CHECKLIST

Oracle Privileged User Auditing Checklist

A practical checklist for validating Oracle auditing of privileged user activity, including logons, privilege use, sensitive access, and investigation-ready context.

Oracle Privileged User Auditing Checklist

Use this checklist to confirm that your Oracle auditing approach captures the privileged activity that matters, reduces noise, and produces records you can actually use during review or investigation.

1) Define the privileged scope

- [ ] Identify all privileged accounts, including DBA, security admin, support, and break-glass accounts.
- [ ] List privileged roles and system privileges that require auditing.
- [ ] Include proxy users, service accounts, automation identities, and scheduled job accounts.
- [ ] Document any accounts that temporarily assume elevated rights.
- [ ] Separate production, test, and maintenance access paths.

2) Decide what must be audited

Audit events that are typically worth capturing for privileged users:

- [ ] Successful logons for privileged accounts.

- [ ] Failed logons for privileged accounts.
- [ ] Privilege grants, revokes, and role changes.
- [ ] Creation, alteration, and drop of users, roles, or security-relevant objects.
- [ ] Changes to audit settings or other security configuration.
- [ ] Access to sensitive tables, views, or schemas.
- [ ] Administrative commands with material impact on the database or security posture.
- [ ] Password changes, account locks, and account unlocks.

3) Make sure each record is useful

A privileged audit event should answer these questions:

- [ ] Who performed the action?
- [ ] What privilege, object, or account was involved?
- [ ] When did the event occur?
- [ ] From which host or client did the action originate?
- [ ] What program, proxy user, or session context was used?
- [ ] Can the event be tied back to a specific session or transaction when needed?

4) Confirm the auditing model in your Oracle environment

- [ ] Verify which auditing features are available in your Oracle version.
- [ ] Confirm whether you are using unified auditing, legacy auditing, or both.
- [ ] Identify every trail where privileged events may be recorded.
- [ ] Document where audit records are stored.
- [ ] Confirm whether local and centralized audit paths are both enabled.
- [ ] Ensure audit configuration is consistent across primary, standby, and failover systems.

5) Reduce false positives and routine noise

- [ ] Exclude routine activity that is fully covered by approved operational procedures.
- [ ] Define maintenance windows for recurring DBA work.
- [ ] Tag scheduled jobs and automation so they can be recognized in review.
- [ ] Establish host or network expectations for privileged sessions.
- [ ] Distinguish normal administration from unusual privilege use.
- [ ] Review whether repeated, low-value events can be summarized instead of fully logged.

6) Test the audit trail before relying on it

Perform safe test actions in a controlled environment or with approved test accounts:

- [ ] Attempt a successful privileged login.
- [ ] Attempt a failed privileged login.
- [ ] Grant or revoke a test role.
- [ ] Perform a harmless administrative action.
- [ ] Access a non-production sensitive object.
- [ ] Change a security-related setting in a test environment.

Then verify:

- [ ] The event is recorded.
- [ ] The audit record contains the expected identity.
- [ ] The timestamp is correct and consistent.
- [ ] The object, privilege, or action is clearly described.
- [ ] Host, program, proxy user, or session context is present when expected.
- [ ] There are no duplicate records that create confusion.
- [ ] No critical event types are missing.

7) Check operational resilience

- [ ] Confirm audit records survive normal log rotation.
- [ ] Confirm records remain available after restart or failover.
- [ ] Validate retention settings meet security and compliance needs.
- [ ] Review who can read, modify, or delete audit data.
- [ ] Ensure audit trails are protected from unauthorized tampering.
- [ ] Confirm exports are possible for incident response or legal review.
- [ ] Verify backup and recovery procedures include the audit trail.

8) Validate reviewability

- [ ] Make sure analysts can query privileged events without database administration access.
- [ ] Create a repeatable search for logon, privilege use, and sensitive access.
- [ ] Confirm the audit trail can be correlated with tickets, change records, and maintenance windows.
- [ ] Define what constitutes normal versus suspicious privileged activity.
- [ ] Document escalation thresholds for unusual privileged behavior.

9) Assign ownership and response steps

- [ ] Name the owner of the audit configuration.
- [ ] Name the owner of audit review and alert triage.
- [ ] Define how often privileged audit data is reviewed.
- [ ] Document the escalation path for suspicious events.
- [ ] Record how long audit evidence must be retained.
- [ ] Document how to preserve records during an investigation.

10) Quick pass/fail checklist

Use this final check to decide whether the audit design is ready for production use:

- [ ] Privileged users are clearly defined.
- [ ] The right high-risk events are being recorded.
- [ ] Each record can be tied to a user, action, and time.
- [ ] Session context is captured where needed.
- [ ] Routine activity is separated from suspicious activity.
- [ ] Records are durable, retrievable, and reviewable.
- [ ] The trail was tested with safe sample actions.
- [ ] Ownership and retention are documented.

Result

If you can check every box above, your Oracle privileged-user auditing is likely strong enough to support operational review and incident investigation. If several boxes remain unchecked, tighten the scope before you depend on the audit trail in production.