1.1.1 Jailbreak Kontrol Eksikliği (Missing Jailbreak Check) (CWE-693)

Açıklık Önem Derecesi: Düşük

Açıklığın Etkisi: Uygulamaya dair son kullanıcı bilgilerinin saldırganlarca elde edilebilmesi, uygulama sunucusuna kurban son kullanıcı üzerinden saldırılar yapılabilmesi

Açıklığın Barındıran Dosyalar/Satırlar:

Proje Dosyası/Dosya Adı	Satır Numarası

Açıklığın Açıklaması:

Jailbreak iOS mobil cihazların sistem güvenlik kilidinin kullanıcısı tarafından kırılarak telefona her yönüyle hükmetme saldırısına denir. Bu saldırıda zararlı işlem uygulayan programlar kullanıldığı için ve son kullanıcıyı dışarıdan gelecek tehditlere karşı koruyacak güvenlik katmanlarının kaldırılması dolayısıyla cihaz ele geçirilme girişimlerine karşı korunaksız olmaktadır. Eğer bir iOS uygulama böylesi bir mobil cihazda çalışırsa ilgili iOS uygulamadaki kritik veriler saldırganların eline geçebilir. Örneğin bir bankacılık uygulaması jailbreak'li bir mobil cihazda çalışırsa kullanıcının bankacılık bilgileri ele geçebilir. Bu v.b. tehditler dolayısıyla mobil uygulamalar tasarlanırlarken cihaz jailbreak'li mi kontrolü uygulamalıdır ve jailbreak'li değilse çalışmalıdır, jailbreak'liyse çalışmayı durdurmalıdır.

Kurum mobil iOS uygulamada cihaza karşı jailbreak kontrolü yapılmadığı tespit edilmiştir.

::::::::BULGU::::::::::

Açıklığın Önlemi:

iOS uygulamalarda (Objective C veya Swift uygulamalarda) cihazı test edici jailbreak kontrol metodu tanımlanmalıdır. Örneğin Objective C için bu şu şekilde yapılabilir:

Objective C:

```
BOOL hasBeenJailBroken()
{
   // 1 - Yaygın Olan Jailbreak Dosyaları Kontrol Edilir.
   if (
       [[NSFileManager defaultManager] fileExistsAtPath:@"/Applications/Cydia.app"] ||
       [[NSFileManager defaultManager]
fileExistsAtPath:@"/Library/MobileSubstrate/MobileSubstrate.dylib"] ||
       [[NSFileManager defaultManager] fileExistsAtPath:@"/bin/bash"] ||
       [[NSFileManager defaultManager] fileExistsAtPath:@"/usr/sbin/sshd"] ||
       [[NSFileManager defaultManager] fileExistsAtPath:@"/etc/apt"] ||
       [[NSFileManager defaultManager] fileExistsAtPath:@"/private/var/lib/apt/"]
    ) {
        return YES;
    }

    // 2 - Cydia AppStore'u Var mı Kontrol Edilir.
    if (
       [[UIApplication sharedApplication] canOpenURL:[NSURL
URLWithString:@"cydia://package/com.example.package"]]
    ) {
        return YES;
    }

    // 3 - /private/ Dizinine Rastgele Bir Dosya Yazarak Yazma İzinleri Kontrol Edilir.
    NSError *error;
    NSString *stringToBeWritten = @"Eğer başarılı şekilde yazdırılırsa cihaz jailbreak'lidir.";
    [stringToBeWritten writeToFile:@"/private/WriteTestJailbreak.txt" atomically:YES
encoding:NSUTF8StringEncoding error:&error];
    [[NSFileManager defaultManager] removeItemAtPath:@"/private/WriteTestJailbreak.txt" error:nil];

    if(error == nil)
    {
        return YES;
    }
    return NO;
}
```

Ardından tanımlanan jailbreak kontrol metodu AppDelegate.m dosyasının didFinishLaunchingWithOptions metodunda çağırılmalıdır.

Objective C:

```
if ( hasBeenJailBroken() ) {
    // Jailbreak Tespit Edildi.
} else {
    // Normal uygulama akışı devam eder.
}
```

Örneğin Swift uygulamalarda bu metot şöyle tanımlanabilir:

Swift:

```
func hasBeenJailBroken() -> Bool {
    let fileManager = FileManager.default

    // 1 - Yaygın Olan Jailbreak Dosyaları Kontrol Edilir
    if fileManager.fileExists(atPath: "/Applications/Cydia.app") ||
       fileManager.fileExists(atPath: "/Library/MobileSubstrate/MobileSubstrate.dylib") ||
       fileManager.fileExists(atPath: "/bin/bash") ||
       fileManager.fileExists(atPath: "/usr/sbin/sshd") ||
       fileManager.fileExists(atPath: "/etc/apt") {
        return true
    }

    // 2 - Cydia AppStore'u Var mı Kontrol Edilir.
    if UIApplication.shared.canOpenURL(URL(string: "cydia://package/com.example.package")!)
{
        return true
    }

    // 3 - /private/ Dizinine Rastgele Bir Dosya Yazarak Yazma İzinleri Kontrol Edilir.
    let stringToWrite = "Eğer başarılı şekilde yazdırılırsa cihaz jailbreak'lidir."
    do {
        try stringToWrite.write(toFile: "/private/JailbreakTest.txt", atomically: true,
encoding: String.Encoding.utf8)
        return true
    } catch {
        return false
    }

    return false
}
```

Ardından tanımlanan jailbreak kontrol metodu AppDelegate class'ının didFinishLaunchingWithOptions metodunda çağırılmalıdır.

Swift:

```swift
func application(_ application: UIApplication, didFinishLaunchingWithOptions
launchOptions: [UIApplication.LaunchOptionsKey: Any]?) -> Bool {
    if hasBeenJailBroken() {
        // Jailbreak Tespit Edildi.
    } else {
        // Normal uygulama akışı devam eder.
    }
    return true
}
```

Böylece objective-c veya swift uygulama jailbreak'li bir mobil cihazda çalıştırılmaya çalışıldığında if() bloğuna girilecektir, jailbreak'li olmayan bir mobil cihazda çalıştırılmaya çalışıldığında else bloğuna girilecektir. if() bloğuna girildiğinde örneğin "Jailbreak'li cihaz kullanıyorsunuz, bu güvenlik riskleri oluşturduğundan bu uygulama kullanılamayacaktır" şeklinde bir toast mesajı ekrana verilebilir ve uygulamanın kullanılırlığı durdurulabilir. Else bloğunda ise cihaz jailbreak kontrolünü atlattığından uygulama normal akışında devam edecek şekilde ayarlanabilir.

Referanslar:

1. https://cwe.mitre.org/data/definitions/693.html
2. https://developer.apple.com/documentation/uikit/uiapplicationdelegate/1622921-application
3. ChatGPT # Question: Where to add jailbreak detection code snippet to swift application?