Assignment 2: A safe backup/parameterised script

Goal: Write a robust, parameterised Bash script that backs up a directory into a timestamped compressed archive, validates its inputs, is safe to re-run, and cleans up after itself.

Where: On the Jumpbox (`10.100.100.254`, Ubuntu, user `ubuntu`). Create `~/backup.sh` and back up something harmless you own, such as `~/scripts` or a test folder you create.

Tasks

1. Create `~/backup.sh` with the shebang `#!/usr/bin/env bash`, the safety header `set -euo pipefail`, and a header comment showing usage: `Usage: ./backup.sh <source-dir> [dest-dir]`.
2. **Arguments & validation:** Read the source directory from `$1` and an optional destination from `$2` (default the destination to `~/backups`). If no source is given, print the usage line to stderr and `exit 1`. If the source is not an existing directory (`-d`), print an error to stderr and exit non-zero.
3. Add a `die()` helper that prints `ERROR: <message>` to stderr and exits 1, and use it for your validation messages.
4. **Idempotent destination:** Create the destination directory with `mkdir -p "$dest"` so re-running never errors.
5. **Timestamped archive:** Build an archive name that includes the source's basename and a timestamp, e.g. `scripts-2026-06-01_142530.tar.gz` (hint: `date +%F_%H%M%S`). This guarantees each run makes a new file rather than overwriting an old backup.
6. **Create the backup:** Use `tar -czf "$dest/$archive" -C "$parent" "$base"` to compress the source. Capture/print the resulting file size (e.g. `du -h "$dest/$archive"`).
7. **trap cleanup:** If you build the archive via a temporary file or staging area, register `trap 'rm -f "$tmp"' EXIT` so partial files are cleaned up if the script fails partway.
8. **Report:** Print a clear success line: `Backup complete: <path> (<size>)`. Exit `0` on success.
9. Make it executable, run it against a real folder, then run it a second time to prove it is idempotent (no errors, a new timestamped file appears).
10. Run `shellcheck ~/backup.sh` and resolve all warnings.

Deliverable

The file `~/backup.sh` on the Jumpbox: executable, ShellCheck-clean, parameterised, input-validated, idempotent, with a `trap` and a `die()` helper. Show your mentor the directory listing of `~/backups` after running the script twice (two timestamped archives).

Acceptance criteria — you're done when:

- ☐ The script starts with `#!/usr/bin/env bash` and `set -euo pipefail`.
- ☐ Running it with no arguments prints a usage message to stderr and exits non-zero.
- ☐ Running it with a non-existent source directory prints an error to stderr and exits non-zero.
- ☐ A valid run creates `~/backups` if missing and writes a `.tar.gz` whose name contains the source name and a timestamp.
- ☐ Running the script twice in a row succeeds both times and produces two distinct archive files (proving idempotency).
- ☐ The archive actually contains the source files (verify with `tar -tzf <archive> | head`).
- ☐ A `trap` is registered for cleanup, and a `die()` helper handles error exits.
- ☐ `shellcheck ~/backup.sh` reports no warnings, and every variable is double-quoted.

Hints

- Default the destination: `dest="${2:-$HOME/backups}"`.
- Split a path: `base="$(basename "$src")"` and `parent="$(dirname "$src")"`; then `tar -C "$parent" "$base"` archives clean relative paths instead of absolute ones.
- Make a safe temp file with `tmp="$(mktemp)"` and remove it via the trap — never hardcode `/tmp/backup`.
- Verify an archive without extracting: `tar -tzf "$dest/$archive" | head`.
- Re-read Lesson 4 (arguments and functions) and Lesson 5 (set -euo pipefail, trap, idempotency, ShellCheck).
- Test on a throwaway folder first (`mkdir -p ~/testdir && touch ~/testdir/a ~/testdir/b`) so you never risk real data.

Blocked for more than ~30 minutes after re-reading the lessons? Bring what you've tried to your mentor.

Revision #3
Created 2026-05-15 15:31:00 UTC by Eslam Shapsough
Updated 2026-05-15 15:35:00 UTC by Eslam Shapsough