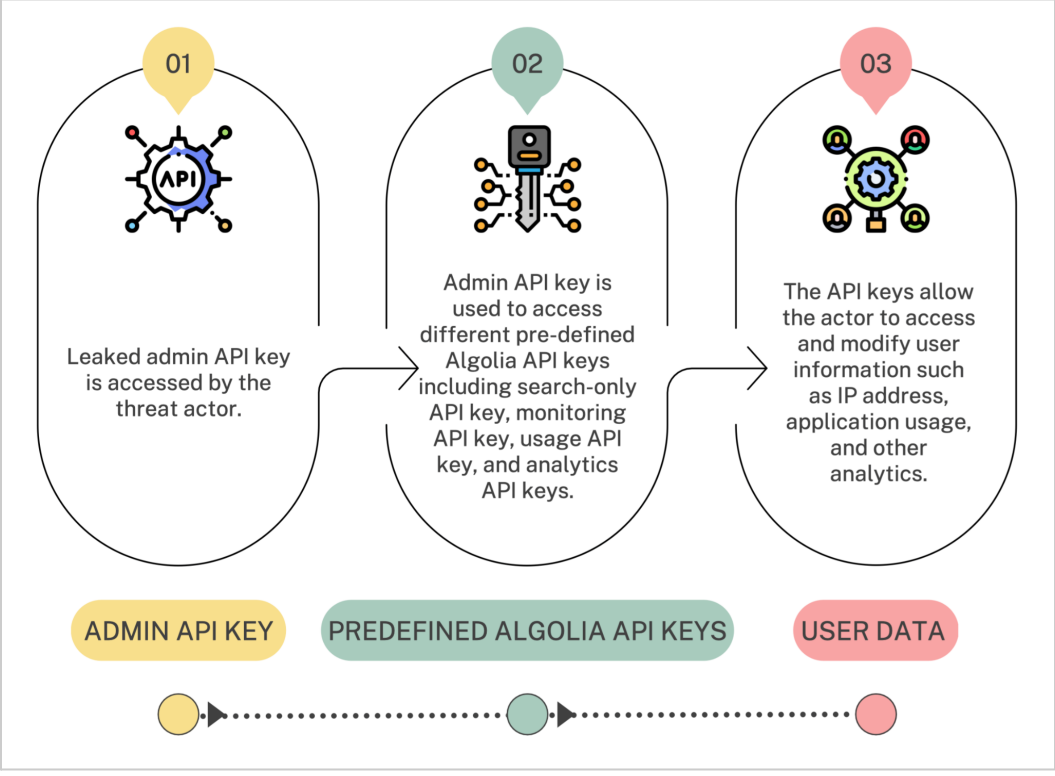
Hardcoded Algolia API Keys Could be Exploited by Threat Actors to Steal Millions of Users' Data

CloudSEK's BeVigil, the world's first security search engine for mobile apps, uncovered 1550 apps, leaking Algolia API key & Application ID. Algolia's API enables developers to implement search, discovery, and recommendations within websites, mobile, and voice applications. It is used by over 11,000 companies, including Lacoste, Stripe, Slack, Medium, and Zendesk to manage ~1.5 trillion search queries a year.

Key Findings

- CloudSEK researchers discovered 1550 applications leaking Algolia API Key and Application ID.
- Furthermore, it was found that a few apps even had hardcoded highly critical Algolia Admin API keys. A total of 32 applications were found to have these critical Admin secrets hardcoded, and so far 57 unique admin keys have been identified.
- The admin API key can be used to access different pre-defined Algolia API Keys including Search-only API key, Monitoring API key, Usage API key, and Analytics API keys.
- This will enable threat actors to:
    - Read users' personal information
    - Modify and delete users' information
    - Access users' IP addresses and other access details
    - View users' app usage and other analytics



Applications Leaking Valid Keys

The 32 applications that were leaking 57 valid unique Admin API Keys are classified under the following 15 categories. The main categories like Shopping, Education, Lifestyle, Business and Medical themselves have a total 2517000 number of downloads.

While this is not a flaw in Algolia or other such services that provide integrations, it is evidence of how API keys are mishandled by app developers. So, it is up to individual companies to address the security concerns associated with payment gateways, AWS services, open firebases, etc.

App	No. of Downloads
Shopping app	2306000
News & Magazines app	300000
Food & Drink app	200000
Education app	110000
Health & Fitness app	105000
Photography app	100000
Lifestyle app	80000
Business app	11000
Entertainment app	11000
Medical app	10000
Parenting app	10000
Books & Reference app	5000
Productivity app	5000
Tool app	5000
Auto & Vehicles app	1000

How the Algolia API Works

In general, API (Application Program Interface) is a way to increase the data and functionality of an application for other developers. To leverage an API, a developer only needs to understand how the interface functions—not how it is implemented. As a result, the backend handles all the messy coding, and the developer is given a tidy interface.

In this sense, the Algolia API allows developers to implement search, discovery, and recommendation services across their websites and mobile apps.

The Algolia API requires the Application ID and API key to be passed through the following headers:

- X-Algolia-Application-Id: the Application ID
- X-Algolia-API-Key: the Authentication Token

The Five Algolia API Keys

Admin API key

With the Admin API key, any of the following services can be used:

- Search (search): allowed to perform search operations.
- Browse Index (browse): allowed to retrieve all index data with the browse endpoint.
- Add records (addObject): allowed to add or update records in the index.
- Delete records (deleteObject): allowed to delete an existing record.
- List indices (listIndexes): allowed to get a list of all existing indices.
- Delete index (deleteIndex): allowed to delete an index.
- Get index settings (settings): allowed to read all index settings.
- Set index settings (editSettings): allowed to update all index settings.
- Use analytics API (analytics): allowed to retrieve data with the Analytics API.
- Use recommendation API (recommendation): allowed to interact with the Recommendation API.
- Use usage API (usage): allowed to retrieve data with the Usage API.
- Access logs (logs): allowed to query the logs.
- Get unretrievable attributes (seeUnretrievableAttributes): allowed to retrieve unretrievableAttributes for all operations that return records.

Search-only API key

This is used to search data. This is the public API key to use in your front-end code. This key is only usable for search queries and sending data to the Insights API.

```
curl -X GET \
     -H "X-Algolia-API-Key: ${API_KEY}" \
     -H "X-Algolia-Application-Id: ${APPLICATION_ID}" \

"https://${APPLICATION_ID}-dsn.algolia.net/1/indexes/imdb?query=george%20clo&hitsPerPage=2&getRankingInfo=1"
```

The search-only key is usable for search queries and sending data to the Insights API

```
{
    "logs": [
        {
            "timestamp": "2013-09-17 13:10:31",
            "method": "GET",
            "answer_code": "200",
            "query_body": "",
            "answer": "{  "items": [    {      "name": "cities",      "createdAt":
"2013-09-16T07:39:29.446Z",      "updatedAt": "2013-09-16T07:39:29.446Z",
"entries": 149096,      "pendingTask": false      }    ]  }",
            "url": "/1/indexes",
            "ip": "127.0.0.1",
            "query_headers": "User-Agent: curl/7.24.0 (x86_64-apple-darwin12.0)
libcurl/7.24.0 OpenSSL/0.9.8x zlib/1.2.5Host: localhost.algolia.com:8080Accept:
*/*Content-Type: application/json; charset=utf-8X-Algolia-API-Key:
20f**************************X-Algolia-Application-Id: MyApplicationID",
            "sha1": "26c53bd7e38ca71f4741b71994cd94a600b7ac68"
        }
    ]
}
```

Valid Response From Server

Monitoring API key

This API enables you to see the inner workings of your clusters/replicas. It is not accessible by the standard API clients.

```
curl -X GET \
     -H "X-Algolia-API-Key: ${API_KEY}" \
     -H "X-Algolia-Application-Id: ${APPLICATION_ID}" \
     --compressed \
    "https://status.algolia.com/1/status"
```

The monitoring API key enables you to see the inner workings of your clusters/replicas

```
{
  "status": {
    "c4-fr": "operational",
    "c2-eu": "operational"
  }
}
```

Valid Response From Server

Usage API key

The Usage API is only available on Premium plans and plans including our Enterprise add-on.

```
curl -X GET \
  -H "X-Algolia-API-Key: ${API_KEY}" \
  -H "X-Algolia-Application-Id: ${APPLICATION_ID}" \
  --compressed \

"https://usage.algolia.com/1/usage/records?startDate=2020-07-15T00:00:00Z&endDate=2020-07-16T00:00:00Z&granularity=daily"
```

The Usage API is only available on Premium plans and plans including the Enterprise add-on

```
{
  "records": [
    {"t": 1455451200000, "v": 53863464},
    {"t": 1455454800000, "v": 53897109},
    ...
  ],
  "max_qps": [
    {"t": 1455451200000, "v": {"c4-fr-1": 35, "c4-fr-2": 40, "c4-fr-3": 37}},
    {"t": 1455454800000, "v": {"c4-fr-1": 34, "c4-fr-2": 35, "c4-fr-3": 33}},
    ...
  ],
  "region_max_qps": [
    {"t": 1455451200000, "v": {"eu": 185}},
    {"t": 1455454800000, "v": {"eu": 186}},
    ...
  ]
}
```

Valid Response From Server

Analytics API key

This is used to fetch different Analytics.

```
curl -X GET \
     -H "X-Algolia-API-Key: ${API_KEY}" \
     -H "X-Algolia-Application-Id: ${APPLICATION_ID}" \
    "https://analytics.algolia.com/2/status?index=${index name}"
```

The Analytics API key is used to fetch different Analytics

```
{
  "attributes": [
    {"attribute": "brand", "count": 2},
    {"attribute": "_tags", "count": 1},
    // {...}
  ]
}
```

Valid Response From Server

Impact

- Perform Search Operation
- Retrieve all index data with the browse endpoint.
- Add or update records in the index
- Delete an existing record
- Read and Update all index settings
- Retrieve data with the Analytics API
- Interact with the Recommendation API
- Retrieve data with the Usage API.
- Query the logs

Mitigation

- The leaked keys should be revoked, and new ones should be generated and stored securely at the backend.
- If you need to communicate with a sensitive, external API, ask the backend to create an endpoint.
- This endpoint should be authenticated with a user token and implement proper security requirements. In this way, attackers should never get those secrets, and only you will be able to communicate with your backend.

Responsible Disclosure

CloudSEK has notified Algolia and the affected apps about the hardcoded API keys.

References

- How Leaked Twitter API Keys Can be Used to Build a Bot Army - CloudSEK
- Hardcoded GitHub Personal Access Tokens Leak 159 Private Repositories - BeVigil Blog