Testing for Weak Encryption (Zayıf Şifreleme Testi)

Summary (Özet)

Şifreleme algoritmalarının yanlış kullanımları, hassas veri maruziyetine, anahtar sızıntısına, kırık kimlik doğrulamasına, güvensiz oturuma ve sahtecilik ataklarına neden olabilir. Zayıf olduğu bilinen ve MD5 ve RC4 gibi kullanım için önerilmemiş bazı şifreleme veya karma algoritmaları vardır.

Güvenli şifreleme veya hash algoritmalarının doğru seçeneklerine ek olarak, parametrelerin doğru kullanımları da güvenlik seviyesi için önemlidir. Örneğin, ECB (Elektronik Kod Kitabı) modu asimetrik şifrelemede kullanılmak üzere önerilmez.

Test Objectives (Test Hedefleri)

- Zayıf şifreleme veya hash kullanımı ve uygulamaları için bir kılavuz sağlayın.

How To Test (Nasıl Test Edilir)

Basic Security Checklist (Temel Güvenlik Kontrol Listesi)

- AES128 veya AES256 kullanırken, IV (Başlangıç Vektör) rastgele ve öngörülemez olmalıdır. Kriptografik Modüller için Güvenlik Gereksinimleri, bölüm 4.9.1 rastgele sayı üreteci testleri. Örneğin, Java'da, `java.util.Random` Zayıf bir rastgele sayı üreteci olarak kabul edilir. `java.security.SecureRandom` yerine kullanılmalıdır `java.util.Random` . .
- Asimetrik şifreleme için, güvenli bir eğri ile Elliptic Curve Cryptography (ECC) kullanın `Curve25519` Tercih edilir.
  - ECC kullanılamıyorsa, en az 2048bit tuşla RSA şifreleme kullanın.
- RSA'nın imza olarak kullanıldığında, PSS dolgusu önerilir.
- Zayıf hash / şifreleme algoritmaları bu tür MD5, RC4, DES,
  Blowfish, SHA1. 1024-bit RSA veya DSA, 160 bit ECDSA (elliptik eğriler), 80/112-bit 2TDEA (iki anahtar üçlü DES) kullanılmamalıdır.
- Minimum Anahtar uzunluğu gereksinimleri:

```
Key exchange: Diffie–Hellman key exchange with minimum 2048 bits
Message Integrity: HMAC-SHA2
Message Hash: SHA2 256 bits
Asymmetric encryption: RSA 2048 bits
```

```
Symmetric-key algorithm: AES 128 bits
Password Hashing: PBKDF2, Scrypt, Bcrypt
ECDH, ECDSA: 256 bits
```

- SSH, CBC modu kullanımları kullanılmamalıdır.
- Simetrik şifreleme algoritması kullanıldığında ECB (Elektronik Kod Defteri) modu kullanılmamalıdır.
- PBKDF2 şifreyi hash etmek için kullanıldığında, yineleme parametresinin 10000'in üzerinde olması önerilir. NIST ayrıca hash fonksiyonunun en az 10.000 yinelemesini önermektedir. Buna
  ek olarak, MD5 karma fonksiyonunun PBKDF2WithHmacMD5 gibi PBKDF2 ile kullanılması yasaktır.

Source Code Review (Kaynak Kod İncelemesi)

- Zayıf algoritmaların kullanımını tanımlamak için aşağıdaki anahtar kelimeleri arayın: `MD4, MD5, RC4, RC2, DES, Blowfish, SHA-1, ECB`
- Java uygulamaları için aşağıdaki API şifreleme ile ilgilidir. Şifreleme uygulamasının parametrelerini gözden geçirin. Örneğin,

```
SecretKeyFactory(SecretKeyFactorySpi keyFacSpi, Provider provider, String algorithm)
SecretKeySpec(byte[] key, int offset, int len, String algorithm)
Cipher c = Cipher.getInstance("DES/CBC/PKCS5Padding");
```

- RSA şifreleme için aşağıdaki dolgu modları önerilmektedir.

```
RSA/ECB/OAEPWithSHA-1AndMGF1Padding (2048)
RSA/ECB/OAEPWithSHA-256AndMGF1Padding (2048)
```

- Aramayı arayın `ECB` Çuvallamada kullanılmasına izin verilmez.
- Farklı IV (başlangıç Vektör) kullanılırsa gözden geçirin.

```
// Use a different IV value for every encryption
byte[] newIv = ...;
s = new GCMParameterSpec(s.getTLen(), newIv);
cipher.init(..., s);
...
```

- Aramayı arayın `IvParameterSpec` IV değerinin farklı ve rastgele oluşturulup oluşturulmadığını kontrol edin.

```
IvParameterSpec iv = new IvParameterSpec(randBytes);
SecretKeySpec skey = new SecretKeySpec(key.getBytes(), "AES");
Cipher cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
cipher.init(Cipher.ENCRYPT_MODE, skey, iv);
```

- Java'da, zayıf hash algoritmasının (MD5 veya CRC) kullanılıp kullanılmadığını kontrol etmek için MesajDöndü arama yapın. Örneğin:

`MessageDigest md5 = MessageDigest.getInstance("MD5");`

- İmza için, SHA1 ve MD5 kullanılmamalıdır. Örneğin:

`Signature sig = Signature.getInstance("SHA1withRSA");`

- Aramayı arayın `PBKDF2` . . Şifrenin hash değerini oluşturmak için, `PBKDF2` Kullanılması önerilmektedir. Oluşturmak için parametreleri gözden geçirin `PBKDF2` Değeri var.

Yinelemeler **10.000'in üzerinde** olmalı ve tuz **değeri** rastgele **değer olarak oluşturulmalıdır.**

```
private static byte[] pbkdf2(char[] password, byte[] salt, int iterations, int bytes)
    throws NoSuchAlgorithmException, InvalidKeySpecException
 {
     PBEKeySpec spec = new PBEKeySpec(password, salt, iterations, bytes * 8);
     SecretKeyFactory skf = SecretKeyFactory.getInstance(PBKDF2_ALGORITHM);
     return skf.generateSecret(spec).getEncoded();
 }
```

- Sert kodlu hassas bilgiler:

`User related keywords: name, root, su, sudo, admin, superuser, login, username, uid`
`Key related keywords: public key, AK, SK, secret key, private key, passwd, password, pwd, share key, shared key, cryto, base64`
`Other common sensitive keywords: sysadmin, root, privilege, pass, key, code, master, admin, uname, session, token, Oauth, privatekey, shared secret`

Tools (Araçlar)

- Nessus, NMAP (sırakralar) veya OpenVAS gibi savunmasızlık tarayıcıları, SNMP, TLS, SSH, SMTP vb. Gibi protokole karşı zayıf şifrelemeyi kullanmayı veya kabul etmeyi tarayabilir.
- Aşağıdaki durumlar için klowork, Fortify, Coverity, CheckMark gibi kaynak kodu incelemesi yapmak için statik kod analiz aracını kullanın.

CWE-261: Weak Cryptography for Passwords
CWE-323: Reusing a Nonce, Key Pair in Encryption
CWE-326: Inadequate Encryption Strength
CWE-327: Use of a Broken or Risky Cryptographic Algorithm
CWE-328: Reversible One-Way Hash
CWE-329: Not Using a Random IV with CBC Mode
CWE-330: Use of Insufficiently Random Values
CWE-347: Improper Verification of Cryptographic Signature
CWE-354: Improper Validation of Integrity Check Value
CWE-547: Use of Hard-coded, Security-relevant Constants
CWE-780 Use of RSA Algorithm without OAEP

Referances (Referanslar)

- NIST FIPS Standards
- Wikipedia: Initialization Vector
- Secure Coding - Generating Strong Random Numbers
- Optimal Asymmetric Encryption Padding
- Cryptographic Storage Cheat Sheet
- Password Storage Cheat Sheet
- Secure Coding - Do not use insecure or weak cryptographic algorithms
- Insecure Randomness
- Insufficient Entropy
- Insufficient Session-ID Length
- Using a broken or risky cryptographic algorithm
- Javax.crypto.cipher API
- ISO 18033-1:2015 – Encryption Algorithms
- ISO 18033-2:2015 – Asymmetric Ciphers
- ISO 18033-3:2015 – Block Ciphers