Encrypted Backup and Restore Bash Script Design Document

Contents

Overview

The provided Bash script is designed to streamline the creation of encrypted backups and the subsequent restoration process. This functionality is achieved through the script's utilization of fundamental file manipulation commands, OpenSSL encryption, and tar compression. The script's development has been guided by specific design choices and underlying assumptions, which are detailed in this document to provide insight into its operation and considerations.

Furthermore, this document addresses an expansion of the script's capabilities to encompass remote server operations through the use of the Secure Copy Protocol (scp). This enhancement empowers users to securely transfer backup files between local and remote machines, broadening the scope of the script's applicability.

Script Purpose

The script serves two primary purposes:

- Backup: It creates encrypted backups of specified directories and files.
- Restore: It restores encrypted backup files to a specified directory after decrypting them.

Design Decisions and Assumptions

Function: print_color

- **Purpose:** Print colored messages to the console for better visual distinction.
- **Design Decision:** Color codes are used to visually differentiate message types.
- **Assumption:** The terminal supports ANSI escape codes for color formatting.

Function: validate_backup_params

- **Purpose:** Validate backup parameters and ensure the required conditions are met.
- **Design Decision:** Parameters include source directory, backup directory, encryption key, and number of days.
- **Assumption:** Users provide the correct number and format of parameters.

Function: backup

- **Purpose:** Perform the backup process.
- **Design Decision:** Directories are backed up separately, and then individual files are backed up within a specified time range.
- **Assumption:** The source directory and backup destination exist, encryption key is valid, and backup is possible.

Function: validate_restore_params

- **Purpose:** Validate restore parameters and ensure the required conditions are met.
- **Design Decision:** Parameters include backup directory, restore directory, and decryption key.
- **Assumption:** Users provide the correct number and format of parameters.

Function: restore

- **Purpose:** Restore encrypted backup files to the restore directory.
- **Design Decision:** Encrypted files are decrypted and extracted to the temporary directory before being moved to the restore directory.
- **Assumption:** The backup directory contains encrypted files, the decryption key is valid, and restore is possible.

Encryption Algorithm

- **Design Decision:** OpenSSL is used to perform encryption and decryption using the AES-256-CBC algorithm with PBKDF2 key derivation and salt.
- **Assumption:** Users have OpenSSL installed and are familiar with the encryption process.

Backup and Restore File Naming

- **Design Decision:** Backed up directories are named with the current date and time. Encrypted files are named using the original directory name and the timestamp.
- **Assumption:** File and directory names are valid and adhere to naming conventions.

Temporary Directory in Restore Phase

- **Design Decision:** A temporary directory is created during the restore process to temporarily hold the decrypted and extracted files before moving them to the restore directory.
- **Assumption:** Sufficient disk space is available for creating temporary directories during the restore process.

Remote Backup

- **Design Decision:** After creating local backups, the script will use the scp command to transfer backup files to a remote server.
- **Assumptions:**
    - Users have SSH access to the remote server.
    - Remote directory exists or can be created.
    - Users have provided valid remote server details (hostname, username, directory).

Assumptions and Limitations

1. Users are responsible for ensuring the script's execution environment supports the required commands and tools (e.g., tar, OpenSSL).
2. The script is designed for basic use cases and may require customization for advanced scenarios.
3. Error messages and exit codes are used to handle errors and terminate the script.
4. The script doesn't handle cases of complex file names with spaces or special characters.
5. No specific logging mechanism is implemented; all output is sent to the console.

Conclusion

The provided Bash script offers a solution for creating encrypted backups and restoring them while adhering to specified parameters. By understanding the design decisions and assumptions outlined in this document, users can confidently utilize the script for their backup and restore needs. However, customization and adaptation might be necessary based on specific use cases and environments.