HEROIC API Reference (7.3.0)

Download OpenAPI specification: Download

URL: https://HEROIC.com | Terms of Service

Introduction

HEROIC offers a powerful suite of enterprise-grade APIs designed to detect and investigate exposed data across billions of breach records. With tens of billions of compromised records indexed, the HEROIC API allows you to search and retrieve breach data across multiple identity types and sources.

Access requirements

To use the HEROIC API, you must have an active HEROIC Enterprise Account. Click here to sign up.

Base URL

The Base URL for our APIs is https://api.heroic.com/v7

Obtaining the API key

To obtain a key:

- Log into your HEROIC Enterprise account.
- Go to API Key Management.
- Create or manage your API keys.

Authentication

All requests must include an API key in the header: x-api-key: YOUR_API_KEY

Error Handling

Code	Error category	Description	Resolution
403	Authentication Failed	Invalid API credentials.	Ensure a valid API key is specified.
404	Path not found	The API path does not exist.	Check the API route.
405	Invalid input	Invalid input provided.	Check your input.
422	Validation Error	Validation failed.	Read the error message and correct your data.
500	Internal server error	Server error.	Contact HEROIC support.

Need help?

Contact support@heroic.com for assistance.

PII Masking

HEROIC is committed to protecting sensitive personal information (PII) in all API responses. To ensure privacy and compliance, all PII fields such as credit card numbers, SSNs, and passwords are masked or redacted in the data returned by our APIs.

- **Credit card numbers**: Randomly masked to show up to 6 digits (e.g., `543210XXXXXX1234`).
- **SSNs**: Only last 2 digits are visible (e.g., `123-456-78**`).
- **Passwords**: Only last 2 characters are visible (e.g., `admin@12**`).

This masking ensures that sensitive data cannot be reconstructed or misused, while still allowing for effective breach investigation and analysis. If you require access to unmasked data for legitimate security or compliance reasons, please contact HEROIC support for more information on our data access policies.

Breach catalog

The breach catalog includes general information about what was exposed, when the breach occurred, and what type of data was involved. It's useful for displaying or investigating breach events at a high level.

Get all breaches

Returns an array of all breaches HEROIC has discovered.

Responses

> **200** Successful response

GET `/breaches`

Response samples

200

Content type
application/json

Copy Expand all Collapse all

```
[
  - {
        "uuid": "f5d77b03-44b4-11eb-9442-1d5c76d5a106",
        "site_name": "Ledger",
        "site_domain": "ledger.com",
        "site_logo": "https://breached-sites-logos.s3.us-west-2.amazonaws.com/le
        "date_leaked": "25-Jun-2020",
        "site_categories": "Crypto",
        "site_country": "United States",
        "site_language": "English",
        "password_types": "None",
        "leaked_data_types": "Email Address, Phone",
        "heroic_article_url": null,
        "description": "The hacker responsible for Ledger's security breach in Ju
        "pwned_count": 1075382
    }
]
```

Get breach details

Provides information associated with a breach. Requires UUID as a parameter.

PATH PARAMETERS

`uuid` required	string <uuid> UUID of the data breach.

Responses

> **200** Successful response

GET /breaches/{uuid}

Response samples

200

Content type
application/json

Copy


```
{
   "uuid": "f5d77b03-44b4-11eb-9442-1d5c76d5a106",
   "site_name": "Ledger",
   "site_domain": "ledger.com",
   "site_logo": "https://breached-sites-logos.s3.us-west-2.amazonaws.com/ledger
   "date_leaked": "25-Jun-2020",
   "site_categories": "Crypto",
   "site_country": "United States",
   "site_language": "English",
   "password_types": "None",
   "leaked_data_types": "Email Address, Phone",
   "heroic_article_url": null,
   "description": "The hacker responsible for Ledger's security breach in July
   "pwned_count": 1075382
}
```


Breach search

APIs for data breach search for accounts.

Breach search

Search for breach details by email, IP address, phone number, etc.

Search Filters

In addition to the required `type` and `account` parameters, you can use any of the supported breach type values as additional filter parameters. These filters accept "yes" or "no" values to refine your search results.

Example Usage

To search for an email address that also has a password exposed:

```
GET /breach-search?type=email&account=mohammad@gmail.com&password=yes
```

This will return all breach records for `mohammad@gmail.com` where a password was also exposed.

Supported Filter Types

You can use any of these values as filter parameters:

- `email` - Filter for records with email addresses
- `email_domain` - Filter for records with email domains
- `phone_number` - Filter for records with phone numbers
- `username` - Filter for records with usernames
- `ip_address` - Filter for records with IP addresses
- `social_security_number` - Filter for records with SSNs
- `password` - Filter for records with passwords
- `password_hash` - Filter for records with password hashes
- `bitcoin_address` - Filter for records with bitcoin addresses

Filter Values

- `yes` - Include only records that have this data type
- `no` - Exclude records that have this data type

QUERY PARAMETERS

`type` `required`	string Enum: `"email"` `"email_domain"` `"phone_number"` `"username"` `"ip_address"` `"social_security_number"` `"password"` `"password_hash"` `"bitcoin_address"` Account type filter.
`account` `required`	string Value for the selected account type.

`paging_token` string
Token for pagination (from previous response).

`number_of_records` integer
Limit the number of records returned.

`[breach_type]` string
Enum: `"yes"` `"no"`
Additional filter parameters. You can use any of the supported breach type values (email, email_domain, phone_number, username, ip_address, social_security_number, password, password_hash, bitcoin_address) as parameter names with "yes" or "no" values to filter results.

Example: `password=yes` will return only records that also have a password exposed.

HEADER PARAMETERS

`x-api-key` string
required
API key for authentication.

Responses

› **200** Successful response

GET `/breach-search`

Response samples

200

Content type
application/json

Copy Expand all Collapse all


```
{
  "records_found": 12,
  "pagination_token": "0053001032e271c0ddeb11ed8a50195359d25748406263356664666
```

```
  - "data": [
      + { … }
    ]
}
```


Credit card search

HEROIC's Credit Card Search API provides access to both legacy (free) and active (paid) stolen credit card data found across darknet markets and hacker forums. Free cards typically originate from older leaks and are useful for identifying previously compromised information, while paid cards are current, often functional, and marketed on underground platforms. HEROIC collects associated metadata such as BINs, expiration dates, issuing countries, prices, and seller reputations. Users can perform single BIN lookups, submit bulk queries, configure automated monitoring, and download results in formats including CSV, Excel, or JSON.

Search Capabilities and Limitations The API enables structured searches, including masked or SHA-256-hashed card numbers, owner names, CVV codes, and expiration dates. Users can apply advanced filters and syntax to refine results and maintain privacy. Monitoring options are available for ongoing updates. While powerful, the system is affected by common challenges—such as disappearing data, duplicate listings, low-quality or fake records, and the migration of card sales to newer platforms like Telegram and Discord. HEROIC continues to improve its scraping and detection capabilities, with regular updates and expanding coverage to adapt to the evolving threat landscape.

What types of credit cards are indexed by HEROIC? HEROIC tracks two primary categories of compromised credit card data uncovered across the deep and dark web:

- **Free Cards** – These are stolen credit card records that have been made publicly available without cost on underground forums and leak sites. While HEROIC's systems have cataloged millions of such entries, the majority are from older breaches and are typically no longer valid for transactions. You can explore this data through the database search feature in the HEROIC Control Center.
- **Paid Cards** – These are active, for-sale credit cards commonly listed on darknet marketplaces. Sold much like regular merchandise, these records are often fresh and functional, requiring a payment to access. HEROIC continuously scans and logs these listings to provide real-time intelligence on emerging threats.

Credit card search

This endpoint enables you to search for exposed credit card records discovered in data breaches and leaks. You can use query parameters to filter results by cardholder, issuing bank, or other criteria. The response includes masked card numbers, CVVs, expiration dates, issuing bank details, and breach-related metadata.

Advanced Search Syntax

Available Fields:

Field	Details
createdAt	Creation date & time.
number	Credit card number (default field), masked with X in the middle except first 6 and last 4 digits.
Hash	SHA-256 of a credit card number
expireDate	Expiration date
cvv	Card verification value
owner	Owner name
bank	Issuer bank name
leakId	Leak ID

Operators: AND, OR, NOT

Examples:

- Search cards starting with 411111: `411111*`
- Search cards starting with 400012 and ending with 7890, containing 16 digits: `400012XXXXXX7890`
- Search cards starting with 400012 and ending with 7890, containing any number of digits: `400012*7890`
- Search cards with Alice Smith as owner and Chase Bank as bank: `owner:"Alice Smith" AND bank:"Chase Bank"`
- Search cards starting with 411111 and belonging to Bob Lee: `number:411111* AND owner:"Bob Lee"`
- Search cards that have a CVV 555 and that expire on or after January 1, 2023: `cvv:555 AND expireDate:[2023-01-01 TO *]`

The credit card search API supports both pagination and advanced search capabilities. Sensitive information, such as full card numbers, is never exposed—only masked versions are included in the response.

QUERY PARAMETERS

`page`	integer Example: `page=0` Page number (zero-based).
`size`	integer Example: `size=10` Number of records per page.
`sort`	string Example: `sort=createdAt,desc` Sort order (e.g., createdAt,desc).
`query`	string Example: `query=owner:Johnson AND bank:Citibank` Search query (e.g., owner, bank, etc). Supports logical operators (AND, OR).

HEADER PARAMETERS

`x-api-key` required	string API key for authentication.

Responses

> **200** Successful response

GET /credit-card-search

Response samples

200

Content type
application/json

Copy Expand all Collapse all


```
{
  "number": 0,
```

```
    "size": 10,
    "totalElements": 352,
    "totalPages": 36,
    "numberOfElements": 10,
    "first": true,
    "last": false,
    "hasContent": true,
  - "content": [
     + { … }
    ]
}
```