ProtoVault Breach – Incident Report (CTF Write-Up)

■ Objective

Trace the origin of a simulated data breach, uncover the vulnerability that led to the exposure, and verify the compromised data.

■ Incident Timeline

- Email ransom received. - Flask app analyzed → hardcoded credentials found in app.py. - Git logs showed 'Remove backup scripts' commits by Walter. - Recovered app/util/backup_db.py from previous commit. - Script used pg_dump + ROT13 + upload to public S3 bucket. - Leak confirmed at s3://protoguard-asset-management/db_backup.xyz - Naomi Adler’s hash recovered after decoding dump.

■ Verified Data Exposure

Public Leak URL:
https://protoguard-asset-management.s3.us-east-2.amazonaws.com/db_backup.xyz
Leaked file name: db_backup.xyz
Extracted hash: pbkdf2:sha256:600000$YQqIvcDipYLzzXPB$598fe450e5ac019cdd41b4b10c5c21515573ee63a8f4881f7d721fd74ee43d59

■ Root Cause Analysis

- Hardcoded database credentials in app.py
- Insecure backup automation (ROT13 + public S3)
- Lack of IAM restriction on bucket
- Git history exposed deleted sensitive scripts.

■■ Recommended Mitigations

1. Remove hardcoded secrets (use env vars / secret manager).
2. Make all S3 buckets private.
3. Encrypt backups properly (AES-256, KMS).
4. Rotate exposed credentials.
5. Use pre-commit secret scanners (git-secrets, truffleHog).
6. Improve audit logging and IR readiness.

■ Summary of Findings

Leaking file: app/util/backup_db.py
Leaked location:
https://protoguard-asset-management.s3.us-east-2.amazonaws.com/db_backup.xyz
Leaked credential: assetdba (PostgreSQL)
Proof: Naomi Adler password hash
Attack vector: Misconfigured backup automation script.