**Defect Leakage and calculation!**

A defect that is discovered during testing and finds its way into the end-users' or production environment's environment is referred to as defect leakage, also known as defect escape or bug leakage. Stated differently, it denotes the percentage of flaws that were detected later in the software development lifecycle and were not detected during the testing stage.

**Calculation of Defect Leakage:**

**Defect Leakage Rate (%) = (Number of Defects Reported by Users / Total Number of Defects Found) x 100**

Where:

• Number of Defects Reported by Users: The total number of defects reported by users or customers after the software has been released.

• Total Number of Defects Found: The total number of defects found during testing (including those found in various testing phases such as unit testing, integration testing, system testing, etc.).

Example:

Suppose during testing, you found 50 defects. However, after releasing the software, users reported an additional 10 defects that were not identified during testing.

• Number of Defects Reported by Users = 10

• Total Number of Defects Found = 50

**Using the formula:**

**Defect Leakage Rate (%) = (10 / 50) x 100 = 20%**

This means that 20% of defects were not caught during testing and leaked into the production environment.

**Importance of Defect Leakage:**

• Quality Assurance: Defect leakage is an essential metric for evaluating the effectiveness of the testing process and the overall quality of the software.

• Customer Satisfaction: High defect leakage rates can lead to customer dissatisfaction, as users may encounter issues with the software after its release.

• Costs: Addressing defects after release can be more expensive and time-consuming compared to fixing them during the testing phase.

**Mitigation Strategies:**

• Improving Testing Processes: Enhance testing techniques, introduce automation, and implement rigorous quality assurance measures to reduce the likelihood of defects escaping detection.

• Feedback Mechanisms: Encourage users to provide feedback and promptly address reported issues to minimize the impact of defect leakage.

• Continuous Improvement: Analyze defect patterns and trends to identify areas for improvement in the development and testing processes.

By monitoring and managing defect leakage effectively, organizations can enhance the quality of their software products and improve customer satisfaction.