Ders 2 - Hardcoding Issues > Part 1

Dersin Hedefi

Diva uygulamasındaki “2. Hardcoding Issues > Part 1” seçeneğine tıklayın. Bu seçenek ile hedefiniz hardcoding (açık açık kodlanmış) bir veri aramaktır ve varsa ne türden bir veri olduğunu tespit etmektir.

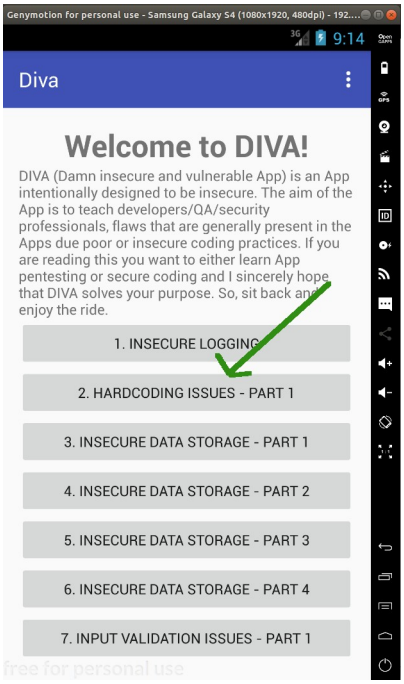
Dersin Açıklaması

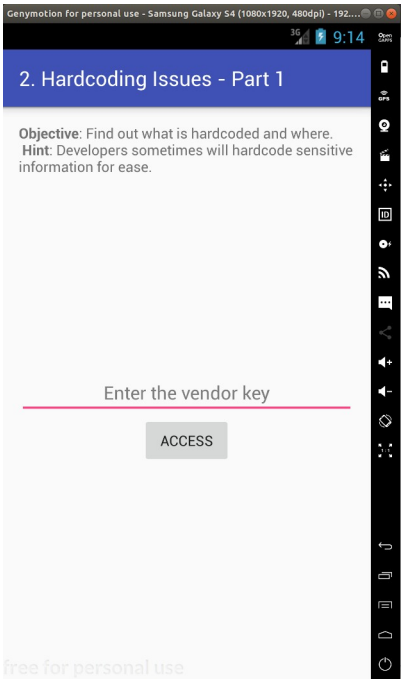
Hardcoding, kelime manasıyla direk Türkçeye çevirdiğimizde pek anlam ifade etmeyen ama anlam itibariyle açık açık kodlama demek olan bir ifadedir. Geliştiriciler (yazılımcılar) genellikle güvenlikçi bakış açısına sahip olmadıklarından ve sadece uygulamalarının çalışır hale kavuşması yolunda kodlamayla meşgul olduklarından hardcoding işlemine sıklıkla başvururlar. Hardcoding (yani açık açık kodlama) hassas verilerin uygulama kaynak koduna olduğu gibi yazılması anlamına gelir. Örneğin bu durum uygulamanın kaynak kodlarındaki veritabanı bağlantısı satırlarında karşınıza gelebilir. Veritabanı bağlantısı satırlarında uzak veritabanı sunucusuna erişim için kullanıcı adı ve şifre bilgileri yer alabilir ve geliştirici bunları açık açık koda yerleştirebilir. Bu v.b. durumlarda uygulamanın kaynak kodlarını elde eden veya bir şekilde görebilen kötü niyetli bir kimse uygulama hakkında hassas verilere erişebilir halde olur ve zarar odaklı faaliyetlerini geliştirebilir. Bu nedenle geliştiricilerin uygulamalarının kaynak kodlarında hassas nitelikte veri bulunduracakları yerlere hardcoding olarak (yani açık açık) hassas veriyi döşemek yerine çeşitli üstünü kapatma tekniklerine (örn; şifrelemeye) başvurması gerekmektedir.

Bu derste “Hardcoding Issues” sayfasında bir hardcoded veri tespiti yapılacaktır ve bu verinin ne türden bir veri olduğu tespiti yapılarak saldırganın bunu faydaya nasıl kullanabileceğine bakılacaktır.

Dersin Çözümü

Ders ekranına bir göz atalım.





Uygulama ekranındaki sayfayı ve davranışını anlamak adına göz attığımızda bir vendor key (yani dağıtıcı anahtarı) metin kutusu mevcuttur ve bir de Access butonu mevcuttur. Anlaşıldığı üzere dağıtıcı anahtar metin kutusuna girilen anahtar doğru olduğu takdirde Access butonu ile bir üçüncü taraf servise erişim hakkı sağlayacağız veya dağıtıcı anahtar metin kutusuna girilen anahtar yanlış olduğu takdirde Access butonu ile bir servise erişim hakkı sağlayamayacağız. Buradaki servise erişmeyi örneğin uzak bir ayrı sunucudaki bir uygulamanın yeteneklerine bu uygulama içerisinde kullanabilme / kavuşabilme olarak hayal edebiliriz.

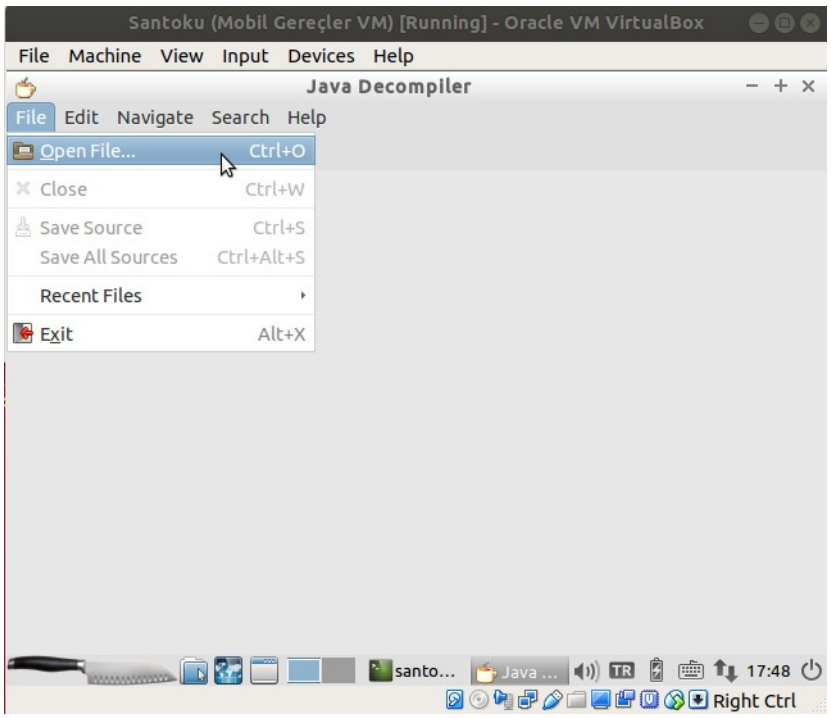
Normal şartlarda herhangi sıradan bir kullanıcı anahtara sahip olamayacağından sayfanın sunduğu servise erişim hakkı elde edemeyecektir. Sadece yetkililer anahtarı bileceklerinden servise erişim hakkı elde edebileceklerdir. Şimdi ders gereği hardcoded bir veri arayacağımız için bunu uygulama kaynak kodlarını inceleyerek yapabileceğimizden mobil cihazdan çalışan uygulamanın kaynak kodlarını elde etmeye çalışalım.

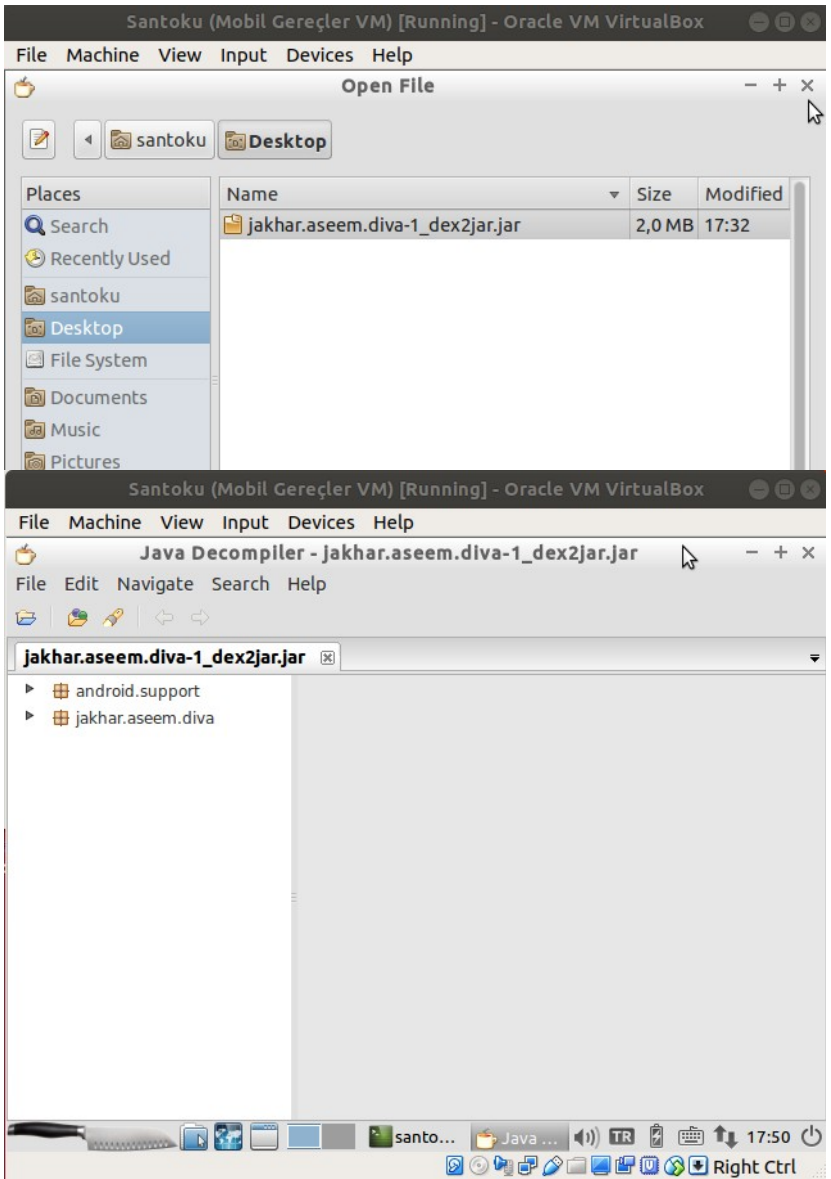
Bu makale dizisinin en başında yaptığımız uygulama binary dosyasını (.apk dosyasını) bilgisayara çekme, üzerinde tersine mühendislik yapma ve uygulamanın okunabilir java dosyalarını elde etme işini tekrarladığımızı varsayalım. Uygulamanın okunabilir java dosyalarını JD-GUI editörü ile inceleyerek bu uygulama sayfasını (“Hardcoding Issues > Part 1” sayfasını) sunan / kontrol eden java dosyasını tespit edelim. Bahsedilen işlemlerin ayrıntısı için bkz. Başlangıç: Android Uygulamalarda Tersine Mühendislik ile Okunabilir Java Kaynak Kodları Elde Etme (Dex2Jar, JD-GUI, ApkTool)

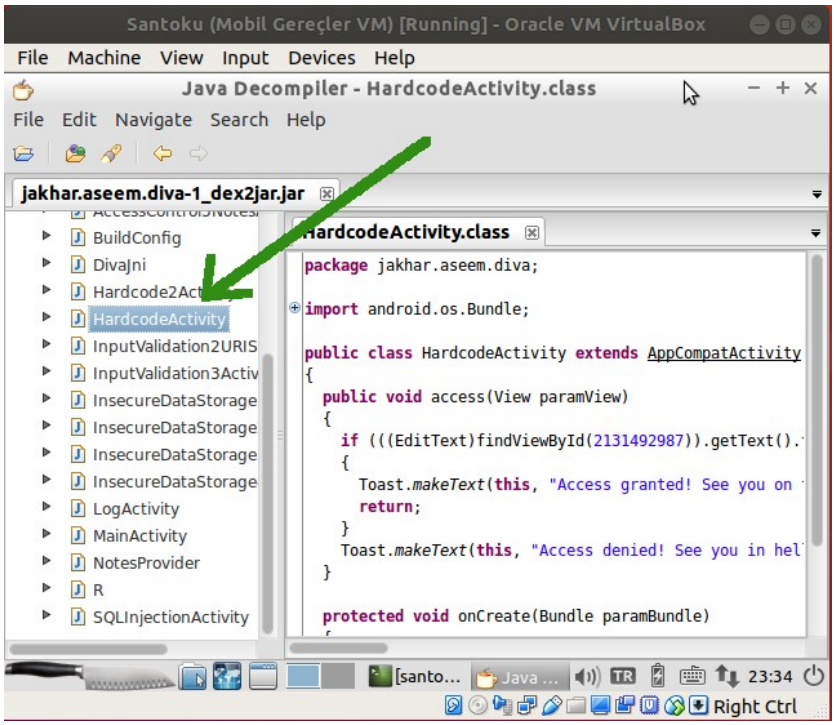
Santoku Linux Terminal:

> jd-gui

Çıktı:

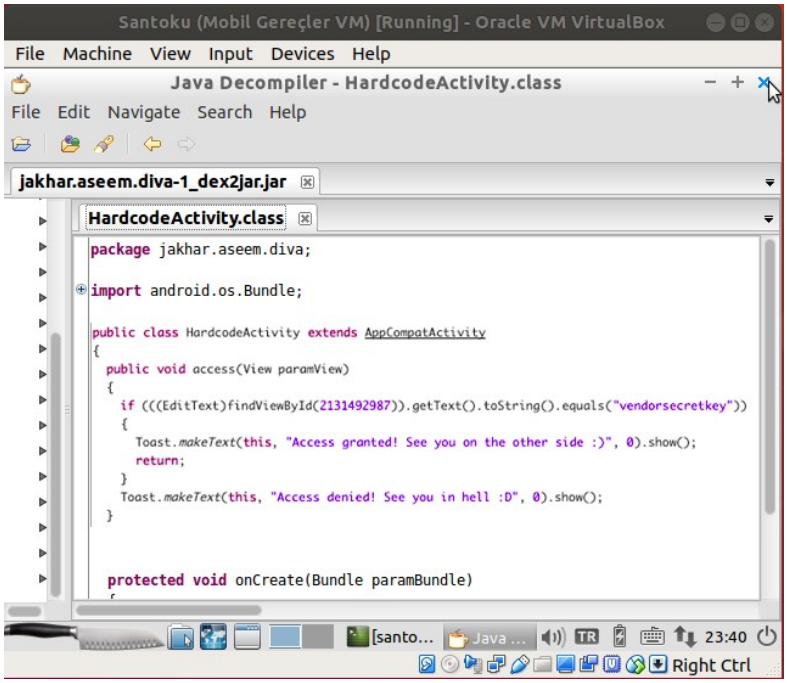

Santoku (Mobil Gereçler VM) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Java Decompiler
File Edit Navigate Search Help
Open File... Ctrl+O
Close Ctrl+W
Save Source Ctrl+S
Save All Sources Ctrl+Alt+S
Recent Files
Exit Alt+X
santo...
Java ...
TR
17:48
Right Ctrl


Santoku (Mobil Gereçler VM) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Open File
santoku
Desktop
Places
Search
Recently Used
santoku
Desktop
File System
Documents
Music
Pictures
Name
Size
Modified
jakhar.aseem.diva-1_dex2jar.jar
2,0 MB
17:32
Santoku (Mobil Gereçler VM) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Java Decompiler - jakhar.aseem.diva-1_dex2jar.jar
File Edit Navigate Search Help
jakhar.aseem.diva-1_dex2jar.jar
android.support
jakhar.aseem.diva
santo...
Java ...
TR
17:50
Right Ctrl



Java kaynak kod dosyaları incelendiğinde uygulamada görüntülüyor olduğumuz ekranın hangi java dosyası tarafından sunulduğu / kontrol edildiği yukarıdaki gibi görülebilir. Bu örnek için java dosyasını bulmak uygulama ekranındaki sayfa ismi ile benzerliği dolayısıyla kolay olacaktır ama gerçek bir senaryoda java dosyaları arasında inceleme yapmak ve doğru java dosyasını bulmak için okumalar yapmak gerekecektir.

Şimdi mobil cihaz ekranında görüntülemekte olduğumuz “Hardcoding Issue > Part 1” sayfasını sunan / kontrol eden HardcodeActivity.class java dosyası içeriğini inceleyelim.



Uygulama ekranındaki Hardcoding Issues - Part 1 sayfasını sunan / kontrol eden java kaynak kodlara (HardcodeActivity.class java dosyasına) bakıldığında access() isimli bir metot vardır. Bu metot Türkçe ifadeyle erişim anlamına gelmektedir. Bu metot içerisinde bir if koşulu konulmuştur ve koşula göre uygulama sayfasında yer alan metin kutusu nesnesinden gelen veri "vendorsecretkey" kelimesiyle kıyaslanmaktadır.

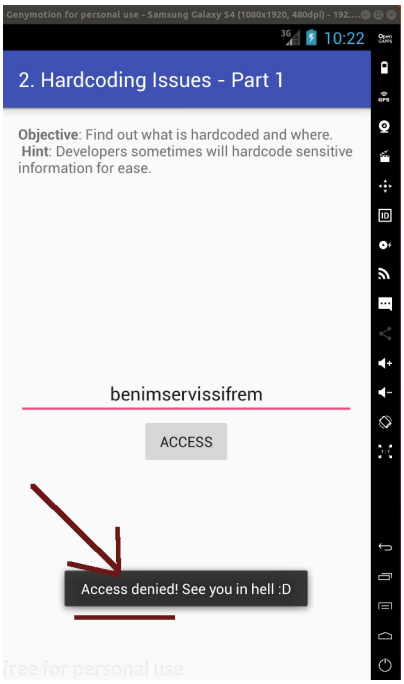
HardcodeActivity.class:

```
...
...
public void access(View paramView)
{
        if (((EditText)findViewById(2131492987)).getText().toString()
        .equals("vendorsecretkey"))
        {
                Toast.makeText(this, "Access granted! See you on the other
                side :)",0).show();
                return;
        }
        Toast.makeText(this, "Access denied! See you in hell :D", 0).show();
}
...
```

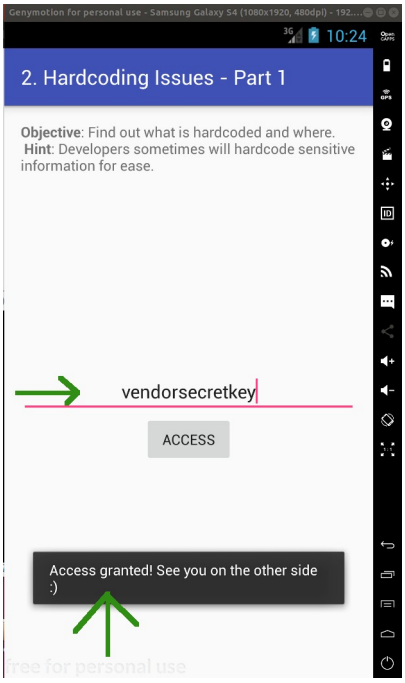
Eğer if koşulundaki kıyaslama başarılı olursa (yani eşleşme sağlanırsa) koda göre if koşulunun içerisine girilecektir ve ekrana "Access granted…" tarzında erişim sağlandı bilgisi yansıyacaktır. Eğer if koşulundaki kıyaslama başarısız olursa (yani eşleşme sağlanamazsa) uygulama ekranındaki if atlanacaktır ve hemen aşağısındaki kod satırı çalışıp ekrana "Access denied…" tarzında erişim reddedildi bilgisi yansıyacaktır.

Sonuç olarak java dosyasındaki kaynak kodlara bakarak hardcoded bir hassas veri var olduğunu söyleyebiliriz: "vendorsecretkey". Bu kelime hassas bir veridir, çünkü ekranda anahtar girilmesini isteyen metin kutusundaki veriler if koşulunda "vendorsecretkey" kelimesiyle kıyaslanmaktadır ve if koşulu başarılı olduğunda (eşleşme olduğunda) içerisinde "Access granted…" (Erişim sağlandı…) şeklinde ekrana çıktı sunmaya hazır bir kod satırı yer almaktadır.

Dolayısıyla uygulama sayfasına rastgele bir anahtar verisi girdiğimizde uygulama erişim başarısız yanıtını dönecekken,



uygulama sayfasına uygulamanın java kaynak dosyalarından elde ettiğimiz hardcoded veriyi (anahtarı) girdiğimizde,



erişim başarılı bilgisi dönecektir.

Böylece binary kodlu uygulama dosyasını bilgisayardan tersine mühendislik yaparak java kodlarını elde edip okuma sonucunda hardcoded (açık açık) bir hassas veri tespit ettik ve bu hassas verinin geçerli anahtar verisi olduğu tespitini yaptık.

Sonuç

Bu derste android bir uygulamanın java kaynak kodlarında hardcoded (açık) hassas veri bulundurması sonucu yetki kontrolü aşamasının nasıl iptal edilebileceği gösterilmiştir. Mobil cihazlarında bu android uygulamayı kullanan ve yetkili olmayan herkes uygulamanın bu yetki kontrolü sayfasını; tersine mühendislik, uygulama java kaynak kodlarını okuma, hardcoded veri tespit etme, ve hardcoded verinin ne iş için var olduğunu tespit etme işlemlerini uygulayarak atlatabileceklerdir. Yani kullanıcılar yetkisizken biraz çabayla servise erişim hakkı elde edebilir durumdalar.

Bu ders bize uygulama geliştiricilerinin uygulama kaynak kodlarına hassas verileri olduğu gibi koymamasını söylemektedir. Aksi takdirde bu derste olduğu gibi kötü niyetli kimseler örneğin yetkisi olmadığı uygulama özelliklerine erişebilirler.

Uygulama geliştiricilerinden bir ödev niteliğinde beklenen şey kolaya kaçmamak ve uygulama kaynak kodlarına hassas verileri hardcoded olarak (yani açık bir şekilde) koymamaktır. Bunun yerine hassas verileri çeşitli şifreleme metotları ile şifreleyerek kaynak kodlara koymaktır. Bu sayede uygulama kaynak kodlarına erişecek / erişme teknik bilgisine sahip kötü niyetli kimseler kaynak kodlardaki hassas verileri elde etme noktasında bir güçlükle karşılaşacaklardır.

EK: Android Uygulamaların Java Kaynak Kodlarının Elde Edilebilmesi Hakkında

Mobil cihazlardaki android uygulamalar binary (.apk) halden okunabilir java dosyaları haline dönüştürülebilmektedir (decompile edilebilmektedir). Android uygulama geliştiricilerinin geliştirdikleri ve mağaza aracılığıyla kullanıcı cihazlarına sundukları uygulamalarının kaynak kodlarının okunabilmesi işleminin önüne geçmek için tek geçerli çözüm geliştiricilerin uygulamalarını bir sunucuda konumlaması ve kullanıcıların, cihazlarından sunucuya bağlanarak uygulamayı kullanmalarıdır. Bu bir çeşit mobil bulut ile servis olarak uygulama sunma çözümüdür. Bu şekilde uygulama kaynak kodları sunucuda kalacaktır ve kullanıcılar / kötü niyetli kimseler uygulama kaynak kodlarını elde edemeyeceklerdir.

Bu çözüm teknik açıdan zorluklar içermektedir. Çünkü kullanıcılar cihazlarından uygulamayı network aracılığıyla kullanacakları için örneğin her kullanıcının internet bağlantı kalitesinin aynı standartta olamayacağı açıktır. Bu durum kullanıcıların uygulama kullanımını çileli hale getirecek bir unsurdur. Aynı şekilde kullanıcıların her bir uygulama işlemi sunucuda onaylanıp cevaba göre hareket edileceğinden ufacık network'sel hataların kullanıcı tarafında uygulama arayüzünün sürekli crash yaşamasına sebebiyet vereceği de açıktır. Bu şekilde kullanım kullanıcıları bıktırmaktan başka bir şey sunmayacaktır. Bu olay net olarak henüz desktop / mobil bulut hizmetlerinin servis olarak uygulama sunma imkanının mevcut sistem ve network kaynaklarımızla mümkün olmayışındandır.

Bu nedenle android, uygulama geliştiricilerine bir başka çözüm sunmaktadır. Uygulama kaynak kodlarının görülmemesini sağlayan bu alternatif çözüm geliştiricilerin uygulamalarını derlerken kaynak kodlarında karartma uygulamalarını (obfuscate etmelerini) sağlar. Yani kaynak kodlar karartılarak (özleştirilmekte, sıkılaştırılmakta ve karıştırılmaktadır) okunamaz hale getirilmektedir. Resmi Android SDK araçları arasında bu işi yapan ProGuard adlı araç mobil uygulama geliştiricilerine uygulamalarının kaynak kodlarını karartma imkanı sunar. Uygulamalar karartılmış halde mağazaya servis edildiklerinde ve kullanıcılar mobil cihazlarına uygulamaları karartılmış halde indirdiklerinde kötü niyetli eller uygulama dosyalarını bilgisayara çekip tersine mühendislik yaparak bu sefer java kaynak kodlarını eskisi gibi elde edemeyeceklerdir. Bunun yerine daha çok, parça parça (belli belirsiz) java kaynak kodları elde edeceklerdir ve böylece kötü niyetlilerin işi biraz güçleşmiş olacaktır.

Kaynak

https://stackoverflow.com/questions/32132434/set-adb-vendor-keys
https://stackoverflow.com/questions/4336637/is-it-really-impossible-to-protect-android-apps-from-reverse-engineering
https://www.perfomatix.com/proguard-android/