**Technical Design Document (TDD)**

**Enterprise Databricks Clusters Architecture & Implementation**

**1. Document Overview**

**Title:**  
Enterprise Databricks Clusters Architecture and Implementation

**Author:**  
[Your Name], Data Platform Architect

**Date:**  
[Date]

**Version:**  
1.0

**2. Objective**

Design and implement a **secure, governed, scalable cluster architecture** in Databricks that:

* Standardizes cluster creation across environments
* Enforces cost controls and security compliance
* Supports diverse workloads (ETL, ML, Ad Hoc Analysis)
* Integrates with enterprise IAM and network controls

**3. Scope**

* Cluster policies and configurations
* Cluster lifecycle management (creation, scaling, termination)
* Access controls for cluster usage
* Network security integration
* Cost management practices
* Non-functional requirements (security, performance, compliance)

**4. Architectural Overview**

**4.1 Key Cluster Types**

| **Cluster Type** | **Description** |
| --- | --- |
| **Interactive Cluster** | On-demand compute for ad hoc analysis and development |
| **Job Cluster** | Ephemeral clusters provisioned automatically by scheduled jobs |
| **High-Concurrency** | Shared clusters optimized for multiple concurrent SQL and notebook workloads |

**4.2 High-Level Architecture Diagram**

*(Visual representation of:)*

* Workspace(s)
* Cluster pools
* Virtual Networks and Private Endpoints
* Unity Catalog integration for data access control
* Audit logging pipeline

**5. Design Considerations**

**5.1 Cluster Policy Strategy**

* **Cluster Policies** enforce standardized configurations:
  + Instance types
  + Auto termination settings
  + Library installation controls
  + Credential Passthrough enforcement

**5.2 Network Security**

* Clusters deployed in **customer-managed VNet**:
  + Private Link to storage and data sources
  + Restricted outbound internet access
* Subnets pre-allocated per environment (Dev, QA, Prod)

**5.3 Compute Sizing & Auto-Scaling**

* Interactive clusters:
  + Small node types (e.g., Standard\_DS3\_v2) for development
* Job clusters:
  + Auto-scaling enabled with min/max workers
* ML workloads:
  + GPU-enabled clusters (e.g., Standard\_NC6s\_v3)

**5.4 Credential Passthrough**

* **Azure AD Passthrough** or **AWS IAM Roles** enforced to ensure:
  + Data access permissions align with user identity
  + No shared credentials embedded in clusters

**5.5 Cluster Pools**

* Create pools for frequently used instance types:
  + Faster startup time
  + Lower cost per job
* Separate pools by workload type (ETL, ML, Analytics)

**5.6 Library Management**

* Cluster-init scripts:
  + Install approved libraries from internal repositories
* Restrict use of unapproved packages via cluster policies

**5.7 Cost Controls**

* **Auto Termination** enforced (e.g., 60-minute idle timeout)
* Daily cost reporting and chargeback tagging
* Budget thresholds and alerts

**5.8 Monitoring & Logging**

* Enable **Ganglia metrics** for resource utilization
* Stream cluster logs to centralized storage/SIEM:
  + Driver logs
  + Executor logs
  + Event logs

**6. Non-Functional Requirements**

| **NFR Area** | **Description** |
| --- | --- |
| **Security** | - Credential Passthrough for all production clusters - Encrypted communication and storage - RBAC enforcement for cluster usage |
| **Scalability** | - Support for 500+ concurrent users - Elastic auto-scaling of clusters - Cluster pools to optimize startup times |
| **Availability** | - SLA target 99.9% uptime for production workloads - Regional redundancy of data dependencies |
| **Compliance** | - GDPR, SOC2 compliance - Secure logging and audit trails retained for 1+ years |
| **Performance** | - Median startup time <3 minutes with cluster pools - Auto-scaling responsive to workload spikes |
| **Observability** | - Near real-time monitoring dashboards - Alerting for failed jobs and resource exhaustion |
| **Cost** | - Automated shutdown of idle clusters - Usage attribution by business unit and project |

**7. Implementation Steps**

**7.1 Prerequisites**

* Network infrastructure (VNets, Subnets, Private Endpoints) configured
* IAM integrations (Azure AD/AWS IAM roles)
* Security baselines approved by compliance teams

**7.2 Cluster Policies Configuration**

1. Create policy for **Interactive Clusters**:
   * Auto termination: 60 min
   * Allowed node types: Standard\_DS3\_v2
   * Credential Passthrough: Required
2. Create policy for **Job Clusters**:
   * Auto termination: 20 min
   * Auto-scaling: Enabled (min 2, max 10 workers)
3. Create policy for **ML Clusters**:
   * GPU node types allowed
   * Restricted libraries pre-installed

**7.3 Cluster Pools Setup**

* Define pools:
  + ETL workloads: Standard\_DS3\_v2
  + ML workloads: Standard\_NC6s\_v3
* Validate pool auto-replenishment
* Monitor utilization

**7.4 Library Management**

* Create init scripts:
  + Install whitelisted libraries
  + Configure logging directories
* Store scripts in secure storage

**7.5 Credential Passthrough Enforcement**

* Enable for all production clusters:

sql

CopyEdit

SET spark.databricks.passthrough.enabled=true

* Validate user-level access controls

**7.6 Cluster Creation Validation**

* Test policy enforcement:
  + Attempt disallowed configurations (should fail)
  + Confirm allowed configurations succeed
* Validate performance benchmarks

**7.7 Logging and Monitoring Setup**

* Enable Ganglia and Spark UI metrics
* Configure log delivery to secure storage
* Integrate with monitoring dashboards

**7.8 Operational Handover**

* Provide operational playbooks:
  + Creating clusters
  + Monitoring utilization
  + Rotating credentials
  + Responding to incidents

**8. Risks & Mitigations**

| **Risk** | **Mitigation** |
| --- | --- |
| Cost overruns due to idle clusters | Enforce auto termination, monitor usage, set alerts |
| Misconfigured clusters bypassing policies | Mandatory cluster policies enforced by workspace admins |
| Credential leakage | Use passthrough authentication and secure init scripts |
| Unauthorized access | RBAC controls for who can create and attach clusters |
| Performance degradation with large jobs | Enable auto-scaling, monitor cluster metrics, adjust sizing recommendations |

**9. Future Enhancements**

* **Attribute-Based Access Control (ABAC)** for cluster usage
* **Terraform Automation** of cluster lifecycle
* **Serverless compute adoption** for lower operational overhead
* **Enhanced GPU workload orchestration**

**10. Appendix**

* Example cluster policy JSON
* Init script samples
* Monitoring dashboard templates
* Terraform modules for cluster deployment

**11. Approval & Sign-Off**

| **Name** | **Role** | **Signature** |
| --- | --- | --- |
| [Name] | Enterprise Data Architect |  |
| [Name] | Security & Compliance Lead |  |
| [Name] | Platform Owner |  |

**Final Architect Notes**

This cluster design establishes:

* **Governed, secure, and standardized compute environments**
* Strong **cost management and operational visibility**
* Flexibility to support **diverse analytics and ML workloads**
* A foundation for **future scalability and automation**