# security: block NQ Vault path traversal on decrypted file writes #669

Merged **stark4n6** merged 2 commits into `abrignoni:main` from `mobasi-team:security/nq-vault-pat…` on Mar 4



**mobasi-team** commented on Mar 2 — Contributor

## Commit

- `0cafd8f` - security: block NQ Vault path traversal on decrypted file writes

## Security issue

`NQ_Vault.py` used attacker-controlled `file_name_from` from DB directly in:

- `open(join(report_folder, decrypted_file_name), 'wb')`

This allowed traversal values like `../../../outside_written.bin` to write outside the report output tree.

## Vulnerability verification

Confirmed as real in a controlled local PoC on pre-fix code:

- Input `old_filename = '../../../outside_written.bin'`
- Output showed:
  - `outside_exists: True`
  - `is_outside_report: True`

## Fix implemented

- Added filename/path hardening helpers in `NQ_Vault.py`:
  - `_sanitize_output_filename(...)`
  - `_build_safe_output_path(...)`
- Enforces:
  - basename extraction (drop path components)
  - invalid character stripping
  - dot/empty fallback handling
  - resolved-path confinement to `report_folder`
  - collision-safe output naming
- Replaced vulnerable write with safe resolved output path.

## Regression tests added

- New test file:
  - `admin/test/scripts/test_nq_vault_path_security.py`
- Covers:
  - traversal sanitization
  - output-path confinement to report folder
  - end-to-end file_decryption traversal-block behavior

## QA run

- `python3 admin/test/scripts/test_nq_vault_path_security.py`
- `python3 -m py_compile scripts/artifacts/NQ_Vault.py admin/test/scripts/test_nq_vault_path_security.py`

**mobasi-team** added 2 commits last month

`security: block NQ Vault path traversal on decrypted file writes` ✗ `0cafd8f`

`chore: resolve NQ Vault lint findings for security PR` ✓ `4924c15`

**abrignoni** requested review from **abrignoni** and **stark4n6** last month

**abrignoni** approved these changes on Mar 3

View reviewed changes

**abrignoni** left a comment — Owner

Code looks good. Sanitizes names and path.

**stark4n6** approved these changes on Mar 4

View reviewed changes

**stark4n6** commented on Mar 4 — Collaborator

@mobasi-team @abrignoni looks good to me as well, I'm wondering if the sanitization should be utilized as a function for other parsers too, something we can look at

**stark4n6** merged commit `9b2a9b8` into `abrignoni:main` on Mar 4

View details

1 check passed

---

**mobasi-team** commented on Mar 4 — Contributor, Author

@stark4n6 @abrignoni

We found these as being places where there's a risk of file path traversal, but it's much harder for an attacker to take advantage of these.

- ALEAPP: `Cello.py` (line 108) uses DB `doc_name` as output path (`os.path.join(report_folder, doc_name)`) without confinement/sanitization.
- ALEAPP: `googlePhotos.py` (line 335) and `googlePhotos.py` (line 397) use DB keys/filenames directly in destination paths.
- ALEAPP: activity/map parsers write files from untrusted activity IDs without path safety checks:
  - `GarminPolyAPI.py` (line 104)
  - `GarminPolyline.py` (line 83)
  - `PumaActivities.py` (line 101)
  - `NikePolyline.py` (line 79)
  - `AdidasActivities.py` (line 90)
  - `RunkeeperActivities.py` (line 102)

    - `MMWActivities.py` (line 139)

- iLEAPP: `vippsContacts.py` (line 58) writes `row[3]` directly to `os.path.join(report_folder, ...)`.
- iLEAPP: `mobileInstall.py` (line 593) writes files from DB `distinctbundle` directly under report output.

Harder to exploit, but present:

- iLEAPP Photos parser family uses DB filename fields in output filenames without sanitization (prefixed, but still unsafe pattern), for example:
    - `Ph10AssetParsedEmbeddedFiles.py` (line 89)
    - `Ph70UserAdjustDateTimezoneLocation.py` (line 247)
    - `Ph15PeopleandDetFacesNAD.py` (line 338)
    - `Ph16AssetPeopleandDetFaces.py` (line 388)
- iLEAPP: `protonMail.py` (line 161) uses decrypted attachment filename in output path (random prefix reduces exploitability but doesn't sanitize).



**Reviewers**

**abrignoni** ✓

**stark4n6** ✓

**Assignees**

No one assigned

**Labels**

None yet

**Projects**

None yet

**Milestone**

No milestone

---

**Development**

Successfully merging this pull request may close these issues.

None yet

---

**3 participants**