Link Box AI API: Internal Security & Development Policy

1.0 Introduction and Policy Mandate

The Link Box AI API provides powerful, programmatic access to our platform's core functionalities and, critically, to sensitive user data, including personal health and financial information. The strategic importance of protecting this data cannot be overstated. This document establishes the mandatory security requirements and development best practices that govern all internal use of the Link Box AI API. Strict adherence to these protocols is non-negotiable.This policy applies to all internal software, services, scripts, and applications developed by our teams that interact in any way with the Link Box AI API. This includes backend services, data processing pipelines, personal automation scripts, and internal tools.Compliance with this policy is mandatory. It is essential for protecting our users' data, ensuring the stability and availability of our services, and maintaining the integrity of the Link Box AI platform. We will now proceed to the foundational element of API security: authentication and authorization.

2.0 Authentication and Authorization Standards

Robust authentication is the primary defense against unauthorized data access and system compromise. A single mishandled credential can lead to a significant data breach. This section defines the mandatory standards for authenticating with the API, which differ based on the application's use case.

2.1 Mandatory API Key Management

For internal scripts, backend services, and personal automation tools where a user is not present to grant permission, API keys are the designated authentication mechanism. The following rules for key management are mandatory:

- **Never Hardcode Credentials:** The hardcoding of API keys directly into source code is strictly prohibited. Similarly, API keys **must never** be committed to any version control system, including Git.
- **Mandate Standardized Environment Variables:** API keys must be loaded from an environment variable named LBX_API_KEY. This is the only approved method for storing and accessing keys in development and production environments.
- **Ensure Credential Isolation:** A distinct and uniquely named API key must be generated for each separate application, service, or script. This practice limits the impact of a potential compromise to a single component.
- **Enforce Key Rotation:** API keys are not permanent. A mandatory key rotation policy is in effect, requiring all keys to be revoked and replaced at least once every 90 days.
- **Immediate Revocation Protocol:** Any API key that is suspected of being compromised, however minor the suspicion, must be revoked immediately in the Link Box AI developer settings.

2.2 OAuth 2.0 Protocol for User-Facing Applications

For any application that will be used by third parties or is designed to act on behalf of other Link Box AI users, the **OAuth 2.0 Authorization Code Flow is the only approved method of authentication and authorization.** This ensures that users explicitly grant consent for an application to access their data.A core principle of this protocol is **least privilege** . Developers must request only the minimum necessary OAuth scopes required for their application to function. Requesting overly broad permissions is a violation of this policy.| Scope | Description || ------ | ------ || entries:read | Grants read-only access to a user's entries. || health:read | Grants access to sensitive personal health data. || finance:read | Grants access to sensitive personal financial data. || collections:write | Grants permission to create, modify, or delete collections. || user:read | Grants read-only access to a user's profile information. |
Properly securing access credentials is the first step. The next is to ensure the integrity of the data your application receives from the API, particularly through asynchronous notifications known as webhooks.

3.0 Webhook Security and Data Integrity

Webhooks provide a powerful mechanism for receiving real-time data updates from the Link Box AI platform. However, an unsecured webhook endpoint creates a significant attack vector, allowing malicious actors to send forged data to your application. Verifying the authenticity and integrity of every incoming webhook payload is therefore mandatory to prevent data corruption and unauthorized actions based on spoofed requests.

3.1 Mandatory Signature Verification

Policy: All application endpoints designed to receive webhooks from the Link Box AI API **must** implement and enforce signature verification for every incoming request. Requests that fail verification must be rejected.The required verification procedure is as follows:

1. **Extract the Signature:** From the incoming webhook request, extract the value of the LinkBox-Signature HTTP header.
2. **Compute the Expected Signature:** Using the secret key provided in your Link Box AI developer webhook settings, compute an HMAC-SHA256 signature of the raw request payload.
3. **Compare Signatures Securely:** Perform a constant-time comparison between the signature received in the header and the signature you computed locally. This method is mandatory to mitigate timing attacks, which could otherwise allow an attacker to forge a valid signature.
4. **Enforce Verification:** Only process requests where the signatures match exactly. Any request that fails this verification check must be immediately discarded, and the event **must** be logged with a CRITICAL severity level as a potential security incident.Maintaining data integrity is parallel in importance to maintaining platform stability. The following section details requirements for responsible API usage.

4.0 Service Stability and Rate Limiting Protocols

Adherence to API rate limits is not merely a best practice; it is a critical component of our collective security and platform stability strategy. Responsible rate limit management is a shared responsibility that prevents service degradation for all users, mitigates the risk of denial-of-service attacks, and ensures that our own internal applications remain performant and reliable.

4.1 Required Protocols for Rate Limit Management

The following protocols for managing API rate limits are required for all applications and scripts.

- **Programmatic Adherence:** Applications **must** programmatically read and adhere to the X-RateLimit-Limit, X-RateLimit-Remaining, and X-RateLimit-Reset headers returned in every API response to manage request frequency and avoid exceeding limits.
- **Graceful Handling:** In the event that a rate limit is exceeded and an HTTP 429 Too Many Requests status code is received, the application **must** implement a graceful retry mechanism using an exponential backoff strategy.
- **Efficient Design:** Developers **must** architect their applications for efficiency. This includes utilizing response caching where appropriate to avoid redundant requests and preferring the use of webhooks for event-driven updates over inefficient, high-frequency polling.These protocols are essential for building robust, scalable, and resilient applications. We conclude with a summary of other foundational security requirements.

5.0 General Security and Compliance

This section covers foundational security principles that are mandatory for all development involving the Link Box AI API.

5.1 Secure Transport Layer

All communication with the Link Box AI API endpoint (https://api.linkbox.ai/v1) **must** occur exclusively over HTTPS. Unencrypted HTTP connections are strictly forbidden, as they expose sensitive data, including authentication credentials and user information, to interception.

5.2 Secure Error Handling

Applications **must** be designed to gracefully handle API errors without exposing sensitive system information in logs or user-facing messages. Security-related HTTP status codes, such as 401 Unauthorized and 403 Forbidden, must trigger secure logging protocols and appropriate user notification mechanisms. Under no circumstances shall applications expose sensitive system information, such as internal system paths, stack traces, or raw database queries, in logs or user-facing error messages.

6.0 Policy Enforcement and Support

Adherence to this policy is not optional. All code interacting with the Link Box AI API will be subject to periodic code reviews and security audits. Any identified instances of non-compliance will require immediate remediation by the responsible team.

6.1 Security Contacts

For assistance, questions, or to report an issue, please use the following contacts:

- **For technical questions about the API:** api@bb23llc.com
- **To report a suspected security vulnerability:** security@bb23llc.com