* Injection flaws:

Injection flaws result from a classic failure to filter untrusted input. It can happen when you pass unfiltered data to the SQL server (SQL injection), to the browser (XSS – we’ll talk about this [later](https://www.toptal.com/security/10-most-common-web-security-vulnerabilities#CommonMistake_XSS)), to the LDAP server (LDAP injection), or anywhere else. The problem here is that the attacker can inject commands to these entities, resulting in loss of data and hijacking clients’ browsers.

* Broken authentication:

1. The URL might contain the session id and leak it in the referer header to someone else.
2. The passwords might not be encrypted either in storage or transit.
3. The session ids might be predictable, thus gaining access is trivial.

* Cross Site Scripting:

This is a fairly widespread input sanitization failure (essentially a special case of [common mistake #1](https://www.toptal.com/security/10-most-common-web-security-vulnerabilities#CommonMistake_Injection)). An attacker gives your web application JavaScript tags on input. When this input is returned to the user unsanitized, the user’s browser will execute it. It can be as simple as crafting a link and persuading a user to click it, or it can be something much more sinister. On page load the script runs and, for example, can be used to post your cookies to the attacker.