API - Public Breaches Search (140+ Billion Records)

? NiamonX API — **Public Breaches Search (140+ Billion Records)**

Comprehensive API documentation for searching public data breaches aggregated from 4,500+ sources.

?? Important Notice

The **Public Breaches Search** API allows you to query a massive database of publicly available leaks (>140 billion records).
Use it **only** for legitimate purposes — such as verifying your own data or with explicit authorization.

> ⚠ Abuse or publication of obtained data will lead to **account suspension** and **permanent API ban**.
> Some results may contain outdated or incomplete data.

? API Endpoint

Main URL:

```
POST https://dash.niamonx.io/api/v2/breaches_search
```

Headers:

```
Content-Type: application/json
X-API-Key: YOUR_API_KEY
```

Body Example:

```
{ "query": "test@example.com" }
```

Example Request via cURL

```
curl -X POST https://dash.niamonx.io/api/v2/breaches_search \
  -H "Content-Type: application/json" \
  -H "X-API-Key: YOUR_API_KEY" \
  -d '{"query":"test@example.com"}'
```

? Query Parameters

Field	Type	Description	Example
`query`	`string`	Search term (email, phone, domain, IP, username, etc.)	`"example@mail.com"`

Supported Search Types

- Email: `john.doe@gmail.com`
- Login or Username: `nickname123`
- Domain: `domain.com`
- IP address: `1.1.1.1`
- Combo (e.g. “email password” or “name phone”)
- Full name, city, social ID

□□ **Tip:** If you get an empty response, try deleting one character from the query and repeat the request — it refreshes the search pipeline.

?? Response Structure

? When Data Is Found

HTTP 200


```
{
  "success": true,
  "data": {
    "query": "niamonx",
    "task_id": "50bab956-4a9c-4548-83ff-a0451be1b18e",
    "status": "ok",
    "detail": "file_downloaded",
    "meta": {
      "blocks_total": 3,
      "emails": ["test@niamonx.io"],
      "names": ["NiaMonx"],
      "first_seen": "2021-01-21T05:32:31+00:00",
      "last_seen": "2021-01-22T01:51:31+00:00"
    },
    "risk": {
      "score": 9,
      "level": "Low"
    },
    "blocks": [
      {
        "id": "p1",
        "title": "TestNia",
        "description": "In November 2021, the TestNia website...",
        "groups_normalized": [
          {
            "fields_map": {
              "email": "test@niamonx.io",
              "nick": "NiaMonx",
              "ip": "2800:***:9824"
            }
          }
        ]
      }
    ],
    "rate": {
      "remaining": 99,
      "reset_in_sec": 600
    },
```

```
    "cooldown_sec": 10
  }
}
```


?? When No Results Found

HTTP 200


```
{
  "success": true,
  "data": {
    "query": "not_found@niamonx.io",
    "status": "not_found",
    "detail": "no results found",
    "meta": { "blocks_total": 0 },
    "risk": { "score": 0, "level": "Low" }
  }
}
```


? When Data Is Protected

HTTP 200


```
{
  "success": true,
  "data": {
    "success": false,
    "error": "[NiamonX | DataGuard] This data has been removed and is no longer indexed by our search engine at the request of the copyright holder."
  }
}
```

? HTTP Status Codes

Code	Meaning
200	Success — request processed
400	Invalid input or malformed body
401	Invalid or missing API key
403	Tool disabled for your account
404	Unknown endpoint or invalid tool name
405	Wrong method — use `POST`
429	Cooldown or daily limit reached

? Features Summary

- □□ **140+ billion** records
- □□ **4500+** aggregated sources
- □□ **Cryptographically protected** data
- □□ **Risk Indicator** (numeric + level)
- ⚡ **Cached results** for faster response
- □□ **Automatic data removal** for sensitive categories (bank/medical)

? Code Examples

1. Python (requests)

```
import requests

url = "https://dash.niamonx.io/api/v2/breaches_search"
headers = {
    "Content-Type": "application/json",
    "X-API-Key": "YOUR_API_KEY"
}
```

```
payload = { "query": "example@mail.com" }

response = requests.post(url, json=payload, headers=headers)
print(response.status_code)
print(response.json())
```

2. JavaScript (Node.js / Axios)

```
import axios from "axios";

const API_KEY = "YOUR_API_KEY";
const url = "https://dash.niamonx.io/api/v2/breaches_search";

async function searchLeak(query) {
  try {
    const res = await axios.post(
      url,
      { query },
      {
        headers: {
          "Content-Type": "application/json",
          "X-API-Key": API_KEY
        }
      }
    );
    console.log(res.data);
  } catch (err) {
    console.error("Error:", err.response?.data || err.message);
  }
}

searchLeak("example@mail.com");
```

3. PHP (cURL)

```php
<?php
$api_key = "YOUR_API_KEY";
$url = "https://dash.niamonx.io/api/v2/breaches_search";

$data = ["query" => "example@mail.com"];
$options = [
  CURLOPT_URL => $url,
  CURLOPT_RETURNTRANSFER => true,
  CURLOPT_POST => true,
  CURLOPT_HTTPHEADER => [
    "Content-Type: application/json",
    "X-API-Key: $api_key"
  ],
  CURLOPT_POSTFIELDS => json_encode($data)
];

$ch = curl_init();
curl_setopt_array($ch, $options);
$response = curl_exec($ch);
curl_close($ch);

echo $response;
?>
```

---

4. Go (net/http)

```go
package main

import (
"bytes"
"fmt"
"io"
"net/http"
)
```

```
func main() {
	apiKey := "YOUR_API_KEY"
	jsonBody := []byte(`{"query":"example@mail.com"}`)
	req, _ := http.NewRequest("POST", "https://dash.niamonx.io/api/v2/breaches_search",
bytes.NewBuffer(jsonBody))
	req.Header.Set("Content-Type", "application/json")
	req.Header.Set("X-API-Key", apiKey)

	client := &http.Client{}
	resp, err := client.Do(req)
	if err != nil {
		panic(err)
	}
	defer resp.Body.Close()

	body, _ := io.ReadAll(resp.Body)
	fmt.Println("Status:", resp.Status)
	fmt.Println(string(body))
}
```

? Best Practices

- Wait **3 seconds** between requests to respect cooldown (anti-spam).
- Combine parameters intelligently:
  - `"email password"`
  - `"name phone"`
- Do not republish or share results publicly.
- If you detect compromised credentials — **change passwords immediately**.
- Sensitive datasets (bank cards, healthcare) are **automatically excluded**.

? Summary

Feature	Description
Endpoint	`https://dash.niamonx.io/api/v2/breaches_search`
Method	POST

Feature	Description
Authentication	`X-API-Key`
Cooldown	3 seconds
Rate Limit Info	Returned in `rate` object
Data Types Supported	Email, phone, IP, domain, username
Data Protection	SHA-256 & DataGuard Compliance

 Now you can safely integrate the NiamonX Public Breaches API into your OSINT tools, cybersecurity dashboards, or monitoring systems — with full ethical and legal compliance.

Revision #3
Created 9 November 2025 21:17:56 by NiamonX Team
Updated 9 November 2025 21:42:53 by NiamonX Team