PBS v2 | Public Breached Search V2 | NiamonX API

? Public Breached Search V2 — NiamonX API Guide

The **Public Breached Search V2** endpoint allows you to search for publicly available breach records in the NiamonX secure data analysis system.
It operates through a **private, encrypted channel** and uses a **minimal indexing model** to protect sensitive data.

Supported identifiers include:

- **Email**
- **Username**
- **Phone**
- **Hash**

All requests must be made securely and responsibly.
Never share or publish personal results — doing so may violate data protection laws and lead to account suspension.

? API Endpoint

```
POST https://dash.niamonx.io/api/v2/breaches_s_v2
```

Headers:

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

Body:

```
{ "value": "test@example.com", "type": "auto" }
```

Available types:
`auto`, `email`, `username`, `phone`, `hash`

? Description

This API searches **closed and anonymized datasets** for breached credential evidence.
Decryption happens at the client level using your master key, ensuring full end-to-end security.

Additional features:

- Passwords are hidden by default (toggleable).
- Supports export to CSV/JSON.
- Local query history.
- Internal quotas are invisible to users.
- Misuse or unauthorized data searches may result in a ban.

? Example Successful Response


```
{
  "success": true,
  "data": {
    "query": {
      "value": "test@niamonx.io",
      "type": "auto"
    },
    "stats": {
      "found": 2,
      "returned": 2,
      "with_passwords": 2,
      "unique_sources": 1,
      "earliest_month": null,
```

```
    "latest_month": null
  },
  "records": [
    {
      "source": {
        "name": "Stealer Logs",
        "breach_date": null,
        "unverified": 0,
        "passwordless": 0,
        "compilation": 1
      },
      "account": "test@niamonx.io",
      "email": "test@niamonx.io",
      "username": null,
      "phone": null,
      "hash": null,
      "password": "z8NiAmOn50H",
      "fields": [
        "password",
        "origin",
        "email"
      ]
    },
    {
      "source": {
        "name": "Stealer Logs",
        "breach_date": null,
        "unverified": 0,
        "passwordless": 0,
        "compilation": 1
      },
      "account": "test@niamonx.io",
      "email": "test@niamonx.io",
      "username": null,
      "phone": null,
      "hash": null,
      "password": "ViptraNiA!",
      "fields": [
        "password",
        "origin",
```

```
          "email"
        ]
      }
    ],
    "niamonx_success": true,
    "status": "ok",
    "fetched_at": "2025-11-09T22:04:00+00:00",
    "api_timing_ms": 146
  }
}
```

?? Possible Outcomes

No Matches Found


```
{
  "success": true,
  "data": {
    "success": false,
    "status": "error",
    "error": "HTTP 400",
    "query": {
      "value": "not_found",
      "type": "auto"
    }
  }
}
```


DataGuard Protection


```
{
  "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	✅ Successful response (check data object)
400	❌ Validation error in input data
401	🔑 Invalid or missing API key
403	🚫 Tool disabled or unavailable
404	❓ Unknown endpoint
405	⚙️ Method not allowed (use `POST`)
429	⏳ Cooldown / daily limit reached

? Example Implementations

1. cURL

```
curl -X POST https://dash.niamonx.io/api/v2/breaches_s_v2 \
  -H "Content-Type: application/json" \
  -H "X-API-Key: YOUR_API_KEY" \
  -d '{"value":"test@example.com","type":"auto"}'
```

---

2. Python (using `requests`)

```
import requests

url = "https://dash.niamonx.io/api/v2/breaches_s_v2"
headers = {
    "Content-Type": "application/json",
    "X-API-Key": "YOUR_API_KEY"
}
data = {
    "value": "test@example.com",
    "type": "auto"
}

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

---

3. JavaScript (Node.js, using `axios`)

```
import axios from "axios";

const url = "https://dash.niamonx.io/api/v2/breaches_s_v2";
const headers = {
  "Content-Type": "application/json",
  "X-API-Key": "YOUR_API_KEY"
};

const data = {
  value: "test@example.com",
  type: "auto"
};

axios.post(url, data, { headers })
  .then(res => console.log(res.data))
  .catch(err => console.error(err.response?.data || err.message));
```

4. PHP

```php
<?php
$url = "https://dash.niamonx.io/api/v2/breaches_s_v2";
$headers = [
    "Content-Type: application/json",
    "X-API-Key: YOUR_API_KEY"
];
$body = json_encode([
    "value" => "test@example.com",
    "type" => "auto"
]);

$ch = curl_init($url);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_POSTFIELDS, $body);
curl_setopt($ch, CURLOPT_HTTPHEADER, $headers);

$response = curl_exec($ch);
curl_close($ch);

echo $response;
?>
```

5. Go

```go
package main

import (
□"bytes"
□"encoding/json"
□"fmt"
□"net/http"
□"io/ioutil"
```

```
)

func main() {
	url := "https://dash.niamonx.io/api/v2/breaches_s_v2"
	body := map[string]string{
		"value": "test@example.com",
		"type":  "auto",
	}
	jsonData, _ := json.Marshal(body)

	req, _ := http.NewRequest("POST", url, bytes.NewBuffer(jsonData))
	req.Header.Set("Content-Type", "application/json")
	req.Header.Set("X-API-Key", "YOUR_API_KEY")

	client := &http.Client{}
	resp, _ := client.Do(req)
	defer resp.Body.Close()

	respBody, _ := ioutil.ReadAll(resp.Body)
	fmt.Println(string(respBody))
}
```

?? Security Recommendations

- Keep your API key private; never hard-code it in shared repositories.
- Use HTTPS exclusively — no plain HTTP connections are allowed.
- If your key is compromised, revoke it immediately.
- Store credentials in environment variables or encrypted vaults.
- Do not reshare or republish search results publicly.

** You're all set!**
With your **NiamonX API Key** and this **Public Breached Search V2** guide, you can perform encrypted and privacy-compliant breach lookups safely and efficiently.

Revision #1
Created 9 November 2025 22:03:24 by NiamonX Team
Updated 9 November 2025 22:13:50 by NiamonX Team