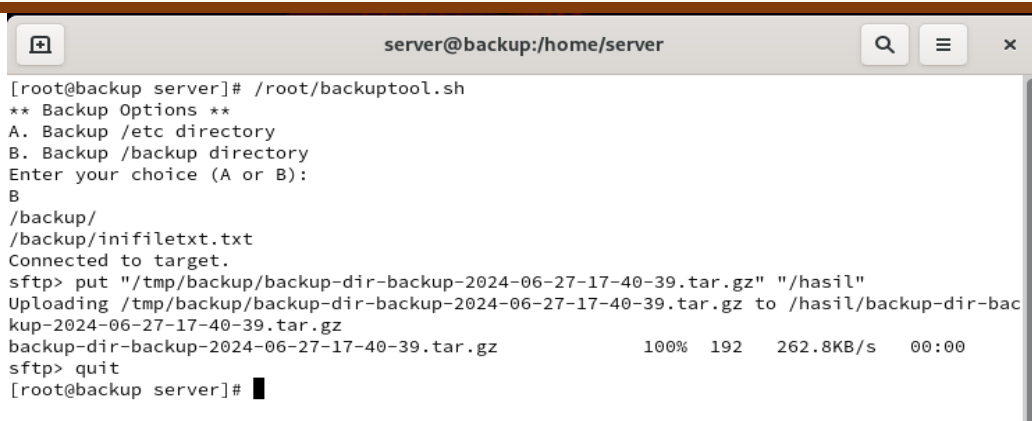
- a. Menit : Menit di mana perintah akan dieksekusi. Dibutuhkan nilai dari 0 hingga 59.
- b. Jam : Jam di mana perintah akan dieksekusi. Dibutuhkan nilai dari 0 hingga 23.
- c. Tanggal : Tanggal bulan (1-31)
- d. Bulan : Bulan dalam setahun (1-12)
- e. Hari : Hari dalam seminggu dibutuhkan nilai dari 0 hingga 6. (0 = Minggu) (1 = Senin, dll).
- f. Perintah : Perintah untuk dijalankan



Gambar 12. Automashi Bash Script
Sumber : Penelitian 2024

Pada gambar ke 12 kita bisa melihat apakah *crontab* kita sudah berjalan atau belum dengan perintah **crontab -l**. option l berfungsi untuk melihat list yang berjalan pada *crontab*.

Berikut adalah proses yang akan ditampilkan jika *script backuptool.sh* di jalankan

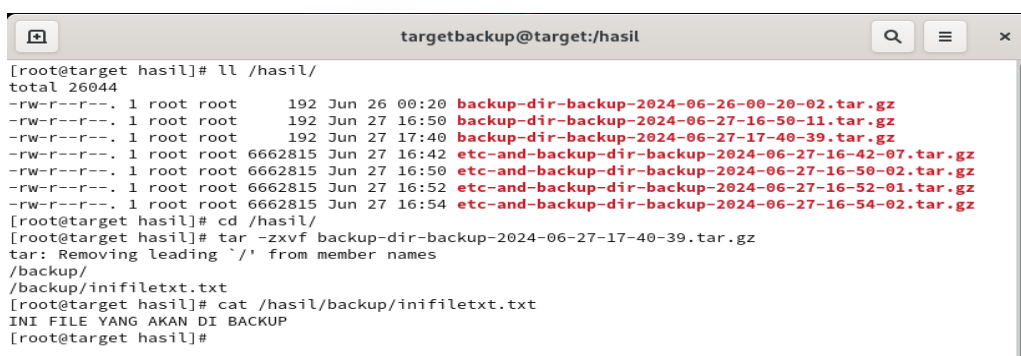


```
[root@backup server]# ./backuptool.sh
** Backup Options **
A. Backup /etc directory
B. Backup /backup directory
Enter your choice (A or B):
B
/backup/
/backup/inifiletxt.txt
Connected to target.
sftp> put "/tmp/backup/backup-dir-backup-2024-06-27-17-40-39.tar.gz" "/hasil"
Uploading /tmp/backup/backup-dir-backup-2024-06-27-17-40-39.tar.gz to /hasil/backup-dir-backup-2024-06-27-17-40-39.tar.gz
backup-dir-backup-2024-06-27-17-40-39.tar.gz          100% 192   262.8KB/s   00:00
sftp> quit
[root@backup server]#
```

Gambar 13. Proses Backup

Sumber : Penelitian 2024

Pada gambar 13 penulis menjalankan *bash script*-nya dengan cara `./backuptool.sh` dalam perintah *Bash shell Linux ./* di awal perintah menunjukkan direktori kerja saat ini. Ini adalah cara untuk memberi tahu shell agar mengeksekusi *file* yang terletak di direktori tempat kita berada saat ini.



```
[root@target hasil]# ll /hasil/
total 26044
-rw-r--r--. 1 root root   192 Jun 26 00:20 backup-dir-backup-2024-06-26-00-20-02.tar.gz
-rw-r--r--. 1 root root   192 Jun 27 16:50 backup-dir-backup-2024-06-27-16-50-11.tar.gz
-rw-r--r--. 1 root root   192 Jun 27 17:40 backup-dir-backup-2024-06-27-17-40-39.tar.gz
-rw-r--r--. 1 root root 6662815 Jun 27 16:42 etc-and-backup-dir-backup-2024-06-27-16-42-07.tar.gz
-rw-r--r--. 1 root root 6662815 Jun 27 16:50 etc-and-backup-dir-backup-2024-06-27-16-50-02.tar.gz
-rw-r--r--. 1 root root 6662815 Jun 27 16:52 etc-and-backup-dir-backup-2024-06-27-16-52-01.tar.gz
-rw-r--r--. 1 root root 6662815 Jun 27 16:54 etc-and-backup-dir-backup-2024-06-27-16-54-02.tar.gz
[root@target hasil]# cd /hasil/
[root@target hasil]# tar -zxvf backup-dir-backup-2024-06-27-17-40-39.tar.gz
tar: Removing leading '/' from member names
/backup/
/backup/inifiletxt.txt
[root@target hasil]# cat /hasil/backup/inifiletxt.txt
INI FILE YANG AKAN DI BACKUP
[root@target hasil]#
```

Gambar 14. Hasil Backup

Sumber : Penelitian 2024

Pada gambar ke 14 kita bisa melihat hasil dari *backup* direktori yang telah di buat. kita juga bisa mengekstrak *file* hasil *backup*-an kita dengan cara `tar -zxvf namafile`. *option z* berfungsi sebagai memberitahu tar untuk menyaring arsip melalui *gzip*, *option x* memberikan instruksi agar *filenya* diekstrak, *option v* singkatan dari "*verbose*," yang membuat tar mencantumkan setiap *file* yang diekstrak, *Option f* menunjukkan bahwa argumen berikutnya (nama *file* yang akan di ekstak).

KESIMPULAN DAN REKOMENDASI

Dalam menghadapi penggunaan *bash script*, membangun *backup* data penting secara otomatis menggunakan *bash script* pada *Red Hat Enterprise Linux*. Script *bash* digunakan untuk melakukan backup data secara berkala, sedangkan *crontab* digunakan untuk menjalankan *script* secara otomatis pada waktu yang ditentukan. Hasil *backup* disimpan pada *device* lain yang bernama *target* dengan metode yang digunakan dalam penelitian ini adalah eksperimental dengan menggunakan *Red Hat Enterprise Linux*. *Script bash* yang dikembangkan dapat melakukan *backup* data secara lengkap dan akurat. *Script* ini juga dapat dijalankan secara otomatis pada waktu yang ditentukan dengan menggunakan *crontab*. Hasil *backup* disimpan pada *device target* dengan aman dan terjamin serta penelitian ini menunjukkan bahwa membangun *backup* data penting secara otomatis menggunakan *bash script* pada *Red Hat Enterprise Linux* merupakan solusi yang efektif dan efisien. Metode ini dapat membantu organisasi dan individu untuk melindungi data mereka dari kehilangan data yang tidak disengaja atau disengaja.

REFERENSI

- Arnomo, I. (2019). Simulasi Backup Dan Restore Database Repository. *Jurnal Sistem Informasi, Teknologi Informasi Dan Komputer*, 9(2), 92–99.
- Chris F. A. Johnson, J. V. (2015). *Pro Bash Programing : Scripting the GNU/Linux Shell*. In Apress (Vol. 4, Issue 1).
- Christanto, F. W., & Rudiyanto, R. (2020). Cron Job Technique pada Integrasi WLAN Controller Device dan Google Maps API Berbasis Website dalam Jaringan Indonesia Wifi. *Matrix : Jurnal Manajemen Teknologi Dan Informatika*, 10(2), 50–57. <https://doi.org/10.31940/matrix.v10i2.1477>
- Farsi, M, Ali, M, Shah, RA, Wagan, AA, & ... (2020). Cloud computing and data security threats taxonomy: A review. *Journal of Intelligent ...*, content.iospress.com, <https://content.iospress.com/articles/journal-of-intelligent-and-fuzzy-systems/ifs179539>
- Guide, C. S. O., & Linux, S. (n.d.). *LINUX Command-Line for Beginners* Sam Chris.
- Kerrisk, M (2022). *crontab (5)-Linux manual page*.

- Kolenko, P (2019). baSHELiXir: A Bash Script for Automated Experimental Phasing Using Shelx C/D/E. *Chemicke Listy, ... Lavka 5, Prague 6 116 68 ...*
- Ludi, I, & Haidar, M Ahmad (2020). Implementasi Penjadwalan Backup Database Postgresql Menggunakan Crontab Berbasis Linux Dengan Metode Time Based repository.binadarma.ac.id, <http://repository.binadarma.ac.id/id/eprint/1119>
- Nurhanif, N, & Maizi, Z (2019). Pembuatan Peta Jaringan Untuk Memonitoring Koneksi Komputer Menggunakan Pemrograman Bash Script. *Journal Of Informatics And ...*, jurnal.uui.ac.id, <http://jurnal.uui.ac.id/index.php/jics/article/view/557>
- Pignalberi, G (2019). Uno script bash di ausilio alla redazione di manoscritti. *ArsTeXnica*, guitex.org, https://guitex.org/home/images/ArsTeXnica/AT028/arstexnica28_low_res.pdf#page=148
- Ramey, C., & Fox, B. (2022). GNU Bash Manual - Bash Builtins (Issue September). https://www.gnu.org/software/bash/manual/html_node/Bash-Builtins.html
- Richter, L. E., Carlos, A., Beber, D. M., & Michael W Lucas. (2017). *SSH Mastery : Second Edition*.
- Shotts Jr, W. E. (2019). *The linux command line: {A} complete introduction*
- Smyth, N. (2019). *Red Hat Enterprise Linux 8 Essentials*. In Payload Media, Inc.
- Taylor, N, & Schneider, K (2024). JCR 20003-UR Through Crontab. *Authorea Preprints*, staging.essopenarchive.org, <https://doi.org/10.22541/advance.171717276.65115131/v1>