0-DAY NO CVE HIGH PUBLIC

n8n, Databricks path join to secrets API

n8n 2.42.0 Databricks Genie getSpace concatenates spaceId with no toPathSegment. Untrusted webhook input becomes GET /api/2.0/secrets/get under the workspace token. Sibling of GHSA-89p4 / GHSA-rqch. No CVE yet.

NAME n8n, Databricks path join to secrets API	TYPE 0-day analysis
CVE n/a	CVE RISK high
DISCLOSURE STATUS public	VENDOR n8n
AFFECTED n8n through 2.42.0 (86c23326); unpublished. Needs a workflow that holds a Databricks credential and binds untrusted input into a path id. Image n8nio/n8n:2.42.0.	PUBLISHED 29 Sept 2026
UPDATED 29 Sept 2026	TAGS 0-day, n8n, databricks, path-traversal, cwe-22, unauthenticated

TL;DR

Lab files and proof-of-concept here.

01 toPathSegment exists. Databricks does not call it

The proof of concept is on GitHub: **abraxas/n8n-databricks-path-join** (loopback client; @abraxas_null). The lab stack is `lab/` : Dockerfile, docker-compose.yml, run.sh, mock/. Authorized lab only. It talks to loopback.

This is **n8n 2.42.0** (`86c23326`), n8n GmbH. No CVE yet. CWE-22. 8.6 High (`CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:C/C:H/I:N/A:N`). Unauthenticated at the webhook if a victim workflow already bound untrusted JSON into Genie `spaceId` (or Vector Search `indexName` , Files `filePath`). Credential leak on the **Databricks** tenant. Not n8n host RCE.

Same product, sibling: hidden Function vm2.

02 How I found it

Same pass as the Function lab: **n8n@2.42.0** after the 16 September 2026 GHSA wave. I was looking for incomplete siblings of GHSA-89p4-6h98-c7xm (n8n Public API resource ids) and GHSA-rqch-9jrh-cr8w (Supabase table name). Those patches introduced `toPathSegment` : `encodeURIComponent` plus reject `''` / `.` / `..` . Elastic, Adalo, Currents, the n8n node itself. The helper exists because they already lost this fight.

`getSpace.operation.ts` concatenates `spaceId` into `${host}/api/2.0/genie/spaces/${spaceId}` . Vector Search `getIndex` and Files `downloadFile` are the same class. No `toPathSegment` . No stay-under-prefix check. LangChain `VectorStoreDatabricks` already wraps the index path. Incomplete sibling **inside** the product.

WHATWG `new URL` is the rest. `../` normalizes. `?` becomes query. Client-side: `https://host/api/2.0/genie/spaces/` plus `../../secrets/get?scope=s&key=k` is `https://host/api/2.0/secrets/get?scope=s&key=k` . undici will send that. The workflow's Databricks PAT goes with it.

I did not aim this at a real workspace. Lab mock on `:18203` answers `GET /api/2.0/secrets/get` with `N8N-DBX-SECRET-WITNESS`. n8n on `:18202`. Webhook `dbx-space` binds `spaceId` from JSON. Mock access log: `GET /api/2.0/secrets/get?scope=s&key=k`. Webhook body contains the witness.

I am not printing a webhook JSON you can paste at someone else's `/webhook/...`. The missing `toPathSegment` is the useful part.

Wrong turns already recorded: request staying under `/api/2.0/genie/spaces/` (then `..` was encoded - lab requires it left); treating n8n host RCE as SUCCESS; hitting a real workspace; a reverse shell. Theatre. The witness is `N8N-DBX-SECRET-WITNESS` plus the mock request line.

03 Concat vs the helper they already wrote

JAVASCRIPT Copy

```javascript
const spaceId = this.getNodeParameter('spaceId', i) as string;
const response = await databricksApiRequest(this, credentialType, {
    method: 'GET',
    url: `${host}/api/2.0/genie/spaces/${spaceId}`,
```

`toPathSegment`:

JAVASCRIPT Copy

```javascript
export function toPathSegment(id: unknown): string {
    if (id === null || id === undefined) {
        throw new UserError('Invalid identifier: a value is required');
    }
    const value = String(id);
    if (value === '' || value === '.' || value === '..') {
        throw new UserError(`Invalid identifier: "${value}" is not allowed`);
    }
    return encodeURIComponent(value);
}
```

LangChain VectorStoreDatabricks already does `${host}/api/2.0/vector-search/indexes/${toPathSegment(indexName)}`. nodes-base Databricks does not.

04 What an attacker can do

Hit a public webhook, chat, or HTTP trigger that flows into Databricks Genie get-space (or Vector Search get-index, or Files download) with an attacker-controlled id. The workflow token then `GET`s `/api/2.0/secrets/get`. If that token may read the scope (typical workspace admin PAT), the node output is the secret `value` JSON. The workflow emits it if it replies on the webhook, a chat, or a later HTTP node.

That is credential leak **on Databricks**, not the n8n database. Not n8n RCE. Token without secrets ACL still 403s; that still proves the request left `/genie`. Files download needs a deeper `../` (longer prefix). Needs a victim who bound untrusted input into that parameter. A locked-down HTTP Request node (`allowedHttpRequestDomains: none`) is not a substitute for encoding the Databricks ids.

05 The lab (run this at home)

Source of truth is `lab/`. Image `n8nio/n8n:2.42.0` plus a loopback Databricks mock. n8n **18202**, mock **18203**. Bind both to loopback. Credential host is the mock, not a real workspace.

DOCKERFILE Copy

```
# Pin n8n 2.42.0. Fallback registry: docker.n8n.io/n8nio/n8n:2.42.0
FROM n8nio/n8n:2.42.0
```

`docker-compose.yml` (truncated; full file on GitHub):

YAML Copy

```yaml
name: n8n-databricks-path-join

services:
  mock:
    build:
      context: ./mock
    image: n8n-databricks-path-join-mock:lab
    ports:
      - "127.0.0.1:18203:8080"
    networks:
      - labnet

  n8n:
    build:
      context: .
      dockerfile: Dockerfile
    image: n8n-databricks-path-join:2.42.0
    ports:
      - "127.0.0.1:18202:5678"
    environment:
      N8N_SECURE_COOKIE: "false"
      N8N_HOST: "127.0.0.1"
      N8N_PORT: "5678"
      N8N_PROTOCOL: http
      N8N_LISTEN_ADDRESS: "0.0.0.0"
      WEBHOOK_URL: "http://127.0.0.1:18202/"
      N8N_EDITOR_BASE_URL: "http://127.0.0.1:18202/"
      N8N_ENCRYPTION_KEY: "lab-n8n-databricks-path-join-key"
      N8N_DIAGNOSTICS_ENABLED: "false"
      N8N_HIRING_BANNER_ENABLED: "false"
    depends_on:
      mock:
        condition: service_healthy
    networks:
      - labnet

networks:
  labnet:
```

PLAIN TEXT Copy

```
git clone https://github.com/abraxas/n8n-databricks-path-join
cd n8n-databricks-path-join/lab
```

```
./run.sh
```

The proof of concept is written for `127.0.0.1:18202`. Do not publish the port off loopback.

06 What the tree actually consumes

Owner setup. Databricks credential pointed at the mock. Webhook to Genie get-space with `spaceId` from JSON. Activate. POST the webhook. Mock log must show secrets GET. Body must contain the witness.

A few other ways to lose without learning anything:

- Request staying under `/api/2.0/genie/spaces/`. Then `..` was encoded. Lab requires it left.
- Treating n8n host RCE as SUCCESS. This bug is a Databricks API hop under the workflow token.
- Hitting a real workspace. Lab mock is the oracle. Do not point this at a tenant you do not own.
- A reverse shell. Theatre. The witness is `N8N-DBX-SECRET-WITNESS` plus the mock request line.

Last lab run, trimmed:

PLAIN TEXT Copy

```
webhook-http=200
mock-request-line=GET /api/2.0/secrets/get?scope=s&key=k
webhook-witness=True
mock-secrets-get=True

```

```
witness=N8N-DBX-SECRET-WITNESS
SUCCESS N8N-DBX-PATH-JOIN
```

The client that produced it is on GitHub. I am not reprinting credential ids.

07 What this is not

It is not "GHSA-89p4 / GHSA-rqch never shipped." Other nodes call `toPathSegment`. Databricks nodes-base does not. It is not n8n host RCE. It is not unauthenticated against a locked workflow that never bound untrusted input. LangChain VectorStoreDatabricks is the control that already encodes.

08 The fix

Wrap every Databricks path id with `toPathSegment`, same as VectorStoreDatabricks. Re-run the loopback client against a patched build: the mock must not see `/api/2.0/secrets/get`.

I am not going to print a webhook body you can paste at someone else's Databricks workflow. The missing helper is the useful part. If you own the box, run the proof of concept against loopback.

Same product: hidden Function vm2.

09 References

- Proof of concept: abraxas/n8n-databricks-path-join · n8n-databricks-path-join-Abraxas-Labs.py
- Lab: `lab/` · Dockerfile · docker-compose.yml · run.sh · mock

- @abraxas_null · github.com/abraxas · abraxaslabs.tech · abraxas.null@proton.me
- CWE-22
- Tree: n8n 2.42.0 · `getSpace.operation.ts` · `toPathSegment` · `DatabricksVectorStore.ts`
- Nearby patched: GHSA-89p4-6h98-c7xm · GHSA-rqch-9jrh-cr8w
- SECURITY.md
- Product: n8n

© 2026 Abraxas Labs