
Devops Shack
TOP 200
SHELL SCRIPTING
DEBUG
COMMANDS
By Devops Shack

Click here for DevSecOps and Cloud DevOps Course

DevOps Shack

Top 200 Shell Scripting Debug Commands

1. Shell Debugging Fundamentals

1. `set -x` → Enables tracing; prints each command before execution.
   Example: `set -x; echo "Debugging"; set +x`

2. `set -e` → Stops script when any command fails (exit code ≠ 0).
   Use when reliability > tolerance for failure.

3. `set -u` → Treats unset variables as an error.
   Catches typos like ***$varible.***

4. `set -o pipefail` → Makes pipelines fail if **any** command fails.
   Example: `set -o pipefail; grep foo file | sort`

5. `trap 'echo "Error at line $LINENO"' ERR` → Print message when any command errors.

6. `trap 'echo "Exiting script..."' EXIT` → Always runs cleanup before exit.

7. `trap 'echo "Running command: $BASH_COMMAND"' DEBUG` → Prints every line before execution.

8. `PS4='+ ${BASH_SOURCE}:${LINENO}:${FUNCNAME[0]}() : '` → Improves `set -x` trace with file and line number.

9. `bash -x script.sh` → Runs entire script with trace.

10. `bash -n script.sh` → Syntax check without executing.

2. Variable & Environment Inspection

11. `declare -p var` → Show value and type of variable.

12. `typeset -p PATH` → Same as declare.

13. `export -p` → List exported environment variables.

14. `set` → Print all variables and functions (huge output).
15. `env` → Print only environment variables.
16. `readonly -p` → List readonly variables.
17. `help` → Shows help for built-ins like `help test`.
18. `type echo` → Shows whether it's a builtin, alias, or external command.
19. `command -V ls` → POSIX-compliant "where" for command.
20. `alias` → List all current aliases.

3. Syntax, Logic & Sanity Checks

21. `shellcheck script.sh` → Static linting (must install).
22. `bash -n file.sh` → Syntax only.
23. `dash -n file.sh` → POSIX compliance test.
24. `time cmd` → Measure execution time.
25. `/usr/bin/time -v cmd` → Verbose runtime + memory stats.
26. `ulimit -a` → Shows process limits that may kill jobs.
27. `echo $?` → Shows exit code of last command.
28. `bash -v script.sh` → Verbose output (print lines before execution).
29. `trap 'echo "$BASH_COMMAND failed at $LINENO"' ERR` → Runtime error hook.
30. `set -Eeuo pipefail` → Combine best safety flags.

4. Files, Paths, and Permissions

31. `stat file` → File metadata (mtime, permissions, etc.).
32. `readlink -f file` → Resolve symlinks to absolute path.
33. `realpath .` → Canonical absolute directory.
34. `file file` → Identify file type.
35. `ls -l` → Check permissions quickly.

36. `id` → See current UID and GID.
37. `groups $USER` → Check group membership.
38. `umask` → See default permission mask.
39. `getfacl / setfacl` → ACL attributes.
40. `lsattr / chattr` → Extended attributes (immutable, append-only).

5. Process & Job Debugging

41. `ps -ef | grep process` → List running processes.
42. `pgrep -a name` → Get PID and full command.
43. `pstree -ap $$` → Show process tree.
44. `jobs -l` → Show background jobs.
45. `fg %1 / bg %1` → Bring job to foreground/background.
46. `disown %1` → Detach job from shell.
47. `strace -f -o trace.log ./cmd` → Trace system calls.
48. `ltrace ./cmd` → Trace library calls.
49. `pstack PID` → Show process stack.
50. `kill -SIGUSR1 PID` → Trigger manual signal handling for debugging.

6. Network & Connectivity Debugging

51. `ip addr` → Shows interfaces and IPs.
52. `ip route` → Routing table.
53. `ss -tulpn` → View open sockets with PIDs.
54. `ping -c 4 host` → Connectivity test.
55. `traceroute host` → Network path.
56. `dig +short domain.com` → DNS resolution.
57. `nslookup domain.com` → Alternative DNS test.

58. `curl -v URL` → HTTP(S) verbose debug.
59. `wget -S URL` → Show HTTP headers only.
60. `openssl s_client -connect host:443` → SSL handshake and certs.

7. Logs & Kernel Debugging

61. `dmesg -T | tail` → Latest kernel logs.
62. `journalctl -xe` → System logs around failures.
63. `tail -F /var/log/syslog` → Follow logs live.
64. `logger "test msg"` → Send test to syslog.
65. `grep -i error /var/log/*` → Quick error search.
66. `journalctl -u service` → Logs for a specific systemd unit.
67. `script -q trace.txt` → Record full terminal session.
68. `scriptreplay` → Replay recorded terminal output.
69. `journalctl --disk-usage` → Space usage by journal.
70. `rsyslogd -N1` → Validate syslog config.

8. Disk, I/O & Filesystem

71. `df -h` → Disk space usage.
72. `du -sh dir` → Folder size summary.
73. `find . -type f -size +100M` → Find large files.
74. `lsof +D /path` → Which process holds a file open.
75. `fuser -v /file` → Process using file/socket.
76. `inotifywait -m /tmp` → Watch live file events.
77. `sync` → Flush filesystem cache.
78. `lsblk -f` → Block devices and mount points.
79. `mount | column -t` → Check mounted FS.

80. `findmnt` → Nicely formatted mount list.

9. Input/Output & Buffer Debugging

81. `tee` → Display and log simultaneously.

82. `stdbuf -oL` → Line-buffered output for live streaming.

83. `timeout 10s cmd` → Kill hung commands.

84. `yes | cmd` → Auto-feed "yes" for testing.

85. `read -r var` → Raw read without backslash escapes.

86. `mapfile -t arr < file` → Load file into array.

87. `IFS=$'\\n\\t'` → Safe word splitting.

88. `xargs -t` → Print command before execution.

89. `tee >(grep ERROR > errors.txt)` → Split log stream.

90. `grep -nE "ERROR|WARN" log.txt` → Print line numbers for quick triage.

10. Safety, Traps & Resilience

91. `trap 'echo Cleaning up' EXIT` → Guaranteed cleanup.

92. `trap 'pkill -P $$' EXIT` → Kill all child processes on exit.

93. `set -m` → Enable job control in scripts.

94. `wait` → Wait for background jobs.

95. `ulimit -t 60` → Limit CPU time.

96. `ionice -c3 cmd` → Lower I/O priority.

97. `nice -n 10 cmd` → Lower CPU priority.

98. `retry(){ n=0; until "$@"; do ((n++)); sleep $((2**n)); [ $n -ge 5 ] && return 1; done; }`
    — Simple retry with exponential backoff.

99. `coproc` → Run background processes with bidirectional pipes.

100. `set -T` → Allow traps inside functions.

11. Text Debugging (grep, sed, awk, formatting)

101. `grep -nR pattern path` → Search recursively with line numbers.
    Example: `grep -nR "ERROR" /var/log`

102. `grep -oE 'regex' file` → Print only matching parts of each line.
    Example: `grep -oE '[0-9]{4}-[0-9]{2}-[0-9]{2}' log.txt`

103. `grep -A3 -B3 "keyword"` → Show context before/after match.
    Example: `grep -A2 -B2 "timeout" config.yaml`

104. `awk '{print NR, $0}' file` → Print line numbers for debugging.

105. `awk -F: '{print $1,$3}' /etc/passwd` → Print selected fields.

106. `awk 'length($0)>100' file` → Identify overly long lines.

107. `sed -n '1,20p' file` → Print only first 20 lines.

108. `sed -n 'l' file` → Show hidden non-printable chars (`\t`, `\r`).

109. `nl -ba file` → Add numbers to all lines including blanks.

110. `sort | uniq -c | sort -nr` → Find duplicate or most frequent lines.

12. Logic & Control Flow Debugging

111. `echo $LINENO` → Current script line (very handy in traps).

112. `caller` → Print call stack inside a function.

113. `return $?` → Return last exit code (keep exact status).

114. `true / false` → Known exit codes for testing branches.

115. `[ -z "$VAR" ]` → Check if variable empty.

116. `[ -f file ]` → File exists and is regular.

117. `[ -x file ]` → File exists and is executable.

118. `[[ -v VAR ]]` → Check if variable set (Bash).

119. `printf '%s\n' "$@"` → Safe print of all arguments.

120. `printf '%q\n' "$string"` → Show quoted version of a variable.

13. Resource & Performance Profiling

121. `ps -eo pid,cmd,%cpu,%mem --sort=-%cpu | head` → Find top consumers.
122. `vmstat 1` → CPU/memory/swap live view.
123. `iostat -xz 1` → Disk performance.
124. `free -h` → Memory summary.
125. `sar -u 1 3` → CPU usage snapshots.
126. `uptime` → Load averages at a glance.
127. `dstat` → Realtime system stats.
128. `pidstat -p PID` → CPU usage per process.
129. `time (cmd)` → Measure single command performance.
130. `/usr/bin/time -v cmd` → Detailed resource breakdown.

14. Security & Permissions

131. `sudo -l` → Show allowed commands.
132. `getenforce / sestatus` → SELinux mode.
133. `ls -Z` → Show SELinux labels.
134. `auditctl -l` → Active audit rules.
135. `lastlog` → Recent login info.
136. `w` → Who’s logged in right now.
137. `whoami` → Current user context.
138. `sudo su - otheruser` → Simulate user environment.
139. `groups` → Verify group permissions.
140. `capsh --print` → View Linux capabilities.

15. Packages, Dependencies & System Info

141. `uname -a` → Kernel version & architecture.
142. `cat /etc/os-release` → Distro details.
143. `lsb_release -a` → Alternative distro info.
144. `dpkg -S /bin/bash` → Package owning file (Debian).
145. `rpm -qf /bin/bash` → Same for RHEL/CentOS.
146. `apt list --installed | grep pkg` → Check installed package.
147. `apt-cache policy pkg` → Package version sources.
148. `yum info pkg` / `dnf info pkg` → Show package details.
149. `which -a cmd` → All occurrences of command.
150. `hash -r` → Clear shell's cached paths.

16. Cron, Services, and Timers

151. `crontab -l` → List current user's cron jobs.
152. `sudo crontab -l -u user` → Others' cron jobs.
153. `systemctl list-timers` → See all systemd timers.
154. `systemctl status service` → Check service status.
155. `journalctl -u service -b` → Logs for current boot.
156. `systemctl show -p ExecStart service` → Real command line.
157. `systemctl cat service` → Show full unit definition.
158. `run-parts --test /etc/cron.daily` → What cron will execute.
159. `atq` / `atrm` → Manage `at` scheduled tasks.
160. `systemd-analyze blame` → Show slow boot units.

17. Network & Internet Troubleshooting

161. `curl -I URL` → Fetch headers only.

162. `curl -v --connect-timeout 5 URL` → Verbose + connection timeout.
163. `curl -sSfo /dev/null URL` → Silent but shows errors.
164. `wget --spider URL` → Check reachability without downloading.
165. `dig +trace domain.com` → Trace full DNS resolution path.
166. `host domain.com` → Simple DNS lookup.
167. `nc -zv host port` → Check if port open.
168. `telnet host port` → Manual socket check (deprecated but quick).
169. `tcpdump -i any -nn port 80` → Trace HTTP traffic.
170. `nmap -p 22,80 host` → Quick open port scan.

18. Container / Docker Debug Commands

171. `docker ps -a` → List all containers.
172. `docker logs name` → Show logs.
173. `docker exec -it name bash` → Enter container shell.
174. `docker inspect name` → Detailed JSON metadata.
175. `docker inspect -f '{{.State.Status}}' name` → State only.
176. `docker top name` → Running processes.
177. `docker events` → Real-time Docker events.
178. `docker network inspect bridge` → Debug networking.
179. `docker system df` → Show space usage.
180. `docker exec -it name env` → Inspect container environment.

19. Advanced Debugging Patterns

181. `BASH_XTRACEFD=9; exec 9>trace.log; set -x` → Write trace to file.
182. `trap 'echo "Err at $LINENO: $BASH_COMMAND"' ERR` → Live error reporting.
183. `export PS4='+ ${BASH_SOURCE}:${LINENO}: '` → Line-aware xtrace.

184. `exec 3>&2 4>&1` → Backup standard streams.

185. `exec 2>error.log` → Redirect stderr globally.

186. `set -T` → Propagate traps to functions.

187. `declare -F` → List all functions.

188. `compgen -A function` → Same, shorter.

189. `dirs -v` → Show dir stack.

190. `pushd /tmp; popd` → Test directory stack navigation.

20. Misc, Recovery & Exit Diagnostics

191. `exit 0` / `exit 1` → Always end with explicit status.

192. `trap 'echo "Exit $?"' EXIT` → Report final exit code.

193. `date '+%F %T'` → Timestamp logs.

194. `mktemp -d` → Safe temporary dir.

195. `trap 'rm -rf "$TMPDIR"' EXIT` → Cleanup temp data.

196. `history | tail` → Review last commands.

197. `alias | grep custom` → Debug overridden aliases.

198. `printf '%(%F %T)T\n' -1` → Timestamp with `printf`.

199. `sleep 0.1` → Throttle tight loops.

200. `echo "Script completed successfully"` → Always close with a clear status.