HEXWORTH PRIME · INSTRUCTOR SOLUTION GUIDE

ALA Scavenger Hunt #1: The Website Is Down

Solution Guide

COURSE	**CTS4321C Advanced Linux Administration**
WEEK	**1**
FORMAT	**In-class team race + printed worksheet**
DURATION	**20 minutes (target)**
FLAGS	**12 (one per worksheet row)**
DIFFICULTY	**Beginner**
POINTS	**600 (50 per flag)**
WORKSHEET	**`~/hexworth-shared/images/advanced linux/scavengerHunt-website-down.docx`**

Objective

The lab is a small CTF wrapper over the ala-l01 engine, retuned so each scavenger-hunt worksheet row maps to one captured flag. The student runs the correct Linux command in the simulated terminal, the engine detects the command pattern, and `awardFlag('cmdN')` fires. The student writes the command on the printed worksheet for the matching row.

The activity is an **in-class team race**:

Team rank	Score
First team to capture all 12 flags AND fill the worksheet correctly	100/100
Other teams who finish correctly	95/100
Each wrong sheet answer	−2 pts

The box itself never enforces an order — students can capture the 12 flags in any sequence. The worksheet rows ARE numbered in incident-investigation order, however, so the natural play is to walk the chain.

Starting State

Inherited from `ala-l01-dead-cell-recovery`:

- Hostname: `cell-071`
- All five services failed (networking/sshd/cron/grid-sync/grid-monitor)
- Login banner asserts DEGRADED state and points at `systemctl status`, `ip link show`, `journalctl -xb`

The point of the lab in this format is NOT to fix the box (though doing so is encouraged — students will naturally run the right commands while fixing). The point is to demonstrate the 12 commands.

Flag-by-Flag Walkthrough

Flag 1 (cmd1) — Persistent session

Worksheet row: "Open a persistent terminal session that will survive if your SSH connection drops."

Command (any of):

```
tmux
tmux new
tmux new-session
tmux new -s incident
screen
```

What happens: Engine `tmux` (or `screen`) handler prints a stub session-created message. `awardFlag('cmd1')` fires.

Flag value: `FLAG{ala-hunt1_cmd01_persistent_session}`

Flag 2 (cmd2) — Service status

Worksheet row: "Show whether a systemd service is active, failed, or stopped, with a one-screen summary including recent log lines."

Command (any service name unlocks the flag):

```
systemctl status nginx
systemctl status grid-monitor
sudo systemctl status grid-sync
service nginx status
```

Trigger: `systemctl` handler with `sub === 'status'` AND `unit` non-empty.

Flag value: `FLAG{ala-hunt1_cmd02_service_status}`

Flag 3 (cmd3) — Last 30 log entries, no pager

Worksheet row: "Show the last 30 log entries from a specific service only, with no pager."

Command (must include `-u SVC` AND `-n 30` AND `--no-pager`):

```
journalctl -u nginx -n 30 --no-pager
journalctl -u grid-sync -n 30 --no-pager
journalctl --no-pager -u grid-monitor -n 30
journalctl -n 30 -u grid-sync --no-pager
```

Trigger: `journalctl` handler detects all three required flags present.

Flag value: `FLAG{ala-hunt1_cmd03_log_last_30}`

Flag 4 (cmd4) — Live-tail service log

Worksheet row: "Watch a service's log live as you make changes."

Command (must combine `-f` and `-u`, in any order or combined):

```
journalctl -fu nginx
journalctl -f -u grid-sync
journalctl -u grid-monitor -f
```

Trigger: `journalctl` handler detects both follow flag (`-f` or `-Xf` combined) AND `-u` flag.

Flag value: `FLAG{ala-hunt1_cmd04_log_live_follow}`

Flag 5 (cmd5) — Bring an interface up

Worksheet row: "Bring a downed network interface back up (e.g. eth1)."

Command:

```
sudo ip link set eth1 up
ip link set eth1 up
```

Trigger: `ip` handler with `sub === 'link'`, `obj === 'set'`, iface match (eth1), `updown === 'up'`.

Side effect: Also sets the engine's `_serviceState.networking = 'active'`, unblocking the rest of the dependency chain (useful for students fixing the box).

Flag value: `FLAG{ala-hunt1_cmd05_interface_up}`

Flag 6 (cmd6) — Show interfaces

Worksheet row: "Show every network interface and its state (UP/DOWN, MAC)."

Command:

```
ip link show
ip addr
ip a
ip -br addr
```

Trigger: `ip` handler with `sub === 'link' && obj === 'show'`, OR `sub === 'addr'` / `'address'`.

Flag value: `FLAG{ala-hunt1_cmd06_interface_show}`

Flag 7 (cmd7) — Listening TCP sockets

Worksheet row: "Show all listening TCP sockets, numeric ports, with the owning process name."

Command (any `ss` with `-l`):

```
ss -tlnp
ss -tln
sudo ss -tlnp
ss -tlp
ss -tunlp
```

Trigger: `ss` handler detects `-l` token among args. `-t` (TCP) + `-l` (listen) returns a richer output that shows the rogue python on port 443.

Flag value: `FLAG{ala-hunt1_cmd07_listening_sockets}`

Bonus learning: The output reveals `python3 (pid=12847)` holding port 443 — a clue mirroring the slide 24 narrative.

Flag 8 (cmd8) — All processes

Worksheet row: "Show the running processes for all users in full long-listing format."

Command:

```
ps -ef
ps aux
ps -aux
ps -e -f
```

Trigger: `ps` handler detects either `-ef` (or `-e` + `-f`) or `aux` / `-aux`.

Output includes: the rogue `python3 /tmp/rogue.py` process at PID 12847.

Flag value: `FLAG{ala-hunt1_cmd08_processes_all}`

Flag 9 (cmd9) — Filter lines

Worksheet row: "Filter lines from a stream where they match a given pattern."

Command (any `grep` use):

```
grep ERROR /var/log/syslog
grep nginx
grep -i error
```

Trigger: Any call to the `grep` handler. The handler awards immediately and prints a hint about piping.

Flag value: `FLAG{ala-hunt1_cmd09_filter_lines}`

Flag 10 (cmd10) — Start a service

Worksheet row: "Start a stopped systemd service."

Command:

```
systemctl start grid-sync
sudo systemctl start sshd
systemctl start grid-monitor
```

Trigger: `systemctl` handler with `sub === 'start'`. Awards regardless of whether the start itself succeeds (so dependency errors don't block the flag).

Flag value: `FLAG{ala-hunt1_cmd10_service_start}`

Flag 11 (cmd11) — Restart a service

Worksheet row: "Restart an already-running systemd service."

Command:

```
systemctl restart nginx
sudo systemctl restart grid-monitor
```

Trigger: `systemctl` handler with `sub === 'restart'`. Same as cmd10 — awards on call regardless of underlying state.

Flag value: `FLAG{ala-hunt1_cmd11_service_restart}`

Flag 12 (cmd12) — Daemon reload

Worksheet row: "After editing a unit file, force systemd to re-read all unit files (BEFORE you start/restart)."

Command:

```
systemctl daemon-reload
sudo systemctl daemon-reload
```

Trigger: `systemctl` handler with `sub === 'daemon-reload'`.

Flag value: `FLAG{ala-hunt1_cmd12_daemon_reload}`

Flag 13 (cmd13) — Verify external connectivity

Worksheet row: "After fixing the network, verify the cell can reach the outside world."

Command:

```
ping 8.8.8.8
ping -c 4 8.8.8.8
ping google.com
ping 1.1.1.1
```

Trigger: `ping` handler called with a non-RFC1918 target (e.g., `8.8.8.8`, `1.1.1.1`, any hostname) AND `_serviceState.networking === 'active'`. The handler awards `cmd13` ONLY when networking is repaired AND the target is not a local/LAN address. A successful external ping = troubleshooting verified.

This is the only flag in the box that rewards a SUCCESSFUL OUTCOME rather than command execution. Pedagogically: real Linux engineers always verify their fix worked. Pings to LAN nodes (10.0.x.x / 192.168.x.x / 172.16-31.x) do NOT award the flag — only true external targets count as "the internet is back."

Flag value: `FLAG{ala-hunt1_cmd13_verify_connectivity}`

Order Dependency

Partial. 12 of the 13 flags are independent — students can capture cmd12 (daemon-reload) before cmd1 (tmux) if they want. The ONLY ordered flag is `cmd13` (verify connectivity), which requires `cmd5` (ip link set eth1 up) or the equivalent service restart to have run first — the network must actually be repaired before a ping to 8.8.8.8 will succeed.

The natural flow IS the slide-24 incident investigation, however, and the worksheet numbering encourages it: - cmd1 tmux → set up - cmd2 systemctl status → diagnose - cmd3, cmd4 journalctl → read logs - cmd5, cmd6, cmd7, cmd8 → ip / ss / ps to dig - cmd9 grep → filter while digging - cmd10, cmd11, cmd12 → fix + restart

Students who fix the box completely will also naturally trigger most flags AND see the underlying ala-l01 inheritance (eth1 down → grid-sync/grid-monitor cascade).

Worksheet Integration

The printed worksheet has 12 numbered rows + 2 flag-value spaces at the bottom:

1. **Row N** description → student writes the command they ran
2. **Lab Flag block** → student copies the captured `FLAG{...}` value into the bottom of the sheet

Auto-captured flags are visible in the box's flag panel (the "Submit Flag" modal shows 13 badges — `cmd1.txt` through `cmd13.txt` — turning from grey to green as each is captured). With the recent CSS update, the badges wrap to two rows so all 13 are visible without modal overflow.

Scoring (instructor side): - Worksheet row matches valid command (per the answer key per flag above) → 8.3 pts (12 × 8.3 = 100) - Wrong answer → −2 pts - All 12 lab flags captured → required for "correct completion"; missing any flag treats as worksheet-only credit - First team to finish correctly: 100/100 - Other correct teams: 95/100

Hints Reference (inherited from ala-l01)

Hint	Text	Penalty
1	"Run systemctl status for each service in the dependency chain listed in ~/notes.txt. Start with networking — check ip link show to see why it is degraded."	−50 pts
2	"The ops.log was being written when power failed. Check the log directory for recovery artifacts: ls -la /var/log/cell-ops/ — look for hidden files."	−50 pts
3	"journalctl -u grid-sync -n 50 will show exactly why grid-sync refuses to start. Trace the dependency failure back to the interface level."	−50 pts

Note: hint 2's content (about ops.log recovery) is inherited but does NOT correspond to any flag in this lab (the old flag 2 was removed). Operator may want to update hint 2's text in a future pass.

Common Student Mistakes

- **Running commands without flag arguments.** `systemctl status` alone (no unit name) does NOT trigger cmd2 — engine requires a unit argument. Be specific.
- **Running `journalctl -u nginx -n 50`.** Misses cmd3 (which requires `-n 30 --no-pager`). Engine is strict on the `-n 30` value and the `--no-pager` presence.
- **Running `journalctl -f`.** Misses cmd4 (which requires both `-f` AND `-u SVC`).
- **Running `ss -tnp` (no -l).** Misses cmd7 (which requires `-l` for "listening").
- **Skipping `--no-pager`.** Common omission. The hint about no pager is in the worksheet description.

Teaching Notes

- **Primary concept:** building muscle memory for the W1 command families through a low-stakes timed race. Each command in the worksheet maps to a tool from the W1 topic decks (CLI Operations, systemd, Network Config, Network Diag).
- **Why a CTF wrapper:** forces students to ACTUALLY run the commands. A multiple-choice quiz could be answered from the deck — this can't.
- **Why a printed sheet:** captures their understanding in handwriting (harder to copy from a neighbor's screen) and gives the instructor a physical artifact for grading.
- **The 12 flag badges:** the post-deploy CSS update makes them wrap to 2 rows, fitting all 12 in the modal width. Students see at a glance which flags they've captured (green) vs. missing (grey).
- **Box engine reused intact:** This is `ala-l01` underneath. The two pre-existing flag rewards (all-services-restored and ops.log-recovered) were removed; the systemd dependency chain and filesystem are still there. A student who chooses to fix the entire box gets full hands-on practice for free — the 12 flags just gamify which commands they HAD to demonstrate.

Validator Checklist

- [x] **BOX-002a** Walkthrough exists at `~/hexworth-shared/Solutions/Advanced Linux Administration/ALA-Hunt1-Website-Down-SOLUTION.md`
- [x] **BOX-002b** Walkthrough cites all 13 flag values verbatim
- [x] **BOX-002c** Walkthrough flag values match `config.js` flags array (13 entries, `cmd1` – `cmd13`)
- [x] **BOX-024** No duplicate flag values across boxes (all use `ala-hunt1_` prefix)

Related Artifacts

- **Lab config:** `_app/houses/matrix/adv-linux/labs/ala-hunt1-website-down/config.js`
- **Lab index:** `_app/houses/matrix/adv-linux/labs/ala-hunt1-website-down/index.html`
- **Hub wiring:** `_app/houses/matrix/adv-linux/index.html` (Week 1 card → In-class Activity subsection)
- **Worksheet (printable):** `~/hexworth-shared/images/advanced linux/scavengerHunt-website-down.docx`
- **Lecture deck reference:** `_app/houses/matrix/adv-linux/presentations/ala-w1-lecture.presentation.html` slide 24 ("The Website Is Down" synthesis)
- **Engine template:** `_app/houses/matrix/adv-linux/labs/ala-l01-dead-cell-recovery/config.js` (the box this was forked from)

Last updated: 2026-06-07