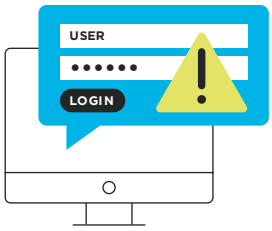
VERACODE

The Top Five Authentication Vulnerabilities

One of the most important parts of a web application is the authentication mechanism, which secures the site and creates boundaries for each user account. Yet authentication mechanisms often have vulnerabilities. Data from our SOSS X report reveals that A2-Authentication flaws rank high in prevalence and frequently play a role in incidents, so staying on top of these flaws is important for prioritization. If web developers are equipped to resolve issues like Authentication flaws before penetration testing, the testers will then have more time to spend tackling complex vulnerabilities. Read on for the most common web application authentication vulnerabilities and how to go about fixing or preventing them.

1 Weak Password Policy



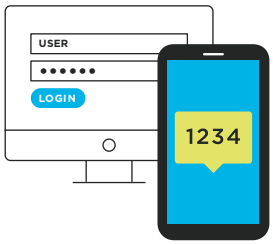
WHAT IT MEANS

Some applications allow end users to set weak passwords like “password” or “1234567.” These are well known to attackers and are frequently found in data leaks where companies have been breached. Additionally, some applications do not allow end users to copy and paste in the password field, which could prevent users from utilizing secure password managers.

THE FIX

Authentication mechanisms should always require strong passwords. NIST guidelines recommend against using complexity rules, and instead rely on password length and checks against common passwords lists to ensure that users choose strong passwords. Consider the following when creating a password policy: Use a minimum of 8 characters, disallow passwords that match commonly used or leaked passwords, and restrict words relating to the application or company. Allow passwords up to at least 64 characters so that users can create secure passphrases. Additionally, permit all printable ASCII characters — including spaces — and Unicode characters in passwords.

2 Two-Factor Authentication (2FA)



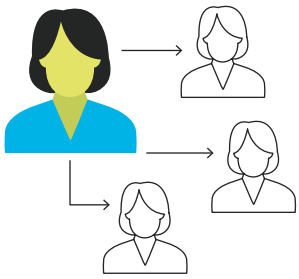
WHAT IT MEANS

A common vulnerability with web authentication mechanisms is the lack of additional security measures. It’s very rare to see developers implementing two-factor authentication, especially on high-privileged accounts.

THE FIX

There are several ways to implement 2FA technology, including RSA tokens, code generators such as Google Authenticator, and Duo and SMS text messaging of one-time codes. Separating administrative functions, such as user-account management, into separate applications with stronger authentication schemes is also a good practice.

3 User Enumeration



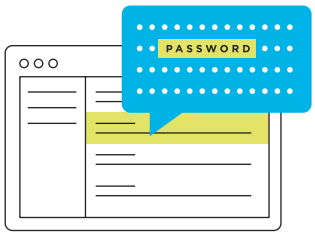
WHAT IT MEANS

This vulnerability is typically found in places within a web application where a username is required, such as a login page. It is mainly due to error messages being presented back to the end user; if an invalid user attempts to authenticate, the error presented back would be different from that presented to a valid user.

THE FIX

Make sure any information presented back to the end user is generic. For example, "If your account was found in our system, you will receive an email shortly." When resolving this on account registration, a CAPTCHA should be implemented on the page and must be filled out correctly before being able to "Submit" the form. While this does not prevent harvesting altogether, it defeats automated attacks and limits probing to manual analysis only.

4 Broken Password Reset



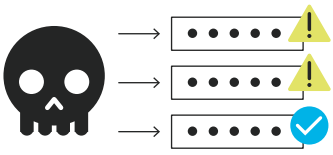
WHAT IT MEANS

Developers sometimes don't use secure methods to protect credentials when attempting to reset passwords. As such, they tend to send passwords via email, which is not secure. This may also indicate that the password is being stored in the database without proper salting and hashing, or is in a reversible format like base64. Base64 can be decoded and is not deemed a secure method to store any sensitive information.

THE FIX

Use the functionality within your chosen framework and ensure all passwords are stored in a secure manner using proper salting and hashing.

5 Brute Force Attacks



WHAT IT MEANS

A brute force attack is where an attacker attempts multiple usernames and passwords until they obtain access to a valid account. This is easier to perform if the application has a user enumeration or a weak password policy. There are several free tools that help an attacker perform such an attack and make it relatively easy to do.

THE FIX

Require the end user to complete a CAPTCHA before attempting authentication. After a failed authentication attempt, the end user should be made to wait for a period of time. This time will increase as the account approaches maximum allowance for consecutive failed attempts (e.g. 30 seconds up to an hour). If possible, restrict access to the application via IP address.

VERACODE

Learn more about common vulnerabilities and how best to address them in our most recent **State of Software Security report**.

Veracode is a leader in helping organizations secure the software that powers their world. Veracode's SaaS platform and integrated solutions help security teams and software developers find and fix security-related defects at all points in the software development lifecycle, before they can be exploited by hackers. Our complete set of offerings helps customers reduce the risk of data breaches, increase the speed of secure software delivery, meet compliance requirements, and cost effectively secure their software assets — whether that's software they make, buy, or sell. Veracode serves more than 1,400 customers across a wide range of industries, including nearly one-third of the Fortune 100, three of the top four U.S. commercial banks, and more than 20 of Forbes' 100 Most Valuable Brands.