ULP (Infostealer Logs) Public Breached ULP Search | NiamonX API

? NiamonX API — **ULP (Infostealer Logs) Public Breached Search**

Comprehensive API documentation for searching **ULP datasets (URL · LOGIN · PASSWORD)** extracted from public infostealer logs and credential leaks.

? What Is ULP?

ULP stands for **URL · LOGIN · PASSWORD** — a triple that represents evidence of compromised credentials captured in stealer logs or public leaks.

Each ULP record links:

- **URL:** the website or endpoint where the credential was used (e.g., `example.com/login`)
- **LOGIN:** the username or email associated with the site
- **PASSWORD:** the stolen or leaked password (masked by default for privacy)

This API enables you to search across massive datasets for specific entries by **email**, **username**, **domain**, **URL**, or **password**.

?? Usage & Ethics

By using this endpoint, you confirm that:

- You **own** the data or have explicit permission to process it.
- You will **not share** results publicly or misuse obtained data.
- You will **act responsibly** — change any compromised credentials and enable MFA.

All queries are **end-to-end encrypted**, and NiamonX **never logs or shares** search data.

? Endpoint

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

? Request Structure

Headers

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

Body Example

```
{ "action": "search", "value": "test@example.com", "type": "auto", "exact": true, "limit": 200
}
```

?? Request Parameters

Field	Type	Description	Example
action	`string`	Must be `"search"`	`"search"`
value	`string`	Your search query (email, domain, password, etc.)	`"user@mail.com"`
type	`string`	Search type (see below)	`"email"`

Field	Type	Description	Example
exact	`boolean`	Whether to match exactly (recommended for emails)	`true`
limit	`integer`	Max number of records (1–1000)	`200`

Available `type` values:

- `auto` *(default)*
- `email`
- `username`
- `domain`
- `url`
- `password`

? Example cURL Request

```
curl -X POST https://dash.niamonx.io/api/v2/ulp_search \
  -H "Content-Type: application/json" \
  -H "X-API-Key: YOUR_API_KEY" \
  -d '{"action":"search","value":"test@example.com","type":"auto","exact":true,"limit":200}'
```

? Successful Response

HTTP 200


```
{
  "success": true,
  "data": {
    "query": {
      "value": "test@niamonx.io",
      "type": "email",
```

```
      "exact": true,
      "limit": 200
    },
    "stats": {
      "total": 1,
      "unique_hosts": 1,
      "with_password": 1
    },
    "records": [
      {
        "id": "ac22b9424f8aab6011fb526c9798e7c3898652d4c7a6eb8c0253212d94a9fec4",
        "url": "niamonx.io/login/index",
        "host": "niamonx.io",
        "login": "test@niamonx.io",
        "pass": "NiaMon750H",
        "score": 8.840368
      }
    ],
    "status": "ok",
    "cached": false,
    "fetched_at": "2025-11-09T21:50:31+00:00",
    "api_timing_ms": 75
  }
}
```

?? When Nothing Is Found

HTTP 200


```
{
  "success": true,
  "data": {
    "query": {
      "value": "not_found",
      "type": "username",
```

```
      "exact": true,
      "limit": 200
    },
    "stats": {
      "total": 0,
      "unique_hosts": 0,
      "with_password": 0
    },
    "records": [],
    "status": "ok",
    "cached": false,
    "fetched_at": "2025-11-09T21:51:56+00:00",
    "api_timing_ms": 63
  }
}
```

? Protected Data Response

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	Description

200	Success — request processed
400	Invalid input or malformed request
401	Invalid or missing API key
403	Tool disabled for your account
404	Unknown endpoint or invalid tool
405	Wrong HTTP method (use `POST`)
429	Cooldown or daily limit exceeded (ToolService message)

? Tips & Notes

- □ **Limit:** up to 1000 results per request
- □□ **Cooldown:** few seconds between requests to prevent spam
- □□ **Domain search:** finds subdomains automatically
- □□ **Duplicate removal** and periodic reindexing ensure fresh results
- □□ If results seem incomplete, retry in several minutes for updated data

---

? Code Examples

1. Python (requests)

```
import requests
import json

url = "https://dash.niamonx.io/api/v2/ulp_search"
headers = {
    "Content-Type": "application/json",
    "X-API-Key": "YOUR_API_KEY"
}
payload = {
    "action": "search",
    "value": "test@example.com",
    "type": "auto",
    "exact": True,
```

```
    "limit": 200
}

response = requests.post(url, headers=headers, json=payload)
print(response.status_code)
print(json.dumps(response.json(), indent=2))
```

---

2. JavaScript (Node.js / Axios)

```
import axios from "axios";

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

async function searchULP(query) {
  try {
    const res = await axios.post(
      url,
      {
        action: "search",
        value: query,
        type: "auto",
        exact: true,
        limit: 200
      },
      {
        headers: {
          "Content-Type": "application/json",
          "X-API-Key": API_KEY
        }
      }
    );
    console.log(JSON.stringify(res.data, null, 2));
  } catch (err) {
    console.error("Error:", err.response?.data || err.message);
  }
```

```
}

searchULP("test@example.com");
```

---

3. PHP (cURL)

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

$data = [
  "action" => "search",
  "value" => "test@example.com",
  "type" => "auto",
  "exact" => true,
  "limit" => 200
];

$options = [
  CURLOPT_URL => $url,
  CURLOPT_RETURNTRANSFER => true,
  CURLOPT_POST => true,
  CURLOPT_HTTPHEADER => [
    "Content-Type: application/json",
    "X-API-Key: $apiKey"
  ],
  CURLOPT_POSTFIELDS => json_encode($data)
];

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

echo $response;
?>
```

4. Go (net/http)

```
package main

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

func main() {
	apiKey := "YOUR_API_KEY"
	body := []byte(`{"action":"search","value":"test@example.com","type":"auto","exact":true,"limit":200}`)

	req, _ := http.NewRequest("POST", "https://dash.niamonx.io/api/v2/ulp_search", bytes.NewBuffer(body))
	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()

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

? Summary

Feature	Description
Endpoint	`https://dash.niamonx.io/api/v2/ulp_search`
Method	`POST`
Auth Header	`X-API-Key`
Limit	up to 1000 records
Cooldown	Short delay between requests
Sensitive Data	Passwords masked, removed on request
Security	Encrypted E2E, DataGuard compliant
Data Source	Public infostealer logs and breach repositories

 With the ULP API, you can ethically and securely analyze exposure of credentials from public infostealer datasets — empowering your investigations, threat intelligence, and personal data protection workflows.

Revision #1
Created 9 November 2025 21:44:56 by NiamonX Team
Updated 9 November 2025 22:01:00 by NiamonX Team