

Reviewer name: Mingxu XUE a1933289

Group: project group 68

Testing approaches / tools used

Static analysis:

```
>>
[main] INFO     profile include tests: None
[main] INFO     profile exclude tests: None
[main] INFO     cli include tests: None
[main] INFO     cli exclude tests: None
[main] INFO     running on Python 3.13.8
Run started:2025-10-15 06:25:31.427378

Test results:
>> Issue: [B608:hardcoded_sql_expressions] Possible SQL injection vector through string-based query construction.
Severity: Medium   Confidence: Medium
CWE: CWE-89 (https://cwe.mitre.org/data/definitions/89.html)
More Info: https://bandit.readthedocs.io/en/1.8.6/plugins/b608\_hardcoded\_sql\_expressions.html
Location: .\datavault.py:165:37
164         for table in data.keys():
165             cur = conn.execute(f"SELECT * FROM {table}")
166             cols = [c[0] for c in cur.description]

>> Issue: [B505:weak_cryptographic_key] RSA key sizes below 2048 bits are considered breakable.
Severity: Medium   Confidence: High
CWE: CWE-326 (https://cwe.mitre.org/data/definitions/326.html)
More Info: https://bandit.readthedocs.io/en/1.8.6/plugins/b505\_weak\_cryptographic\_key.html
Location: .\quarantinepoc_weak_key_register.py:38:10
37         # generate a 1024-bit RSA key (INTENTIONAL WEAK KEY for PoC)
38         key = rsa.generate_private_key(public_exponent=65537, key_size=1024)
39         pub = key.public_key()

Code scanned:
Total lines of code: 2481
Total lines skipped (#nosec): 0

Run metrics:
Total issues (by severity):
  Undefined: 0
  Low: 11
  Medium: 2
  High: 0
Total issues (by confidence):
  Undefined: 0
  Low: 0
  Medium: 2
  High: 11
```

High-level findings

[High] Private key files (.pem) committed to the repository.

[High] Built-in backdoor flags (weak RSA keys and unsigned message acceptance).

[Medium] Non-standard password hashing (single SHA-256 with salt, no iterations).

Detailed findings, impact, and remediation

[High] Private keys included in repository

Impact: Private and public key pairs are stored in the '.keys/' directory, allowing impersonation and data decryption.

Remediation: Remove all keys from version control, regenerate keys securely at deployment, and add CI secret scanning.

[High] Backdoor flags enabled

Impact: Code defines BACKDOOR_WEAK_KEYS and BACKDOOR_TRUST_GOSSIP that bypass security validation.

Remediation: Disable these flags in production, separate them into a lab-only branch, and enforce strict key validation.

[Medium] Weak password hashing scheme

Impact: Single SHA-256 + salt used in datavault.py without iterations.

Remediation: Replace with Argon2id or PBKDF2 ($\geq 200,000$ iterations) and migrate existing hashes gradually.

Recommendations

1. Remove all private keys and regenerate securely using KMS or Vault.
2. Eliminate or isolate backdoor logic; require build-time flags for any lab use.
3. Adopt modern password hashing (Argon2id or PBKDF2).
4. Replace all try/except: pass with specific error handling and logging.
5. Enable TLS (wss://) for all client-server communication.