VULNERABILITY DECODER

Broken Access Controls

The Vulnerability

Weak security controls could allow unauthenticated or unauthorized users to access things you don't want to leave unsecured, like sensitive files and systems, or to escalate user privileges. Flaws in the handling of user credentials could permit attackers to bypass access controls. Some common errors include hardcoded passwords and plaintext passwords in config files.

48.3%

of applications have a credentials management vulnerability on initial scan.

Source: SOSS v11

The Risks

Attackers can manipulate data and systems, dupe users into loading malware onto the system, or steal personal and confidential data, such as banking information and Social Security numbers. Poor password management practices can lead to account takeovers.

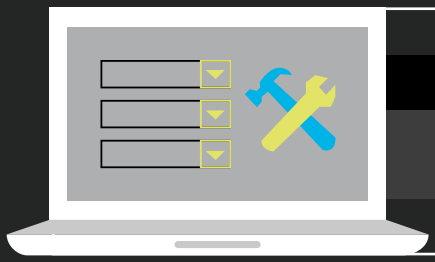
Example Breach

Twitter, Netflix, GitHub, and other big websites were knocked offline by an enormous botnet of hijacked Internet of Things (IoT) devices infected by Mirai, a malware targeting IoT devices that use hardcoded passwords.

Prevention & Remediation

Weak access controls and credentials management vulnerabilities are preventable with secure coding practices. Lock down administrative accounts and controls, and use multi-factor authentication, such as FIDO or dedicated apps, to reduce the risk of compromised accounts. Follow the principle of least privilege, and don't give users more rights than they need.

Code-based validation checks for user account permissions should verify that a user has access to specific features or functionality rather than solely relying on role-based access checks. In practice, verifying a user's role, as well as any granular permissions associated with a specific feature, can usually be included in the same line of validation code.



Consider using the following string:

```
if (user.isRole('Admin') and user.hasAccess('DELETE_ACCOUNT')){
        deleteAccount();}
```

Recommendations

Nobody writes perfect code the first time around. You can avoid vulnerabilities and prevent breaches when you:

- ☑ Get training in secure coding best practices, through on-demand eLearning courses, in person security consultations, and professional development certifications and conferences.
- ☑ Scan early and often to detect flaws while you code. Use application security tools that allow you to scan small batches of code instantaneously, and can provide remediation guidance within your development workflow.

Download the Secure Coding Best Practices Handbook

Join the Veracode Community
community.veracode.com

VERACODE