Önemli Güvenlik Gereksinimleri

www.sourceflake.com
Bedirhan Urgun

Ocak 3

2016

Bu doküman hem kurum içi hem de üçüncü parti şirketlerin mobil uygulamalar geliştirirken dikkat edilmesi gereken temel güvenlik maddelerini detaylandırır. Bu anlamda doküman, minimum gereksinim dokümanı olarak kullanılabilir.

Mobil Uygulamalar

Önemli Güvenlik Gereksinimleri - Mobil Uygulamalar

Giriş

Lokasyon bazlı servisler, mobil sosyal ağlar, mobil bilgi arama, mobil ödeme (NFC), nesne tanıma, mobil mesajlaşma ve e-posta, mobil video gibi trendler mobil uygulamaların artarak geliştirilmesini sağlamaktadırlar.

Bu noktada, her geliştirme platformunda olması gerektiği gibi mobil uygulama platformlarında da güvenli uygulama geliştirme tekniklerine ihtiyaç vardır.

Mobil uygulama pazarı değişime oldukça müsait gözükmekte olmasına rağmen üretilen uygulamaların o veya bu şekilde pazardaki akıllı cihazlarda uzun süre yaşayacağı unutulmamalıdır. Her durumda uygulamaların mobil uygulamalara özel riskleri belirlenmeli ve güvenli geliştirilmeye çalışılmalıdır.

Doküman mobil uygulamalar geliştirilirken dikkat edilmesi gereken güvenlik kontrollerinden bahsetmektedir.

Hassas Bilgi

Doküman içinde hassas bilgi olarak sınıflandırılan bilgiler aşağıda tanımlanmaya çalışılmıştır. Bir veri tipinin hassaslığı hakkında şüpheye düşüldüğünde proje analisti ile görüşülüp, bilgi güvenliği birimleri ile iletişime geçilmelidir.

- Aboneye ait muhtelif haberleşme bilgileri (CDR) (Abone hangi tarihte, kiminle, nereden, hangi saatte, ne kadar süreyle görüştüğü)
- Konum bilgisi,
- Şifre, PIN, PUK
- OTP onay kodları
- Kimlik ve özel bilgileri (adresi, kredi kartı numarası, rehber kayıtları, TCKno vb.)

Güvenlik Politikaları

Uygulama

Native mobil uygulamalar genellikle uzak servisleri kullanırlar. Uygulama sunucularında yaşayan bu uygulamaların geliştirilmelerinde web uygulama güvenlik prosedürlerine (girdi denetimi, oturum yönetimi, yetkilendirme, veri güvenliği v.b.) uyulmalıdır [3].

Özellikle kısa olarak IDOR şeklinde isimlendirilen en kritik güvenlik zafiyetlerinden biri olan Insecure Direct Object Reference zafiyetinin uygulamalarda olmaması için, belirleyici ID'ler olarak veritabanlarındaki kayıtlar ile eşleştirilen parametrelerin (örneğin customerNo, faturaID, statementID, xyzID, v.b.) sahiplik kontrolleri sunucu tarafında mutlaka ama mutlaka gerçekleştirilmelidir [4]. Örneğin kullanıcı aşağıdaki 233 değerini 234 yapıp başka bir kullanıcının order'ına ulaşamamalıdır.

https://www.cokguvenliuygulama.com/products/orders/233

Ayrıca sunucu tarafındaki uygulamaların yönetim arayüzlerinde kimlik doğrulama ve IP kısıtlaması mutlaka uygulanmalıdır. Özellikle eğer mümkün ise Internet'e erişimleri kapatılmalıdır. Örneğin Wordpress veya özel uygulama yönetim admin sayfaları.

Kimlik doğrulama, yetkilendirme ve oturum yönetimi (session mantığı) sanki bir web uygulaması geliştiriliyormuş gibi gerçeklenmelidir. Giriş ve rol kontrolleri atlanmamalı ve bütün dışarıya açılmış metotlar için gerçeklenmelidir. Kimlik doğrulama mekanizması sadece görünürde olmamalıdır. Örneğin, uygulama kullanıcı adı ve şifre alıp daha sonra bu ikiliyi sunucuda kontrol ettikten sonra bu oturum için bir sessionID ile iletişime devam etmelidir. Bu sessionID logout olana kadar her istekte sunucu tarafında kontrol edilmelidir.

Client tarafına sadece gönderilmesi gereken bilgiler gönderilmeli, başka herhangi bir veri extra'dan gönderilmemelidir. Örneğin sadece sunucu tarafında oluşturulması gereken SMS onay kodu, kullanıcının telefonuna SMS olarak out-of-band şeklinde gönderilmeli, kesinlikle native uygulamaya http veya başka bir yol ile iletilmemelidir.

Kontroller her zaman sunucu tarafında gerçekleştirilmelidir.

Güvenli Veri Depolama

Uygulamaların kullandığı hassas bilgiler depolama mekanizmaları tarafından kullanılmamalıdır. Ayrıca veri depolama dosyaları, yapılandırma dosyaları, fatura PDF'leri cihaz üzerinde örneğin SDCARD gibi herkes tarafında ulaşılabilir yerlerde tutulmamalıdır.

Verilerin tutulduğu dosya işletim sistemi hakları sadece uygulama erişebilecek şekilde verilmelidir. Dosya sistemi izolasyonu en güvenli opsiyonları ile gerçekleştirilmelidir. Örneğin Android cihazlarda default opsiyon olan MOD_PRIVATE gibi.

Açık Yönetim Arayüzleri

Mobil uygulamaların bağlandığı backend uygulamaların sadece kısıtlı sayıda yetkililerin bağlandığı yönetim arayüzlerinin olması çok yaygındır. İçerik yönetiminin ve yapılandırmanın dinamik gerçekleştirmesi için geliştirilen bu arayüzlere erişim mutlaka ama mutlaka kullanıcı adı/şifre hatta mümkünse IP kısıtlamaları ile yetkilendirme arkasına alınmalıdır.

HotSpot Riski

Mobil cihazlarda tethering/hotspot yöntemi ile internet bağlantısının paylaşılması mümkündür. Özellikle 3G/4G bağlantılarda kullanılan ve kullanıcıları herhangi bir kullanıcı adı/şifre girmeksizin backend uygulamalarda kimliklerinin doğrulanabileceği Radius kimlik doğrulama metodu, internet bağlantısını paylaşan kullanıcılar için ciddi risk barındırmaktadır.

Bu riski düşürebilmek adına mobil uygulamalar internet bağlantı tipini kontrol etmeli ve WIFI tipi kullanılmakta ise kullanıcıyı mutlaka kullanıcı adı/şifre alınan arayüze yönlendirmelidirler.

Kayıt Tutma

Logcat, NSLog veya diğer client mobil uygulamalarda kayıt alınırken hassas bilgilerin yazılmaması gerekmektedir.

Debug Amaçlı Kodların veya Gereksiz Metotların Kaldırılması

Reverse engineering ile uygulama kodları tekrar oluşturulabileceğinden test ortamlarında DEBUG amaçlı yazılmış kodların prod ortamı için uygulama içeriden çıkarılması gerekmektedir.

Veri Güvenliği

Mobil uygulamalar ve web servisleri ile HTTPS kullanarak haberleşmelidir. Bu haberleşme yapılırken native kod tarafında kesinlikle validation kuralları by-pass edilmemelidir. Örneğin Android cihazlarda özel HostnameVerifier'lar, iOS'de NSURLAuthenticationMethodServerTrust gibi yapılar kullanılmamalıdır.

Genellikle bu bypass teknikleri test sistemlerinde self-signed SSL sertifikalara sahip uygulama sunuculara sorunsuz bağlanmak için uygulanmaktadır. Prod sistemlerinde bu bypass teknikleri kesinlikle uygulamadan kaldırılmalıdır.

Anti Kaba Kuvvet

Özellikle kimlik doğrulama işlemlerinde deneme-yanılma işlemlerini engellemek amacı ile anti kaba kuvvet mekanizmaları gerçekleştirilmelidir. Belirlenebilecek hatalı belirli deneme sayısından sonra native tarafta CAPTCHA gösterilmesi örnek bir kaba kuvvet önleme tekniğidir. Ancak bu mekanizma kesinlikle Session'a bağlı olmamalı, hem IP ve hem de kullanıcı adı'na bağlı olmalıdır. Yani kullanıcı adının sabit tutulup şifrelerin değiştirilmeye çalışılması veya kullanıcı adının değiştirilerek şifrelerin sabit tutulması gibi saldırı metotların engellenmeleri gerekmektedir.

Anti kaba kuvvet sadece kimlik doğrulama fonksiyonlarında değil, hassas olabilecek bütün işlemler de gerçeklenebilir. Örneğin fatura ID'lerinin birer birer arttırılarak gerçek bir fatura ID'si bulunması veya arka tarafta uzun zaman alan bir işlemin defalarca çağırılabilmesi, v.b.

Uygulama Bileşenleri veya Uygulamalar Arası Güvenli İletişim

Özellikle Android uygulamalarda broadcast edilen verilerin güvenliği çok önemlidir. Intent'lerin broadcast edilmesi uygulamalar arası iletişimin bir yöntemidir. Bu şekilde sisteme gönderilen Intent'ler, doğru Intent Filter'lar yardımı ile Broadcast Receiver'ler tarafından yakalanırlar ve işlenirler.

Bu şekilde veri ve bilgilendirme iletişiminde dikkatli olunması gereklidir. Çünkü, gönderilen veriler veya bilgilendirmeler herhangi bir uygulama tarafından kayıt edilmiş Broadcast Receiver tarafından da yakalanabilirler. Uygulamalar bu durum söz konusu olduğunda, broadcast edilen Intent'ler ile hassas veriler göndermemelidirler veya permission'lar bu Intent'lerin her filter tarafından alınamayacağı şekilde implement edilmelidir.

SQL Injection, XSS

Client side veritabanını kullanan native uygulamalar sql sorgularında kesinlikle Prepared Statement kullanmalıdırlar.

Ayrıca webview gibi html/javascript yüklenebilecek bileşenler kullanıldığında mutlaka bu yapılarda render edilecek verilerin uygun encode işleminden geçirildiğine emin olunmalıdır. Veriler javascript yapıları içine gidecek ise Javascript encoding, HTML içerisine gidecek ise HTML encoding, URL içine parametre olarak gidecek ise URL encoding, HTML attribute içerisinde gidecek ise HTML Attribute encoding uygulanmalıdır [6, 7].

İletişim

bedirhan.urgun@sourceflake.com

Referanslar

1. https://www.owasp.org/index.php/IOS_Developer_Cheat_Sheet
2. http://www.webguvenligi.org/docs/Guvenli_Android_Gelistirme_Ipuclari.pdf
3. http://code.google.com/p/owasp-development-guide/
4. https://www.owasp.org/index.php/Top_10_2013-A4-Insecure_Direct_Object_References
5. https://www.owasp.org/index.php/XSS_%28Cross_Site_Scripting%29_Prevention_Cheat_Sheet
6. https://www.owasp.org/index.php/OWASP_Java_Encoder_Project