What are Business Logic Bugs?

Business logic vulnerabilities are set of flaws found in the design and implementation of web applications. While they have legitimate business features, they can also be exploited by attackers to cause unexpected behaviour. These errors are often the result of applications failing to securely identify and handle unexpected user actions.

Identifying business logic flaws usually requires collaboration between business teams and people with in-depth knowledge of the business or manual penetration testing.

For example, lets Imagine a website that sells t-shirts.

A typical user workflow would be as follows:

1. Adds t-shirts to the basket.
2. Pays with their credit card.
3. Finalises the order.

An attacker comes to the website and tries to exploit a logic flaw like the following:

1. Adds 2 t-shirts to the basket.
2. Maliciously adds more t-shirts (10) to the basket.
3. Pays with their credit card.
4. Finalise the order and gets 12 t-shirts, for the price of 2

Logic flaws actually exist in real life as well. You can compare this e-commerce example with what can happen in a real supermarket. The normal "workflow" of a supermarket assumes that all items that consumers want to buy are placed in the shopping cart and then placed on the conveyor belt.

But what if an attacker hides an item in their shopping cart during the checkout process? The cashier might not see the item and the consumer might eventually not pay for it.

Vulnerabilities that can be considered Business Logic Bugs

- **Excessive Trust in Client-Side Controls**

A fundamentally wrong assumption is that the user interacts with the application only through the provided web interface. This is particularly dangerous because it introduces the additional assumption that client-side validation will prevent the user from providing malicious input.

However, an attacker can use proxy tools to manipulate data after it is sent from the browser and before it is passed to server-side logic. This effectively disables client-side control which will eventually allow exploitation by attacker**.**

- **Making Flawed Assumptions About User Behaviour**

One of the most common causes of business logic errors is misunderstandings about user behaviour. In general, developers do not consider dangerous situations that violate these assumptions.

For example, an application may seem secure because it provides a reliable way to follow business rules. However, some developers are unaware that they cannot trust user and application data indefinitely after passing these rigorous tests.

Applying constraints only at the beginning of the interaction and not checking them afterwards is not enough, because these can open a vulnerability path to privilege escalation.

- **Failing to handle unconventional input**

Most of the time the application is set up to follow some business rule like the user cannot enter an unrealistic value, e.g. if the user tries to order the quantity more than the quantity available in the stock, the server will redefine that value by showing an error.

If proper input validation is not presented on the server side, an irregular input (such as "negative value") might potentially create a major business logic flaw.

- **Domain-specific flaws**

Many logic errors are related to a specific business domain of an application. An example is the discount feature in an e-commerce website. This might be an important attack surface, as it allows attackers to discover fundamental logical holes in how discounting is applied.

It is important to understand the algorithms that the application uses to make these changes and the circumstances under which these changes occur. A good way to test this is to use user input to control the type of function and lead to unexpected results.

 How do you prevent Business Logic bugs?

1. It is important for business analysts, application architects, and developers to fully understand the complexity of their business. Then develop a test case and thoroughly test the business logic, taking into account all these assumptions made during the design phase.

2. The planning and design phases of the application development are very important. Failure to identify and document the predictions can lead to poor application design.

3. Application architects must identify user personas and application interaction scenarios. Access to functional modules should always be based on the success or failure of certain conditions or states passed from one stage to another.

References:

https://www.vaadata.com/blog/what-are-business-logic-flaws-on-web-applications/

https://brightsec.com/blog/business-logic-vulnerabilities/

https://portswigger.net/web-security/logic-flaws/examples

https://cyberbakery.net/business-logic-vulnerabilities/