Lab 1C – Terraform EC2 → RDS Integration Verification Report

Overview

This report documents the successful verification of **Lab 1C (Terraform)**, demonstrating an end-to-end, infrastructure-as-code deployment in AWS where an EC2 instance securely connects to an RDS MySQL database using **IAM roles** and **AWS Secrets Manager**, with **no static credentials**. All resources were provisioned via Terraform and validated using AWS CLI and in-instance checks.

This report follows the structure and intent outlined in the provided markdown templates (INDEX.md, 00_START_HERE.md, FILE_MANIFEST.md) and incorporates the captured outputs from `1A Verification Checks2.txt`.

Architecture Summary

Pattern Implemented: EC2 (Application Host) → IAM Role → Secrets Manager → RDS (MySQL)

Key Design Principles: - Infrastructure as Code (Terraform only) - Least-privilege IAM - No hard-coded secrets - Private database access - Verifiable via AWS CLI and Systems Manager

Core AWS Services Used: - Amazon EC2 (Amazon Linux 2023) - Amazon RDS (MySQL) - AWS Secrets Manager - AWS Systems Manager (Session Manager) - IAM Roles & Instance Profiles - VPC, Subnets, Security Groups

---

Deployment Method (Lab 1C)

- All infrastructure was provisioned using **Terraform** (`main.tf`)
- No resources were created manually in the AWS Console
- IAM permissions, networking, secrets, and compute were declared explicitly

This satisfies the Lab 1C requirement of **Terraform-only deployment**.

Verification Strategy

Verification was performed in three layers:

1. **Control Plane Checks (AWS CLI from local machine)**
2. **Data Plane Checks (inside EC2 via SSM Session Manager)**

3. **Security & Policy Validation**

Automated and manual checks align with the lab's verification guidance.

Verification Results

1. Secrets Manager

Check: Secret existence

- Secret ID: `lab1a/rds/mysql`
- Result: ✅PASS

The secret exists and contains structured database connection data (username, password, host, port, dbname).

---

2. EC2 Instance IAM Attachment

Check: EC2 instance has an IAM instance profile

- Instance ID verified via `describe-instances`
- Instance Profile ARN present
- Result: ✅PASS

This confirms the EC2 instance does not rely on static credentials.

---

3. Instance Profile → Role Resolution

Check: Instance profile correctly resolves to IAM role

- Instance Profile: `arcanum-instance-profile01`
- IAM Role: `arcanum-ec2-role01`
- Result: ✅PASS

---

4. IAM Policy Validation (Secrets Manager)

Check: Role has permission to read Secrets Manager

Attached policies include: - `AmazonSSMManagedInstanceCore` - `CloudWatchAgentServerPolicy` - Custom policy: `secrets_policy`

The custom policy grants: - `secretsmanager:GetSecretValue` - `secretsmanager:DescribeSecret`

Result: ✅PASS (least-privilege access confirmed)

---

5. Systems Manager Connectivity

Check: EC2 instance is managed by SSM

- PingStatus: `Online`
- Session Manager access confirmed
- No SSH keys required

Result: ✅PASS

---

6. In-Instance Identity Verification (SSM)

Command (inside EC2):

```
aws sts get-caller-identity
```

Observed: - ARN contains `assumed-role/arcanum-ec2-role01`

Result: ✅PASS

This confirms the EC2 instance is assuming the intended IAM role.

---

7. In-Instance Secrets Access

Checks (inside EC2): - `secretsmanager:DescribeSecret` - `secretsmanager:GetSecretValue`

Both operations succeeded using **role-based credentials only**.

Result: ✅PASS

---

8. Database Connectivity

- RDS is deployed in private subnets
- `publicly_accessible = false`
- Security group allows MySQL (3306) **only from EC2 security group**
- Credentials retrieved dynamically from Secrets Manager

Result: ✅PASS

Security Posture Review

Control	Status
No plaintext credentials	✅
IAM role for EC2	✅
Least-privilege secrets access	✅
Private RDS	✅
SG-restricted DB access	✅
SSM instead of SSH	✅

No wildcard principals were detected in secret resource policies.

Files Used for Verification

- `main.tf` – Infrastructure definition
- `verify_secrets_and_iam.sh` – Control-plane verification
- `verify_ec2_secrets_access.sh` – In-instance checks
- `1A Verification Checks2.txt` – Captured proof output

Documentation references: - `INDEX.md` - `00_START_HERE.md` - `FILE_MANIFEST.md`

Lab 1C Completion Statement

This lab demonstrates a complete, production-grade AWS pattern:

> **An EC2 application securely connects to an RDS database using IAM roles and AWS Secrets Manager, fully provisioned and reproducible via Terraform.**

All required verification checks have passed. The system meets security, automation, and architectural requirements for **Lab 1C – Terraform**.

Final Status

Lab 1C (Terraform): ✅COMPLETE

All infrastructure deployed, verified, and documented successfully.