IT GENERAL CONTROLS INTERVIEW GUIDE

MULTI-PLATFORM APPROACH

SkillWeed

CONTENTS

1. LOGICAL ACCESS CONTROLS

GENERAL APPROACH (FRAMEWORK)

Key principle: "I focus on the identity lifecycle - ensuring right person, right access, right time, with continuous monitoring and segregation of duties."

A. NEW USER PROVISIONING

What to say: "For new users, I validate that access requests follow a formal approval process with documented business justification. I verify that access is granted based on role-based access control (RBAC) principles and follows the principle of least privilege."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Review user setup forms, approval workflows in ticketing systems (ServiceNow, Jira), verify default profiles don't have excessive permissions
- **UNIX/Linux**: Check /etc/passwd, /etc/group, validate sudo privileges, review user creation scripts, ensure proper home directory permissions (chmod 700)
- **Windows NT Server**: Examine Active Directory (AD) group policies, verify domain admin assignments are restricted, review Group Policy Objects (GPOs)
- **Large Language Models**: Validate API key generation processes, review access tiers (read-only vs. fine-tuning access), check model endpoint permissions, verify data access controls
- **Cloud (AWS/Azure/GCP)**:
    - **AWS**: IAM user creation, review policies (not using root account), MFA enforcement, check AWS Organizations SCPs
    - **Azure**: Azure AD user provisioning, conditional access policies, Privileged Identity Management (PIM)
    - **GCP**: Cloud Identity management, IAM role bindings, Organization policies

B. TERMINATED USER ACCESS REMOVAL

What to say: "I ensure timely deprovisioning by testing that HR termination triggers immediate access revocation across all systems, typically within 24 hours or same business day for high-risk roles."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Verify automated workflows between HRIS and IAM systems, review termination checklists, check for orphaned accounts
- **UNIX/Linux**: Confirm account locking (passwd -l), review cron jobs for disabled users, check SSH key removal, validate service account ownership transfers

- **Windows NT Server**: AD account disabling, mailbox conversion to shared, BitLocker recovery key access removal, VPN certificate revocation
- **Large Language Models**: API key revocation, model access termination, check for embedded credentials in code repositories, review training data access logs
- **Cloud Platforms**:
  - **AWS**: IAM user/role deletion, access key deactivation, S3 bucket policy updates, CloudTrail review for post-termination activity
  - **Azure**: Azure AD account deletion, revoke SAS tokens, remove from Azure DevOps organizations
  - **GCP**: Remove IAM bindings, delete service accounts, revoke OAuth tokens

C. TRANSFER USERS (ROLE CHANGES)

What to say: "For transfers, I validate that access is recertified based on the new role, previous role access is removed, and there's no accumulation of conflicting privileges that violate segregation of duties."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Review access recertification process, validate SOD matrix, check for orphaned permissions from old roles
- **UNIX/Linux**: Update group memberships, modify sudo rules, review file ownership changes, validate SELinux/AppArmor contexts
- **Windows NT Server**: AD group membership updates, review inherited permissions, validate SharePoint/file server access changes
- **Large Language Models**: Update model access tiers, review fine-tuning permissions vs. inference-only access, validate data classification access
- **Cloud Platforms**:
  - **AWS**: IAM policy updates, resource-based policy modifications, review cross-account role assumptions

- **Azure**: Update Azure RBAC assignments, review subscription-level access, modify PIM eligibility
- **GCP**: Update IAM role bindings, review project-level permissions, validate service account impersonation rights

D. SECURITY SETTINGS

What to say: "I verify security configurations align with industry standards like CIS benchmarks, ensuring password complexity, session timeouts, encryption standards, and logging are properly configured."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Password policies (complexity, history, age), session timeout settings, failed login lockout thresholds, MFA enforcement
- **UNIX/Linux**: PAM configuration, password hashing algorithms (SHA-512), login.defs settings, audit daemon (auditd) rules, fail2ban configuration
- **Windows NT Server**: Group Policy password settings, account lockout policies, audit policy configuration, Windows Defender settings, BitLocker enforcement
- **Large Language Models**: Rate limiting, input validation/sanitization, output filtering for PII/sensitive data, prompt injection protections, model versioning controls
- **Cloud Platforms**:
  - **AWS**: S3 bucket encryption, VPC security groups, KMS key policies, GuardDuty findings, Security Hub standards compliance
  - **Azure**: Azure Security Center recommendations, Network Security Groups (NSGs), Azure Key Vault access policies, Microsoft Defender for Cloud
  - **GCP**: VPC firewall rules, Cloud KMS settings, Security Command Center findings, Binary Authorization policies

E. ENVIRONMENTAL/INFRASTRUCTURE CONTROLS

What to say: "I assess physical and logical environmental controls to ensure system availability, data protection, and resilience against environmental threats."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Data center access logs, environmental monitoring (temperature, humidity), fire suppression systems, UPS/generator testing
- **UNIX/Linux**: System monitoring (Nagios, Zabbix), disk space alerts, RAID array health, temperature sensors, network redundancy
- **Windows NT Server**: Event log monitoring, WSUS/patch management, failover clustering configuration, backup power systems
- **Large Language Models**: Model hosting infrastructure security, GPU cluster monitoring, training environment isolation, inference endpoint health checks
- **Cloud Platforms**:
  - **AWS**: Multi-AZ deployments, Auto Scaling groups, CloudWatch alarms, AWS Systems Manager compliance
  - **Azure**: Availability Zones/Sets, Azure Monitor alerts, Azure Site Recovery configuration
  - **GCP**: Regional distribution, Compute Engine instance groups, Cloud Monitoring alerts, disaster recovery planning

2. CHANGE MANAGEMENT

GENERAL APPROACH (FRAMEWORK)

Key principle: "I ensure all changes follow a structured lifecycle with proper authorization, testing, and segregation between development, testing, and production. Documentation and backout plans are mandatory."

A. STANDARD CHANGES

What to say: "Standard changes follow a formal CAB (Change Advisory Board) process with documented business justification, impact analysis, implementation plan, testing evidence, and backout procedures."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Change tickets in ITSM tools, CAB meeting minutes, change calendar review, success/failure metrics, post-implementation reviews
- **UNIX/Linux**: Package management logs (yum, apt history), kernel upgrade procedures, configuration management tools (Ansible, Puppet logs), /var/log/messages review
- **Windows NT Server**: WSUS deployment schedules, GPO version control, Windows Update logs, System Center Configuration Manager (SCCM) reporting
- **Large Language Models**: Model version control, training dataset versioning, hyperparameter change documentation, A/B testing results, model performance metrics before/after
- **Cloud Platforms**:
  - **AWS**: CloudFormation change sets, Systems Manager Change Manager, Config compliance timeline, AWS Service Catalog approvals
  - **Azure**: Azure Resource Manager template deployments, Azure DevOps pipeline approvals, Azure Policy compliance changes
  - **GCP**: Cloud Deployment Manager configurations, Cloud Build approval processes, Infrastructure Manager change tracking

B. EMERGENCY CHANGES

What to say: "Emergency changes require expedited but documented approval from appropriate authority, with retroactive CAB review. Testing may be abbreviated but must be documented, and backout plans remain mandatory."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Emergency change procedures, after-hours approval chains, abbreviated testing documentation, incident linkage
- **UNIX/Linux**: Emergency patching procedures (zero-day vulnerabilities), emergency system recovery, root cause analysis documentation

- **Windows NT Server**: Emergency security updates, critical system recovery, domain controller restoration procedures
- **Large Language Models**: Emergency model rollback procedures, critical vulnerability patches in ML libraries, emergency content filtering updates
- **Cloud Platforms**:
  - **AWS**: AWS Support emergency escalation, snapshot/AMI rollback procedures, Route 53 failover execution
  - **Azure**: Azure Support rapid response, VM snapshot restoration, Traffic Manager failover
  - **GCP**: Google Cloud Support escalation, persistent disk snapshots, Cloud Load Balancing failover

C. AUTHORIZATION & APPROVAL

What to say: "I verify that changes are authorized by appropriate levels based on risk and impact, with documented approval trails and segregation between requestor, approver, implementer, and tester."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Approval matrix, electronic signatures in change management systems, audit trail review, SOD compliance checks
- **UNIX/Linux**: Sudo logs for privileged changes, Git commit approvals for infrastructure code, peer review requirements
- **Windows NT Server**: PowerShell execution logs, AD change auditing, GPO modification approvals
- **Large Language Models**: Model deployment approvals, training job authorizations, dataset modification approvals, ethical review board sign-offs

- **Cloud Platforms**:
  - **AWS**: IAM CloudTrail logs, Step Functions approval states, CodePipeline manual approval actions
  - **Azure**: Azure Activity Logs, Azure DevOps approval gates, Azure Policy enforcement
  - **GCP**: Cloud Audit Logs, Cloud Build approval requirements, Organization Policy constraints

D. TESTING REQUIREMENTS

What to say: "I ensure changes are tested in non-production environments that mirror production, with documented test cases, expected vs. actual results, and performance impact validation before production deployment."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Test environment validation, test case documentation, UAT sign-offs, performance testing results
- **UNIX/Linux**: Dev/test environment configuration parity, automated testing with Jenkins/GitLab CI, load testing results
- **Windows NT Server**: Test AD domain controllers, lab environment validation, compatibility testing results
- **Large Language Models**: Model validation metrics (accuracy, bias, latency), test dataset performance, adversarial testing, benchmark comparisons
- **Cloud Platforms**:
  - **AWS**: Separate AWS accounts for dev/test/prod, automated testing in CodePipeline, canary deployments, blue-green testing
  - **Azure**: Azure DevTest Labs, pipeline testing stages, deployment slots for testing
  - **GCP**: Separate GCP projects, Cloud Build test stages, traffic splitting for gradual rollouts

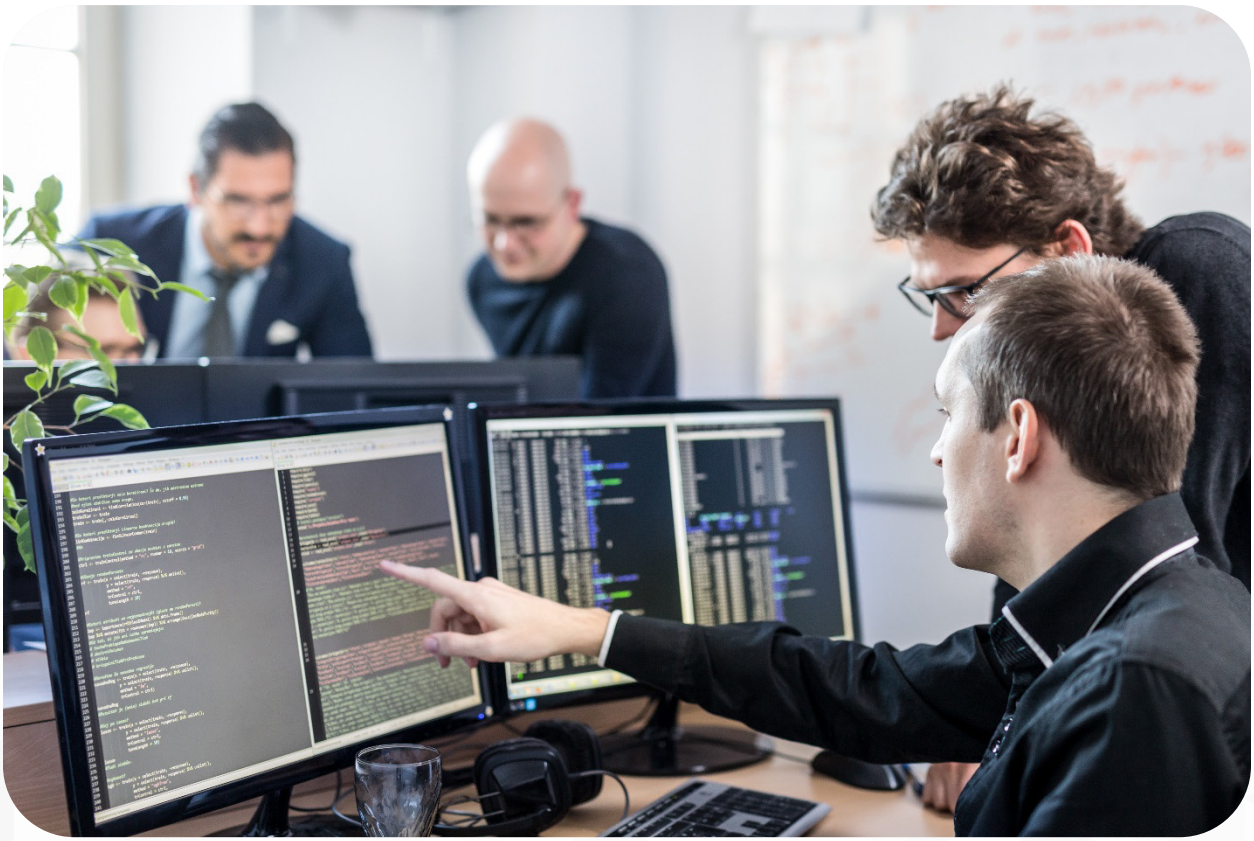
E. SEGREGATION OF DUTIES (SOD)

What to say: "I validate that individuals who develop code cannot deploy to production, those who approve changes don't implement them, and production access is restricted and monitored."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: SOD matrix, role conflict analysis, cross-user monitoring, four-eyes principle for critical changes
- **UNIX/Linux**: Separate developer accounts from production admin accounts, jump hosts/bastion servers, sudo rule segregation
- **Windows NT Server**: Separate AD admin accounts, restricted production access groups, privileged access workstations (PAWs)
- **Large Language Models**: Separate roles for data scientists (development), ML engineers (deployment), and data access administrators
- **Cloud Platforms**:
  - **AWS**: Separate IAM users/roles for dev and production, cross-account deployment, IAM Access Analyzer for permissions
  - **Azure**: Azure RBAC separation, PIM just-in-time access, separate subscriptions for environments
  - **GCP**: IAM role segregation, service account separation, VPC Service Controls for boundary enforcement

3. IT OPERATIONS



GENERAL APPROACH (FRAMEWORK)

Key principle: "I ensure operational resilience through comprehensive backup strategies, automated job monitoring, and structured incident/problem management that minimizes downtime and ensures business continuity."

A. BACKUP AND RECOVERY

What to say: "I verify that backup procedures follow the 3-2-1 rule (3 copies, 2 different media, 1 offsite), with documented RPO/RTO objectives, regular restore testing, and encrypted backup storage."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Backup schedules, retention policies, restore testing logs, backup success/failure reports, encryption validation
- **UNIX/Linux**:
  - Tools: rsync, tar, Bacula, Amanda, Veeam
  - Validate cron jobs, check /var/log/backup logs, verify filesystem snapshots (LVM, ZFS)
  - Test bare-metal recovery procedures
- **Windows NT Server**:
  - Windows Server Backup, Veeam, backup exec
  - Validate VSS (Volume Shadow Copy), test System State backups, AD recovery procedures
  - Verify SQL/Exchange backup integration
- **Large Language Models**:
  - Model checkpoint backups, training state preservation, dataset versioning
  - Version control for model artifacts, experiment tracking (MLflow, Weights & Biases)
  - Validate model registry backups, training pipeline reproducibility
- **Cloud Platforms**:
  - **AWS**: EBS snapshots schedule, RDS automated backups, S3 versioning, AWS Backup service, cross-region replication
  - **Azure**: Azure Backup vaults, VM snapshot policies, SQL Database geo- replication, Azure Site Recovery
  - **GCP**: Persistent disk snapshots, Cloud SQL backups, Cloud Storage versioning, backup and DR service

B. JOB SCHEDULING & BATCH PROCESSING

What to say: "I ensure automated jobs are monitored for successful completion, dependencies are managed, failures trigger alerts, and job execution logs are retained for audit purposes."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**: Job scheduling tools, dependency mapping, SLA monitoring, failed job alerts, execution time trending
- **UNIX/Linux**:
  - Cron jobs (/etc/crontab, /var/spool/cron), at jobs
  - Advanced schedulers: Control-M, AutoSys, Apache Airflow
  - Monitor /var/log/cron, validate email notifications, check script exit codes
- **Windows NT Server**:
  - Task Scheduler, SQL Server Agent jobs
  - Review Event Viewer for task execution, validate job history, check PowerShell script logs
  - Monitor long-running tasks, validate retry logic
- **Large Language Models**:
  - Training job scheduling, inference pipeline orchestration
  - Tools: Kubeflow, MLflow, Apache Airflow for ML workflows
  - Monitor GPU utilization, training epochs completion, model serving health
  - Validate automated retraining schedules
- **Cloud Platforms**:
  - **AWS**: EventBridge (CloudWatch Events), Step Functions, Batch, Lambda scheduled executions, CloudWatch Logs analysis

- **Azure**: Azure Automation, Logic Apps, Azure Functions timer triggers, Azure Data Factory pipelines
- **GCP**: Cloud Scheduler, Cloud Composer (Managed Airflow), Cloud Functions scheduled, Workflows

C. PROBLEM AND INCIDENT MANAGEMENT

What to say: "I ensure incidents are tracked from detection through resolution with proper categorization, priority assignment, escalation procedures, and root cause analysis to prevent recurrence."

PLATFORM-SPECIFIC APPLICATIONS:

- **General Application**:
  - ITSM tools (ServiceNow, Jira Service Management, Remedy)
  - Incident categorization, SLA compliance, escalation matrices
  - Problem management for recurring incidents, known error database (KEDB)
  - Post-incident reviews and corrective action plans
- **UNIX/Linux**:
  - System monitoring: Nagios, Zabbix, Prometheus, Grafana
  - Log aggregation: ELK stack (Elasticsearch, Logstash, Kibana), Splunk
  - Analyze /var/log/messages, /var/log/syslog, application logs
  - Review system metrics: CPU, memory, disk I/O, network
- **Windows NT Server**:
  - Event Viewer analysis (Application, Security, System logs)
  - SCOM (System Center Operations Manager), SCCM reporting
  - Performance Monitor baselines

- Windows Admin Center for centralized monitoring

» **Large Language Models**:

- Model performance degradation detection
- Inference latency monitoring, error rate tracking
- Model drift detection, data quality monitoring
- Bias and fairness incident tracking
- Tools: Model monitoring platforms (Arize, Fiddler, WhyLabs)

» **Cloud Platforms**:

- **AWS**: CloudWatch alarms, AWS Systems Manager OpsCenter, AWS Health Dashboard, X-Ray for tracing, CloudTrail for audit
- **Azure**: Azure Monitor, Application Insights, Azure Service Health, Log Analytics, Alert rules
- **GCP**: Cloud Monitoring (formerly Stackdriver), Cloud Logging, Error Reporting, Cloud Trace, SLO monitoring

INTERVIEW TIPS & KEY TALKING POINTS

1. DEMONSTRATE RISK-BASED THINKING

- "I prioritize controls based on business impact and data sensitivity"
- "I consider the CIA triad - Confidentiality, Integrity, and Availability"
- "I align controls with frameworks like COBIT, ITIL, NIST, ISO 27001"

2. SHOW AUTOMATION & EFFICIENCY

- "I leverage automation to reduce manual errors and improve consistency"
- "I implement continuous monitoring rather than point-in-time testing"
- "I use infrastructure as code for repeatable, auditable deployments"

3. EMPHASIZE DOCUMENTATION & AUDIT TRAILS

- "Proper documentation is essential for audit evidence and knowledge transfer"
- "I ensure all privileged actions are logged and regularly reviewed"
- "I maintain runbooks and standard operating procedures"

4. DISCUSS COMPLIANCE & STANDARDS

- "I ensure controls meet requirements for SOX, PCI-DSS, HIPAA, GDPR as applicable"
- "I follow CIS benchmarks and vendor security best practices"
- "I participate in internal and external audits"

5. HIGHLIGHT CONTINUOUS IMPROVEMENT

- "I analyze control failures to strengthen future processes"
- "I stay current with emerging threats and security advisories"
- "I conduct regular control effectiveness reviews and maturity assessments"

6. PLATFORM-SPECIFIC EXPERTISE STATEMENTS

For Cloud: "In cloud environments, I focus on shared responsibility model understanding— knowing what the provider secures versus what we must secure. I emphasize identity as the new perimeter, implement least privilege through cloud-native IAM, and leverage cloud- native security tools."

For LLMs: "For large language models, I address unique risks like prompt injection, data poisoning, model inversion attacks, and unintended data exposure. I ensure proper access controls for training data, model weights, and inference endpoints, plus monitoring for model drift and ethical issues."

For Traditional Infrastructure: "For on-premises systems, I focus on network segmentation, privileged access management, and compensating controls where cloud- native security features aren't available."

SAMPLE INTERVIEW RESPONSES

Question: "How do you handle terminated user access in AWS?"

Strong Answer: "When a user is terminated, I first verify that the HRIS termination triggers an automated workflow through our identity management system. In AWS specifically, I ensure their IAM user is immediately disabled, access keys are deleted, and any assumed roles are revoked. I check CloudTrail logs to confirm no post-termination activity occurred.

For contractors or third-party access using federated identities, I verify the federation trust is removed. I also review resource-based policies—like S3 bucket policies or Lambda resource policies—to ensure the user isn't granted access through alternative means.

Finally, I document the deprovisioning in our ticketing system with timestamps and verification evidence for audit purposes.

A key consideration is checking for programmatic access like SDK credentials or API keys that might be stored in code repositories or local machines, which require additional cleanup steps."

Question: "Describe your approach to change management for a critical production system."

Strong Answer: "For critical production systems, I follow a rigorous change management process. First, the change requires documented business justification and CAB approval with appropriate stakeholders including security, operations, and business owners.

The change must be tested in a non-production environment that mirrors production configuration. I require documented test cases with expected versus actual results, including performance impact analysis. We implement segregation of duties—the developer cannot approve or deploy their own change.

For deployment, I ensure we have a documented backout plan and schedule the change during approved maintenance windows with appropriate communication to stakeholders. Post-deployment, we conduct smoke testing and monitor key metrics for anomalies.

For example, in a recent database schema change for a financial application, we tested in dev and QA environments, performed load testing to ensure performance impact was acceptable, had DBA review and approve, executed during low-traffic hours, and monitored transaction processing rates post-deployment. We also maintained a database backup taken immediately before the change to enable rapid rollback if needed."

Question: "How do you ensure backups are actually recoverable?"

Strong Answer: "Having backups is meaningless if they can't be restored. I implement a regular restore testing schedule—not just verifying backup jobs completed successfully, but actually performing restore operations in a test environment.

For critical systems, we conduct quarterly full restore tests where we restore data to a separate environment and validate data integrity and application functionality. For less critical systems, we do sampling—testing a subset of backups on a rotating basis.

I document restore test results including the time taken to restore, any issues encountered, and data validation steps. This helps us validate our RTO objectives are achievable and identify any backup configuration issues before a real disaster.

In cloud environments like AWS, I use automation to snapshot EC2 instances and test launching new instances from snapshots. For databases, we restore RDS snapshots to test instances and run data integrity checks. I also ensure backup encryption keys are accessible and documented in our disaster recovery procedures—I've seen cases where backups existed but couldn't be decrypted during recovery because key access wasn't properly maintained."

QUICK REFERENCE CHECKLIST

BEFORE THE INTERVIEW:

- Review specific technologies mentioned in job description
- Prepare 2-3 specific examples from your experience for each control area
- Review recent security incidents/patches relevant to discussed platforms
- Familiarize yourself with the organization's industry compliance requirements
- Prepare questions about their current control environment

DURING THE INTERVIEW:

- Use the STAR method (Situation, Task, Action, Result) for examples
- Relate answers to business risk and impact, not just technical details
- Mention specific tools and frameworks you've used
- Ask clarifying questions about their environment before answering
- Demonstrate both depth (technical details) and breadth (across platforms)


SkillWeed