**LAPORAN PRAKTIKUM**

**SISTEM OPERASI**



Modul 3

Asfal Fuad

240306031

Program Studi Teknologi Informasi

Fakultas Dakwah dan Ilmu Komunikasi

Universitas Islam Negeri Mataram

2025

|  |  |
| --- | --- |
| Pertemuan | 3 |
| Topik | Shell Scripting dan Automasi |
| Repository |  |
| Tanggal |  |

1. Tujuan

Tujuan praktikum Shell Scripting Dan Automasi

* 1. Memahami Dasar-dasar Shell Scripting
  2. Mengotomatisasi Tugas-tugas Berulang
  3. Meningkatkan Keterampilan Pemrograman Sistem
  4. Mengoptimalkan Proses Administrasi Sistem
  5. Peningkatan Keamanan Sistem

1. Requirement
   1. Sistem Operasi yang digunakan : MacOS Big Sur 11.2.3
   2. Browser : Google Chrome Version

129.0.6668.89 (Official Build)  
 (x86\_64)

* 1. Tools yang digunakan : Virtual Box

1. Dasar Teori

Definisi Shall Scripting dan Automasi

Shell scripting dalam lingkungan Linux merupakan alat yang sangat berharga untuk mengotomatisasi berbagai tugas, mulai dari pengelolaan file sederhana hingga administrasi sistem yang kompleks. Pengantar tentang shell scripting menekankan kemampuannya untuk memungkinkan interaksi langsung dengan sistem operasi, melewati antarmuka grafis dan menawarkan alur kerja yang lebih efisien, terutama untuk tugas-tugas yang berulang, pengelolaan berkas, dan pemrosesan teks. Fitur utama dari shell scripting mencakup keunggulannya dalam pemrosesan teks dan manajemen file, di mana alat-alat seperti grep, sed, awk, dan cut dimanfaatkan untuk memanipulasi dan mengekstrak data secara efisien. Selain itu, shell scripting memungkinkan otomatisasi tugas-tugas rutin seperti pencadangan, pemantauan sistem, dan konfigurasi jaringan, yang secara signifikan menghemat waktu dan mengurangi upaya manual. Fleksibilitasnya juga ditingkatkan dengan kemampuannya untuk berintegrasi dengan skrip dari bahasa pemrograman lain seperti Python, Perl, dan Ruby.

Bahasa Scripting yang Umum Digunakan dalam Automasi

* 1. Bash (Bourne Again Shell): Salah satu bahasa scripting yang paling umum digunakan pada sistem operasi berbasis Unix/Linux. Bash digunakan untuk menulis skrip yang mengelola tugas-tugas sistem, seperti backup data, instalasi perangkat lunak, atau pemeliharaan server.
  2. PowerShell: PowerShell adalah shell scripting yang dikembangkan oleh Microsoft, khusus untuk sistem operasi Windows. PowerShell memungkinkan pengguna untuk menjalankan perintah yang sangat kuat dan mengotomatiskan berbagai

tugas administratif, mulai dari manajemen file, pengelolaan server, hingga integrasi dengan layanan cloud seperti Azure.

* 1. Python: Python dikenal sebagai bahasa pemrograman yang sangat fleksibel dan mudah dipahami, sehingga banyak digunakan untuk automasi di berbagai platform. Python digunakan dalam banyak aplikasi automasi, dari tugas-tugas sederhana hingga automasi yang lebih kompleks, termasuk pengelolaan server dan otomatisasi proses bisnis.

Manfaat Automasi dan Scripting

* + 1. Efisiensi Waktu dan Sumber Daya
       - Automasi memungkinkan proses yang biasanya memerlukan waktu lama dilakukan secara otomatis, tanpa campur tangan manusia. Sebagai contoh, sebuah skrip dapat digunakan untuk secara otomatis mengunduh file, mengubah format, dan mengunggahnya ke server, yang akan menghemat banyak waktu dibandingkan melakukannya secara manual.
    2. Pengurangan Kesalahan Manusia
       - Dengan melakukan tugas secara otomatis melalui skrip, peluang terjadinya kesalahan manusia berkurang. Tugas-tugas yang mengulang, seperti pemeliharaan database atau backup file, bisa dilakukan tanpa risiko kelalaian manusia.
    3. Peningkatan Konsistensi
       - Skrip memastikan bahwa proses yang sama dilakukan dengan cara yang sama setiap kali, menjaga standar kualitas dan keandalan sistem yang lebih konsisten.
    4. Scalability
       - Dengan automasi, tugas yang dilakukan bisa diterapkan ke banyak sistem sekaligus, yang jauh lebih sulit dilakukan dengan cara manual. Misalnya, pengelolaan patch perangkat lunak pada ratusan server yang dapat dilakukan hanya dengan menulis skrip dan menjalankannya pada semua mesin yang terhubung.

Jenis-jenis Tugas yang Dapat Diotomatisasi

1. Administrasi Sistem
   * Sebagai administrator sistem, banyak tugas yang dapat diotomatisasi, seperti memonitor kesehatan server, memperbarui perangkat lunak, membersihkan log file yang tidak diperlukan, dan bahkan melakukan reboot otomatis pada server.
2. Pengujian Perangkat Lunak
   * Dalam pengembangan perangkat lunak, automasi pengujian adalah hal yang sangat penting. Skrip dapat dibuat untuk menjalankan pengujian unit, pengujian fungsional, atau pengujian integrasi secara otomatis, memastikan kualitas perangkat lunak tetap terjaga tanpa harus dilakukan secara manual setiap saat.
3. Manajemen Konfigurasi
   * Dalam lingkungan cloud atau server yang dinamis, penggunaan alat seperti Ansible, Chef, atau Puppet memungkinkan pengelolaan konfigurasi server secara otomatis di seluruh infrastruktur IT tanpa intervensi manual.
4. Deployment Aplikasi
   * Automasi deployment aplikasi juga menjadi salah satu penggunaan yang sangat penting dalam pengembangan perangkat lunak. Dengan menggunakan skrip dan alat automasi seperti Jenkins, GitLab CI, atau CircleCI, pengembang dapat mengotomatiskan proses pembuatan, pengujian, dan pengiriman aplikasi ke lingkungan produksi.

Prinsip-Prinsip Dasar dalam Menulis Skrip Automasi

1. Modularitas
   * Skrip yang baik harus dapat dibagi menjadi bagian-bagian kecil yang mudah dikelola dan dipelihara. Ini memungkinkan Anda untuk menambahkan atau memodifikasi bagian skrip dengan cara yang tidak memengaruhi keseluruhan sistem.
2. Dokumentasi
   * Setiap skrip harus dilengkapi dengan dokumentasi yang jelas. Hal ini sangat penting bagi siapa pun yang akan membaca atau memodifikasi skrip tersebut di masa mendatang. Dokumentasi yang baik membantu menjaga agar skrip tetap dapat digunakan dan dipahami oleh tim lain.
3. Error Handling
   * Skrip yang baik harus memiliki pengendalian kesalahan yang memadai. Ini berarti bahwa jika terjadi kesalahan, skrip harus dapat menangani kesalahan tersebut dengan cara yang aman (seperti log kesalahan atau pemberitahuan) dan tidak menyebabkan kegagalan sistem secara keseluruhan.
4. Pengujian dan Pemeliharaan
   * Skrip tidak hanya harus diuji, tetapi juga harus dirawat secara berkala. Ketika sistem atau perangkat lunak yang digunakan berubah, skrip perlu diperbarui untuk memastikan tetap berfungsi dengan baik.

Alat dan Framework untuk Automasi

1. Ansible: Alat yang sangat populer untuk manajemen konfigurasi dan automasi infrastruktur. Ansible memungkinkan untuk mengelola perangkat keras dan perangkat lunak dalam jumlah besar menggunakan kode deklaratif yang mudah dipahami.
2. Puppet: Sebuah alat untuk automasi pengelolaan konfigurasi yang digunakan untuk menjaga agar semua server dalam kondisi yang konsisten dan mengelola pembaruan perangkat lunak secara otomatis.
3. Jenkins: Platform automasi yang banyak digunakan untuk mengelola pipeline CI/CD (Continuous Integration/Continuous Deployment). Jenkins memungkinkan

untuk otomatisasi build, testing, dan deployment aplikasi dalam pengembangan perangkat lunak.

Tantangan dan Pertimbangan dalam Automasi

Meskipun banyak keuntungan dari automasi, ada beberapa tantangan yang perlu dipertimbangkan:

1. Keamanan: Skrip yang mengotomatiskan tugas sistem sangat rentan terhadap kesalahan yang bisa mengakibatkan kerusakan jika tidak diprogram dengan hati- hati. Skrip yang tidak aman bisa mengekspos sistem pada risiko, terutama dalam lingkungan yang sensitif seperti produksi.
2. Pengelolaan dan Pemeliharaan: Setelah automasi diterapkan, sistem dan skrip yang ada harus terus dipelihara, terutama ketika ada pembaruan teknologi atau perubahan pada infrastruktur yang digunakan.

Menulis Shell Script Sederhana

Shell script merupakan kumpulan perintah yang ditulis dalam sebuah file teks dan dieksekusi oleh shell seperti Bash atau Zsh. Tujuan utama dari shell script adalah untuk mengotomatisasi tugas-tugas yang bersifat repetitif. Struktur dasar sebuah shell script umumnya dimulai dengan baris shebang #!/bin/bash yang memberitahu sistem operasi untuk menggunakan Bash sebagai interpreter perintah. Baris-baris selanjutnya dalam script berisi perintah-perintah yang akan dieksekusi secara berurutan. Komentar dapat ditambahkan menggunakan tanda pagar (#) dan tidak akan dieksekusi oleh shell. Perintah echo sering digunakan untuk menampilkan teks atau output ke layar.

Variabel dalam shell script berfungsi untuk menyimpan nilai yang dapat digunakan kembali di dalam script. Untuk mendeklarasikan sebuah variabel, Anda dapat menuliskan nama variabel diikuti dengan tanda sama dengan (=) dan nilai yang ingin disimpan, tanpa adanya spasi di sekitar tanda sama dengan. Untuk mengakses nilai dari sebuah variabel, Anda perlu menambahkan tanda dolar ($) di depan nama variabel. Penggunaan kurung kurawal (${nama\_variabel}) disarankan untuk menghindari potensi ambiguitas, terutama ketika variabel diikuti oleh karakter lain.

Pernyataan kondisional memungkinkan script untuk mengambil keputusan berdasarkan evaluasi suatu kondisi. Struktur dasar pernyataan if terdiri dari if [ kondisi ], diikuti oleh then dan blok kode yang akan dieksekusi jika kondisi benar. Anda juga dapat menggunakan elif [ kondisi ] untuk kondisi alternatif dan else untuk blok kode yang dieksekusi jika semua kondisi sebelumnya salah. Blok kondisi diakhiri dengan fi. Operator perbandingan numerik seperti -eq, -ne, -gt, -ge, -lt, dan -le digunakan untuk membandingkan nilai angka, sementara operator perbandingan string seperti ==, !=, -z, dan -n digunakan untuk membandingkan string.

Perulangan memungkinkan Anda untuk mengeksekusi blok kode secara berulang kali. Terdapat beberapa jenis perulangan dalam shell script, di antaranya adalah for, while, dan until. Perulangan for biasanya digunakan untuk mengulangi sekumpulan item, seperti daftar string atau rentang angka. Perulangan while akan terus menjalankan blok

kode selama kondisi yang ditentukan bernilai benar. Sebaliknya, perulangan until akan terus menjalankan blok kode sampai kondisi yang ditentukan menjadi benar.

Mengotomatisasi Tugas Menggunakan Cron Job

cron adalah sebuah daemon atau layanan latar belakang pada sistem operasi mirip Unix yang memungkinkan penjadwalan perintah atau script untuk dieksekusi secara otomatis pada waktu dan tanggal tertentu. Konfigurasi jadwal cron disimpan dalam file yang disebut crontab, dan setiap pengguna pada sistem memiliki crontab mereka sendiri. Untuk mengelola crontab, Anda dapat menggunakan perintah crontab -l untuk melihat daftar pekerjaan terjadwal, dan crontab -e untuk mengedit crontab menggunakan editor teks.

Setiap baris dalam file crontab merepresentasikan sebuah pekerjaan terjadwal dan memiliki format yang terdiri dari lima bagian waktu (menit, jam, hari bulan, bulan, hari minggu) diikuti oleh perintah yang akan dijalankan. Nilai untuk setiap bagian waktu memiliki rentang tertentu, dan Anda juga dapat menggunakan karakter wildcard (\*) untuk menunjukkan semua nilai dalam rentang tersebut. Contohnya, 0 3 \* \* \*

/path/ke/backup.sh akan menjalankan script backup.sh setiap hari pada pukul 3 pagi. Penting untuk memastikan bahwa script yang dijalankan oleh cron memiliki izin eksekusi (chmod +x) dan menggunakan path absolut untuk perintah dan file. Output dari script cron biasanya dikirim melalui email kepada pengguna yang menjalankan cron, namun Anda juga dapat mengarahkannya ke file log jika diperlukan menggunakan operator pengalihan output seperti > dan 2>&1.

Membaca Log Sistem dan Melakukan Debugging pada Shell Script

Log sistem adalah catatan berbagai peristiwa yang terjadi pada sistem operasi dan aplikasi, dan memahaminya sangat penting untuk memecahkan masalah. File-file log umum terletak di direktori /var/log/, seperti /var/log/syslog atau /var/log/messages untuk log sistem umum, /var/log/auth.log untuk log otentikasi, dan direktori lain untuk log aplikasi spesifik. Beberapa perintah umum yang digunakan untuk membaca log termasuk cat untuk menampilkan seluruh isi file, less untuk navigasi yang lebih baik, tail untuk melihat baris terakhir (termasuk opsi -f untuk mengikuti secara real-time), head untuk melihat baris pertama, dan grep untuk mencari pola tertentu dalam log. Pada sistem yang menggunakan systemd, perintah journalctl menyediakan cara yang lebih terstruktur untuk melihat dan memfilter log sistem.

Debugging shell script adalah proses mengidentifikasi dan memperbaiki kesalahan dalam script Anda. Beberapa teknik debugging yang efektif meliputi penggunaan perintah echo untuk mencetak nilai variabel dan pesan status di berbagai bagian script, menjalankan script dalam mode debug Bash menggunakan opsi -x (bash -x nama\_script.sh atau set -x di dalam script), yang akan menampilkan setiap perintah yang dieksekusi.

































































