Software Security Assurance

Software Security Assurance

SSA Knowledgebase

Knowledgebase Search

Software Security Test Cases

Rows per page: 30

Search

Authentication Testing

Authentication Testing

1. Test for transport of credential (or any other authentication secret)

Objective: To test if application transport credentials via encrypted channel.

Test case ID	Description	Expected results
Auth.1.1	Validate browser and web server communication	Credentials are not be displayed into sniffer
Auth.1.2	Validate communication between web server and application server	Credentials are not be displayed into sniffer
Auth.1.3	Validate communication between application server and database server	Credentials are not be displayed into sniffer
Auth.1.4	Validate communication between application and external application	Credentials are not be displayed into sniffer
Auth.1.5	Validate communication between application and external device	Credentials are not be displayed into sniffer

2. Test for storage of credential (or any other authentication secret)

Objective: To test if application stores credentials (or any other authentication secret) in unencrypted format.

Test case ID	TestCase	Expected results
Auth.2.1	Validate credential storage at database / directory	Credentials are not visible to system administrators in plain text
Auth.2.2	Validate credential storage into property / configuration files	Credentials are stored into encrypted format
Auth.2.3	Validate credential storage into binary file	Credentials are not stored into binary
Auth.2.4	Check for hardcoded secret in plaintext into code	Secrets are not hardcoded into code
Auth 2.5	Check if the authentication token is unique for each user while creation	Authentication token (e.g., User ID / Email ID /Account No etc) if found already while creating a new, it should show appropriate error message

3. Test for user enumeration

Objective: To test if application reveals valid usernames or confirms valid user names

Test case ID	TestCase	Expected results
Auth.3.1	Check if application display different response valid username / wrong password and wrong username / wrong password combinations	Application responds with same content Same content (with same message length) in both scenarios
Auth.3.2	Check redirection Urls, redirection page title and response code for valid username / wrong password and wrong username / wrong password combination	Application response is same for both cases
Auth.3.3	Validate if application generate user identifier in predictable way	User has to register for account where user specifies their identifier

4. Test for default or guessable user account or credentials

Objective: To test if default, guessable or test accounts/credentials

Test case ID	TestCase	Expected results
Auth.4.1	Validate if default accounts are renamed / changed for all servers, infrastructure components and application	All default accounts are changed and test accounts are removed
Auth.4.2	Validate if password complexity is sufficient to avoid password prediction	Strong password complexity policy is applied

Test case ID	TestCase	Expected results
Auth.4.3	Check if source code contains comments that give clue for account information	All comments are removed from production code
Auth.4.4	Validate if common password is used for all new accounts	Each user is assigned different one time password and user is forced to change password during first login
Auth.4.5	Validate if password history is checked to avoid reuse of recent passwords	Password once used will not be allowed to used for the next 5 password changes for respective login.

5. Account lockout and reopening

Objective: To test if account lockout and reopening is present

Test case ID	TestCase	Expected results
Auth.5.1	Validate if account lockout and reopening functionality is present	Accounts get locked out after specified number of unsuccessful attempts and can be reopened only after verifying users identity
Auth.5.2	Validate if administrative accounts can be locked	Administrative accounts can't be locked out however access through such account is restricted by IP based access control

6. Test for brute force

Objective: To test if username / password combination can be predicted

Test case ID	TestCase	Expected results
Auth.6.1	Validate if user accounts can accessed with dictionary username / password combination & permutations	Usernames are selected by users and complex password policy is mandated and accounts get locked out of specified period of time
Auth.6.2	Check if session identifier can be brute forced	Session identifiers are generated at server side using framework methods which has sufficient randomization
Auth.6.3	Validate if hash mechanism can be broken	Publically scrutinized latest hash algorithms are used to generate hash
Auth.6.4	Validate if predefined authentication tokens are used	Application do not use predefined authentication token

7. Test for bypassing authentication schema

Objective: To test if application allows user to access information without getting authenticated

Test case ID	TestCase	Expected results
Auth.7.1	Validate if application allow direct resource access without getting authenticated	Application do not allow to access any resource without authentication
Auth.7.2	Check if application uses magic numbers for authentication	Application do not user magic numbers during authentication
Auth.7.3	Validate if session identifier generation is predictable	Session identifier is generated using framework methods and it is not predictable
Auth.7.4	Validate if authentication mechanism can be bypassed by supplying malicious input	All inputs are checked and filtered with validation library so authentication mechanism can't be bypassed

8. Test for password management

Objective: To test if application wrongly implemented password management functionality

Test case ID	TestCase	Expected results
Auth.8.1	Validate if application allow local password storage by default	Application do not allow password storage locally by default however if user want to store password locally, application allow user to after displaying warning and getting confirmation from user
Auth.8.2	Validate if password can be reset without complete verification of user's identity	Application verifies user's identity before resetting password
Auth.8.3	Validate if password is displayed to user or can be sent to user specified email address	Application do not display password to user or send password to user. If user want to reset password, user answers multiple secret questions and password is sent to user's alternate email address which was specified into user's profile
Auth.8.4	Validate if user is forced to change password periodically	User is forced to change password periodically. If user hasn't used account for long account gets disabled and when user starts using account, user has to change password to enable account
Auth.8.5	Check if user can change password without supplying old password	User must supply current password to change it. Passwords are not displayed into plain text so user must type new password twice for verification

9. Test for logout and browser cache management

Objective: To test if application implement logout and browser cache management functionality properly

Test case ID	TestCase	Expected results
Auth.9.1	Validate if after logout user can access application by pressing back button in browser	After logout browser ask user to close the window. User may select not to close window however if user presses back button in browser, user gets session expiry message
Auth.9.2	Validate if application allow offline page viewing through browser cache	Application do not allow offline page viewing through browser cache
Auth.9.3	Validate if after re-login, user supply old session identifier, user is allowed to access old session	Application do not allow user to reuse old session identifier once they logout
Auth.9.4	Validate if application logout user after specified period of inactivity. Also check if activity enforced from server	Application force logout user after specified period of inactivity. This enforcement is done through server side code
Auth.9.5	Validate if user is assigned same session identifier when re-login	User is assigned new session identifier each time user login
Auth.9.6	Validate if session identifier from server is immediately removed after user logout	Session identifier for user is immediately removed from server once server receives logout notification for user
Auth.9.7	Validate if it's possible to store in the browser any sensitive information provided in the user input fields	Application has browser caching disabled wherever any sensitive information such login ID, password, secret question or answer, email address, etc. that could be provided. HTML contains AUTOCOMPLETE=OFF for all those input fields.

10. Test for CAPTCHA

Objective: To test if CAPTCHA implementation is correct

Test case ID	TestCase	Expected results
Auth.10.1	Validate if application keeps track of each CAPTCHA image generated for each user	Application keeps track of each image generated for each user
Auth.10.2	Check if application accepts old CAPTCHA id and old CAPTCHA value	Application do not accept old CAPTCHA id and value
Auth.10.3	Check if application accepts old CAPTCHA id and old session identifier	Application do not accept old CAPTCHA id and old session identifier

11. Test for multifactor authentication

Objective: To test if multifactor authentication implemented properly

Test case ID	TestCase	Expected results
Auth.11.1	Validate if all the authentication mechanism (factors) happens through different communication channel	All authentication happens through different communication channels (i.e. One through computer network and other through GSM cell phone)
Auth.11.2	Validate if one of the authentication mechanism (factor) uses continuously changing random number	One of the authentication mechanism is using continuously changing random number (i.e. RSA tokens)
Auth.11.3	Validate if an alert mechanism deployed for intimating users about transactions?	Alert mechanism is deployed and users are given intimation through multiple communication channels
Auth.11.4	Validate if risk scoring systems enabled for judging transactions?	There is a risk scoring system in place to identify fraudulent transactions

12. Test for Profile Cloning

Objective: To test if multiple profiles of the same user is created

Test case ID	TestCase
Auth.12.1	Validate before creating a profile of a new user if the similar profile is already exists
Auth.12.2	Validate if the user's private information while being sent to the server or vice-e-versa is encrypted while sending

Authorization Testing

Testing Audit Trails

Testing Session Management

Data Security Testing

Testing for Client Side Attacks