wib

OWASP API3:

Excessive Data Exposure

OWASP top 10

Excessive Data Exposure occurs when an application API sends more data than necessary. This vulnerability is widespread in real-world applications because it is very so to overlook.

If a smart client needs some data, it sends a request to the backend to extract the data from the database and returns it.

For example, an e-commerce app may include a product catalog with many product attributes. A client developer might want to display model name, price, and rating. A backend developer would then implement an API that returns the three fields. In the next version, another client developer wants to retrieve order numbers, which requires updates to the API.

As a result, backend developers often just send everything to the client and let client developers do what they want. Then, the client can render whatever it needs. This approach can make developers' jobs easier. However, it is also a security issue because the API might send proprietary data that no one should be able to see. While the mobile app does not display sensitive data, attackers can bypass the app and use the API directly.

Lack of understanding and awareness are frequently the source of this vulnerability. In an attempt to be ready for future use, developers try to implement APIs generically without considering the sensitivity of all exposed data. Therefore, it is critical to consider API security from the beginning during the development stage.

How Hackers Exploit It

The theft of personal data from social media sites is a typical example of this type of attack. Consider the following example of a social network where users can create posts and leave comments.

This API is available for getting information about comments:
/api/post/{id}/comments/{commentId}

It enables metadata retrieval for comments, including the author's name. Besides basic information about the author, this API call also returns the API token, Passport number, and address of that author. When sniffing the data, an attacker can also observe personal information about the author, such as their passport number (PPN), API token, and address

```
GET /api/post/1/comments/2
[
   {
      "comment": "I like it !",
      "postId": "1",
      "id": 2,
      "user": "Michael",
      "ppn": "5553388",
      “Phone”: “5553322”
      "api_token": "2a66c14b32d94d81ca565e8f32dhf6db",
      "user email": "michael@yahoo.com"
      "address": "372 Allison St, San Francisco, CA, USA"
   }
]
```

When attackers obtain these details, then they can carry out malicious attacks that directly affect not only company reputation but also people's lives

Why You Should Care

If an attacker sniffed a passport number and address in the scenario above, they could apply for credit cards or loans or file fraudulent income tax returns.
Moreover, companies could attract investigations and fines from regulators for data breaches. For example, hotel operator Marriott had to pay a $124 million fine because of violations of the GDPR (General Data Protection Regulation).
Finally, loss of confidence is one of the most severe hits to a business, as it usually means a loss of client trust.

Traditional Tools Will Not Protect You

Traditional security measures like WAFs and API Gateways cannot distinguish between acceptable API data and sensitive data that requests should return. However, as threats become more complicated and sophisticated, these tools miss attacks that target the logic of APIs. They could, in theory, use pattern matching to identify sensitive data types, but they have no idea what this data represents and have no understanding of API context. Even if they can detect passport number, API token, or address as in our example, they cannot distinguish between legitimate and malicious API usage.

How to Combat Excessive Data Exposure Threats

You must do several things to protect your APIs from Excessive Data Exposure Threats:

1. Never rely on client-side data filtering
2. Examine API replies to ensure they only include legitimate information
3. Consider the consumer of the data before exposing a new API endpoint
4. Avoid using generic methods such as to_json() and to_string() in favor of specific properties
5. Classify the sensitive and personally identifiable information (PII) that your application stores and uses, and examine any API requests that return such data to evaluate security risks
6. As an added layer of protection, implement a schema-based response validation system that defines and enforces all data returned by API methods, including errors

How Wib Can Help You

Wib is a full-lifecycle API security platform, defending against API security threats. The platform ensures comprehensive protection across the entire API software lifecycle from development to testing to production.

Wib utilizes state-of-the-art proprietary AI and ML to analyze millions of requests in real-time. It provides complete visibility of existing APIs, with actionable insights and comprehensive protection across the entire lifecycle. It learns your APIs inside and out and can provide complete visibility of your existing APIs, analyze their integrity, identify vulnerabilities, and detect attacks in real-time.

If you are building or using APIs in your applications (and you probably are), sign up for a demo to see Wib in action and learn more.