Terraform Cli Output with local-exec

Terraform changes the CLI output because of the way it internally handles strings, commands, and logging. The key reasons why the CLI output looks different are:

1. Terraform's String Escaping

Terraform treats `local-exec` commands as strings. To ensure proper parsing and execution, Terraform escapes characters like `\n` and quotes (`"`) in the logs. This behavior is intrinsic to how Terraform represents and executes commands, particularly for multi-line strings.

---

2. Why Can't Terraform Avoid This?

- **Portability**: Terraform needs to ensure that the commands work across various shells, operating systems, and configurations.
- **Consistency**: Terraform outputs a sanitized, escaped version of your command to avoid ambiguity in how the command is interpreted by the shell.
- **HCL Design**: HashiCorp Configuration Language (HCL) is designed to work with string interpolation and templating. This leads to Terraform always representing multi-line commands as escaped strings.

Unfortunately, this behavior cannot be overridden or turned off directly.

---

3. Workarounds to Control the Output

While Terraform will always log the command it runs in an escaped format, you can make the output more readable **after execution**. Here are two approaches to minimize the impact:

Option 1: Use a Logging Step

Manually log the commands in their intended format by including `echo` or using a file-based approach.

```
provisioner "local-exec" {
  command = <<-EOT
    echo "Running the following script:"; \
    echo "if [ '${data.aws_instance.ejb-
webserver.instance_state}' = 'running' ];
then"; \
    echo "  echo 'EC2 instance is now in
running state! Waiting for around 2
minutes...'"; \
    echo "  sleep 40"; \
    echo "else"; \
    echo "  echo 'EC2 instance is not in
the running state. Exiting with error.'"; \
    echo "  exit 1"; \
    echo "fi"; \
    if [ "${data.aws_instance.ejb-
webserver.instance_state}" = "running" ];
then echo "EC2 instance is now in running
state! Waiting for around 2 minutes...";
sleep 40; else echo "EC2 instance is not in
the running state. Exiting with error.";
exit 1; fi
  EOT
}
```

This will log each part of your script before execution, ensuring a readable output in the CLI.

Option 2: Write the Script to a Temporary File

Instead of directly running the commands, write them to a temporary file and execute the file. The logged file path and its contents will be easier to read.

```
provisioner "local-exec" {
  command = <<-EOT
    temp_script=$(mktemp)
    echo "Writing script to $temp_script"
    cat <<'EOF' > $temp_script
    if [ "${data.aws_instance.ejb-
webserver.instance_state}" = "running" ];
then
      echo 'EC2 instance is now in running
state! Waiting for around 2 minutes...'
      sleep 40
    else
      echo 'EC2 instance is not in the
running state. Exiting with error.'
      exit 1
    fi
    EOF
    echo "Executing script:"
    cat $temp_script
    bash $temp_script
    rm -f $temp_script
  EOT
  interpreter = ["/bin/bash", "-c"]
}
```

4. Why Doesn't Terraform Change This Behavior?

- Terraform's primary goal is **state management and resource provisioning**, not being a scripting tool. The `local-exec` and `remote-exec` provisioners are meant for specific, limited use cases and are not optimized for complex scripting or human-readable logging.
- Terraform assumes that the actual command output (what the command prints) is more important than the command's representation in logs.

If the readability of commands in logs is critical, consider using a configuration management tool like Ansible, which provides better control over script formatting and logging.