Design Document

Secure Backup and Restore Script

Introduction

The provided Bash script is designed to handle backup and restore operations for directories and files. It includes functions to validate parameters, perform backups, and restore backups. This document explains the design decisions and assumptions made in the script's development.

Script Structure

The scripts are structured as follows:

- `backup.sh` - Main entry point for backup
- `restore.sh` - Main entry point for restore
- `backup_restore_lib.sh` - Contains shared utility functions
- The main scripts source the library

Assumptions

1. The script assumes that the user is familiar with using a command-line interface and runs a Unix-like operating system where Bash scripting is available.
2. The script assumes that the user has execution permission to run without errors, if not, please run this command: `chmod +x backup.sh restore.sh backup_restore_lib.sh`
3. Backup and restore operations involve encryption and decryption, so the script assumes that the `gpg` command-line tool is installed and configured with the required keys.
4. For remote backup, the script assumes that the user has access to the `scp` command for secure file transfer.
5. All backup and restore directories are local to the machine running the script, and remote hosts are accessed via SSH.

Design Decisions

Function `validate_backup_params()`

This function is designed to validate the parameters provided for a backup operation. It checks the number of arguments, ensures the existence of source and destination directories, and validates the encryption key and days parameter.

Function `validate_restore_params()`

Similar to `validate_backup_params()`, this function validates parameters for a restore operation. It checks the number of arguments, verifies the existence of backup and restore directories, and validates the decryption key.

Function `backup()`

This function is responsible for performing backup operations. It takes the source directory, backup directory, encryption key, and days as input parameters. The backup process is divided into two sections:

1. **Dealing with Directories**: For each directory within the source directory, the script identifies directories that have been modified within the specified number of days. It creates compressed and encrypted archives for these directories using the `tar` and `gpg` commands. The original compressed files are deleted after encryption.

2. **Dealing with Files**: This section handles loose files within the source directory. It first creates a tar archive containing the first file found. Then, it iterates through all files, excluding directories, and adds them to the existing tar archive. Finally, the tar file is compressed, encrypted, and the original files are deleted.

After performing the backup, the script prompts the user to specify a remote host. If a valid host is provided, the script validates its reachability and then prompts the user for a remote directory. The backup is then copied to the remote host using `scp`.

Function `restore()`

The `restore()` function is responsible for restoring files from encrypted backup archives. It takes the backup directory, restore directory, and decryption key as input parameters. The function creates a temporary directory to hold decrypted files and proceeds with the following steps:

1. It decrypts each `.gpg` backup file in the backup directory and places the decrypted files into the temporary directory.

2. It iterates through the decrypted files, which are assumed to be compressed tar archives, and extracts their contents into the restore directory.

After restoring the files, the temporary directory is removed.

Conclusion

The provided Bash script is designed to facilitate backup and restore operations for directories and files. The design decisions and assumptions outlined in this document provide context for the script's intended usage and functionality. Users should ensure that the script is used in a compatible environment and with the required tools installed for encryption, decryption, and secure file transfer.