

# FIFTY QUESTIONS TO IMPROVE SOFTWARE SECURITY

## AUTHENTICATION

1. Does each web request validate authentication?
2. Are credentials presented securely (i.e. using SSL, not using the GET method)?
3. Are passwords stored in an encrypted or hashed format?
4. Is password complexity enforced, including minimum length, non-guessable words, special characters, numbers?
5. Do user credentials expire after a period of time?
6. Are standards used for authentication and identity management (i.e. SAML, WS-Security, LDAP, NTLM, Kerberos)?
7. Are user accounts locked after a certain number of failed authentication attempts?

## AUTHORIZATION

8. Are permissions defined to create fine-grained user access?
9. Are permissions defined for fine-grained administrator access?
10. Are permissions enforced consistently in the application?
11. Can permissions be grouped or organized to user roles for simplified access management?
12. Are roles and permissions consistent with standards or other applications in the enterprise?

## DATA VALIDATION

13. Are all user inputs validated?
14. Does validation check data length?
15. Does validation filter or escape special characters?
16. Does validation of web input remove tags before displaying it back to the user?
17. Does the application validate the data type of user input before operating on it?
18. Is XML received from outside of the application validated?
19. Is the integrity of files sent and received by the application validated?

## SESSION MANAGEMENT

20. Is session data excluded from the URL using the GET method?
21. Does data in the browser cookie contain only the session ID and exclude other session information?
22. Are session IDs hashed to prevent attackers from guessing valid session IDs?
23. Are session IDs guaranteed to be unique?
24. Are sessions validated on each page request?
25. Do sessions expire after a period of inactivity?
26. Are expired sessions deleted on the server?

## LOGGING

27. Are security-related events logged consistently?
28. Is sensitive information, such as passwords, kept from logs?
29. Are security events stored in a secure location and not mixed with common application logging?
30. Are events logged in a format and location that is compatible with security monitoring/event correlation software?

## ERROR HANDLING

31. Are exception handling mechanisms used consistently?
32. Does the application fail securely? If so, how?
33. Are open transactions processed appropriately if an error is encountered during processing?
34. Are error messages displayed to the users informative without revealing information about system internals or other sensitive data?
35. For function-based error handling, are return values of functions tested?
36. For exception-based error handling, are specific exceptions caught, rather than broad exception handlers (i.e. Throwable in Java)?
37. Are exceptions that are caught managed and logged (i.e. no empty catch{ } blocks)?

## CRYPTOGRAPHY

38. What is the sensitivity of the data being processed by the application?
39. Is encryption required for the data? If so, in transit, at rest, or both?
40. Does the application comply with your organization's standards regarding encryption?
41. Are standard, accepted encryption protocols being used, rather than home-grown algorithms?
42. Are passwords encrypted in transit and at rest?
43. Are keys used with encryption protocols managed securely in the application?

## PERFORMANCE

44. Is the application thread-safe?
45. Are variables encapsulated to limit their scope and prevent sharing between processes?
46. Are efficient algorithms used?
47. Are database transactions clearly defined and not subject to deadlocks?
48. Are database tables indexed appropriately?
49. Are file handles and connections to external systems explicitly closed?
50. Are all variables that are initialized actually used?

