Mobil Penetrasyon Testi Nedir ve Neleri Kapsar?

Mobil penetrasyon testi mobil cihazlardaki (telefon, tablet, kiosk, akıllı ev eşyaları,…) barınan apk uygulamalara yapılan test / analiz işlemlerine denmektedir. Mobil penetrasyon testleri kapsam olarak mobil cihazlardaki uygulamaların (apk'ların) decompile edilmiş kodlarında (.java kodlarında) bir tür statik analiz yapmaya, uygulama (.apk) çalışır haldeyken dinamik analiz yapmaya, uygulamada tersine mühendislik ile gömülü zararlı var mı tespiti / incelemesi yapmaya ve uygulamanın bir web sunucuyla haberleşmesi varsa haberleşmedeki trafiği gözlemleyerek kullanılan istemci veya sunucu taraflı web api'ını test etmeye dayanır.

Mobil penetrasyon testleri kapsam olarak çeşitlilik arz ettiğinden birçok araç kullanımı söz konusu olabilir. Örneğin; mobil uygulamadan (.apk'dan) decompile edilmiş kodlarda bir tür statik analiz yapmak için araçlar (AndroBugs, AndroGuard,...), mobil uygulama (.apk) çalışır haldeyken dinamik analiz yapmak için araçlar (Drozer, Mobile Security Framework (MobSF), Frida, ...), mobil uygulamada tersine mühendislik ile gömülü bir zararlı var mı tespiti / incelemesi yapmak için araçlar (Dex2Jar, ApkTool, ...), ve mobil uygulama bir web sunucu ile haberleşme halindeyse haberleşme trafiğini kullanarak istemci veya sunucu tarafındaki web api'ını analiz etmek için araçlar (Burpsuite, Owasp ZAP, …) gibi.

Mobil penetrasyon testleri tıpkı masaüstü yazılımlara yapılan penetrasyon testleri gibidir. Mobil penetrasyon testlerinde uygulamaya (.apk'ya) dinamik analiz, tersine mühendislik ile statik analiz, tersine mühendislik ile zararlı tespit analiz ve uzak bir sunucu haberleşmesi varsa haberleşme üzerinden uzak sunucu analizi olduğu gibi aynı şekilde masaüstü yazılım penetrasyon testlerinde yazılıma (.exe, ...) dinamik analiz, tersine mühendislik ile statik analiz, tersine mühendislik ile zararlı tespit analiz ve uzak bir sunucu haberleşmesi varsa haberleşme üzerinden uzak sunucu analizi vardır.

Örneğin mobil cihazlardaki bir uygulamanın (söz gelimi Spotify'ın) tersine mühendislik yoluyla kodları elde edilebilir ve lisanslama satırları keşfedilerek uygulama lisansı geçerli ve kalıcı olsun manipulasyonu yapılabilir. Bu şekilde full lisanslı uygulama sürümü (söz gelimi crack'li Spotify) hazırlanabilir. Tıpkı masaüstü yazılıma tersine mühendislik yoluyla kodları elde edilmesi ve lisanslama satırları keşfedilerek yazılım lisansı geçerli ve kalıcı olsun manipulasyonu yapılabilmesi gibi. Bu şekilde full lisanslı yazılım sürümü (söz gelimi crack'li Adobe Photoshop) hazırlanması gibi.

Örneğin mobil cihazlardaki uygulamalar tıpkı masaüstü uygulamalar gibi uzak bir sunucu ile haberleşme halinde olabilirler. Mobil cihazda bu bir kütüphane uygulaması olabilir ve web sunucu ile haberleşerek arşiv taraması yapma imkanı sunabilir. Veya mobil cihazda bu bir web tarayıcı uygulaması olabilir ve her türlü ziyaret edilen web sunucu ile haberleşme imkanı sunabilir. Mobil cihazlardaki bu v.b. uygulamaların uzak sunucu ile haberleşme trafiği arasına proxy yazılımlar ile girerek http trafiği yakalama ve manipulasyonlar gerçekleştirilerek testler yapılabilir. Tıpkı desktop PC'lerdeki uygulamaların uzak sunucu ile olan haberleşme trafiği arasına sistem / tüm trafik proxy yazılımları ile girerek http trafiği yakalama ve manipulasyonlar gerçekleştirme testleri yapılabilmesi gibi.

Dolayısıyla mobil penetrasyon testleri masaüstü yazılımlara yapılan penetrasyon testleri ile aynı kapsama sahiptirler. Biri mobil cihazdaki uygulamaları baz alırken diğeri desktop PC'deki uygulamaları baz alır.

Kaynaklar

https://www.perforce.com/blog/sca/what-static-code-analysis
https://resources.infosecinstitute.com/how-to-get-started-as-a-mobile-penetration-tester/#gref
https://en.wikipedia.org/wiki/Web_API
https://resources.infosecinstitute.com/android-penetration-tools-walkthrough-series-androguard/#gref
https://gurelahmet.com/mobil-android-s%C4%B1zma-testine-giri%C5%9F/
https://resources.infosecinstitute.com/android-penetration-tools-walkthrough-series-drozer/#gref
https://stackoverflow.com/questions/1249973/decompiling-dex-into-java-sourcecode