MATRIX HOUSE | OPERATOR QUICK REFERENCE

Week 4 Operator Field Manual

Integrity & Performance — The Final Week

COURSE	**CTS4321C Advanced Linux Administration**
WEEK	**4 of 8**
TOPICS	**File Integrity | Performance | Log Management | Incident Playbooks**
LABS	**L10 Ghost in the Cell | L11 Flatline | L12 Full Cell Audit**
PLATFORM	**Ubuntu 22.04 LTS**
AUDIENCE	**Operator / Instructor Quick Reference**

CTS4321C Advanced Linux Administration — Hexworth Prime

Quick reference for lab sessions and incident response. Covers AIDE, auditd, performance monitoring, log management, and capstone audit workflow.

WEEK 4 OPERATOR QUICK REFERENCE

> *File Integrity | Performance | Log Management | YOUR CELL IS YOUR RESPONSIBILITY*

Integrity keeps the cell honest. Performance keeps it alive. Logs tell the story after the fact.

Problem	First Question	Command
Was a file tampered?	Does the hash match the baseline?	`aide --check 2>/dev/null`
Which files changed?	What does AIDE report?	`aide --check | grep Changed`
Who changed it?	What does auditd say?	`ausearch -k sshd-config -i`
Node running slow?	CPU, memory, or I/O?	`top` then `vmstat 1 3`
Disk full?	Where is it?	`df -h` then `du -sh /var/log/* | sort -rh`
CPU pegged?	Which process?	`top` (M = sort mem, P = sort cpu)
Memory leak?	What's growing?	`ps aux --sort=-%mem | head -8`
Swap usage rising?	Is the node paging?	`free -h` (check Swap: used)
I/O saturated?	Which device?	`iostat -xz 2 3`
Logs stopped?	Is rsyslog running?	`systemctl status rsyslog && rsyslogd -N1`
Can't find the event?	Is journal persistent?	`journalctl --disk-usage`

Daily First Commands

Command	Use
`uptime`	Load average + uptime in one line
`free -h`	Memory state at a glance
`df -h`	Disk usage across all filesystems
`aide --check 2>/dev/null`	Integrity check against baseline
`journalctl -p err --since "1 hour ago"`	Errors in the last hour
`aureport --auth --failed`	Failed authentications today

Week 4 Goal: detect what changed, measure what is slow, and reconstruct what happened -- before the incident commander asks.

FILE INTEGRITY MONITORING

> *sha256sum | AIDE | Tripwire | auditd | chattr*

Problem to Command

Question	Command
Hash a single file	`sha256sum /etc/ssh/sshd_config`
Hash multiple files	`sha256sum /etc/ssh/sshd_config /etc/passwd /etc/sudoers`
Save a manifest	`sha256sum /etc/ssh/* /usr/local/bin/* > /var/lib/integrity/baseline.sha256`
Verify against manifest	`sha256sum --check /var/lib/integrity/baseline.sha256`
Protect baseline from root	`chattr +i /var/lib/integrity/baseline.sha256`
Confirm immutable flag	`lsattr /var/lib/integrity/baseline.sha256`
Check all packages	`dpkg --verify`
Show only modified non-config files	`dpkg --verify 2>&1 | grep -v ' c '`

Hash Algorithm Reference

Algorithm	Bits	Status	Use
MD5	128	BROKEN -- collisions engineered	Download verification only
SHA-1	160	DEPRECATED -- SHAttered 2017	Do not use for FIM
SHA-256	256	CURRENT	All production integrity work
SHA-512	512	CURRENT	High-security / compliance

AIDE Lifecycle

```
aideinit                              # Step 1: build new database (takes several minutes)
cp /var/lib/aide/aide.db.new \
   /var/lib/aide/aide.db              # Step 2: activate

aide --check                          # Step 3: scan and compare (exit 0=clean 1=changed 2=error)
aide --check 2>/dev/null              # suppress stderr noise from /proc and /sys
aide --check --verbose=2              # show detailed diff for each changed file
aide --check --limit=/etc/ssh         # check specific directory only

aide --update                         # Step 4: after authorized changes, write aide.db.new
mv /var/lib/aide/aide.db.new \
   /var/lib/aide/aide.db              # promote to active
```

AIDE Configuration (key sections of `/etc/aide/aide.conf`)

```
# Attribute groups
FIPSR  = p+i+n+u+g+s+m+c+acl+selinux+xattrs+sha256
NORMAL = p+i+n+u+g+s+m+c+sha256
LOGCHECK = p+u+g+n+S          # permissions only -- content changes constantly

# Monitor binaries and critical config for everything
/usr/bin     FIPSR
/usr/sbin    FIPSR
/usr/local/bin FIPSR
/etc/ssh     FIPSR
/etc/cron.d  FIPSR
/etc/sudoers FIPSR

# Logs: permissions only
/var/log     LOGCHECK

# Exclude noise
!/var/log/wtmp
!/var/log/lastlog
!/var/run
!/proc
!/sys
```

Tripwire Reference

```
apt install tripwire
twinstall.sh                          # generate site-key + local-key passphrases
tripwire --init                       # build signed database (prompts for passphrases)
tripwire --check | tee /var/log/tripwire/check.log
tripwire --update --twrfile /var/lib/tripwire/report/hostname-date.twr
twadmin --print-polfile               # view current policy in plain text
```

auditd: Who Changed It

```
# Install and enable
apt install auditd audispd-plugins
systemctl enable --now auditd

# Runtime rules (lost on reboot)
auditctl -w /etc/passwd        -p wa  -k identity
auditctl -w /etc/ssh/sshd_config -p wa -k sshd-config
auditctl -w /etc/sudoers       -p wa  -k sudoers
auditctl -w /etc/cron.d        -p wa  -k cron
auditctl -w /usr/bin/sudo      -p x   -k sudo
auditctl -l                           # list active rules
auditctl -D                           # delete all rules

# Persistent rules: /etc/audit/rules.d/sector-security.rules
-D
-b 8192
-w /etc/passwd   -p wa -k identity
-w /etc/shadow   -p wa -k identity
-w /etc/ssh/sshd_config -p wa -k sshd-config
-w /etc/sudoers  -p wa -k sudoers
-w /etc/cron.d   -p wa -k cron
-w /usr/bin/sudo -p x  -k sudo
-e 2    # lock rules until reboot

augenrules --load                     # compile and load all rules.d files
systemctl restart auditd              # reload after rule edits
```

ausearch and aureport

```
# ausearch: query by key, user, time, path
ausearch -k sshd-config -i            # -i = interpret UIDs as names
ausearch -k identity --start today
ausearch -ua deploy --start today
ausearch -f /usr/local/bin/grid-backup -i
ausearch --start 04/09/2026 02:14:00 --end 04/09/2026 02:16:00 -i | less

# aureport: summary tables
aureport                              # overall summary
aureport --auth --failed              # failed logins
aureport --executable                 # command usage
aureport --file                       # file access events
aureport --anomaly                    # anomaly events
aureport --auth --start today
```

chattr: Filesystem Immutability

```
chattr +i /var/lib/aide/aide.db       # immutable: root cannot modify without removing flag
chattr +i /var/lib/integrity/baseline.sha256
chattr +a /var/log/auth.log           # append-only: can write, cannot truncate or delete

lsattr /var/lib/aide/aide.db          # ----i---------e-- = immutable
chattr -i /var/lib/aide/aide.db       # remove flag before AIDE update, re-add after
```

AIDE vs auditd

Tool	Answers	Mechanism
AIDE	*What* changed (hash/size/permissions)	Periodic comparison to database
auditd	*Who* changed it, *when*, from *which terminal*	Kernel-level syscall auditing

Use both. AIDE for detection, auditd for attribution.

PERFORMANCE MONITORING

> *top* | *htop* | *vmstat* | *iostat* | *sar* | *free* | *uptime*

Investigation Model

```
Observe (top/htop) --> Isolate subsystem --> Quantify with baseline --> Fix
                       CPU  MEM  IO  NET
```

USE Method (Brendan Gregg): for every resource -- check Utilization, Saturation, and Errors.

Load Average

```
uptime                    # uptime + 1/5/15-min load average
cat /proc/loadavg         # 2.34 1.87 1.42 3/287 14823
nproc                     # number of logical CPUs

# Normalize: load / nproc
# 4.0 on 4-core = saturated   |   4.0 on 32-core = light
# Trend: 1-min > 15-min = rising load
```

top

```
top                       # live, default refresh (q to quit)
top -bn1                  # batch mode, 1 sample: script-friendly
top -p 14823              # monitor specific PID
top -u nginx              # filter by user
top -d 0.5                # 0.5-second refresh

# Interactive keys (while running):
# 1 = per-CPU breakdown    M = sort by memory    P = sort by CPU
# k = kill PID             u = filter by user    W = write to ~/.toprc
```

top CPU fields:

Field	Meaning	Alert if
us	User-space CPU	Sustained high with known rogue process
sy	Kernel CPU	High = excessive syscalls
wa	I/O wait	> 20% consistently = disk bottleneck
st	Steal (VM only)	Non-zero = hypervisor taking CPU

Process states (S column): `R` = running, `S` = sleeping (normal), `D` = uninterruptible I/O wait, `Z` = zombie, `T` = stopped. Many D-state processes = I/O saturation.

vmstat

```
vmstat 1 5                # 1-second intervals, 5 samples
vmstat -d 1 3             # per-disk I/O stats
vmstat -s                 # event summary since boot

# Key columns:
# r  = processes in run queue (> nproc = CPU saturation)
# b  = uninterruptible sleep (> 0 consistently = I/O bottleneck)
# si/so = swap in/out per second (non-zero = memory pressure)
# wa = I/O wait %
# cs = context switches per second
```

free and Memory

```
free -h                   # human-readable
watch -n2 'free -h'       # live update every 2 seconds
cat /proc/meminfo | head -20

# IMPORTANT: "free" column is not the answer.
# "available" = free + reclaimable cache = what a new process can use.
# buff/cache being large is healthy (OS using spare RAM for disk cache).
# Swap used > 0 AND growing = memory leak or undersized RAM.
```

iostat

```
apt install sysstat
iostat -xz 2 5            # extended stats, 2s intervals, 5 samples
iostat -c 1               # CPU I/O wait only
iostat -d sda 1           # single device

# Key fields:
# await   = average I/O latency ms  (> 10ms HDD, > 1ms NVMe = slow)
# %util   = device busy %           (> 80% consistently = saturated)
# aqu-sz  = average queue depth     (> 1 = requests piling up)
# r/s w/s = reads and writes per second
```

sar (Historical)

```
apt install sysstat
sed -i 's/ENABLED="false"/ENABLED="true"/' /etc/default/sysstat
systemctl enable --now sysstat

# Query today's data
sar -u                    # CPU history (10-min intervals)
sar -r                    # memory history
sar -b                    # disk I/O history
sar -d                    # per-device disk history
sar -n DEV                # per-interface network history
sar -n TCP                # TCP segment rates
sar -q                    # run queue and load average history

# Live mode (like vmstat)
sar -u 1 5                # CPU every second, 5 samples

# Query a specific day's data file
sar -u -f /var/log/sysstat/sa08            # April 8
sar -q -f /var/log/sysstat/sa08 | sort -k5 -rn | head -5   # peak load

# Incident window on specific day
for flag in -u -r -b -n DEV; do
    echo "=== sar $flag ==="
    sar $flag -f /var/log/sysstat/sa08 -s 02:00:00 -e 04:00:00
done
```

mpstat and iotop

```
mpstat -P ALL 1 5         # per-CPU breakdown (single core saturated = single-threaded bottleneck)
taskset -pc 14823         # check CPU affinity of a process

apt install iotop
iotop -o                  # show only processes with active I/O
iotop -b -n5              # batch mode, 5 iterations
cat /proc/14823/io        # per-process I/O counters without iotop
```

df, du, and Disk Space

```
df -h                     # filesystem usage
df -hT                    # include filesystem type
df -i /var/log            # inode usage (can be full even if bytes are free)

du -sh /var/log/*         # sizes of items under /var/log
du -sh /var/log/* | sort -rh | head -10   # largest first

# Find top 20 largest files under /var
find /var -type f -printf '%s %p\n' 2>/dev/null | sort -rn | head -20 | \
    awk '{printf "%.1f MB\t%s\n", $1/1048576, $2}'
```

Key Bottleneck Signatures

Symptom	Tool Confirms	Action
Load > nproc, us% high	`top` `mpstat -P ALL`	Find rogue process; check crontab
High wa%, b > 0 in vmstat	`iostat -xz` `iotop -o`	Check `await` and `%util` per device
Swap used + growing	`free -h` `vmstat si/so`	Find memory leak: `ps aux --sort=-%mem`
Disk 90%+	`df -h` `du -sh /var/log/*`	Logrotate, clean old files
Single CPU saturated	`mpstat -P ALL`	Single-threaded process; check `taskset`

LOG MANAGEMENT

> *journalctl | rsyslog | logrotate | grep/awk analysis | centralization*

Two Logging Systems

	journald	rsyslog
Storage	Binary, `/var/log/journal/`	Plain text, `/var/log/`
Query	`journalctl` (rich filters)	`grep` , `awk` , `cut`
Use for	Service investigation, real-time following, boot messages	SIEM forwarding, legacy integration, long-term retention
Config	`/etc/systemd/journald.conf`	`/etc/rsyslog.conf` , `/etc/rsyslog.d/`

journalctl Core Commands

```
journalctl                          # all entries (oldest first)
journalctl -f                       # follow live
journalctl -n 50                    # last 50 lines
journalctl -n 50 -f                 # last 50 + follow

# By unit
journalctl -u nginx
journalctl -u nginx -u mysql        # multiple units
journalctl -fu nginx                # follow nginx specifically

# By priority (emerg=0 .. debug=7)
journalctl -p err                   # error and above
journalctl -p err..warning          # error through warning

# By boot
journalctl -b                       # current boot
journalctl -b -1                    # previous boot
journalctl --list-boots             # all available boots

# Kernel messages
journalctl -k
```

journalctl Time Filtering

```
journalctl --since "2026-04-09 02:30"
journalctl --since "2026-04-09 02:44" --until "2026-04-09 02:52"
journalctl --since "1 hour ago"
journalctl --since today
journalctl --since yesterday

# Incident pattern: all errors across all services in the window
journalctl --since "2026-04-09 02:40" --until "2026-04-09 03:00" -p err --no-pager

# Combine: unit + time + priority
journalctl -u nginx --since "2026-04-09 02:40" --until "2026-04-09 03:00"
```

journald Configuration (`/etc/systemd/journald.conf`)

```
[Journal]
Storage=persistent          # write to /var/log/journal/ (survives reboot)
SystemMaxUse=2G             # max disk for all journals
SystemMaxFileSize=256M      # max size of one journal file
MaxRetentionSec=30day       # oldest entry to keep
ForwardToSyslog=yes         # feed rsyslog
Compress=yes

# Apply
mkdir -p /var/log/journal
systemctl restart systemd-journald
journalctl --disk-usage
journalctl --vacuum-time=30d
journalctl --vacuum-size=1G
```

rsyslog Facility.Priority Reference

Priority	Severity	Route example
emerg (0)	System unusable	`:omusrmsg:*` (alert all users)
crit (2)	Critical	`/var/log/syslog` + SIEM
err (3)	Error	`/var/log/syslog`
warning (4)	Warning	`/var/log/syslog`
info (6)	Normal	`/var/log/syslog`

rsyslog Key Commands

```
rsyslogd -N1                          # syntax check (dry run)
systemctl restart rsyslog             # apply config changes
journalctl -u rsyslog -n 30          # rsyslog own errors

# Routing examples in /etc/rsyslog.conf
auth,authpriv.*         /var/log/auth.log
kern.*                  /var/log/kern.log
cron.*                  /var/log/cron.log
local0.*                /var/log/sector/app.log
*.warning;mail.none     /var/log/syslog

# Forward with disk queue (TCP, production standard)
action(type="omfwd"
       target="logs.sector.local" port="514" protocol="tcp"
       action.resumeRetryCount="-1"
       queue.type="Disk"
       queue.filename="fwd-buffer"
       queue.maxDiskSpace="2g"
       queue.saveOnShutdown="on")

# Property-based routing
:programname, isequal, "nginx"        /var/log/nginx/syslog.log
:msg, contains, "CRITICAL"            /var/log/sector/critical.log
```

logrotate

```
# Force rotation now
logrotate -f /etc/logrotate.d/cell-stream

# Test without acting
logrotate -d /etc/logrotate.d/cell-stream

# Standard config template: /etc/logrotate.d/cell-stream
/var/log/cell-stream/*.log {
    daily
    rotate 14
    compress
    delaycompress
    missingok
    notifempty
    postrotate
        systemctl reload cell-stream.service 2>/dev/null || true
    endscript
}

# NOTE: chattr +a (append-only) files cannot be rotated --
#       remove flag in postrotate, re-add after, or drop the flag entirely.
```

Log Analysis Patterns

```
# Failed SSH logins by source IP
grep 'Failed password' /var/log/auth.log | \
    awk '{print $NF}' | sort | uniq -c | sort -rn | head -20

# nginx 5xx responses and which URLs
awk '$9 >= 500' /var/log/nginx/access.log | \
    cut -d'"' -f2 | sort | uniq -c | sort -rn

# Errors per service (last 24h) via journal
journalctl -p err --since "24 hours ago" -o json | \
    jq -r '.SYSLOG_IDENTIFIER' | sort | uniq -c | sort -rn | head -10

# Event count by hour (today)
journalctl -p err --since today -o json | \
    jq -r '.__REALTIME_TIMESTAMP | tonumber / 1000000 | todate | .[0:13]' | \
    sort | uniq -c

# Timeline of all sudo activity today
grep 'sudo:' /var/log/auth.log | awk '{print $1" "$2" "$3" "$6" "$NF}'

# Count events per minute (plain syslog)
awk '{print $1" "$2" "$3}' /var/log/syslog | cut -c1-16 | sort | uniq -c | tail -30
```

rsyslog Troubleshooting

Check	Command
Is it running?	`systemctl status rsyslog`
Config syntax?	`rsyslogd -N1`
Can it reach SIEM?	`nc -zv logs.sector.local 514`
Queue backed up?	`ls -lh /var/lib/rsyslog/*.qi`
Disk full?	`df -h /var/log`

INCIDENT PLAYBOOKS

> *L10 Ghost in the Cell | L11 Flatline | L12 Full Cell Audit*

L10 Playbook: Ghost in the Cell (File Integrity Forensics)

Scenario: AIDE reports 14 modifications. Eleven are legitimate. Find the three unauthorized ones.

```
sudo aide --check 2>/dev/null                              # get all 14 modifications

# Categorize: expected vs suspicious
# Expected: /var/log/ changes, /tmp/ files, /var/cache/apt/, .bash_history
# Suspicious: binaries in /usr/local/bin/, critical configs, hidden directories

diff /usr/local/bin/grid-backup \
    /var/lib/aide/originals/grid-backup                    # compare binary to original

grep PermitRootLogin /etc/ssh/sshd_config                  # check for hardening regression

find /home -name '.*' -type d                              # surface hidden directories

tar -tzf /home/operator/.hidden-cache/payload.tar.gz       # inspect archive without extracting

/opt/verify/check-findings.sh \
    /usr/local/bin/grid-backup \
    /etc/ssh/sshd_config \
    /home/operator/.hidden-cache/payload.tar.gz            # submit all three to award flags
```

Post-investigation remediation (not required for flags, reference only):

```
sudo cp /var/lib/aide/originals/grid-backup /usr/local/bin/grid-backup
sudo sed -i 's/PermitRootLogin yes/PermitRootLogin no/' /etc/ssh/sshd_config
sudo systemctl reload sshd
sudo rm -rf /home/operator/.hidden-cache/
sudo aide --init && sudo mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db
```

Key lessons: AIDE detects *what*, auditd identifies *who*. Triage first -- categorize all changes before investigating any. Inspect archive contents with `tar -tz` before extracting.

L11 Playbook: Flatline (Performance / Resource Exhaustion)

Scenario: Three active resource problems -- rogue CPU cron job, memory-leaking service with no limit, and a 4.2 GB log file at 94% disk.

Flag 1: CPU Spike

```
top                                                    # watch for spike every ~4 minutes
# grid-index.sh spawns: find / -type f | sha256sum

ps aux | grep grid-index                               # confirm process + path

cat /opt/cell-services/grid-index.sh                   # understand what it does

sudo crontab -l                                        # find the cron entry: */4 * * * *

sudo crontab -e                                        # delete the offending line

/opt/verify/check-cpu.sh                               # verify: FLAG 1
```

Flag 2: Memory Leak

```
free -h                                                # see 2.1 GB used
ps aux --sort=-%mem | head -8                          # cell-cache at 27% memory

systemctl cat cell-cache.service                       # no MemoryMax directive

sudo systemctl edit cell-cache.service
# Add:
# [Service]
# MemoryMax=256M

sudo systemctl daemon-reload
sudo systemctl restart cell-cache.service

/opt/verify/check-memory.sh                            # verify: FLAG 2
```

Flag 3: Disk Full

```
df -h                                                    # /var at 94%

du -sh /var/log/cell-stream/*                            # stream.log = 4.2 GB

sudo nano /etc/logrotate.d/cell-stream
# /var/log/cell-stream/*.log {
#     daily
#     rotate 7
#     compress
#     delaycompress
#     missingok
#     notifempty
#     postrotate
#         systemctl reload cell-stream.service 2>/dev/null || true
#     endscript
# }

sudo logrotate -f /etc/logrotate.d/cell-stream           # force immediate rotation

df -h                                                    # verify /var dropped below 80%

/opt/verify/check-disk.sh                                # verify: FLAG 3
```

L12 Playbook: Full Cell Audit (Capstone)

Scenario: `cell-abandoned` -- 32 issues across four domains. Read `~/MISSION.txt` and `~/audit-checklist.txt`. Work domain by domain. Run each verify script only after completing the full domain.

Domain	Flag	Key Actions	Verify Script
Network	FLAG 1	Fix netplan typo; `netplan apply`; fix DNS resolver	`check-network.sh`
Security	FLAG 2	Harden sshd; remove shadow root account; enable UFW	`check-security.sh`
Services	FLAG 3	Secure BIND (`recursion no`); stop beacon service; start grid-sync	`check-services.sh`
Integrity	FLAG 4	Install AIDE + auditd; init baseline; configure logrotate	`check-integrity.sh`

Network (FLAG 1)

```
ip link show                                             # eth1 is DOWN
sudo nano /etc/netplan/00-config.yaml                    # fix: addressess -> addresses
sudo netplan apply
sudo nano /etc/resolv.conf                               # replace 1.2.3.4 with 10.0.1.1
/opt/verify/check-network.sh
```

Security (FLAG 2)

```
sudo nano /etc/ssh/sshd_config
# PermitRootLogin no | PasswordAuthentication no | PermitEmptyPasswords no
sudo systemctl reload sshd

awk -F: '$3 == 0 {print $1}' /etc/passwd                    # find UID 0 shadow accounts
sudo userdel -r svc-ghost
sudo visudo                                                  # remove svc-ghost ALL=(ALL)
NOPASSWD:ALL
sudo rm /var/spool/cron/crontabs/svc-ghost 2>/dev/null || true
sudo ufw enable && sudo ufw allow 22/tcp && sudo ufw allow 443/tcp
/opt/verify/check-security.sh
```

Services (FLAG 3)

```
sudo systemctl stop cell-beacon && sudo systemctl disable cell-beacon
sudo rm /tmp/beacon.sh /etc/systemd/system/cell-beacon.service
sudo systemctl daemon-reload

sudo nano /etc/bind/named.conf.options
# recursion no;
# allow-query { 10.0.0.0/8; localhost; };
sudo systemctl reload named

sudo systemctl start grid-sync                               # now works: networking is up
/opt/verify/check-services.sh
```

Integrity (FLAG 4)

```
sudo apt install aide auditd
sudo aide --init
sudo mv /var/lib/aide/aide.db.new /var/lib/aide/aide.db

sudo systemctl enable --now auditd
sudo auditctl -w /etc/passwd -p wa -k user_accounts
sudo auditctl -w /etc/ssh/sshd_config -p wa -k ssh_config

# Persist rules: /etc/audit/rules.d/cell-hardening.rules
# -w /etc/passwd -p wa -k user_accounts
# -w /etc/ssh/sshd_config -p wa -k ssh_config
# -w /etc/sudoers -p wa -k sudoers
sudo systemctl restart auditd

sudo nano /etc/logrotate.d/cell-ops
# /var/log/cell-ops/*.log {
#     daily
#     rotate 14
#     compress
#     delaycompress
#     missingok
#     notifempty
# }
sudo logrotate -f /etc/logrotate.d/cell-ops

/opt/verify/check-integrity.sh
```

Domain ordering is not arbitrary. Network must come first -- grid-sync requires `network-online.target`. Services domain depends on networking being operational.

Professional operators do not guess. They gather evidence until the failure becomes obvious. Your cell is your responsibility. The grid depends on you.

Matrix House | CTS4321C Advanced Linux Administration | Week 4 Operator Field Manual