PhishIQ - API & Integration Overview

Introduction to the PhishIQ API

The PhishIQ API is an advanced interface for real-time detection and blocking of phishing attacks. The system enables flexible integration with a variety of environments - from organizational email servers and network traffic monitoring to mobile-based services.

The API is suitable for organizations of all sizes and across multiple sectors - including private companies, public institutions, infrastructure, and tech firms - looking to identify and monitor malicious links and enhance protection for users and critical data.

PhishIQ is powered by advanced Machine Learning (ML) and Artificial Intelligence (AI) technologies that identify phishing threats even when they are new and previously unknown. In parallel, the system continuously builds and updates an internal repository of malicious links, allowing faster response to recurring threats and continuous improvement of protection levels.

Integration with the PhishIQ API is simple and fast via a RESTful interface, and includes full professional support throughout implementation and ongoing usage.

Prerequisites

Before starting to work with the PhishIQ API, please ensure the following conditions are met:

- **API Key Access:** Each user receives a unique API key that grants access to the service. Authentication is performed simply via this key alone, with no additional protocols required.
- **Simple Access with High Security:** The connection process is designed to be easy and developer-friendly, while maintaining robust internal security measures and continuous monitoring of suspicious activity-all transparent to the user.
- **Usage Plans:** Each client accesses the API according to their purchased usage plan. There is no need for special configuration from the client's side, only adherence to the request limits defined by the plan.
- **Request Method:** Requests are sent via HTTP POST, with each request containing a single link for inspection. Multiple requests can be sent concurrently, each with a separate link.
- **Unified API for All Clients:** There is no differentiation between client types regarding the API structure-everyone uses the same interface. However, each request can include an internal identifier parameter (e.g., user ID or campaign ID) for the client's internal analytics.

Authentication

PhishIQ API uses a simple yet secure authentication method based on a unique API key.

- Each client is issued a dedicated API key which must be included in every request to the API to authenticate the call.
- Authentication is performed exclusively via this API key, with no additional protocols or token exchanges required.
- The API key should be sent in the HTTP header of each request, typically using the x-api-key header field.
- This approach ensures ease of integration for developers while maintaining high internal security standards and continuous monitoring of suspicious activity behind the scenes.

Example of an authenticated API request using curl:

```
curl -X POST "https://api.phishiq.com/v1/predict" \
-H "Content-Type: application/json" \
-H "x-api-key: YOUR_API_KEY_HERE" \
-d '{"url": "http://suspicious-link.example"}'
```

Passing the API Key in Requests

To make API calls, an access key (API Key) issued by the company must be used. This key is used to authenticate the requests and grant access to the services.

To use the API, the key must be passed as a parameter in every request. While we recommend using the POST method, the GET method is also supported.

Example:

POST request

```
curl -X POST "https://api.phishiq.com/predict" -d "url=https://example.com -H "x-api-key:YOUR_API_KEY"
```

GET request

```
curl "https://api.phishiq.com/predict?url=https://example.com&x-api-key=YOUR_API_KEY"
```

> ***Note:*** *For the security of the key, it is recommended to use the POST method and avoid passing the key in URLs whenever possible.*

Endpoints and Request Structure

PhishIQ API provides a single unified endpoint for URL inspection and threat detection.

Base URL:

api.phishiq.com/health

Request Method:

- The API accepts POST requests (recommended).
- GET requests are also supported but are less secure.

Request Parameters:

- **url:** The URL to be checked for phishing or malicious content.
- **api_key:** Your assigned API key (see Authentication section).
- **client_type:** Internal client classification such as business, private, or subscription tiers like silver, gold, platinum. This parameter is returned in the response for reference.
- Additional custom parameters can be included in the request. These parameters will be returned unchanged in the response, allowing clients to associate metadata or perform internal tracking.

Example (POST):

```
{ "url": "http://example.com", "api_key": "YOUR_API_KEY", "client_type": "business", "custom_param1": "value1" }
```

Example (GET):

```
https://api.phishiq.com/v1/check?
url=http://example.com&api_key=YOUR_API_KEY&client_type=business&custom_param1=value1
```

Response Format:

The response is a JSON object containing:

- **url:** The checked URL.
- **url_hash:** A hash value representing the URL (used for internal tracking).
- **is_malicious:** Boolean indicating whether the URL is detected as malicious or phishing.
- **client_type:** The client type sent in the request.

- Any additional custom parameters sent in the request will be returned as-is.

Error Handling

PhishIQ API uses standard HTTP status codes to indicate the success or failure of an API request. In addition, the response body provides detailed error information to help diagnose issues.

Common HTTP Status Codes:

- **200 OK** - The request was successful, and the response contains the requested data.
- **400 Bad Request** - The request is malformed or missing required parameters.
- **401 Unauthorized** - Authentication failed due to missing or invalid API key.
- **403 Forbidden** - The API key is valid but does not have permission to access the requested resource.
- **429 Too Many Requests** - The client has exceeded their allowed request rate or quota.
- **500 Internal Server Error** - An unexpected error occurred on the server.

Error Response Format:

When an error occurs, the API returns a JSON response with the following fields:

- **error_code:** A short string identifying the type of error.
- **message:** A human-readable description of the error.
- **details (optional):** Additional information that may help resolve the issue.

Example Error Response:

> { "error_code": "INVALID_API_KEY", "message": "The API key provided is invalid or has expired." }

Guidelines for Proper and Secure Use of the API

- **Always prefer using POST:** Sending requests via POST ensures better security and allows data transmission in a safe and controlled manner.
- **Separate environments:** Use a dedicated API key for each environment (production and test) to prevent errors and enhance security.
- **Handle error responses:** Ensure every server error is properly handled to guarantee a smooth user experience and quick fault detection.
- **Keep your keys secure:** Do not expose API keys on the client side. Store them only on the server and follow accepted security practices.

- **Stay updated on changes:** We regularly publish improvements, changes, and alerts. It is recommended to follow system updates to ensure continuous compatibility.

For any issues or questions, please contact our support at:
support@ntrigo.com