

Recent Linux & Automation Work Overview

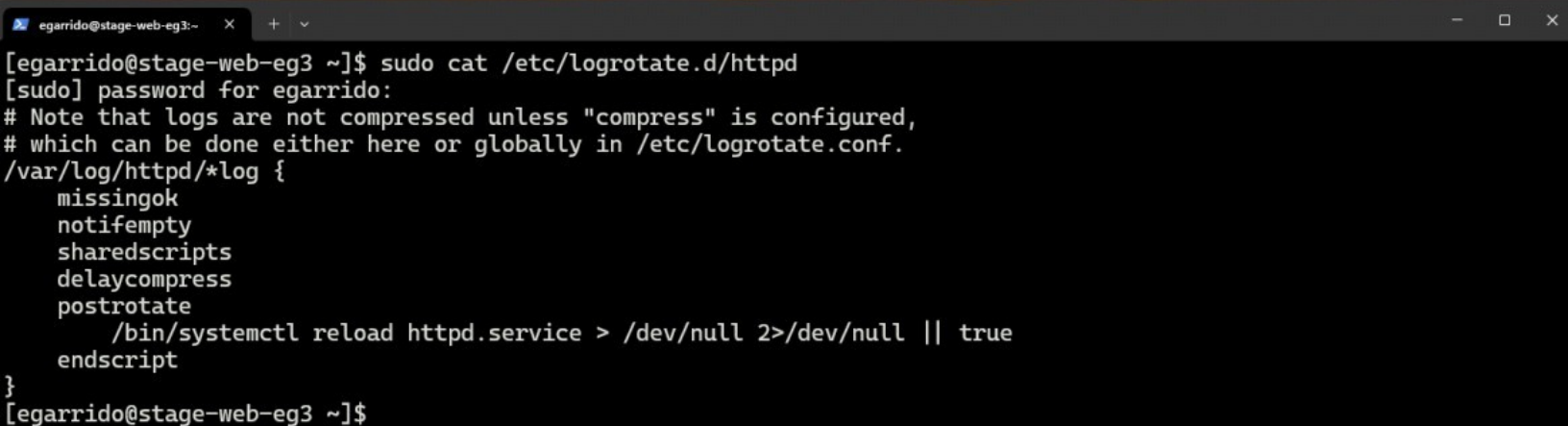
This collection represents a focused set of recent hands-on Linux system administration and automation tasks performed in an enterprise-style environment. The work emphasizes operational reliability, security hardening, service management, and repeatable automation using standard Linux tooling and Ansible. Each item reflects real-world scenarios commonly encountered in production systems, with an emphasis on validation, troubleshooting, and clean implementation.

The scope of this work includes configuring and verifying log rotation policies for Apache HTTP services, enforcing retention and compression standards, and validating post-rotation service reload behavior. System log handling was tested both through configuration review and forced execution to confirm expected outcomes. Additional tasks include managing service permissions, validating filesystem ownership and access controls, and confirming operational changes through command-line verification.

Automation-focused work demonstrates structured Ansible usage for system configuration, user management, and service control, paired with supporting Bash scripts to handle operational checks and reporting. Version control practices were applied throughout, with changes organized, committed, and tracked in GitLab to reflect a disciplined workflow aligned with production standards.

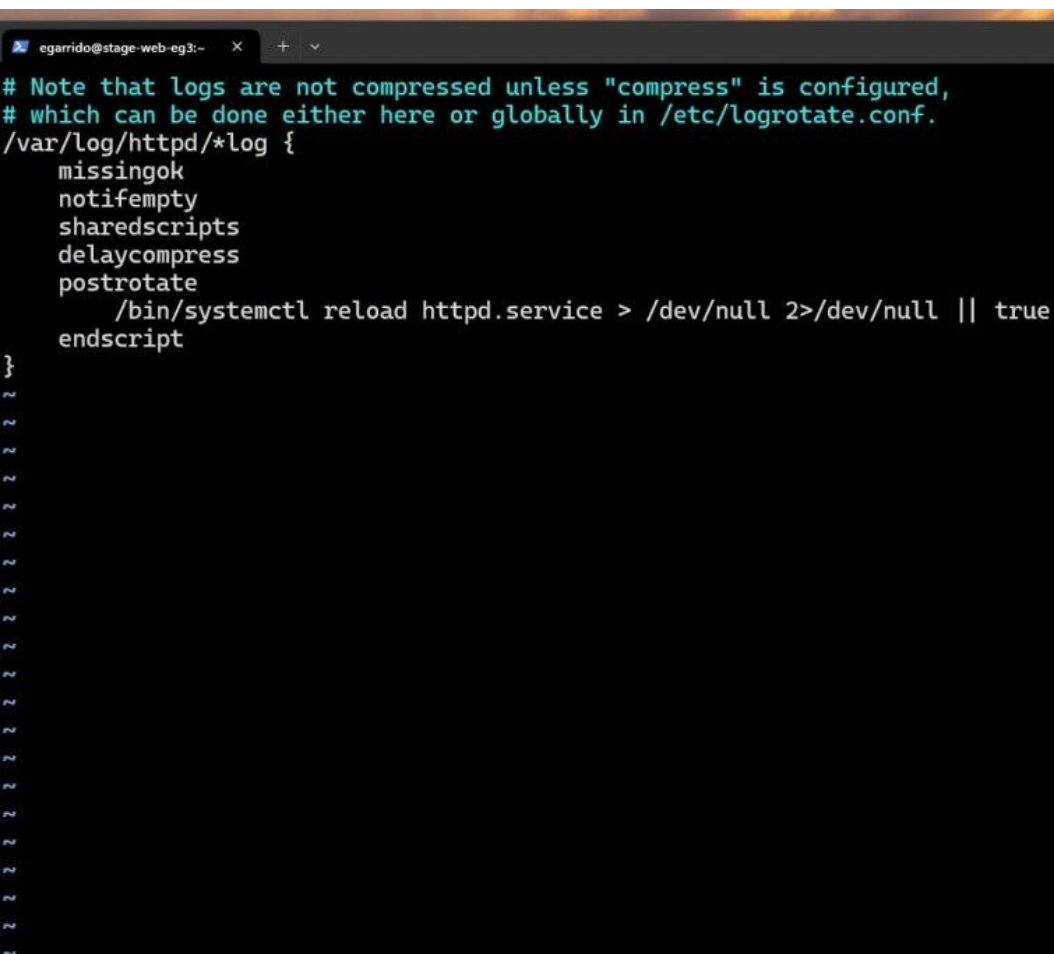
Overall, this set of work highlights practical Linux administration skills, attention to operational detail, and the ability to implement, test, and document infrastructure changes in a controlled and repeatable manner.

A terminal session shows the contents of the Apache logrotate configuration file being viewed with elevated privileges. The configuration applies to `/var/log/httpd/*log` and includes options to skip missing logs, ignore empty files, use shared scripts, and delay compression. After log rotation, the configuration runs a post-rotate command that reloads the `httpd` service using `systemctl`, suppressing any output or errors.



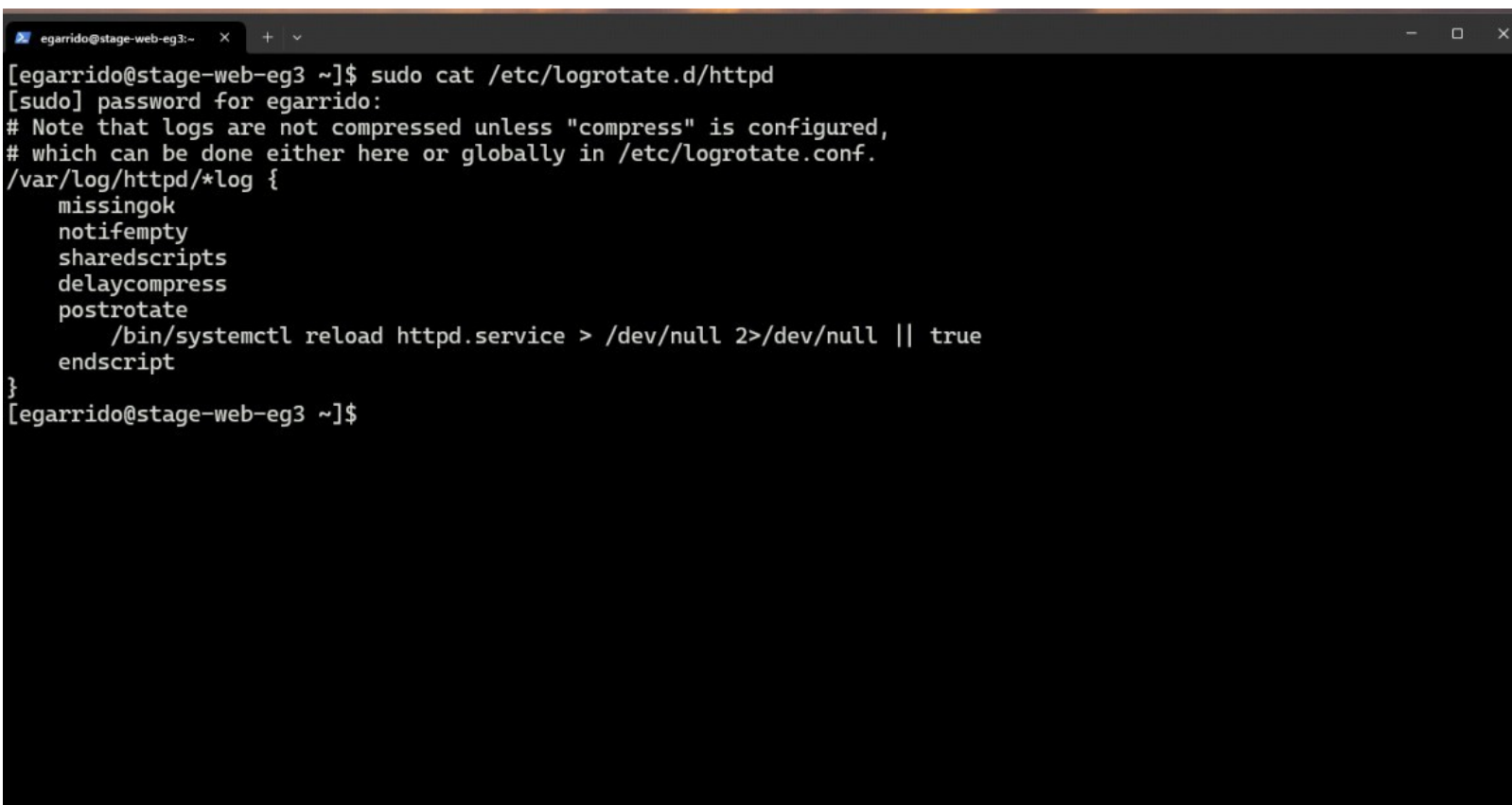
```
egarrido@stage-web-eg3:~$ sudo cat /etc/logrotate.d/httpd
[sudo] password for egarrido:
# Note that logs are not compressed unless "compress" is configured,
# which can be done either here or globally in /etc/logrotate.conf.
/var/log/httpd/*log {
    missingok
    notifempty
    sharedscripts
    delaycompress
    postrotate
        /bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true
    endscript
}
[egarrido@stage-web-eg3 ~]$
```

The Apache logrotate configuration is shown, defining rotation behavior for all files under /var/log/httpd/*log. The configuration skips missing or empty logs, uses shared scripts, and delays compression until a later rotation. After log rotation completes, a post-rotate script reloads the httpd service using systemctl, with all output and errors suppressed to avoid interruptions.

A terminal window with a dark background and a title bar showing 'egarrido@stage-web-eg3:~'. The terminal displays the Apache logrotate configuration for /var/log/httpd/*log. The configuration includes comments about compression, a list of options (missingok, notifempty, sharedscripts, delaycompress, postrotate), a postrotate script to reload httpd.service, and an endsript line. The terminal output is as follows:

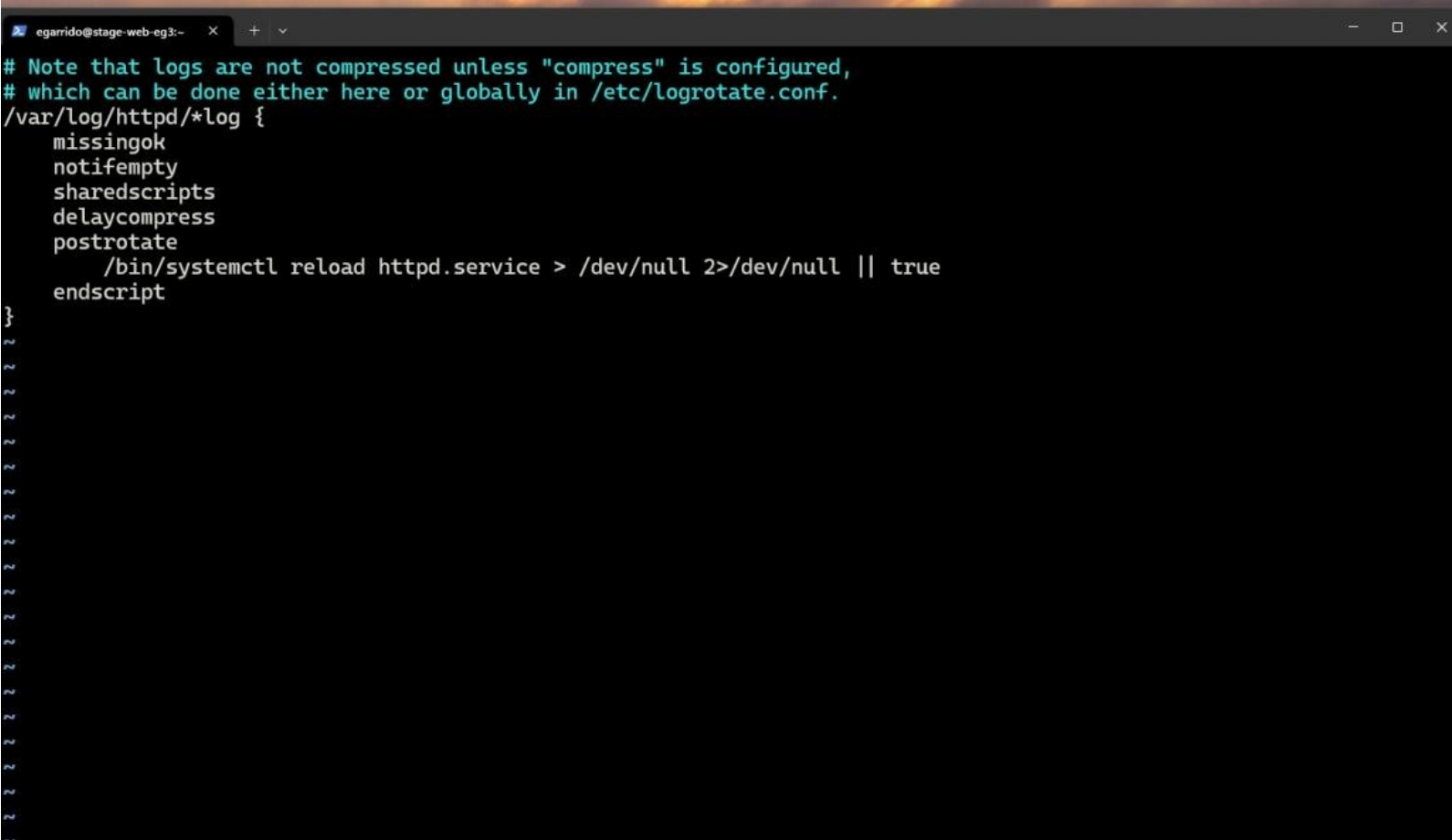
```
# Note that logs are not compressed unless "compress" is configured,
# which can be done either here or globally in /etc/logrotate.conf.
/var/log/httpd/*log {
    missingok
    notifempty
    sharedscripts
    delaycompress
    postrotate
        /bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true
    endsript
}
```

A terminal window shows the Apache log rotation configuration being viewed with elevated privileges. The file targets `/var/log/httpd/*log` and includes directives to ignore missing or empty logs, use shared scripts, and delay compression. A `postrotate` section reloads the `httpd` service with `systemctl`, redirecting all output and errors to `/dev/null` to ensure the reload runs quietly.

A terminal window with a dark background and light text. The window title bar shows 'egarrido@stage-web-eg3:~'. The terminal content shows a user running 'sudo cat /etc/logrotate.d/httpd'. After a password prompt, the configuration for /var/log/httpd/*log is displayed, including directives like missingok, notifempty, sharedscripts, delaycompress, and a postrotate section that reloads httpd.service using systemctl. The terminal ends with the prompt '[egarrido@stage-web-eg3 ~]\$'.

The Apache logrotate configuration is displayed in a text editor, defining rotation rules for all log files under `/var/log/httpd/*log`. The configuration skips missing and empty logs, enables shared scripts, and delays compression until a later rotation cycle.

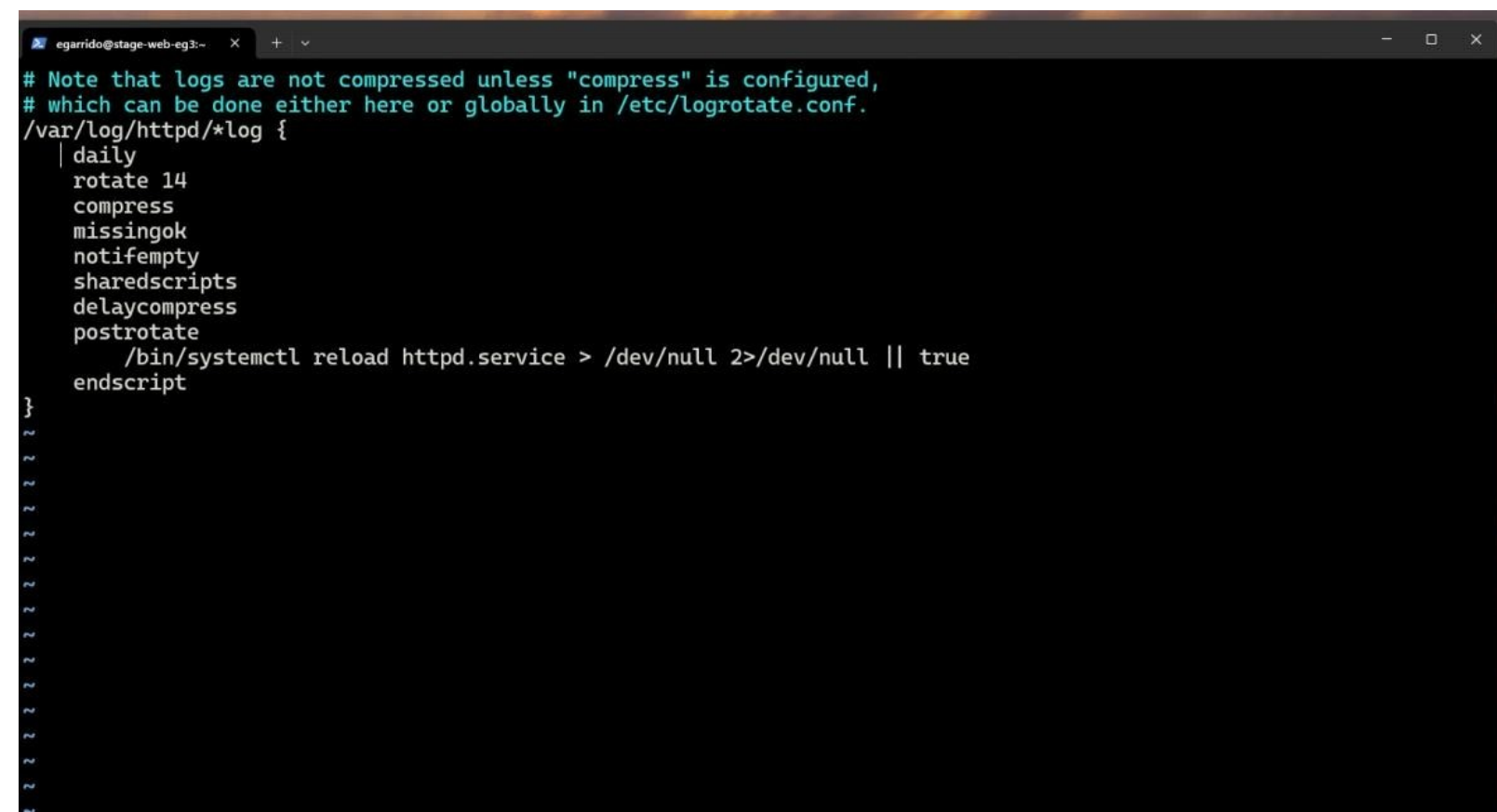
After log rotation completes, a post-rotate script reloads the `httpd` service using `systemctl`, with both standard output and errors redirected to `/dev/null` to ensure the reload runs quietly without interrupting log rotation.

A terminal window with a dark background and light green text. The window title is 'egarrido@stage-web-eg3:~'. The content shows the configuration for the logrotate service for httpd logs. The configuration includes comments about compression and a list of options: missingok, notifempty, sharedscripts, delaycompress, and postrotate. The postrotate script uses systemctl to reload httpd.service, redirecting output to /dev/null. The configuration ends with 'endscript' and a closing brace. The terminal shows several tilde characters (~) at the bottom, indicating the end of the file or a scroll buffer.

```
egarrido@stage-web-eg3:~  
# Note that logs are not compressed unless "compress" is configured,  
# which can be done either here or globally in /etc/logrotate.conf.  
/var/log/httpd/*log {  
    missingok  
    notifempty  
    sharedscripts  
    delaycompress  
    postrotate  
        /bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true  
    endscript  
}
```

The Apache logrotate configuration is shown with help comments at the top and rules applied to /var/log/httpd/*log. Logs are rotated daily, kept for 14 rotations, and compressed, with compression delayed until the next cycle. The configuration ignores missing or empty log files and uses shared scripts.

After each rotation, a post-rotate script reloads the httpd service using systemctl, redirecting all output and errors to /dev/null so the reload runs quietly without affecting log rotation.

A terminal window with a dark background and light-colored text. The window title bar shows 'egarrido@stage-web-eg3:~' and standard window controls. The terminal displays the Apache logrotate configuration for /var/log/httpd/*log. The configuration includes a note about compression, a daily rotation cycle, 14 rotations kept, compression, and a post-rotate script that reloads the httpd.service using systemctl.

```
egarrido@stage-web-eg3:~  
# Note that logs are not compressed unless "compress" is configured,  
# which can be done either here or globally in /etc/logrotate.conf.  
/var/log/httpd/*log {  
| daily  
  rotate 14  
  compress  
  missingok  
  notifempty  
  sharedscripts  
  delaycompress  
  postrotate  
    /bin/systemctl reload httpd.service > /dev/null 2>/dev/null || true  
  endscript  
}
```

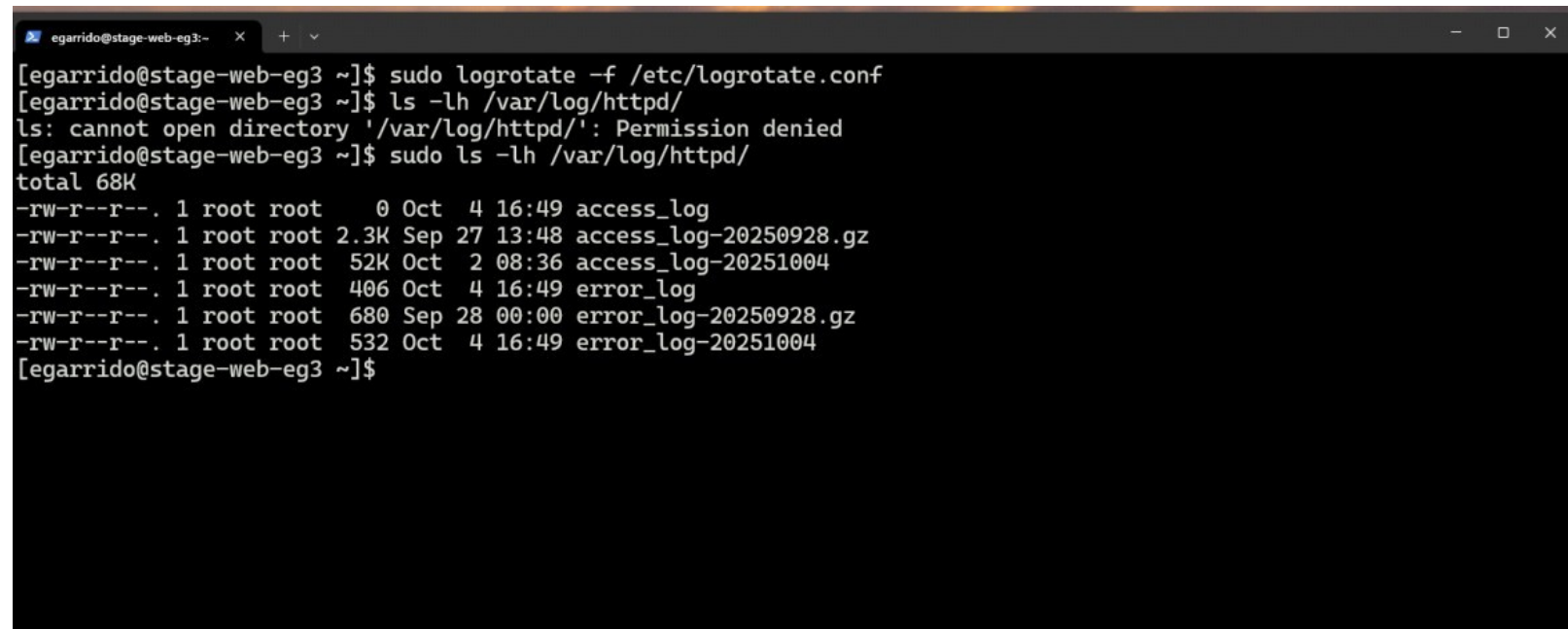
The terminal output shows a verbose logrotate run evaluating multiple system log files. Each entry displays the current timestamp, the last rotation time, and a decision on whether rotation is required. Several sssd logs are checked and skipped because they were rotated less than a week ago, so no post-rotate scripts are executed.

Additional rotation rules are evaluated for vsftpd, xferlog, and wtmp. Missing logs are skipped, weekly and monthly rotation policies are confirmed, and size-based conditions are enforced. The output confirms that all logs are currently within their rotation thresholds and no rotations were performed during this run.

```
egarrido@stage-web-eg3:~  
Now: 2025-10-04 16:48  
Last rotated at 2025-09-28 00:00  
log does not need rotating (log has been rotated at 2025-09-28 00:00, which is less than a week ago)  
considering log /var/log/sssds/sssds_pam.log  
Now: 2025-10-04 16:48  
Last rotated at 2025-09-28 00:00  
log does not need rotating (log has been rotated at 2025-09-28 00:00, which is less than a week ago)  
considering log /var/log/sssds/sssds_procore.dev.log  
Now: 2025-10-04 16:48  
Last rotated at 2025-09-28 00:00  
log does not need rotating (log has been rotated at 2025-09-28 00:00, which is less than a week ago)  
considering log /var/log/sssds/sssds_ssh.log  
Now: 2025-10-04 16:48  
Last rotated at 2025-09-28 00:00  
log does not need rotating (log has been rotated at 2025-09-28 00:00, which is less than a week ago)  
considering log /var/log/sssds/sssds_sudo.log  
Now: 2025-10-04 16:48  
Last rotated at 2025-09-28 00:00  
log does not need rotating (log has been rotated at 2025-09-28 00:00, which is less than a week ago)  
not running postrotate script, since no logs were rotated  
  
rotating pattern: /var/log/vsftpd.log weekly (4 rotations)  
empty log files are rotated, old logs are removed  
considering log /var/log/vsftpd.log  
log /var/log/vsftpd.log does not exist -- skipping  
  
rotating pattern: /var/log/xferlog weekly (4 rotations)  
empty log files are rotated, old logs are removed  
considering log /var/log/xferlog  
Now: 2025-10-04 16:48  
Last rotated at 2025-10-01 00:00  
log does not need rotating (log has been rotated at 2025-10-01 00:00, which is less than a week ago)  
  
rotating pattern: /var/log/wtmp monthly (1 rotations)  
empty log files are rotated, only log files ≥ 1048576 bytes are rotated, old logs are removed  
considering log /var/log/wtmp  
Now: 2025-10-04 16:48  
Last rotated at 2025-09-25 11:00  
log does not need rotating ('minsize' directive is used and the log size is smaller than the minsize value)  
[egarrido@stage-web-eg3 ~]$
```

A terminal session shows a forced log rotation being executed using `logrotate -f` with the main configuration file. An initial attempt to list the Apache log directory fails due to insufficient permissions, followed by a successful listing using elevated privileges.

The directory contents confirm that Apache logs were rotated as expected: current `access_log` and `error_log` files are present alongside timestamped rotated logs, with older logs compressed using `gzip`. File ownership is set to `root`, and timestamps indicate the rotation occurred at the time the command was run.

A terminal window titled 'egarrido@stage-web-eg3:~' with standard window controls. The terminal shows a sequence of commands and their outputs. First, 'sudo logrotate -f /etc/logrotate.conf' is executed. Then, 'ls -lh /var/log/httpd/' is run, resulting in a 'Permission denied' error. Finally, 'sudo ls -lh /var/log/httpd/' is run, displaying a detailed directory listing of the Apache log files, including current logs and their compressed, timestamped counterparts.

```
[egarrido@stage-web-eg3 ~]$ sudo logrotate -f /etc/logrotate.conf
[egarrido@stage-web-eg3 ~]$ ls -lh /var/log/httpd/
ls: cannot open directory '/var/log/httpd/': Permission denied
[egarrido@stage-web-eg3 ~]$ sudo ls -lh /var/log/httpd/
total 68K
-rw-r--r--. 1 root root    0 Oct  4 16:49 access_log
-rw-r--r--. 1 root root 2.3K Sep 27 13:48 access_log-20250928.gz
-rw-r--r--. 1 root root 52K Oct  2 08:36 access_log-20251004
-rw-r--r--. 1 root root 406 Oct  4 16:49 error_log
-rw-r--r--. 1 root root 680 Sep 28 00:00 error_log-20250928.gz
-rw-r--r--. 1 root root 532 Oct  4 16:49 error_log-20251004
[egarrido@stage-web-eg3 ~]$
```

Summary

Configured and validated Apache log rotation policies with defined retention, compression, and delayed compression to ensure consistent and manageable log storage. Verified post-rotation behavior by reloading the httpd service to maintain uninterrupted logging. Executed forced rotation tests and confirmed correct log creation, compression, and retention through filesystem validation. Reviewed logrotate behavior across multiple system services to ensure rotation rules were applied correctly and unnecessary rotations were skipped. Implemented and tested Ansible playbooks for user management, service configuration, and system hardening. Developed and committed supporting Bash scripts for operational checks, performance data collection, and process monitoring. Maintained clean version control practices in GitLab with structured commits and verified repository integrity after each update.