İzin ve Sistem Tabanlı Android Antivirüs Uygulaması

Ezel KOC

January 2018

1 Giriş

Açık kaynak kodlu ve Linux tabanlı bir mobil işletim sistemi olan Android, son raporlara göre dünyada en çok kullanılan mobil işletim sistemidir. Bu popülarite ve açık kaynak kodlu yapı sonucunda Android, kötücül saldırıların ve saldırganların hedefi haline gelmiştir. Cisco 2014 güvenlik raporuna göre mobil kötücül yazılımların %99'u Android işletim sistemini hedef almaktadır.

Android uygulamaları genellikle resmi uygulama marketi olan Play Store'dan temin edilmektedir. Play Store, çeşitli geliştiriciler tarafından yüklenen uygulamaları yeterli güvenlik taramasına tabi tutmadan yayınlamaktadır. Android, izin tabanlı güvenlik mekanizmasına sahiptir. Bunun yanı sıra kullanıcı bir uygulamayı yükleme girişiminde bulunduğu zaman Android, bu uygulamalar yüklenirken kullanıcılara uygulamanın talep ettiği izinleri sunmakta ve sonrasındaki tüm sorumluluğu kullanıcıya bırakmaktadır. Yapılan çalışmalarda Android kullanıcıların büyük bir çoğunluğunun bu izinlerden habersiz olduğu veya bu izinlerin ne tür etkilerinin olduğunu bilmediği ortaya çıkmaktadır. Bu sebeple, uygulamaların kötücül bir içerik içerip içermediğine dair bir güvenlik taraması yapılmasına ve kullanıcıların bilgilendirilmesine ihtiyaç duyulmaktadır. Bu ihtiyacı karşılamak üzere literatürde çeşitli yaklaşımlar ve yöntemler yer almaktadır.

Bu projede amacım tasarım projemde yaptığım trojan uygulamasının android cihazda hangi açıklardan yüklenebileceğini ve çalışabileceğini vurgulamak ve google play ve antivirus programlarının maalesef yetersiz olması ve asıl güvenliğin aslında bilinçli kullanıcıdan geçtiğini vurgulamak amaçlı kullanıcıya bir bilinç ve güvenlik oluşturmak amaçlı tasarlanmıştır.

2 Kötü Amaçlı Yazılım Yükleme

Kötü amaçlı yazılım yükleme başlığında tasarım projemden bahsetmemin uygun olduğunu düşünüyorum hangi problem üzerine projemi geliştirdiğimi daha net anlatabilmem için. Android uygulamalarının belirli izinler verildiğinde çok fazla kişisel bilgiye erişebileceğini biliyoruz. Uygulamalar, kişileri okuyabilir ve düzenleyebilir, metin ve telefon görüşmeleri gönderip alabilir, telefon numaranızı ve e-posta hesap bilgilerinizi okuyabilir, fiziksel konumunuzu izleyebilir ve çok daha fazlasını yapabilir. Çoğu durumda, bu özellikler kullanıcıya olumlu bir hizmet sunarlar. Bununla birlikte, bu izinler sessizce kişisel bilgileri toplamak ve çok kötü niyetle hareket etmek için kolayca istismar edilebilir. Android Truva Atı projem, Android uygulamalarına yaygın olarak verilen izinlerin bazılarının kötü niyetli bir şekilde nasıl kullanılabileceğini göstermek için tasarlanmıştır. Uygulamamdaki kötü amaçlı kod blokları, herhangi bir Android uygulamasına gömülür ve zararlı olmayan bir uygulamanın arkasında sessizce çalışır. Bu projede trojanın çalışma mantığını göstermek amaçlı basit düzeyde tasarlanmış bir mp3 player uygulaması oluşturuldu ve gerekli yerlerine bu zararlı kod parçaları gömüldü. Kötü amaçlı kod tamamen arka planda çalışır ve kullanıcıya ulaşmaz. Mp3 player yüklenir yüklenmez trojan sahip olduğu izinlerle hedefin telefonunu dinlemeye alır ve tetiklenmeyi bekler. Bu trojan, kullanıcının telefon rehberine ve metin mesajlarına erişebilir; eriştiği her mesajı hedefe gönderir. Uygulama yayılmak için bir sosyal mühendislik içermektedir ve gelen arama ile tetiklenir, arama sonlandıktan belli bir süre sonra o kişiye uygulamanın indirme linkini içeren bir oltalama mesaj gönderilir. Ayrıca uygulama tetiklendikten 5 dk sonra her 15 dakikada bir kullanıcının GPS bilgilerini okur ve hedefe gönderir. Uygulamanın kötü kod parçaları açılışta direk çalışmamaktadır bir tetikleme üzerine çalışmaya başlamaktadır bu sayede de antivirus programları tarafından yakalanmamaktadır. Tasarlarken düşündüğüm başka bir şey ise telefonun ısınma sorunu ve batarya sorunu olmuştur bunları da aynı şekilde optimize ettiğim için kullanıcı tarafından takibi zorlaşmıştır.

3 Antivirüs Uygulamaları Yerince İyi Mi?

Bugün akıllı telefonların %80'inden fazlası Android kullanıyor. Google'ın işletim sistemini kullanan kitlenin bu kadar büyük olması, kötü amaçlı yazılım geliştiricileri için daha uygun bir hedef haline geldi. Soru şu ki, şu anda antivirüsler etkili mi kullanılıyor? Cevap evet ya da hayırdan daha karmaşıktır, ancak cevap hayır a daha yakın durmaktadır.

Bir antivirüs programının görevi, kötü amaçlı yazılımları sistemde algılamak, izole etmek ve ortadan kaldırmaktır. Algılama, herhangi bir antivirüs programı için en önemlisidir ve Android OS'nin kendine özgü belirli faktörleri, görevlerini yerine getirmeye çalışırken aslında antivirüs programlarını karışık hale getirir. Android OS, uygulama paketlerini sandbox ile inceler; bu, çalışan programları ayırmak için bir güvenlik mekanizmasıdır. Uygulamanın diğer uygulamaların dizin içeriğini listelemesine izin vermez. Yani antivirüsün, kurulumdan sonra diğer uygulamaların dizinlerini listelemesine izin vermeyerek, yüklendiğinde şüpheli davranış göstermeyen uygulamalar güvenli olarak algılanır. Sonradan uygulamanın zararlı kod parçaları etkinleştiğinde ise antivirüsün, bu uygulamanın zararlı olduğunu bilmesinin yolu kalmamaktadır. AISEC'in "Android antivirus cannot monitor dynamic behavior of other apps and working directories " raporuna göre, Android'in uygulama ekosisteminin korunduğu bir yapıya sahip olduğu için antivirüs yazılımının bu tür faaliyetleri tamamen yapması imkânsızdır. Dolayısıyla, Bir Android cihaza birden çok saldırı vektörü vardır ve bunların çoğu tipik olarak Google'ın Bouncer zararlı yazılım algılama sistemi kapsamındadır. Bu hizmet, Android işletim sistemindeki karmaşıklığa bağlı olarak Google Play Mağazasına yüklenen her uygulamayı tarar ve kötü

amaçlı yazılım istismarı olan bazı uygulamalar bu hizmete yakalanırken, bazı kötü amaçlı uygulamalar ise yakalanmadan bunu başarabilir.

Google'ın Bouncer hizmeti, Google'ın bulut altyapısında bulunan uygulamaları test ederek Google Play Store'daki bariz malwarelerin çoğunu ortadan kaldırmaya çalışıyor. Google bu hizmeti etkinleştirdiğinde, 2011 yılında birinci ve ikinci çeyrekler arasında uygulama mağazasındaki kötü amaçlı yazılımların oranı %40 oranında azaldı.

Malware yazarları, Bouncer'a yakalanmamak için nispeten basit bir yol bulmuşlardır. 'dropper' tekniğini kullanan görünüşte zararlı bir uygulama, hedef telefona yüklendikten sonra kötü amaçlı yazılımları indirebilir. Bu tekniğin kullanılması, hemen hemen tüm antivirüs uygulamalarını işe yaramaz hale getirir; çünkü sandboxing Android özelliği, antivirüsün diğer uygulamaların içeriğini okuma yeteneğini reddetmesiyle sonuçlanır. Bu nedenle kötü amaçlı yazılımın, Android sisteminize zarar verecek ve kişisel verileri çalmasına izin verecektir.

Başka popüler bir saldırı vektörü de yeniden paketlenmiş uygulamalar olup, yaygın olarak Google'ın Bouncer sistemine takıldığından, üçüncü taraf uygulama mağazalarında bulunur. Bu üçüncü taraf uygulama mağazalarının bazıları, çoğunlukla, doğru/meşru görünmesi için gizlenmiş kötü amaçlı kodlarla oluşturulmuş uygulamaları içerir ve cihazınızı rooting yaparak, onu OS'nin tüm bölümlerine root erişimi sağlamak için tasarlanmış kötü amaçlı yazılımlara maruz bırakabilir. Bunlar, AV uygulamalarının karşılaştığı zorluklardan sadece birkaçı.

4 Özet Proje Tanıtımı

Önerilen yöntemim de, birçok nokta üzerine değinilerek bunların statik analizi üzerine odaklanıyor. Uygulamanın çalışması esnasında tanımladığım 3 çeşit json dosyası üzerinde tarama yapılıyor bu dosyalar sırasıyla whiteList başlığı altında tanımladığım güvenli paket olarak listelediğim, güvenli şirketler tarafından onaylanmış paket adlarıdır, diğer bir dosya blackActivityList başlığı altında listelediğim kötü amaçlı kullanılabilecek reklam paketleri adlarıdır ve son dosyam ise permissions başlığı altında listelediğim birçok etkeni düşünerek oluşturduğum kötü amaçlı kullanılabilecek izin adlarından oluşmaktadır ve uygulama içinde tanımladığım sistem problemleri altında listelediğim cihazı tehlikeye açık halde bırakacak device özelliklerinin açık veya kapalı tespitine dayalı bir yöntem/tarama geliştirilmiştir.

4.1 Uygulama Ve Sistem Problemleri İçin Risk Analizi

Aslında, uygulama yüklenmeden önce, istenen izinler kullanıcıya bir liste halinde gösterilir. Maalesef, çoğu kullanıcı, yalnızca bir izin listesinin okunmasıyla bir uygulamanın kötü amaçlı olup olmadığını anlamak için yeterli uzmanlığa sahip olmayabilir. Dahası, çok sayıda kullanıcı izinleri okumuyor ve yalnızca uygulamayı yüklüyor. Bu durumda, izin sistemi bu tür kullanıcıların kötü niyetli uygulamalardan korunmalarına yardımcı olmuyor. İzin, kodun bir bölümüne veya cihazdaki verilere sınırlı erişimi sınırlama rolüne sahiptir. Kritik veriler ve kod, bu sınırlamanın yanlış amaçla kullanılması durumunda, kullanıcı deneyimini bozmak veya zarar vermek için kötüye kullanılabilir. Bu nedenle, Table 1 de verilen verilere göre yöntemim, kullanıcı için uygulama riskini değerlendirmek için uygulamaların paket içindeki izinlerini taramaya dayanır. Özellikle, bir grup[1] izin için, yanlış[2] kullanılmasının kolay olduğu özel muamele ihtiyacını kabul ederek onlara özel bir risk ağırlığı belirledim.

	Uygulama İzinlerinin Adı	Risk Durumu
1	READ_SMS	Moderate-High
2	SEND_SMS	High
3	RECEIVE_SMS	High
4	CAMERA	Medium
5	READ_CALENDAR	Medium
6	WRITE_CALENDAR	Medium
7	READ_CONTACTS	Medium-High
8	GET_ACCOUNTS	Medium-High
9	ACCESS_FINE_LOCATION	Moderate-High
10	ACCESS_COARSE_LOCATION	Moderate-High
11	BODY_SENSORS	Medium
12	RECORD_AUDIO	Medium-High
13	READ_PHONE_STATE	Moderate-High
14	CALL_PHONE	Medium-High
15	PHONE	Moderate-High
16	PROCESS_OUTGOING_CALLS	Very High
17	READ_EXTERNAL_STORAGE	Moderate-High
18	WRITE_EXTERNAL_STORAGE	Moderate-High
19	WAKE_LOCK	Medium-High
20	ACCESS_NETWORK_STATE	Medium

Table 1: İzin Adları ve Risk Durumu

Öncelikle, bu izinleri görüldüğü gibi projeye dahil etmek için risk durumu derecelendirilmesi yapıldı. Yüksek seviye ve Orta seviye risk diye iki duruma ayırarak derecelendirme yapılmıştır.

Sistem problemleri başlığı altında:

Device özelliği	Risk Durumu
USB Hata Ayıklama	Medium-High
Bilinmeyen Kaynaklar	Moderate-High
Konum Paylaş	Moderate-High
Bluetooth	Moderate-High

Table 2: Device özelliği ve Risk Durumu

Proje içinde izin yapısında Table 1 deki risk durumları göz önüne alınarak json dosyasına tehlike derecesi

[1] https://developer.android.com/guide/topics/permissions/requesting.html
[2] https://www.csc2.ncsu.edu/faculty/xjiang4/pubs/OAKLAND12.pdf
[3] https://www.politesi.polimi.it/bitstream/10589/119402/3/2016_04_Preti.pdf

tanımlamıştır. Eğer yüksek riskli izin için “1”, orta risk için ise “0” sayısı tanımlanmıştır[3]. Uygulama içinde bu durum:

```
for (int i = 0; i  izinDizisi.length(); i++) {
...
int tehlikeli=temp.getInt(”dangerous”);
...
}
public boolean tehlikeli() {
for(UygulamadakiIzinBilgileri uib : _izinler) {
if(uib.tehlikeliMi()==1) //getInt değeri
return true;
}
return false;
}
```

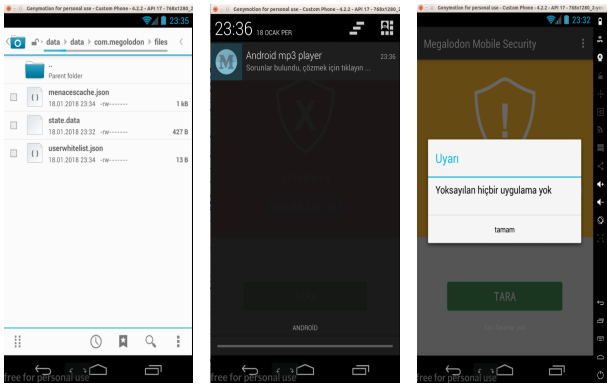
Bu kod parçasından elde edilen sonuç eğer true ise yüksek risk değilse orta risk olarak işlem yapılır.

4.2 Uygulama Paketi İçindeki Java Dosyalarının Tanıtımı

Proje dosyaları oluşturulurken genel hatlar üzerine duruldu. Proje MainActivity dosyasının çalışması ve AntivirusActivity dosyasını çağırması ile başlamaktadır. MainActivity dosyası, hem ilk açılış xml dosyasını ve AntivirusActivity dosyasını çağırmak da görevlidir.

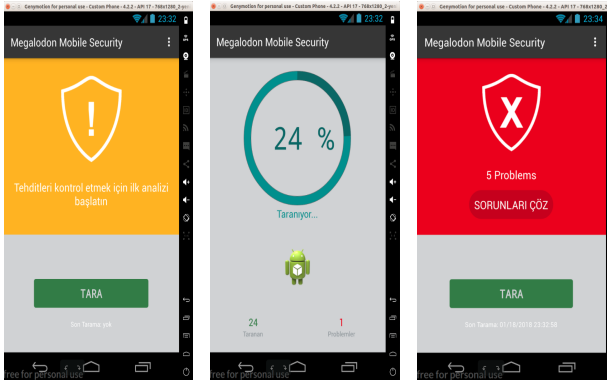
ilk açılış xml sayfası

AntivirusActivity dosyası, projesinin genel hattını oluşturmak üzerine tasarlanmıştır. AntivirusActivity dosyası içinde telefonun genel durumunu izlemek üzerine bir servis[4] çağırmakta, bu servis içinde de eklenen veya kaldırılan paketlerin durumu[5] dinlenme, tehditleri izleme ve raporlama, sonuçlara göre 2 tane json dosyası oluşturulması için gerekli sınıfları çağırma ve ekleme işlemi yapmakdır. Bu json dosyalarından birincisi kullanıcının telefonunun taraması sonucunda tehdit içermeyen userwhitelist json dosyası ikincisi ise tehdit bulunduğu takdirde oluşturulacak menacescache json[6] dosyasıdır, ayrıca içinde önceden tanımladığım json dosyalarına göre tarama yapmaktadır başka bir işlevi ise paket yüklendiğinde cihaza kullanıcıya bu uygulama hakkında bildirim yayınlamakdır. AntivirusActivity diğer hatlarında ise çalışmalarına göre oluşturduğum Fragment[7] sınıflarını yönetmekle ilgilenmektedir çünkü fragment arası data alışverişi direk yapılamıyor bağlı bulundukları Activity üzerinden yapılabiliyor ve bu bağlantıyı sağlamak için interface yapısı kullanılıyor.



AntivirusActivity.java dosyası işlemleri screenshots

İlk çalışacak fragment sınıfım ise MainFragment java dosyasıdır. Bu sınıf tarama işlemleri sırasında uygulamanın düzgün ve hızlı yönetimi için AsyncTask[8] olarak tanımladığım sınıfı çağırmaktadır. Bu AsyncTask dosyası, tararken problem tipine göre ekleme yapmakta, tarama anı ekranı oluşturma burada tarana uygulamalara ait verileri göstermektedir. MainFragment açılışta ve sonrasında tarama[9] işlemlerini yapmak veya sorunları görmek için butonlar , tarama anındaki ilerleyişi görmek için tasarımlar içermekte ve tehdit durumuna göre arka plan rengi oluşturmaktadır. Ayrıca işlem bittikten sonra sonuçların listelenmesi için ResultsFragment dosyasını çağırmaktadır.

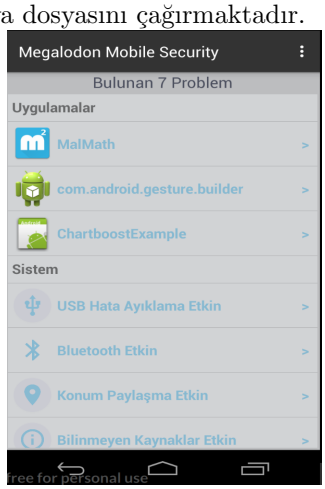


MainFragment.java dosyası işlemleri screenshots

ResultsFragment dosyası, içinde ArrayAdapter sınıfı bağlantısı yapılmıştır . Bu adapter sınıfı, bulunan problemleri sistem ve uygulama problemi olarak gruplayıp uygun başlık altında listeler[10] ve tehditi detaylı görüntüleyebilmek için seçim durumları oluşturur. Bu durumların listelenmesi ve seçimleri için kendi içinde interface ve verileri almak için bir sınıf

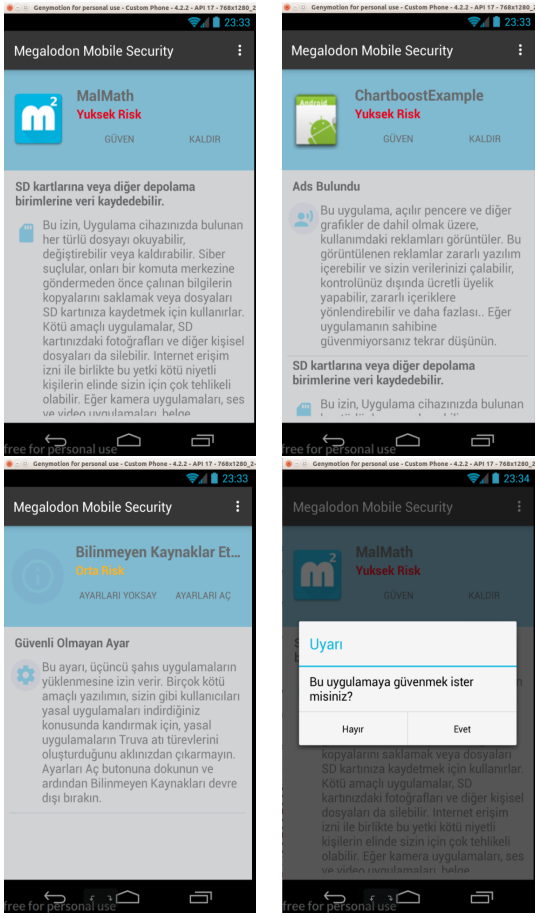
[4]https://stackoverflow.com/questions/20594936/communication-between-activity-and-service
[5]https://github.com/nickaknudson/android-nickaknudson/blob/master/src/com/nickaknudson/android/receivers/PackageReceiver.java
[6]https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-101/javada-dosya-islemleri
[7]https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-201/fragment-olusturmak
[8]https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-301/asynctask
[9]https://github.com/jakob-grabner/Circle-Progress-View/blob/master/ExampleApp/src/main/res/layout/activity$_main.xml$
[10]https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-301/listview-ozellestirme

çağrılmıştır. Ayrıca tespit edilen tehditlerle ilgili detaylı bilgi için oluşturduğum UygulamaBilgisiFragment java dosyasını çağırmaktadır.



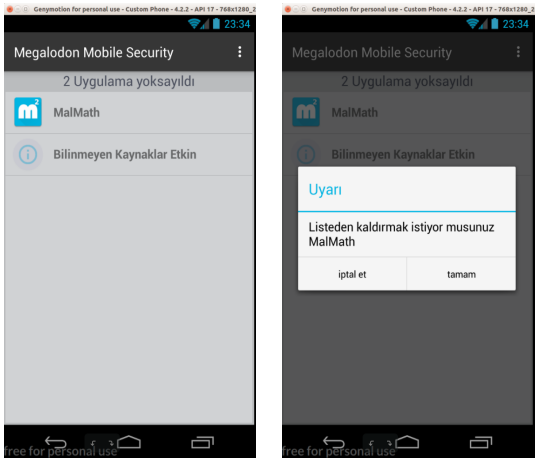
ResultsFragment.java dosyası işlemleri screenshot

UygulamaBilgisiFragment java dosyası, paket ile ilgili tehdit oluşturan izinlerin verilerini ve açıklamalarını içermektedir. ayrıca bu sayfada uygulamayı telefondan kaldırabilir[11] veya tehdit olduğu düşünülmüyorsa güven butonu tercihi yapılarak tehdit listesinden kaldırılabilmesi sağlanmaktadır.



UygulamaBilgisiFragment.java dosyası işlemleri screenshots

YoksayilanlarFragment java dosyası, her zaman erişilebilsin diye ayrı bir menu tuşu olarak tanımlanmıştır ve AntivirusActivity sınıfı üzerinde tanımlanmıştır. YoksayilanlarFragment gelen verileri listelemesi için aynı şekilde ayrı bir adapter sınıfı tanımlanmıştır. YoksayilanlarAdapter, uygulama paket verilerini içerir ayrıca bu listeden tehdit listesine transfer için kaldır butonu içermektedir. YoksayilanlarFragment dosyası diğer fragment dosyalarında olduğu gibi yapılan işlemlere bağlı olarak uygulama paketlerinin seçilen işleme bağlı olarak iki json dosyasında da güncelleme işlemlerini gerçekleştirmektedir.

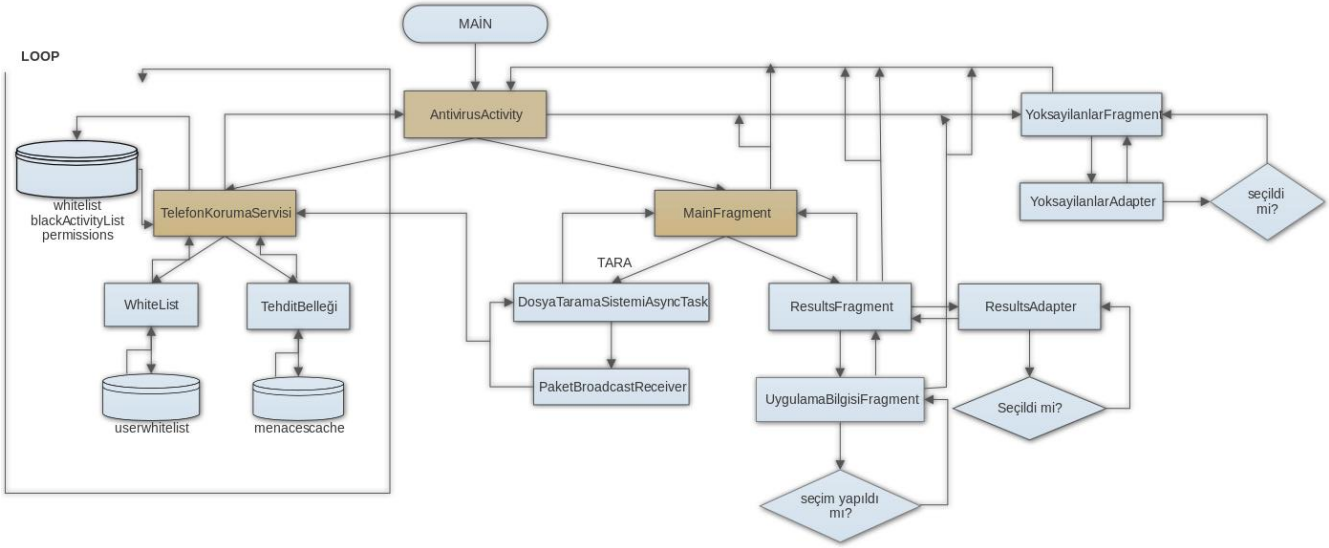


YoksayilanlarFragment.java dosyası işlemleri screenshots

Tarama sonucu kullanıcı işlemlerine göre oluşturulan menacescache ve userwhitelist json[12] dosya yapısı aşağıdaki gibidir:

kullanıcı telefonunda oluşturulan json dosya screenshots

[11] https://gist.github.com/menht/2698877
[12] https://gelecegiyazanlar.turkcell.com.tr/konu/android/egitim/android-101/javada-dosya-islemleri



SEKİL 1

Proje ile ilgili arka planda çalışan yardımcı java dosyaları hariç çalışma yapısı yukarıdaki şekilde gösterilmeye çalışılmıştır. Projenin mimari tasarımının genel yapısını göstermek amacıyla iki adet ağaç yapısı oluşturulmuştur. Birinci yapı uygulamanın ilk başladığı zamanki çalışma[13] gösterimidir. İkinci yapı ise uygulamanın TARA[14] denildikten sonraki çalışma yapısı gösterilmeye çalışılmıştır.

5 Sonuç

Bugün akıllı telefonların %80'inden fazlası Android kullanıyor. Google'ın işletim sistemini kullanan kitlenin bu kadar büyük olması, kötü amaçlı yazılım geliştiricileri için daha uygun bir hedef haline geldi. Bir antivirüs programının görevi, kötü amaçlı yazılımları sistemde algılamak, izole etmek ve ortadan kaldırmaktır fakat android OS, uygulama paketlerini sandbox ile inceler; bu, çalışan programları ayırmak için bir güvenlik mekanizmasıdır. Uygulamanın diğer uygulamaların dizin içeriğini listelemesine izin vermez. Yani antivirüsün, kurulumdan sonra diğer uygulamaların dizinlerini listelemesine izin vermeyerek, yüklendiğinde şüpheli davranış göstermeyen uygulamalar güvenli olarak algılanır. Sonradan uygulamanın zararlı kod parçaları etkinleştiğinde ise antivirüsün, bu uygulamanın zararlı olduğunu bilmesinin yolu kalmamaktadır.

Sonuç olarak benim projemde bu sorunlara dayanarak kullanıcıya sağduyu ve eleştirel düşünme, yapısı sunularak kötü amaçlı yazılımlardan korunması için bir bilinç kazanması hedeflenmiştir. Ayrıca, önemli olan, Google'ın uygulama tarama hizmeti tarafından sunulan korumadan yoksun ve genellikle kötü amaçlı yazılımlar ve casus yazılımlarla dolu üçüncü taraf uygulama mağazalarından kaçınmaktır. Bu araştırmalara dayanarak yine de tespit edilen malwareler için etkili çözüm antivirus programlarını kullanmaktır. Android kullanıcılarının bu tür truva atlarından kendilerini korumalarının tek yolu, yeni uygulamalar yüklerken her zaman uygulama izinlerini okumak ve yalnızca güvenilir kaynaklardan uygulama yüklemektir. Bir şeyler size garip veya şüpheli geliyorsa o uygulamayı kurmaktan kaçınmalısınız.

[13] https://drive.google.com/open?id=1W8-k9YRhEejNgqE7nETTLMSnlhSvEO1p
[14] https://drive.google.com/open?id=1nRrE7lslz4vuHdh1jsIF7USyBNC5pBwi