Ders 5 - Insecure Data Storage - Part 3

Dersin Hedefi

Diva uygulamasındaki “5. Insecure Data Storage - Part 3” seçeneğine tıklayın. Bu seçenek ile açılan kayıt sayfasında üçüncü taraf servise kaydolmak için girilen kullanıcı adı ve parola bilgilerinin nerede ve ne şekilde depolandığını keşfedin. Ardından bu kayıt sayfasındaki girilen hesap bilgilerini güvensiz depolayan kod satırlarını belirtin. Not: Bir önceki makaleye göre bu sefer farklı bir yöntemle üçüncü taraf servise kayıt bilgileri depolanmaktadır.

Dersin Açıklaması

*Uyarı: Bir önceki Insecure Data Storage > Part 2 makalesini okuduysanız bu makaledeki **Dersin Açıklaması** başlığını atlayabilirsiniz. Çünkü bir önceki makaledeki aynı bilgiler bu başlık altında yer almaktadır.*

Android uygulamalarda android uygulama verilerini depolama ikiye ayrılmaktadır: Yerel veri depolama ve uzak veri depolama. Yerel veri depolama birbirinden amaç ve kapsam açısından farklılık arzeden birçok tekniği barındırmaktadır. Uzak veri depolama ise uzakta çalışan bulut veritabanının teknolojisine göre değişiklik gösterir.

Bu “Insecure Data Storage > Part 1-4” (Güvensiz Veri Depolama > Part 1-4) mini serisinde yerel veri depolama yöntemleri arasından birbirinden farklı, fakat diva zafiyetli mobil uygulamasında üçüncü taraf bir servise kayıt olma sayfasındaki girilen bilgileri (kullanıcı adı ve şifre bilgilerini) depolama noktasında ortak işlev görecek dört farklı yöntem üzerinden gidilecektir. Bunlar; “Shared Preferences” (Paylaşımlı Tercihler), “SQLite Database” (SQLite İlişkisel Veritabanı), “Internal File Storage” (Dahili Dosya Depolama) ve “External File Storage” (Harici Dosya Depolama) şeklindedir. Bu yerel depolama yöntemlerine ilave olarak “Saved Instance State”, “Internal Cache Files”, “Realm Database”,... şeklinde listeye eklemeler yapılabilir. Fakat zafiyetli diva uygulamamızı ilgilendirmediğinden başka yöntemlere girilmeyecektir.

Yerel depolama yöntemi SharedPreferences (Paylaşımlı Tercihler) uygulama ayar / girdi / … tercihlerinin saklanmasında kullanılır. SQLite uygulamadaki kullanıcı taraflı girdilerin / verilerin veritabanı teknolojisi ile depolanmasında kullanılır. Internal File Storage uygulamadaki kullanıcı taraflı girdilerin / verilerin cihazın dahili depolama ünitesinde metin dosyası halinde depolanmasında kullanılır. External File Storage uygulamadaki kullanıcı taraflı girdilerin / verilerin cihazın harici depolama ünitesinde metin dosyası halinde depolanmasında kullanılır.

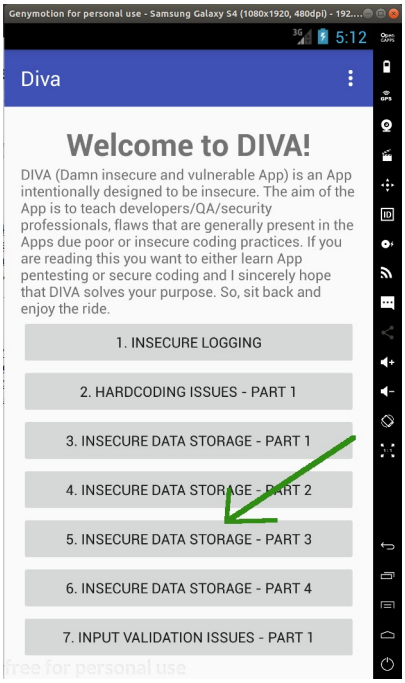
Android cihazlarda yerel veri depolama yöntemleri birbirlerinden **amaç ve kapsam** olarak farklıdırlar. Örneğin bu mini seride bahsedeceğimiz “Shared Preferences” tekniği sadece string, float, integer verileri depolar (çünkü amacı kullanıcı uygulama tercihlerini depolamaktır ve bu nedenle xml formatında veri depolar), resim-ses-video gibi verileri depolamaz, örneğin bu mini seride değinmeyeceğimiz “Saved Instance State” yerel depolama yöntemi sadece uygulama nesnelerinde yapılan değişiklikleri anlık kaydeder ve kazara sayfada geri gitme gibi durum olduğunda mevcut durumun / verilerin korunurluğunu sürdürür, veya bu mini seride değinmeyeceğimiz bir diğer yerel depolama yöntemi “Internal Cache Files” sadece kısıtlı süreli veri tutmaya yarar (çünkü amacı cihaz depolamasını verimli kullanmaktır)... gibi.

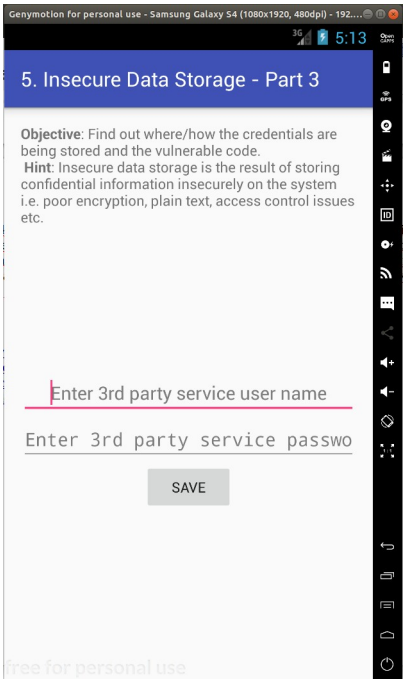
Bu güvensiz veri depolama mini (part 1 - 4) serisinde dört farklı yerel depolama yönteminin yanlış şekilde kullanımı nedeniyle kullanıcı hassas verilerinin (diva uygulamasındaki üçüncü taraf servis kullanıcı hesap bilgilerinin) ele geçirilebileceği gösterilecektir.

Dersin Çözümü

Uyarı: Bir önceki Insecure Data Storage > Part 2 makalesini okuduysanız bu makaledeki ***Dersin Çözümü*** *başlığı çoğunlukla aynı metne sahiptir. Ancak diva uygulamasındaki Insecure Data Storage > Part 3 dersinin sunduğu metot gereği bir noktada ayrışmaktadır.*

Diva uygulamasının "5. Insecure Data storage - Part 3" sayfasında senaryo gereği bizi uygulama içi üçüncü taraf bir servise kaydolma sayfası karşılamaktadır. Bu kayıt işlemi ile diva uygulaması içerisinden üçüncü taraf bir servisin işlevlerini / özelliklerini kullanabilir hale geleceğiz.

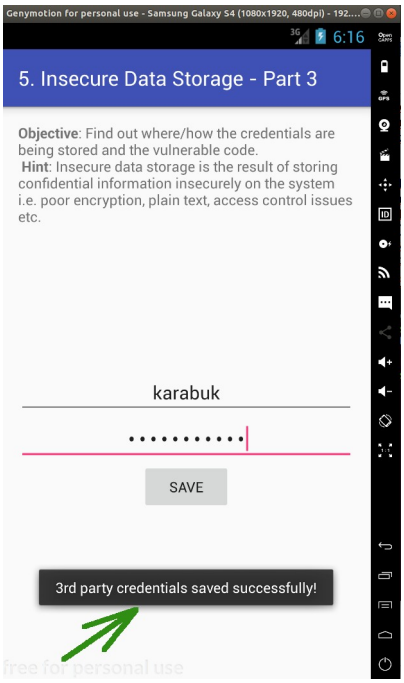




Bu sayfaya gireceğimiz kayıt bilgileri normal şartlarda uzak sunucuda kayıt altına alınacaktır. Fakat ders sayfasındaki senaryoya göre uygulamayı başka zamanlarda kullanırken üçüncü taraf servise her defasında elle giriş yapmayalım diye kayıtlı hesabımızın mobil cihazda yerel olarak depolanması söz konusudur. Bu şekilde diva uygulamasını kullanırken üçüncü taraf servise her daim bağlı halde kalmış olacağız ve uygulamayı kullanırken her daim üçüncü taraf servisin işlevlerini / özelliklerini kullanabilir halde olacağız.

Bizim amacımız ise yerel android sistemde güvensiz şekilde depolanan üçüncü taraf servis hesabımızın nerede ve ne şekilde depolandığını tespit etmektir.

Şimdi uygulama sayfasında üçüncü taraf servise kayıt olmak için bir kullanıcı adı ve şifre girelim;
Kullanıcı adı: karabuk, Şifre: password123



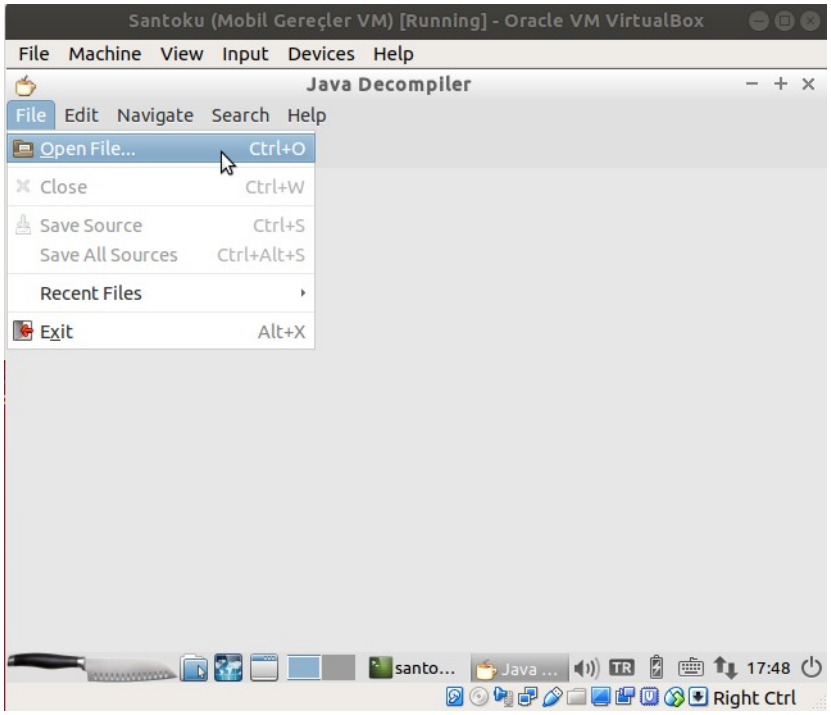
Bu bilgiler uzak sunucuda bir tür veritabanına kaydolur. Yerel mobil cihazda ise sonradan kullanım için bir dosya halinde saklanır / kaydedilir.

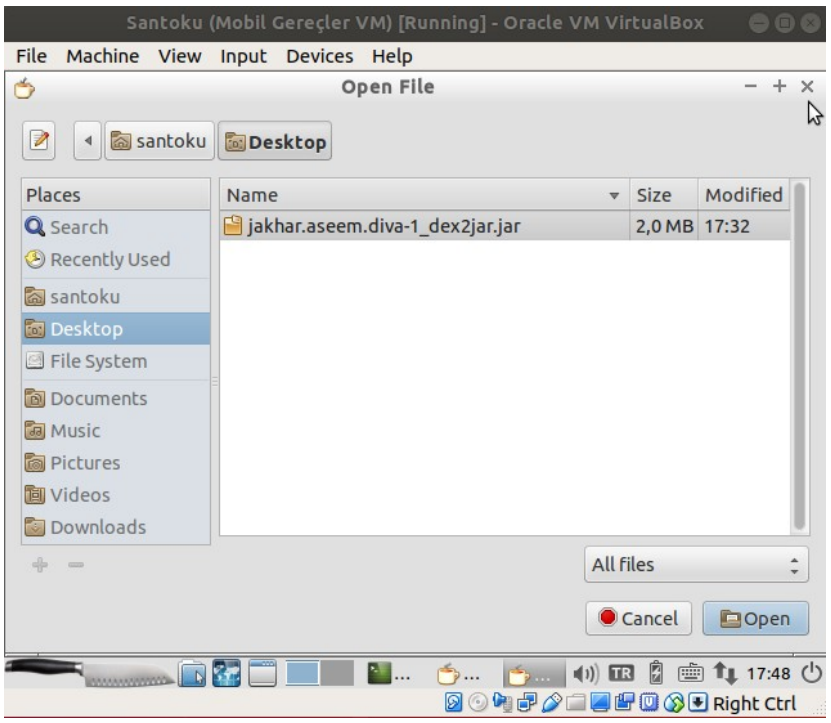
Şimdi bu makale dizisinin en başında yaptığımız uygulama binary dosyasını (.apk dosyasını) bilgisayara çekme, üzerinde tersine mühendislik yapma ve uygulamanın okunabilir java dosyalarını elde etme işini tekrarladığımızı varsayalım. Uygulamanın okunabilir java dosyalarını JD-GUI editörü ile inceleyerek bu uygulama sayfasını (“Insecure Data Storage - 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)

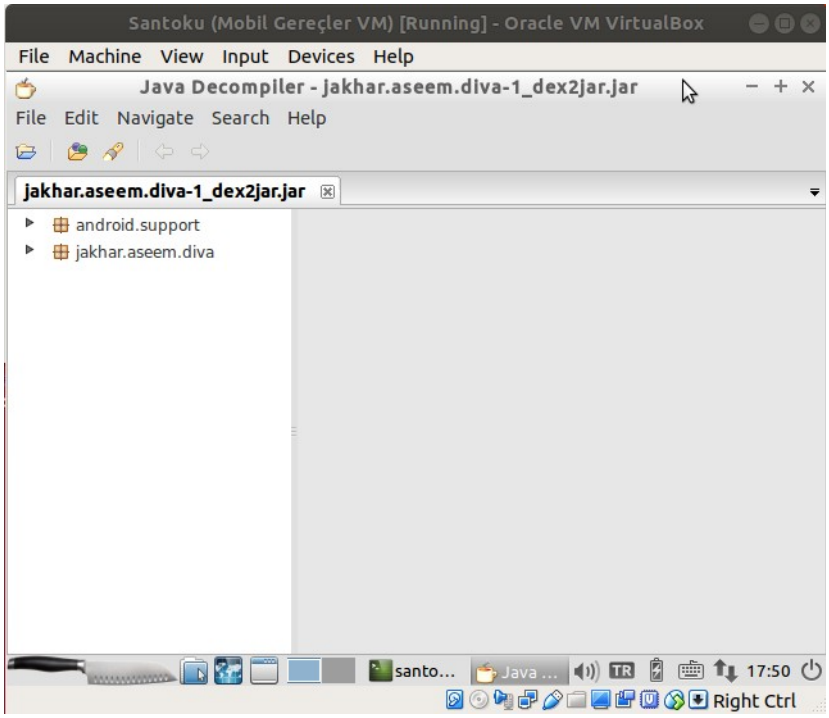
Santoku Linux Terminal:

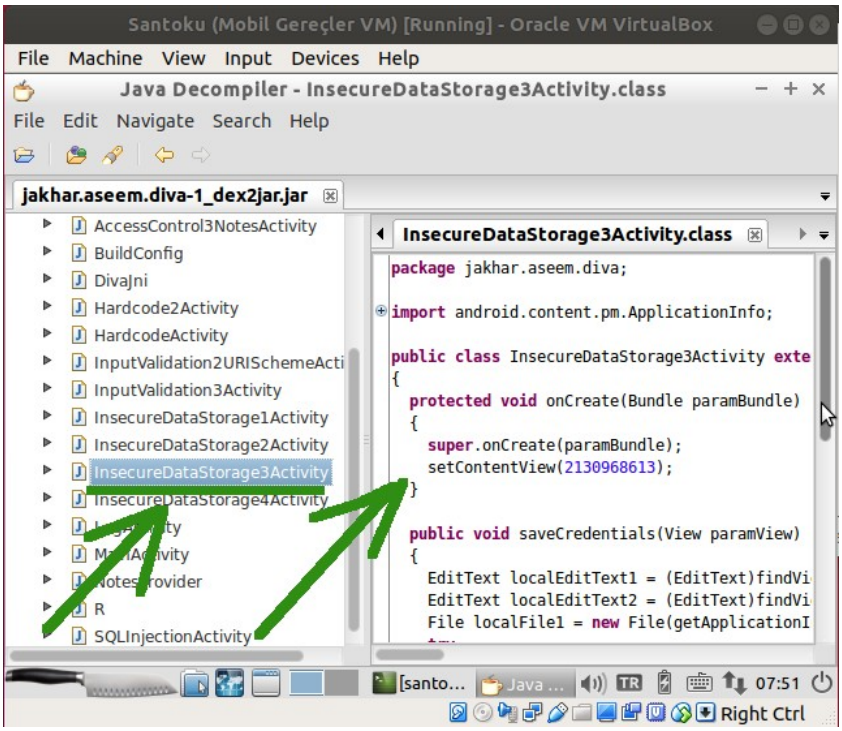
> jd-gui

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

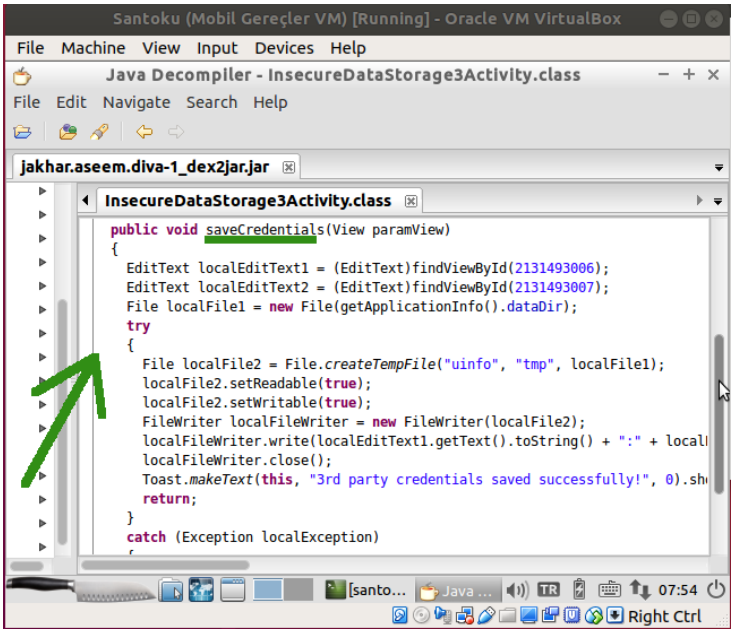

Santoku (Mobil Gereçler VM) [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Open File
santoku
Desktop
Places
Search
Recently Used
santoku
Desktop
File System
Documents
Music
Pictures
Videos
Downloads
Name
Size
Modified
jakhar.aseem.diva-1_dex2jar.jar
2,0 MB
17:32
All files
Cancel
Open
TR
17:48
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 daha önceki makalelerde de bahsedildiği üzere uygulama ekranındaki sayfa ismi ile benzerliği dolayısıyla kolay olmuştur ama gerçek bir senaryoda java dosyaları arasında inceleme yapmak ve doğru java dosyasını bulmak için okumalar yapmak gerekecektir.

Üçüncü taraf servise kaydolduğumuz mobil uygulama sayfasını sunan / kontrol eden java dosyası yukarıdaki gibi InsecureDataStorage3Activity.class'tır. Bu java dosyası incelendiğinde saveCredentials() isimli (yani Türkçe ifadeyle hesap bilgilerini kaydet isimli) bir metot kullanımı görünmektedir.



Metot içerisi incelendiğinde uygulama sayfasındaki metin kutuları nesneler halinde, içerdikleri verilerle beraber localEditText1 ve localEditText2'e atanmaktadırlar. Yani bu nesnelerden biri kullanıcı adını, diğeri parolayı kelime halinde tutmaktadır. Sonra localFile1 nesnesine diva uygulamasının android sistemde bulunduğu dizin yolu kelime olarak atanmaktadır. Bu şekilde try try isimli bloğa girildiğinde File sınıfı createTempFile() metodu ile bir önceki satırda elde edilen diva uygulaması dizin yolu içerisinde (yani mobil cihazın dahil diskinde) geçici bir yerel uinfo isminde dosya oluşturulmaktadır, dosyaya read ve write (okuma / yazma) izinleri tanımlanmaktadır, ve diskteki bu dosya dosya içeriği yazma nesnesine sunularak diskteki dosyaya metin kutularındaki kullanıcı adı ve parola bilgileri yazdırılmaktadır.

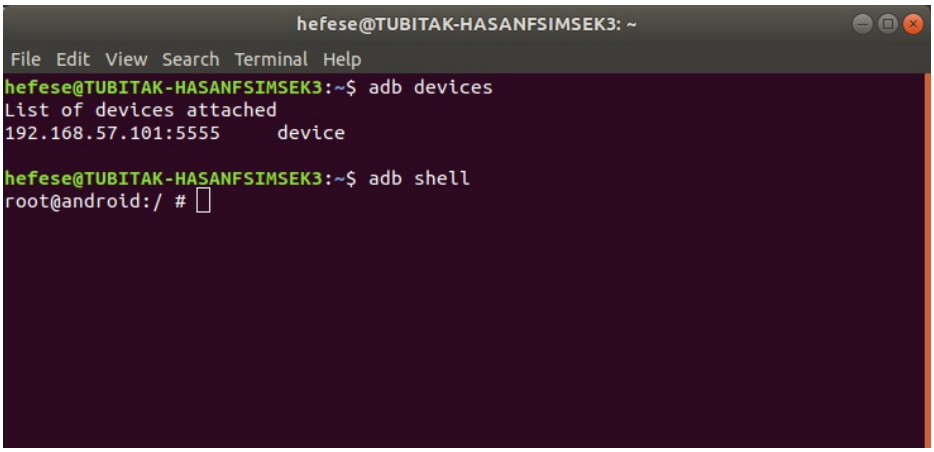
Bu uygulama sayfasını sunan / kontrol eden java dosyasındaki saveCredentials() metodu içerisinde servis hesap bilgilerini depolamak için mobil cihazın dahili diskinde geçici dosya oluşturma yöntemi tercih edilmiştir. Fakat geçici kelimesi kulağa yanıltıcı gelebilir. Java programlama dili resmi dökümantasyonuna göre java'da geçici dosya oluşturma metodu createTempFile() esasında kısıtlı süreli ömre sahip dosya oluşturmak için değil, geçici dosyaların geçiciliğini ara yolla kullanmak için vardır. Yani geçici dosyaların özelliğine sahip (kolaylıkla diskten silinebilen), fakat müdahale yapılmadığında kalıcı olarak kalmaya devam eden dosya oluşturmaya yaramaktadır. Sonuç olarak bu java dosyasında servis hesap bilgilerini depolamak için mobil cihazın dahili diski tercih edildiğinden kullanılan yerel depolama yöntemi "Internal File Storage" (Dahili Dosya Depolama) şeklindedir.

Uygulama java kaynak kodunu inceleyerek kullanıcı hassas verilerinin nerede tutulduğu bilgisine (Dahili diskte mobil uygulamanın bulunduğu dizin altında olduğu bilgisine) eriştik. Şimdi ne şekilde tutulduğu bilgisine erişmek için bilgisayardan mobil cihaza bağlanalım ve kullanıcı hassas verilerinin tutulduğu dahili diskteki diva uygulaması dosya yoluna gidip dosyayı görüntüleyelim.

Ubuntu 18.04 LTS Linux Terminal:

> adb devices
> adb shell

Çıktı:



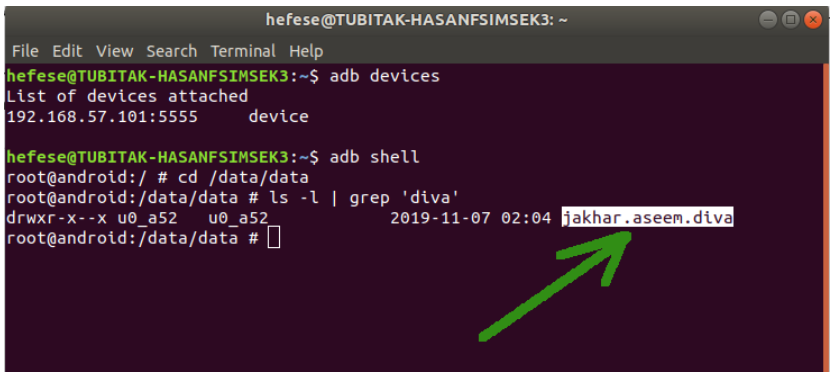
Resmi Android SDK aracı adb ile linux bir bilgisayardan mobil cihaza bağlandık ve shell komutu ile mobil cihazın komut satırını aldık. Şimdi yapılacak şey mobil cihazdaki Diva uygulamasının paket ismini öğrenmektir, sonra android sistemdeki /data/data dizini altında öğrendiğimiz paket ismindeki dizinin içerisine girmektir. Bu şekilde uygulama paket ismindeki klasör altında yer alan dosyalar listelenecektir.

Linux bilgisayardan mobil cihazla olan komut satırı oturumuzda Diva uygulamasının android sistemdeki paket ismini tespit edelim.

Ubuntu 18.04 LTS Terminal:

root@android:/ # cd /data/data/
root@android:/data/data # ls -l | grep 'diva'

Çıktı:

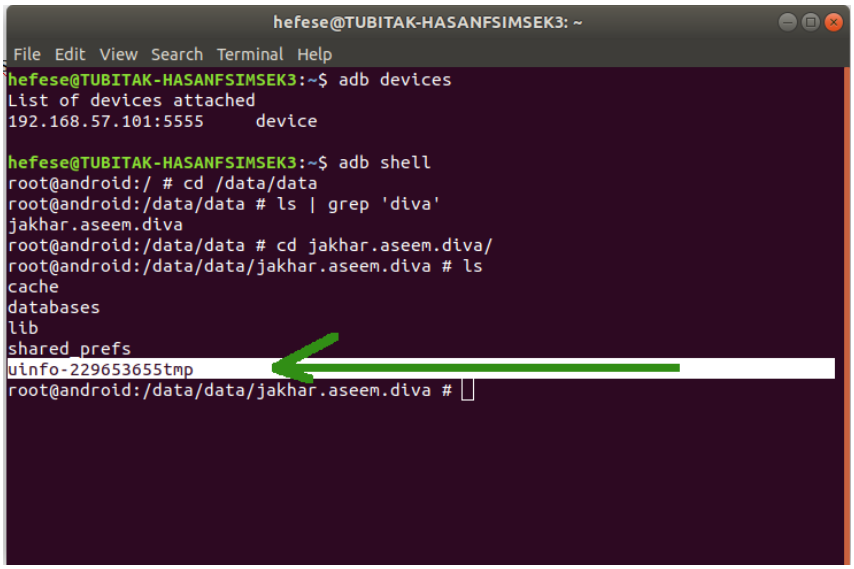


Ardından /data/data dizini altındaki diva ugulamasının android sistemdeki paket isminde olan klasöre girelim ve dosyaları listeleyelim.

Ubuntu 18.04 LTS Terminal:

root@android:/data/data # cd jakhar.aseem.diva/
root@android:/data/data/jakhar.aseem.diva # ls

Çıktı:

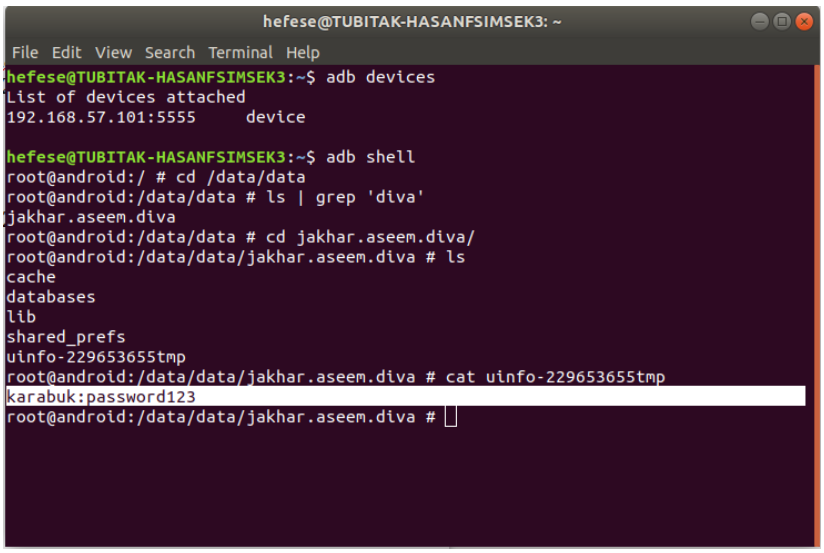


Dahili diskteki uygulama dizini altında listelenen dosyalar ve klasörler arasında uygulama sayfasının oluşturduğu yerel dosyayı görüntülemekteyiz: uinfo-229653655tmp. Bu dosya içeriğini ekrana bastığımızda,

Ubuntu 18.04 LTS Terminal:

root@android:/data/data/jakhar.aseem.diva # cat uinfo-229653655tmp

Çıktı:



Görüldüğü üzere üçüncü taraf servis hesap bilgileri mobil cihazda dahili diskte bir yerel dosyada açık bir şekilde (güvensiz bir şekilde) depolanmıştır. Bu mobil cihaza veya başka söz gelimi diva uygulamasını kullanan mobil cihazlara sızmayı başaran saldırganlar diva uygulaması klasörü içerisinde kritik bir veri elde edebilir miyim diye inceleme yaptıklarında yerel uinfo-229653655tmp dosyası onlara kullanıcının diva uygulamasındaki bağlı olduğu üçüncü taraf servisin hesap bilgilerini verecektir.

Şimdi uygulama sayfasındaki kullanıcı hesap bilgilerini kaydetme metodundaki güvensiz veri depolayan kod satırlarını gösterelim ve başlığı bitirelim.

```
public void saveCredentials(View paramView)
{
        EditText localEditText1 = (EditText)findViewById(21314930000);
        EditText localEditText2 = (EditText)findViewById(21314930001);
        File localFile1 = new File(getApplicationInfo().dataDir);
        try
        {
                File localFile2  File.createTempFile("uinfo", "tmp", localFile1);
                localFile2.setRreadable(true);
                localFile2.setWritable(true);
                FileWriter localFileWriter = new FileWriter(localFile2);
                localFileWriter.write(localEditTExt1.getText().toString() + ":" +
localEditText2.getText().toString() + "\n");
                localFileWriter.close();
                Toast.makeText(this, "3rd party credentials saved successfully!", 0).show();
                return;
}
```

Vurgulanan satırda görüldüğü üzere uygulama sayfasındaki metin kutularından gelen veriler olduğu gibi dosyaya yazdırılmaktadır. Bu satır metin kutusu verilerini normalde bir çeşit şifreleme metoduyla şifreleyip o şekilde dosyaya yazdırılmalıydı.

>> Güvensiz Veri Depolama Kod Satırları

```
localFileWriter.write(localEditTExt1.getText().toString() + ":" +
localEditText2.getText().toString() + "\n");
```

Bunun yerine direk olduğu gibi metin kutusundan gelen kullanıcı adı ve şifre bilgileri dosyaya yazdırıldığından Internal File Storage yerel depolama yöntemi kullanıcı adı ve şifreyi ifşa eder durumda kalmıştır.

Sonuç

Diva uygulamasının Insecure Data Storage - Part 3 sayfasındaki girilen üçüncü taraf servise kaydolma hesap bilgilerinin nerede ve ne şekilde kaydolduğu bilgisi tersine mühendislik ile elde edilen uygulama kaynak kodlarının incelenmesiyle tespit edilmiştir. Nerede olduğu bilgisi kaynak kodlarda kullanılan File java sınıfı ismi ve createTempFile() metodu isminden, ne şekilde tutulduğu bilgisi ise (yani şifreli mi, yoksa açık metin halinde mi bilgisi) android sistemdeki dahili diskte yer alan diva uygulaması klasörü altındaki dosya içeriği görülerek anlaşılmıştır.

Bu derste vurgulanmak istenen açıklık diva uygulamasındaki kullanıcı servis hesap bilgilerinin yerel diskte dosya içerisinde açık metin halinde / şifrelenmeden tutulmasıdır. Kullanıcı servis hesap bilgileri bu şekilde yerel diskte tutulduğu için mobil cihaza ilerde sızacak bir saldırgan kullanıcının hassas bu verilerini alabilir durumdadır.

Örneğin android dünyası jargonunda “root”lamak (root izne çıkmak) denilen ve iOS dünyası jargonunda “jailbreak”lemek (kafesi geçmek / kırmak) denilen metot ile mobil cihazları üzerinde tüm erişim yetkilerini açan birçok kullanıcı vardır. Bu sayede kullanıcılar resmi uygulama mağazası dışında yabancı yerlerden uygulama indirip kullanabilmeyi, veya mobil cihazlar üzerinde kendini geliştirmek adına sistemsel işlemler yapabilmeyi, veya bunun gibi verilen sınırların ötesinde mobil cihazda işlemler yapabilmeyi hedeflerlerken fark etmeden cihazlarını korumasız hale getirmektedirler ve bu derste olduğu gibi bir uygulama kullanıcı hassas verilerini eğer cihazda güvensiz depoluyorsa cihaz root’lu / jailbreak’li olduğunda cihaza erişen herkes kullanıcı hassas verilerinin olduğu uygulama dizinlerine erişim hakkına sahip olacaktır. Normal şartlarda mobil cihazlardaki uygulama dizinlerine sadece kullanıcı cihazı (kullanıcı android hesabı) erişir / görebilir durumdadır. Bu şekilde bir saldırgan cihaza sızsa dahi izin engeline takılıp uygulama dizinlerine geçemeyebilir ve uygulama dizinlerinde şifresiz / açık metin halinde hassas veriler dursa dahi o dizinlere geçiş yapamadığından okuyamayabilir / elde edemeyebilir. Ancak root’lanan / jailbreak’lenen cihazlarda tüm erişim izinleri açıldığından ilerde mobil cihaza sızacak bir saldırgana cihazdaki her dosya yoluna gidebilirsin izni verilmiş olmaktadır ki bu şekilde saldırganlar kullanıcıların uygulamalara dair hassas verilerini elde edebilir durumda olacaklardır.

> Bilgi:
>
> Bundan önceki güvensiz veri depolama makalelerinde bahsedilen kullanıcıların uygulamalara dair şifresiz / açık metin halindeki hassas verilerinin elde edilebilir olduğu bilgisi çeşitli başka atak vektörleri ile (örn; kullanıcı android sistem hesap bilgisinin ele geçmesi ile) uygulama dizinlerine geçiş hakkı kazanıldığı durum varsayılarak denmiştir. Uygulama dizinlerine geçiş koşulu sağlandığı takdirde kullanıcının uygulamaya dair hassas verileri şifresiz / açık metin halinde olduğundan elde edilebilirdir demek daha açık bir ifade olabilir.

Sonuç olarak burada güvenliğin kademelerden oluştuğu farkındalığı sunulmuştur. Fakat nihayetinde mobil cihazlardaki android uygulamaların yerel diskte kullanıcı hassas verilerini şifrelemeden tutmaları kullanıcının mobil cihazına ilerde sızacak saldırganlara karşı “kaptırma imkanı doğurur”. Bu nedenle mobil cihazlardaki android uygulamalarda yerel olarak depolanacak kullanıcı verileri hassas niteliğindeyse şifreli olarak yerel sistemde tutulmalıdırlar.

Kaynak

https://www.loginworks.com/blogs/top-10-ways-know-store-data-android/
https://stackoverflow.com/questions/16691437/when-are-java-temporary-files-deleted
https://source.android.com/devices/tech/connect/third-party-call-apps
https://developer.android.com/guide/topics/data/data-storage
https://developer.android.com/training/data-storage/shared-preferences
https://www.tutorialspoint.com/android/android_shared_preferences.htm