GÜVENLİ KOD GELİŞTİRMEK İÇİN TEMEL ÖNERİLER

Bu öneriler kod güvenlik taramalarında sık karşılaşılan temel zafiyet sorunlarından derlenmiştir. Bunların dışında pek çok madde daha mevcuttur. Konu ile ilgili ekstra kaynaklardan faydalanılması önerilir.

1) Sistem ayrıntıları, oturum veya hesap bilgileri, SQL sorguları gibi hassas bilgiler içeren hata mesajları direkt sayfa üzerinde veya kaynağında kullanıcıya gösterilmemelidir.

Riskli
Warning: mysql_pconnect(): Access denied for user: 'root@localhost' (Using password: N1nj4) in /usr/local/www/wi-data/includes/database.inc on line 4

2) Girdi/çıktı doğrulaması:

Uygulamaya ve ilgili veri tabanına giren ve çıkan tüm değerlerin doğru yerde, doğru özellikler dikkate alınarak doğrulanması gerekmektedir.

Bunun için girdiler sisteme girmeden kontrol edilerek alfanumerik veya numerik olması, tanımlanan özellikleri sağlaması (e-posta, url, telefon numarası vs. için farklı formatlar) gibi kısıtlar getirilebilir. Uygun karakter setleri tanımlanmalıdır (UTF-8 gibi). Buna ek olarak potansiyel tehlikeli karakterlere engel olacak kara listeler hazırlanabilir, listedeki bu karakterler elimine edilecek veya işlem reddedilecektir. Bu karakterlere örnek olarak şunlar gösterilebilir:

Riskli
< > " ' % ( ) & + \ \' \"

Programlama dillerinde bu karakterlere engel olmak için yazılmış bazı hazır fonksiyonlar kullanılabilir.

Örnek: PHP’de htmlspecialchars(), htmlentities()

Riskli	Önerilen
<? $id=$_GET[“id”]; echo “Öğrenci numarası: “.$id; ?>	<? $id=$_GET[“id”]; $id=htmlspecialchars($id); echo “Öğrenci numarası: “.$id; ?>

3) SQL sorgularında içerikleri değiştirilebileceği için dinamik değerler kullanılmamalıdır:

Riskli
“select first_name, last_name from $test_users”;

SQL sorgularında parametreli sorgular (prepared statement) veya “stored procedure”’lar kullanabilir:

Riskli
$userName = $_SESSION['userName']; $itemName = $_POST['itemName']; $query = "SELECT * FROM items WHERE owner = '$userName' AND itemname = '$itemName';"; $result = mysql_query($query);
Önerilen
$mysqli = new mysqli($host,$dbuser, $dbpass, $db); $userName = $_SESSION['userName']; $itemName = $_POST['itemName']; $query = "SELECT * FROM items WHERE owner = ? AND itemname = ?"; $stmt = $mysqli->prepare($query); $stmt->bind_param('ss',$username,$itemName); $stmt->execute();

4) Üçüncü partilere ait kütüphaneler ve framework’ler kullanılıyorsa bunların güvenilir olduğundan emin olunmalıdır.

Bu kaynak kodların kontrolü kullanıcı üzerinde olmalıdır. Müdahale ihtimaline karşı dışarıdan link verilmemeli, dosyalar lokal sunucuda tutulmalıdır. Ayrıca güncelliği kontrol edilmelidir.

Riskli	Önerilen
<script src=”https://code.jquery.com/jquery-3.4.1.js”></script>	<script src=”scripts/jquery-3.4.1.js”></script>

5) Şifre işlemlerinde ve hassas veri transferinde HTTP GET yerine HTTP POST kullanılmalıdır.

Şifreler açık metin olarak kod içine gömülmemelidir:

Riskli
// varsayılan veri tabanı kullanıcısı: "admin" // varsayılan veri tabanı şifresi: "12345"

6) Dosya yükleme işlemlerinde bu dosyaların sadece kabul edilebilir isimde, formatta, boyutta ve içerikte olmasına dikkat edilmelidir.

7) Web uygulama dosyalarının bulunduğu ortam altında gereksiz dosya bırakılmamalıdır. Yedekler, test versiyonları ve kopyaları web üzerinden ulaşılamayacak bir dizinde saklanmalıdır. Kullanılmayan tüm dosyalar ilgili proje dizini altından silinmelidir.

8) Saldırganlar ".." ve "/" gibi ayırıcı karakterleri kullanarak sistemin başka bir yerinde bulunan dosyalara veya dizinlere erişmek için kısıtlanan konumun dışına ulaşabilir, güvensiz okuma ve yazma işlemleri yapabilirler. Dizin aşım saldırılarına engel olmak için tüm girdilerin doğrulanması (numerik, alfanumerik), kötü amaçla kullanılabilecek girdilerin içeriğine göre elimine edilmesi gerekir. Çeşitli dillerde mevcut olan “realpath”, “getCanonicalPath” gibi dosya yolu fonksiyonları kullanılabilir.

9) PHP’de “exec”, ASP’de “wscript.shell” gibi işletim sistemi komutlarını çalıştıran fonksiyonlar saldırganlar tarafından istedikleri komutları girdi alanlarına girerek kötü amaçla kullanılabilir. Buna engel olmak için tanımlanmış komut seti dışında komut çalıştırılmaması, girdilerin doğrulanması (numerik, alfanumerik) gerekir. İşletim sistemi komutlarını direkt çalıştırmak yerine kullanılan program diline ait API ve fonksiyonlar kullanılabilir.

10) Wordpress, Drupal vs. gibi hazır bir içerik yönetim sistemi kullanılacaksa versiyon kontrolleri düzenli olarak yapılmalı, her zaman son sürümler kullanılmalı, içerdiği plugin’ler de aynı şekilde güncellenmelidir.

Faydalı kaynaklar:

CWE/SANS TOP 25 Most Dangerous Software Errors
https://www.sans.org/top25-software-errors

OWASP Top Ten:
https://owasp.org/www-pdf-archive/OWASP_SCP_Quick_Reference_Guide_v2.pdf
https://owasp.org/www-pdf-archive/OWASP_Top_10-2017_%28en%29.pdf.pdf