[Skip to main content](https://lms.alnafi.com/xblock/block-v1:alnafi+DCCS102+2025_DCCS+type@vertical+block@8c345e0ed2af4a7b82ebe21da9fe8c52?exam_access=&recheck_access=1&show_bookmark=0&show_title=0&view=student_view#main)

**Summary**

The password change and reset function of an application is a self-service password change or reset mechanism for users. This self-service mechanism allows users to quickly change or reset their password without an administrator intervening. When passwords are changed they are typically changed within the application. When passwords are reset they are either rendered within the application or emailed to the user. This may indicate that the passwords are stored in plain text or in a decryptable format.

**Test Objectives**

* Determine the resistance of the application to subversion of the account change process allowing someone to change the password of an account.
* Determine the resistance of the passwords reset functionality against guessing or bypassing.

**How to Test**

For both password change and password reset it is important to check:

1. if users, other than administrators, can change or reset passwords for accounts other than their own.
2. if users can manipulate or subvert the password change or reset process to change or reset the password of another user or administrator.
3. if the password change or reset process is vulnerable to [CSRF](https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/06-Session_Management_Testing/05-Testing_for_Cross_Site_Request_Forgery).

**Test Password Reset**

In addition to the previous checks it is important to verify the following:

* What information is required to reset the password?

The first step is to check whether secret questions are required. Sending the password (or a password reset link) to the user email address without first asking for a secret question means relying 100% on the security of that email address, which is not suitable if the application needs a high level of security.

On the other hand, if secret questions are used, the next step is to assess their strength. This specific test is discussed in detail in the [Testing for Weak security question/answer](https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/04-Authentication_Testing/08-Testing_for_Weak_Security_Question_Answer) paragraph of this guide.

* How are reset passwords communicated to the user?

The most insecure scenario here is if the password reset tool shows you the password; this gives the attacker the ability to log into the account, and unless the application provides information about the last log in the victim would not know that their account has been compromised.

A less insecure scenario is if the password reset tool forces the user to immediately change their password. While not as stealthy as the first case, it allows the attacker to gain access and locks the real user out.

The best security is achieved if the password reset is done via an email to the address the user initially registered with, or some other email address; this forces the attacker to not only guess at which email account the password reset was sent to (unless the application show this information) but also to compromise that email account in order to obtain the temporary password or the password reset link.

* Are reset passwords generated randomly?

The most insecure scenario here is if the application sends or visualizes the old password in clear text because this means that passwords are not stored in a hashed form, which is a security issue in itself.

The best security is achieved if passwords are randomly generated with a secure algorithm that cannot be derived.

* Is the reset password functionality requesting confirmation before changing the password?

To limit denial-of-service attacks the application should email a link to the user with a random token, and only if the user visits the link then the reset procedure is completed. This ensures that the current password will still be valid until the reset has been confirmed.

**Test Password Change**

In addition to the previous test it is important to verify:

* Is the old password requested to complete the change?

The most insecure scenario here is if the application permits the change of the password without requesting the current password. Indeed if an attacker is able to take control of a valid session they could easily change the victim’s password. See also [Testing for Weak password policy](https://owasp.org/www-project-web-security-testing-guide/v42/4-Web_Application_Security_Testing/04-Authentication_Testing/07-Testing_for_Weak_Password_Policy) paragraph of this guide.

**Remediation**

The password change or reset function is a sensitive function and requires some form of protection, such as requiring users to re-authenticate or presenting the user with confirmation screens during the process.