10 Scariest Vulnerabilities

1. **Information Leakage**
   - **Prevalence**: 72%
   - **Cause**: Applications that let attackers view sensitive information like usernames, passwords, and session IDs.
   - **Cure**: Use SSL to encrypt all traffic. Implement pop-up blockers to prevent users from accidentally clicking malicious links.

2. **Cryptographic Issues**
   - **Prevalence**: 65%
   - **Cause**: Applications that use encryption algorithms or key management incorrectly.
   - **Cure**: Use industry-standard encryption algorithms and keys, and stronger encryption modules.

3. **Bad Code Quality**
   - **Prevalence**: 62%
   - **Cause**: Applications with bugs that allow attackers to execute commands, hijack sessions, and steal sensitive information.
   - **Cure**: Implement static code review and automated unit tests.

4. **CRLF Injection**
   - **Prevalence**: 53%
   - **Cause**: An attacker introduces an unexpected &quot;Line Feed&quot; character to split text streams into discrete characters or software log entries to split lines.
   - **Cure**: Never trust user input. Always properly encode and sanitize user input.

5. **Cross-Site Scripting**
   - **Prevalence**: 50%
   - **Cause**: An attacker can inject malicious scripts into websites that a user's web browser executes as trusted code.
   - **Cure**: Implement delegation of authorization and do other data sanitization checks on the client before sending requests to the web server.

6. **Directory Traversal**
   - **Prevalence**: 49%
   - **Cause**: An attacker can access files and directories by sending modified URLs to the application.
   - **Cure**: Always properly encode and sanitize user input.

7. **Insufficient Input Validation**
   - **Prevalence**: 44%
   - **Cause**: An attacker can input creepy code to read and steal information about the user.
   - **Cure**: Wrap private data in classes to keep implementation details out of sight.

8. **Credentials Mis-Management**
   - **Prevalence**: 41%
   - **Cause**: Applications that store passwords, payment data and a whole lot more insecurely.
   - **Cure**: Implement delegation of authorization and do other data sanitization checks on the client before sending requests to the web server.

9. **SQL Injection**
   - **Prevalence**: 32%
   - **Cause**: Applications that let attackers execute SQL commands through input fields.
   - **Cure**: Use whitelists to define valid input data.

10. **Encapsulation Errors**
    - **Prevalence**: 25%
    - **Cause**: Applications that sanitize user input incorrectly, leading to CRLF injection attacks.
    - **Cure**: Implement delegation of authorization and do other data sanitization checks on the client before sending requests to the web server.

---

Dreadful defects in applications are lurking everywhere you look. In fact, more than 65% of applications fail security testing the first time they reach production — a truly horrifying and scary — application vulnerabilities today.

**How Vulnerable Are Your Applications?**

You can prevent all of these scary vulnerabilities and more by routinely assessing your applications through automated static and dynamic testing.

State of Software Security: [Download the Report at Veracode.com/SSS](https://www.veracode.com/SSS)