Journal of the Faculty of Engineering and Architecture of Gazi University 32:4 (2017) 1015-1024

Mühendislik Mimarlık Fakültesi Dergisi
Journal of The Faculty of Engineering and Architecture of Gazi University

Elektronik / Online ISSN : 1304 - 4915
Basılı / Printed ISSN : 1300 - 1884

Android kötücül yazılımlar için izin tabanlı tespit sistemi

Anıl Utku[1]*, İbrahim Alper Doğru[2]

Gazi Üniversitesi Mühendislik Fakültesi Bilgisayar Mühendisliği Bölümü, Ankara, 06570, Türkiye
Gazi Üniversitesi Teknoloji Fakültesi Bilgisayar Mühendisliği Bölümü, Ankara, 06500, Türkiye

ÖNEÇIKANLAR

- Mobil kötücül yazılım tespit sistemi tasarımı
- Naive Bayes ve KNN algoritmaları ile mobil kötücül yazılım tespiti
- Statik analizler ile mobil kötücül yazılım tespiti

Makale Bilgileri

Geliş: 01.03.2016
Kabul: 04.08.2017

DOI:
10.17341/gazimmfd.368788

Anahtar Kelimeler:
Mobil kötücül yazılım, kötücül yazılım tespit teknikleri, mobil güvenlik, izin tabanlı analiz

ÖZET

Günümüzde mobil cihazların kullanım alanlarında ve işlevselliklerinde büyük bir artış yaşanmaktadır. Mobil cihazlar kullanıcıların kişisel bilgilerini sakladıkları platformlar haline gelmişlerdir. Bu gibi özelliklerinden dolayı mobil cihazlar saldırganların hedefi konumuna gelmiştir. Bu çalışmada, Naive Bayes ve KNN algoritmaları kullanılarak izin tabanlı bir kötücül yazılım tespit sistemi geliştirilmiştir. Geliştirilen sistemin test sonuçları her bir algoritma için analiz edilerek karşılaştırılmıştır. Analiz sonuçları Naive Bayes sınıflandırıcısının %97,29 oranında kötücül yazılım tespitinde başarılı olduğunu, Knn sınıflandırıcısının ise %97,74 oranında kötücül yazılım tespitinde başarılı olduğunu göstermiştir.

Permission based detection system for android malware

HIGHLIGHTS

- Mobile malware detection system design
- Detection of mobile malware with Naive Bayes and KNN algorithms
- Mobile malware detection with static analysis

Article Info

Received: 01.03.2016
Accepted: 04.08.2017

DOI:
10.17341/gazimmfd.368788

Keywords:
Mobile malware, malware detection techniques, mobile security, permission based analysis

ABSTRACT

Nowadays uses and functionality of mobile devices show a large increase. Mobile devices have become platforms where users keep personal information. Due to such characteristics, mobile devices have become the target of the attackers. In this study, a permission-based malware detection system have been developed using Naive Bayes and KNN algorithms. The test results of the proposed system were analyzed and compared for each algorithm. Analysis results showed that Naive Bayes classifier was successful in detecting malware with 97.29% And Knn classifier was successful in detecting malware with 97.74%.

*Sorumlu Yazar/Corresponding Author: anilutku@gazi.edu.tr / Tel: +90 312 582 3130

1. GİRİŞ (INTRODUCTION)

Günümüzde gittikçe yaygınlaşan mobil cihazlar, geleneksel cep telefonlarının sağladığı işlevselliklerin yanı sıra destekledikleri multimedya iletişimi ve uygulamalar sayesinde birer avuç içi bilgisayara dönüşmüşlerdir [1]. Kullanıcılar mobil cihazlarını internet, çevrimiçi bankacılık, sosyal ağlar, dosya paylaşımı ve eğlence gibi birçok amaç için kullanabilmektedir. Mobil cihazlar SMS/MMS, Bluetooth ve kablosuz ağ erişimi gibi kişisel bilgisayarların sunduğu birçok bağlantı özelliklerini kullanıcılarına sunmaktadır. Fonksiyonları artan mobil cihazlar üzerinden finansal işlemler, çevrimiçi alışveriş ve hassas veri transferleri gibi işlemlerin yapılabilmesi, mobil cihazları saldırganların hedefi konumuna getirmektedir [2].

Saldırganlar kötücül yazılımlar vasıtasıyla kullanıcı cihazlarından elde ettikleri bilgileri satabilir, kullanıcıların finansal işlemleri ile ilgili şifreleri ele geçirebilir ya da mobil cihaz ve servis sağlayıcı arasındaki güven ilişkisinden yararlanarak kurumsal verilere ve veri kaynaklarına erişebilir. Ayrıca botnet haline getirilmiş cihazlar üzerinden diallerware saldırısı olarak bilinen ücretli aramalar ve SMS gönderimleri yapılabilmektedir [3]. Diğer mobil işletim sistemlerine göre %84,7 pazar payı ile en popüler işletim sistemi olan Android, iOS gibi işletim sistemlerinin aksine kullanıcılarını sadece belirli uygulama marketlerinden yükleme yapmaları konusunda kısıtlamamaktadır. Uygulamalar Android cihazlara Google Play Store, üçüncü parti uygulama marketleri, torrentler ya da direk indirme yoluyla yüklenebilmektedir. Bu kolaylık, kötücül uygulama yazarlarını cezbetmektedir. Popüler uygulamalar saldırganlar tarafından kötücül kodlar eklenip yeniden paketlenerek dağıtılmaktadır [1]. Google'ın pazarlama stratejisinin sonucu olarak resmi uygulama marketi üzerinde yeterli derecede güvenlik önlemlerinin alınmaması ve Android işletim sisteminin popülerliği saldırganların bu platforma odaklanmalarına sebep olmuştur. Android kötücül yazılımların tespit edilmesi için temel olarak statik analiz ve dinamik analiz yöntemleri kullanılmaktadır. Statik analiz, uygulamalar cihaza yüklenmeden önce gerekli analizlerin yapılarak uygulamaların kötücül olup olmadığının belirlenmesi sürecidir. Dinamik analiz ise uygulamalar cihaza yüklendikten sonra uygulamanın davranış analizlerine göre kötücül olup olmadığının belirlenmesidir. Literatürde bu yöntemleri birlikte kullanarak geliştirilmiş hibrid yaklaşımlar ve makine öğrenmesi yöntemlerine dayalı yaklaşımlar mevcuttur [4]. Kabakuş vd. [5] tarafından 2015 yılında yapılan çalışmada, Android uygulamaları kötücül ya da iyicil olarak sınıflayabilmek için statik analiz tabanlı bir sistem geliştirilmiştir. Geliştirilen sistem uygulama bilgilerinin ve analiz sonuçlarının depolandığı imza veritabanı, son kullanıcılar tarafından analiz isteklerinin yapıldığı Android istemcisi ile analiz sürecini yönetme ve iletişim kurmadan sorumlu merkezi bir sunucudan oluşmaktadır. Analiz sonuçları, geliştirilen sistemin %88 doğruluk oranı ile kötücül yazılım tespiti gerçekleştirdiğini göstermektedir. Burguera vd. [6] tarafından 2011 yılında yapılan çalışmada, Android kötücül yazılım tespiti için dinamik analiz yöntemlerine dayalı bir yaklaşım sunulmuştur. Sunulan yaklaşım, test için üretilen kötücül yazılımlar ve gerçek kötücül yazılımlar olmak üzere iki farklı veri seti için kullanıcılardan toplanan örüntülerin depolandığı bir merkezi sunucu bileşenine sahiptir. Önerilen yaklaşım ile kullanıcı cihazlarını kötücül yazılımlardan izole etmek ve kötücül yazılım algılandığında kullanıcıları bilgilendirmek amaçlanmıştır. Shabtai vd. [7] tarafından 2011 yılında yapılan çalışmada, mobil cihazlardan elde edilen çeşitli özellikleri ve olayları sürekli olarak izleyen ve toplanan verileri makine öğrenmesi yöntemleri ile sınıflandıran anomali tabanlı bir tespit sistemi sunulmuştur. Kötücül yazılım tespitinde en iyi performansı elde etmek için anomali tespiti algoritmaları ve özellik seçimi yöntemleri kullanılmıştır.

Wu vd. [8] tarafından 2012 yılında yapılan çalışmada, Android kötücül yazılım tespiti için statik analiz yöntemlerine dayalı özellik tabanlı bir sistem geliştirilmiştir. Geliştirilen sistem Android uygulamalarının davranışlarını karakterize etmek için uygulama izinleri, bileşen dağıtımı, niyet mesajları ve API çağrıları gibi statik bilgileri değerlendirmektedir. Kötücül yazılımların hedeflerini belirlemek amacıyla kümeleme algoritmaları kullanılmıştır. Arp vd. [9] tarafından 2014 yılında yapılan çalışmada, uygulamalardan mümkün oldukça fazla özellik toplayarak geniş statik analizler gerçekleştiren bir sistem geliştirilmiştir. Uygulamalardan elde edilen özellikler ortak bir vektör uzayına gömülerek kötücül yazılımların tipik örüntülerinin belirlenmesi hedeflenmektedir. 123.453 uygulama ve 5560 kötücül yazılım için geliştirilen sistemin %94 başarı oranına sahip olduğu tespit edilmiştir. Bu çalışma kapsamında Android platformu için makine öğrenmesi yöntemlerine dayalı olarak kötücül yazılım tespit sistemi geliştirilmiştir. Çalışmanın 2. ve 3. bölümlerinde mobil kötücül yazılımlar ve kötücül yazılım tespit yöntemleri hakkında açıklamalar yapılmıştır. 4. bölümde kullanılan yöntem ve veri setleri açıklanmıştır. Veri seti olarak Drebin veri setindeki uygulamalar ve Google Play Store üzerinden elde edilen uygulamalar kullanılmıştır. Geliştirilen yöntemde öncelikle .apk uzantılı Android uygulamaları ayrıştırma işlemine tabi tutularak AndroidManifest dosyaları elde edilmiş ve bu dosyalardan uygulama izinleri çıkarılarak izin verileri her bir uygulama için kategorik hale getirilmiştir. Kategorik veriler, Drebin veri setindeki uygulamalar için kötücül, Google Play Store'dan indirilen uygulamalar için iyicil olarak etiketlenmiştir. Geliştirilen sistem Naïve Bayes ve KNN algoritmaları ile test edilerek sonuçları analiz edilmiştir.

2. MOBİL KÖTÜCÜL YAZILIMLAR (MOBILE MALWARES)

Kötücül yazılımlar yetkisiz bir biçimde cihazları uzaktan yönetmek, bilgi sızdırmak ya da kötücül faaliyetler yürütmek üzere tasarlanmış kod parçalarıdır. Kötücül yazılımlar, genel olarak dosya ekleri ya da virüs barındıran Web siteleri yoluyla yayılmaktadır. Kötücül yazılımlar özelliklerine göre

virüs, solucan, trojan, rootkit ve botnet olarak gruplanabilmektedir. Virüsler kendi kendilerini çoğaltan ve çalıştıran kod parçalarıdır. Solucanlar herhangi bir kullanıcı müdahalesi olmadan var olan ağ üzerindeki farklı taşıma mekanizmalarını kullanarak, bir cihazdan diğerine kendisini kopyalayan programlardır. Rootkitler sistemlere bulaşarak kullanıcı işlemlerini ve dosyalarını gizleme, güvenlik duvarı ve antivirüs yazılımlarını devre dışı bırakma gibi faaliyetler yürütmektedir. Botnetler ise bir virüs tarafından etkilenmiş ve saldırganların uzaktan kontrol yetkisine sahip olduğu cihazları ifade etmektedir [10]. Mobil kötücül yazılımlar, virüslü dosya ekleri, Bluetooth yoluyla alınan dosyalar, SMS ve MMS içeriğinde bulunan ve kötücül kod içeren sitelere linkler gibi çeşitli ve farklı yollarla yayılabilmektedir. Mobil cihazları hedef alan bu kötücül yazılımların temel hedefi, cihazda depolanan kullanıcı verilerine ve bankacılık gibi hassas finansal işlemlerde kullanılan kullanıcı bilgilerine erişmektir. Sağladıkları imkân ve kolaylıklar neticesinde günlük hayattan iş hayatına kadar neredeyse her alanda kullanılmaya başlayan mobil cihazlara yönelik riskler de kullanım durumuna göre değişebilmektedir. Mobil cihazlara yönelik tehditler kişisel bilgiler, kurumsal fikri mülkiyetler, gizli bilgiler, finansal varlıklar, cihaz ve hizmetlerin kullanılabilirliği ve işlevselliği ile kişisel ve siyasi itibar alanlarına yönelik olabilmektedir [10].

Mobil kötücül yazılımlar yayılma davranışları, uzaktan kontrol edilebilme davranışları ve saldırı davranışlarına göre kategorize edilebilmektedir. Yayılma davranışı, kötücül yazılımların kurbanları arasında nasıl ve ne gibi yollarla yayıldığını ifade etmektedir. Uzaktan kontrol edilebilme davranışı, kötücül yazılım bulaşmış mobil cihazların uzak sunucu bağlantıları yoluyla kontrol edilebilmesini ifade etmektedir. Saldırı davranışı ise Bluetooth gibi farklı iletişim kanalları yoluyla hedef cihazlara saldırının nasıl gerçekleştirileceğini ifade etmektedir. Kötücül yazılımlar sistemlere bulaştıktan sonra cihazlarda depolanan verilere erişim, cihazların normal fonksiyonlarına müdahale etme veya güvenlik açıkları oluşturma gibi faaliyetler yürütebilmektedir. Kötücül yazılımlar yolu ile kimlik avı (phishing) saldırıları, casus yazılımlar (spyware), gözetim saldırıları, diallerware saldırıları, finansal saldırılar, solucan tabanlı saldırılar ve botnet saldırıları gerçekleştirilebilir. Kimlik avı saldırıları gerçek gibi görünen uygulamalar, SMS mesajları ya da e-postalar yoluyla kullanıcılara ait kredi kartı gibi hesap bilgilerini elde etmeye yönelik gerçekleştirilmektedir. Casus yazılımlar kullanıcı cihazlarının izlenerek kişisel bilgilerin elde edilmesidir. Casus yazılımların sürdürülebilirlik saldırılarından farkı hedef tek bir cihazın olmayışıdır. Sürdürülebilirlik saldırıları kullanıcı cihazlarının sensörler vasıtasıyla izlenmesi ve kullanıcılara ait bilgilerin elde edilmesine yönelik gerçekleştirilmektedir. Diallerware saldırıları, saldırganlar tarafından kullanıcı cihazlarından Premium aramalar yapılması ve Premium SMS'lerin gönderilmesi ile yüksek fatura şoku yaşanmasını ifade etmektedir. Finansal saldırılar kullanıcıların gerçekleştirdiği mali işlemlerde ortadaki adam saldırısı (man in the middle) ile araya girerek kimlik bilgilerinin elde edilmesini amaçlamaktadır. Solucan tabanlı saldırılar bir cihazdan diğerine kullanıcı müdahalesi ve bilgisi olmadan kendi kendini kopyalayan ve cihazlar arasında yayılan solucanlar ile gerçekleştirilmektedir. Botnet saldırıları, kötücül yazılım bulaşmış zombi cihazların saldırgan tarafından uzak bir sunucudan yönetilmesi yoluyla gerçekleştirilmektedir [1].

3. MOBİL KÖTÜCÜL YAZILIM TESPİT YÖNTEMLERİ (MOBILE MALWARE DETECTION TECHNIQUES)

Kötücül yazılım analizi, yazılımın kodunu anlaşılması, davranışlarının ve işlevselliklerinin belirlenerek kötücül olup olmadığının belirlenmesi süreçleridir. Kötücül yazılım tespit yöntemleri statik analiz, dinamik analiz ve izin tabanlı analiz olarak kategorize edilebilmektedir. Statik analizler statik kod analizi, kusur izleme ve kontrol akış bağımlılıkları yolu ile gerçekleştirilmektedir. Dinamik analizlerde ise ağ trafiği, native kod ve kullanıcı etkileşimi gibi parametreler göz önünde bulundurulmaktadır. İzin tabanlı analizler ise AndroidManifest dosyalarında bulunan uygulama içi izinler kullanılarak gerçekleştirilmektedir [11].

3.1. Statik Analiz (Static Analysis)

Statik analizler, uygulamaların yazılım özelliklerinin ve kaynak kodlarının incelenmesi yoluyla gerçekleştirilen analizlerdir. Statik analizler uygulamayı yürütmeden ve uygulamanın kötücül davranışından etkilenmeden kod içindeki kötücül alanları bulmak için kullanılan basit bir yöntemdir. Ayrıştırma, şifre çözme, örüntü eşleştirme ve statik sistem çağrısı analizi gibi birçok teknik statik analiz için kullanılabilmektedir. Ancak yazılım içine gömülü olan şifreleme ve gizleme teknikleri statik analizleri zorlaştırmaktadır. Ayrıca statik analizler geleneksel anti-virüsler tarafından kullanılan kötüye kullanım tespiti ve anomali tespiti olarak kategorize edilebilmektedir.

Kötüye kullanım tespitinde, güvenlik politikaları ve kural setlerine dayalı olarak imza eşleştirmesi yapılmaktadır. Statik analizlerde, veri akış bağımlılıkları ve akış kontrol bağımlılıkları uygulama davranışlarını anlamak için kullanılmaktadır. Anomali tespiti ise bilinen kötücül yazılımları öğrenmek ve bilinmeyen kötücül yazılımları tahmin etmek için makine öğrenmesi algoritmalarını kullanmaktadır. Bu yaklaşım ile uygulamaların zararlı davranışlarının yerine örüntüleri belirlenmektedir. Uygulamaların şüpheli davranışları araştırılarak veri tabanlarına kaydedilmekte ve daha sonra yeni uygulama örüntüleri bu veri tabanları ile karşılaştırılarak anomali tespiti yapılabilmektedir [12].

3.2. Dinamik Analiz (Dynamic Analysis)

Dinamik analizler, uygulama davranışlarının gözlemlenmesi için izole bir ortamda uygulamanın yürütülmesini gerektirmektedir. Statik analizlerin aksine, dinamik analizler kod yürütme işlemlerinin olması nedeniyle kötücül yazılımların doğal davranışlarının açığa çıkarılmasını

sağlamaktadır. Ağ etkinliği izleme, dosya değişikliklerini izleme ve sistem çağrılarının belirlenmesi işlemleri dinamik analizler kullanılarak gerçekleştirilmektedir [13]. Android uygulamaları kişisel bilgisayarlar üzerinde çalıştırılabilen ve mobil cihazların yazılım ve donanım özelliklerinin bir simülasyonu şeklinde tasarlanan emülatörler yardımıyla test edilebilmektedir. Test amaçlı emülatörler Android Virtual Device (AVD) yapılandırmalarını desteklemektedir. Uygulamalar emülatör üzerinde çalıştırıldığında, ağ durumuna erişim, ses ve video gibi uygulamaları yürütme, uygulama mağazasına bağlanma ve veri depolama gibi işlemler gerçekleştirilebilmektedir [11].

3.3. İzin Tabanlı Analiz (Permission Based Analysis)

Android uygulamalarının analizinde, uygulama içi izinler önemli rol oynamaktadır. Uygulamaların yüklenmesi esnasında istenen izinler, AndroidManifest.xml dosyasında listelenmektedir. Android uygulamalarının yüklenmesi sırasında istenen izinlere erişim verilip verilmemesi kullanıcıya bağlıdır. Ancak Android cihazlarda kaynakların kullanımı, bu izin kümesine dayalıdır. Literatürde, AndroidManifest dosyalarında bulunan izinler kullanılarak kötücül yazılım tespiti gerçekleştiren çalışmalar mevcuttur [11].

4. İZİN TABANLI ANALİZ YÖNTEMLERİ İLE MOBİL KÖTÜCÜL YAZILIM TESPİTİ
(DETECTION OF MOBILE MALWARE WITH PERMISSION BASED ANALYSIS METHODS)

Güncel mobil cihazlar, kişisel bilgisayarlar tarafından sunulan hizmet ve uygulamaların büyük çoğunluğunu kullanıcılarına sunmaktadır. Aynı zamanda, mobil cihazları hedef alan güvenlik tehditleri de giderek artmaktadır. Son zamanlarda akıllı telefonlar, tabletler ve PDA gibi mobil cihazların sayısında ve yeteneklerinde yaşanan artış ile birlikte gelişen mobil uygulamalar, saldırganlardan tarafından kullanıcıların kişisel bilgilerini elde etme ve kullanıcıları mali açıdan zarara uğratma amaçları ile kullanılmaktadır. Kullanıcılar mobil cihazlarında banka ve hesap bilgileri, konum bilgileri, rehber, kısa mesajlar ve e-posta gibi kişisel bilgilerini depolamaktadır. Saldırganlar, mobil cihazların sınırlı yeteneklerini ve standart güvenlik mekanizmalarının eksikliklerinden faydalanarak kullanıcıların bu gibi hassas bilgilerine erişme, kullanıcı cihazlarından aramalar yaparak ve SMS'ler göndererek yüksek fatura şoku yaşatma veya cihazları işlevsiz bırakma gibi faaliyetler yürütebilirler. Android uygulamaların resmi marketi olan Google Play Store üzerine üçüncü kişilerin uygulamalarını yükleyebilmesi, bu market üzerinden sunulan uygulamaların kalitesi ve güvenliği üzerine yeterli kontrollerin yapılmadığı sonucunun çıkarılmasına neden olmaktadır. Ayrıca Google Play Store üzerinden indirilen resmi uygulamalara kötücül kodlar eklenmesi ile elde edilen kötücül uygulamalar, üçüncü parti uygulama marketlerinden indirilebilmektedir. Bu gibi kötücül kod enjekte edilmiş uygulamaların orijinal olup olmadığını belirlemek zordur. Bu sebeple uygulamaların güvenilirliği, uygulama mağazası tarafından uygulanan güvenlik önlemlerine bağlıdır. Resmi marketler ve üçüncü parti uygulama marketlerindeki kötücül uygulamaları verimli bir şekilde tespit edebilmek için mobil cihazların yapısını ve geçmişte piyasaya sürülmüş kötücül uygulamaların incelenmesi üzerine birçok çalışma yapılmıştır. Google, geliştirdiği Bouncer servisi ile kötücül olabilecek uygulamaları test etmektedir. Bouncer, Google'ın bulut alt yapısı içinde sanal bir Android ortamı ile Google Play Store'a yüklenen uygulamaları test etmektedir. Bouncer sistemi kullanılmaya başlandığından beri indirilen kötücül yazılım sayısında azalma yaşanmış olsa da bu sistem modern saldırı yaklaşımlarına karşı güvenlik sağlayamamaktadır [14].

Android platformu, 1,9 milyondan fazla mobil cihazda kullanılan bir işletim sistemi olması ve Android uygulamalarının tersine mühendislik teknikleri ile kolayca düzenlenebilmesi ve yeniden paketlenebilmesi nedeniyle saldırganların hedefi konumuna gelmiştir. Bu gibi saldırı durumları nedeniyle Android sistem mimarisi çeşitli güvenlik ve kimlik doğrulama mekanizmaları içerecek şekilde tasarlanmıştır. Android işletim sistemi sağlam güvenlik mimarileri ve katmanlı mimarilere çeşitli düzeylerde güvenlik sağlayan bir yapıya sahiptir. Temel mekanizmalardan biri olan izin çerçevesi, yeni yüklenen uygulamaların çalışması sırasında gerekli olacak erişimler için kullanıcı onaylarının alındığı bir mekanizmadır. Her Android uygulamasının içinde belirli kaynaklara erişimin kontrol edildiği bir izin kümesi bulunmaktadır [15]. Geliştiriciler uygulamaları tasarlarken uygulamanın hangi izinleri kullanacağını belirlemekten sorumludurlar. Kullanıcılar, uygulamaları yüklerken istenen izinleri dikkatli bir şekilde incelemeli ve izinlerin hassas bilgilere erişim hakkı verebileceğini göz ardı etmemelidir. Uygulama içi izinlerden özellikle INTERNET, READ_PHONE_STATE ve WRITE_SETTINGS izinleri belirli kaynaklara erişim için kullanılan popüler izinlerdendir. INTERNET izni, uygulamanın bütün domainlere http(s) istekleri göndererek uygulamanın hedef bağlantılara ve portlara bağlanmasını sağlamaktadır [14].

4.1. Android İşletim Sisteminin Arka Plânı
(Background of the Android Operating System)

Android işletim sistemlerinde kullanılan güvenlik modeli esas olarak uygulama içi izinlere dayanmaktadır. İzinler, uygulamaların API'lere ve kaynaklara erişimi sağlamak ya da kısıtlamak için kullanılmaktadır. Örneğin, INTERNET izni uygulamaların ağ iletişimi gerçekleştirmesi için gereklidir. Diğer bir deyişle, ağ bağlantısı açma durumu INTERNET izni ile sınırlıdır. Ayrıca, uygulamaların kullanıcı girdilerinin yanı sıra telefon rehberini okuyabilmesi için READ_CONTACTS iznine sahip olması gerekmektedir.

Uygulama izinleri geliştiriciler tarafından AndroidManifest.xml dosyası içerisine Şekil 1'de görüldüğü gibi <uses-permission> bölümleri ile eklenmektedir. Android uygulamaları META-INF, lib, res, assets klasörlerinden ve AndroidManifest.xml, classes.dex,

```
<uses-permission android:name="android.permission.RECEIVE_BOOT_COMPLETED" />
<uses-permission android:name="android.permission.ACCESS_DOWNLOAD_MANAGER" />
<uses-permission android:name="android.permission.ACCESS_CACHE_FILESYSTEM" />
<uses-permission android:name="android.permission.SEND_DOWNLOAD_COMPLETED_INTENTS" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.ACCESS_ALL_DOWNLOADS" />
<uses-permission android:name="android.permission.UPDATE_DEVICE_STATS" />
<uses-permission android:name="android.permission.CONNECTIVITY_INTERNAL" />
<uses-permission android:name="android.permission.MODIFY_NETWORK_ACCOUNTING" />
```

Şekil 1. AndroidManifest.xml dosyası içerisinde izinlerin kullanımı (Use of permissions within the AndroidManifest.xml file)

resources.arsc dosyalarından oluşmaktadır. META-INF klasöründe manifest dosyası, uygulamanın sertifikası ile kaynak listesi ve SHA-1 özeti bulunmaktadır. Lib klasörü derlenmiş kodları, res klasörü resources.arsc dosyasının derlenmemiş halini, assets klasörü ise uygulama içinde kullanılan resim gibi dosya varlıklarını içermektedir. classes.dex dosyası, uygulama içinde kullanılan sınıfların Dalvik sanal makinesinin anlayabileceği formattaki halini içermektedir. Resources.arsc dosyası binary XML gibi derlenmiş kaynakları içermektedir. AndroidManifes.xml dosyası uygulamanın adı, sürümü, erişim hakları, başvurulan kütüphane dosyalarını gibi ek bilgileri içermektedir.

Binary XML formatında olan AndroidManifest.xml dosyası AXMLPrinter2, apktool veya Androguard gibi araçlar kullanılarak insanların okuyabileceği formata dönüştürülmektedir. AndroidManifest.xml dosyasında, uygulamaların kullanıcılar tarafından yüklenmesi sırada sorulan izinler bulunmaktadır. Şekil 2'de örnek bir kötücül yazılım uygulaması yüklenirken kullanıcılardan istenen izinler görülmektedir.



Şekil 2. Kötücül bir yazılımın kullanıcılardan istediği izinlere örnek
(Example of the permissions that a malicious software wants from users)

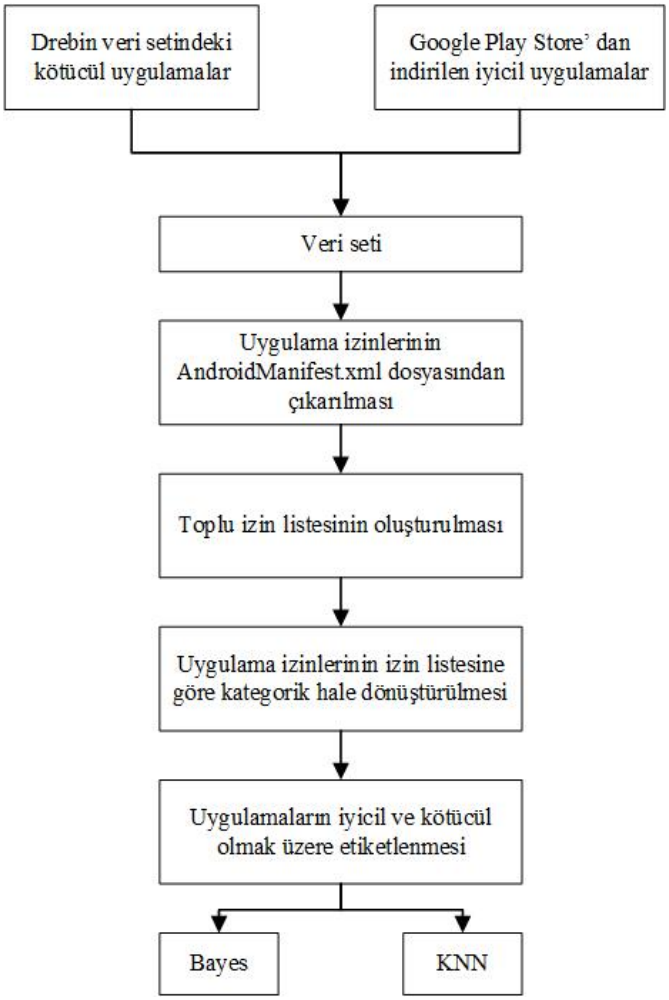
Normalde e-kitap uygulaması olan yazılım, kullanıcıdan mesajlar (SMS veya MMS düzenleme ve okuma), konum (ağ tabanlı konum ve GPS konumu), kişisel bilgiler (tarayıcı geçmişi ve yer imleri), ağ iletişimi (tam internet erişimi), kayıt yeri (SD kart içeriklerini değiştirme veya silme), telefon görüşmeleri ve sistem araçları (telefonun uyku moduna geçmesini önleme) gibi verdiği hizmet ile ilgisi olmayan izinler istemektedir.

Android sistemlerde kullanılan izinler aşağıdaki dört koruma seviyesinden biri ile ilişkili olmaktadır.

- Normal: Uygulamaların API çağrılarına erişmesini sağlayan (duvar kâğıdı ayarlamak için kullanılan SET_WALPAPER izni gibi) kullanıcılara zarar vermeyen izinlerdir.
- Tehlikeli: Uygulamaların cihazda bulunan kullanıcı verileri ya da kontrol gibi potansiyel olarak zararlı olabilecek API çağrılarına erişmesini sağlayan izinlerdir (READ_CONTACTS gibi). Bu izinler uygulamanın yüklenmesi esnasında kullanıcıya gösterilmektedir. Yüklemeye devam edilip edilmemesi kullanıcının kontrolündedir.
- İmza: Bu izinler, eğer istekte bulunan uygulama imzaların tanımlandığı uygulama ile aynı sertifika ile imzalanmışsa verilmektedir.
- İmza veya sistem: Bu izinler İstekte bulunan uygulama imzaların tanımlandığı uygulama ile benzer işlevde veya aynı sertifika ile imzalanmışsa verilmektedir [14].

4.2. Materyal ve Metot (Material and Method)

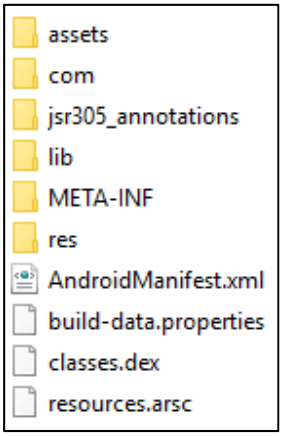
Android işletim sistemi, katmanlı mimarilerine çeşitli düzeylerde güvenlik sağlamaktadır. Android platformunun izin çerçevesi erişim denetimlerini sağlamak için kullanılan önemli özelliklerden biridir. Her uygulamanın ulaşabileceği kaynakların belirtildiği izin kümeleri mevcuttur. Android işletim sistemlerinde kullanılan güvenlik modelinin temel olarak uygulama içi izinlere dayanması ve uygulama içi izin kullanımının hassasiyeti sebebiyle, bu çalışma kapsamında izin tabanlı bir kötücül yazılım tespit sistemi geliştirilmiştir.



Şekil 3. Geliştirilen sistemin akış diyagramı (Flow chart of the developed system)

Geliştirilen sistemin akış diyagramı Şekil 3'te görülmektedir. Geliştirilen sistemde ilk olarak Drebin veri setindeki kötücül uygulamalar ve Google Play Store üzerinden indirilen iyicil uygulamalar ile veri seti oluşturulmuştur. Oluşturulan veri setindeki uygulamaların izinleri AndroidManifest.xml dosyalarından çıkarılarak toplu bir izin listesi olıuşturulmuştur. İzin listesindeki izinlerin, her bir uygulamada bulunma durumları incelenmiş ve izinler kategorik hale dönüştürülmüştür. Her bir uygulama, iyicil ve kötücül olmak üzere etiketlenmiştir. Kategorik haldeki ve etiketlenmiş olan veriler, Naïve Bayes ve K en yakın komşu sınıflandırma algoritmaları kullanılarak sınıflandırılmıştır.

Android uygulamaları .apk formatında olan sıkıştırılmış dosyalardır. Apk dosyaları META-INF, lib, res, assets klasörlerinden ve AndroidManifest.xml, classes.dex, resources.arsc dosyalarından oluşmaktadır. Bu çalışmada uygulama izinlerini elde etmek için apk dosyaları Şekil 4'te görüldüğü gibi öncelikle decompile işlemine tabii tutularak bileşenlerine ayrıştırılmıştır. Her bir uygulama için AndroidManifest.xml dosyaları elde edilmiş ve uygulamalarda kullanılan izinler çıkarılmıştır.



Şekil 4. Apk dosyası (Apk file)

Veri seti olarak, 179 farklı kötücül yazılım ailesinden ve 5560 adet kötücül yazılımdan oluşan Drebin veri setindeki uygulamalardan, çalışan 5555 adet uygulama ile Google Play Store'dan indirilen ve 5 üzerinden en az 4 yıldıza sahip 1339 adet iyicil uygulama kullanılmıştır. Drebin veri setinde bulunan kötücül uygulamaların hassas bilgi aktarımı için

kullanabilecekleri izinler ve izinlerin mevcut olduğu uygulama sayıları Tablo 1'de görülmektedir.

Tablo 1. Hassas bilgi aktarımında kullanılabilecek önemli izinler
(Important permissions to be available in precision information transfer)

Uygulama izni	İzni isteyen uygulama sayısı	İstenen iznin oranı (%)
INTERNET	5340	96,12
READ_SMS	2085	37,53
SEND_SMS	2999	53,98
RECEIVE_SMS	2139	38,50
WRITE_SMS	1244	22,39
READ_CONTACTS	1299	23,38
CALL_PHONE	735	13,23
READ_LOGS	519	9,34
ACCESS_LOCATION	700	12,60

Veri setindeki uygulamaların tamamına yakını internet izni isterken yüksek bir çoğunluğu da ağ durumuna erişme izni ve SMS'ler ile ilgili izinleri istemektedir. Bu sonuçlar, kötücül uygulamaların ağ üzerinden hassas verilerin aktarımı yapma konusunda gerekli izinleri elde ettiklerini göstermektedir. Uygulamalar internet izni elde ettikten sonra uzak sunucu bağlantıları kurabilmekte, kurulan bağlantılar üzerinden dosya ve veri transferi gerçekleştirebilmekte ve bu sayede kullanıcı bilgilerinin sızdırılmasına neden olabilmektedir.

Manifest dosyalarından çıkarılan izinler bütün uygulamalar için toplanarak bir izin listesi oluşturulmuştur. Şekil 5'te uygulamalardan elde edilen izinlerin listesi görülmektedir.

```
WRITE_EXTERNAL_STORAGE
INTERNET
READ_SMS
SEND_SMS
BROADCAST_SMS
ACCESS_CACHE_FILESYSTEM
ACCESS_FINE_LOCATION
ACCESS_COARSE_LOCATION
CHANGE_NETWORK_STATE
GET_TASKS
ACCESS_NETWORK_STATE
SYSTEM_ALERT_WINDOW
ACCESS_DOWNLOAD_MANAGER
MODIFY_NETWORK_ACCOUNTING
```

Şekil 5. Uygulamalardan elde edilen izinlerin listesi
(List of permissions obtained from applications)

Her bir uygulamanın Manifest dosyası, elde edilen izin listesindeki izinler ile karşılaştırılarak kategorik hale getirilmiştir. Listede bulunan izinler, karşılaştırılan Manifest dosyasında mevcut ise 'E', mevcut değilse 'H' harfi ile belirtilmiştir. Şekil 6'da uygulama izinlerinin kategorik hali görülmektedir.

```
E,E,H,H,H,E,E,H,H,H,
E,E,H,H,H,H,H,H,H,H,
E,E,H,H,H,E,E,H,H,H,
E,E,H,H,H,E,E,H,H,E,
E,E,H,H,H,E,E,H,H,H,
E,E,H,H,H,E,E,H,H,H,
E,E,E,E,E,E,E,E,E,E,
E,E,H,H,E,H,H,H,E,E,
H,E,H,H,H,H,H,H,H,H,
E,E,H,H,H,H,H,H,H,H,
```

Şekil 6. Uygulamalardan elde edilen izinlerin kategorik hali
(Categorical form of permissions obtained from applications)

Uygulama izinleri kategorik hale getirildikten sonra alındıkları veri setine göre kötücül ve iyicil olarak etiketlenmişlerdir. Uygulama izinleri kategorik hale getirildikten ve etiketlendikten sonra, makine öğrenmesi yöntemlerinden olan Naïve Bayes ve K en yakın komşu (Knn) sınıflandırma algoritmaları kullanılarak kötücül yazılım tespiti gerçekleştirilmiştir.

Naïve Bayes sınıflandırıcısı Bayes teoremini kullanan istatistiksel bir sınıflandırıcıdır. Her bir örneğin, eğitim verisindeki örneklere göre hangi sınıfta olabileceği olasılıksal olarak hesaplanmaktadır. Naïve Bayes sınıflandırıcısı, örneklerin her bir niteliğinin bağımsız olduğu ve her bir nitelik değerinin, diğer nitelikler hakkında bilgi içermediği temeline dayanmaktadır. Bayes teorisinde sınıf etiketi bilinmeyen bir X örneği, bir H hipotezine göre C sınıfındadır. H hipotezinin doğruluğunu belirlemek için P(H|X) değerinin hesaplanması gerekmektedir [16]. Bayes teorisi, bu çalışma kapsamında kullanılan veri seti için örneklenecek olursa P(H) değeri herhangi bir test verisinin kötücül yazılım olma ihtimalini, P(X) değeri veri setindeki herhangi bir uygulamanın test verisindeki X örneğinin kullandığı izinleri kullanıyor olma olasılığını, P(X|H) ise H hipotezinin doğru olması durumunda P(X) olasılığını ifade etmektedir.

X test verisi için, H hipotezinin sonraki olasılığı P(H|X) değeri Eş. 1 kullanılarak elde edilmektedir.

$$P(H \mid X) = \frac{P(X \mid H)P(H)}{P(X)} \tag{1}$$

Birbirinden bağımsız nitelikler için X test verisinin C_1 sınıfına ait olma olasılığı Eş. 2 kullanılarak elde edilmektedir.

$$P(X \mid C_1) = \prod_{k=1}^{n} P(X_k \mid C_i) = P(X_1 \mid C_i)\ x\ P(X_2 \mid C_i)\ x \ldots x\ P(X_n \mid C_i) \tag{2}$$

Knn algoritması ise depolanmış olan eğitim setindeki veriler ile sınıflandırılacak olan yeni örneği, uzaklık fonksiyonu gibi bir benzerlik ölçütü kullanarak sınıflandıran makine öğrenmesi yöntemidir. Sınıflandırılacak örneğin her bir eğitim verisine olan uzaklığı ölçüldükten sonra k değeri kadar en yakın komşu bulunur ve komşu sınıflarının durumuna göre sayısı yüksek olan sınıf, örneğin sınıfı olarak atanır. Uzaklık fonksiyonu olarak Öklid, Manhattan ya da Minkowski uzaklık fonksiyonları kullanılabilir. Bu çalışma kapsamında Knn algoritmasında benzerlik ölçütü olarak Öklid uzaklık fonksiyonu kullanılmıştır. Öklid uzaklığı Eş. 3 kullanılarak hesaplanmaktadır.

$$\sqrt{\sum_{i=1}^{k}(x_i - y_i)^2} \quad (3)$$

4.3. Tasarım ve Uygulama (Design and Implementation)

Oluşturulan veri setinde, Drebin veri setinden elde edilen 5555 uygulama ve Google Play Store'dan indirilen 1339 uygulama olmak üzere toplamda 6894 adet uygulama bulunmaktadır. Naïve Bayes ve Knn sınıflandırıcıları için oluşturulan veri setinin yaklaşık olarak %80'i eğitim (5555 adet uygulama), %20'si ise test (1339 adet uygulama) için kullanılmıştır.

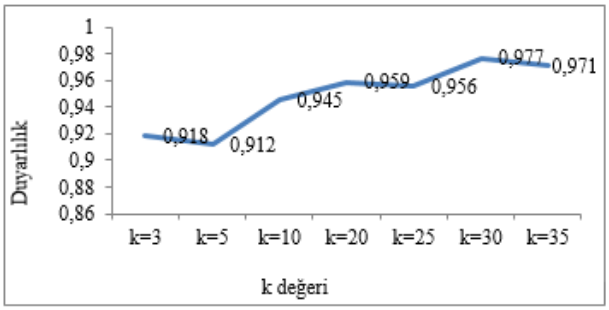
Naïve Bayes sınıflandırıcısı ile geliştirilen sistemin test sonuçları Tablo 2'de bulunan karışıklık matrisinde görülmektedir. Tablo 2'de görüldüğü gibi doğru sınıflandırılan kötücül uygulama sayısı 1080 (TP), doğru sınıflandırılan iyicil uygulama sayısı 178 (TN), yanlış sınıflandırılan kötücül uygulama sayısı 30 (FN), yanlış sınıflandırılan iyicil uygulama sayısı ise 51 (FP) olarak elde edilmiştir.

Sınıflandırıcıların değerlendirilmesinde ise hassasiyet (precision), duyarlılık (recall) ve doğruluk (accuracy) paraetreleri kullanılmaktadır. Hassasiyet parametresi gerçekte kötücül olup sınıflandırıcı tarafından kötücül olarak sınıflandırılan uygulama sayısının, kötücül olarak sınıflandırılan tüm uygulama sayısına olan oranı olarak TP/TP+FP eşitliği ile elde edilmektedir. Naïve Bayes sınıflandırıcısı için hassasiyet değeri: Hassasiyet = TP/TP+FP = 0,954 olarak elde edilmektedir.

Duyarlılık parametresi gerçekte kötücül olup sınıflandırıcı tarafından kötücül olarak sınıflandırılan uygulama sayısının, gerçekte kötücül olan tüm uygulama sayısına olan oranı olarak TP/TP+FN eşitliği ile elde edilmektedir. Naïve Bayes sınıflandırıcısı için duyarlılık değeri: Duyarlılık = TP/TP+FN = 0,972 olarak elde edilmektedir.

Doğruluk parametresi sınıflandırıcının doğru olarak sınıflandırdığı uygulama sayısının, toplam uygulama sayısına olan oranı olarak, (TP+TN)/(TP+FP+FN+TN) eşitliği ile hesaplanmaktadır. Doğruluk = (TP+TN)/ (TP+FP+FN+TN) = 0,939 olarak elde edilmektedir.

Knn sınıflandırıcısı için k değerinin doğru seçilmesi sınıflandırma başarısını doğrudan etkilemektedir. Bu sebeple farklı k değerleri kullanılarak sonuçlar analiz dilmiş ve en uygun k değerinin bulunması hedeflenmiştir. K değeri için 3,5, 10, 20, 25, 30 ve 35 sayıları kullanılmıştır. k değerleri ve k değeri için Knn algoritmasının duyarlılık değerlerinin değişimi Şekil 7'de görülmektedir.



Şekil 7. Knn algoritması için k değerine göre duyarlılık parametresinin değerlerinin değişimi
(Change of values of sensitivity parameter according to k value for Knn algorithm)

Şekil 7'de kötücül olarak sınıflandırılan uygulama sayısının, gerçekte kötücül olan tüm uygulama sayısına olan oranı ile hesaplanan duyarlılık parametresinin k değeri 30 olduğu zaman en yüksek değere sahip olduğu görülmektedir. Bu sebeple k değeri 30 olarak seçilmiştir. Knn sınıflandırıcısı ile geliştirilen sistemin test sonuçları Tablo 3'te bulunan karışıklık matrisinde görülmektedir.

Tablo 3'te görüldüğü gibi doğru sınıflandırılan kötücül uygulama sayısı 1085 (TP), doğru sınıflandırılan iyicil uygulama sayısı 112 (TN), yanlış sınıflandırılan kötücül uygulama sayısı 25 (FN), yanlış sınıflandırılan iyicil uygulama sayısı ise 117 (FP) olarak elde edilmiştir. Knn sınıflandırıcısı için:

Tablo 2. Naïve Bayes sınıflandırıcısı için karışıklık matrisi (Confusion matrix for Naïve Bayes classifier)

		Tahmin Edilen C^+	C^-	Toplam
	C^+	TP (1080)	FN (30)	N^+ Gerçek Kötücül Yazılım Sayısı
Gerçek	C^-	FP (51)	TN (178)	N^- Gerçek İyicil Yazılım Sayısı
Toplam		P^+ Kötücül Tahmin Edilen Yazılım Sayısı	P- İyicil Tahmin Edilen Yazılım Sayısı	N Toplam Örnek Sayısı

Tablo 3. Knn sınıflandırıcısı için karışıklık matrisi (Confusion matrix for Knn classifier)

		Tahmin Edilen C^+	C^-	Toplam
	C^+	TP (1085)	FN (25)	N^+ Gerçek Kötücül Yazılım Sayısı
Gerçek	C^-	FP (117)	TN (112)	N^- Gerçek İyicil Yazılım Sayısı
Toplam		P^+ Kötücül Tahmin Edilen Yazılım Sayısı	P^- İyicil Tahmin Edilen Yazılım Sayısı	N Toplam Örnek Sayısı

Hassasiyet = TP/TP+FP= 0,902
Duyarlılık = TP/TP+FN= 0,977
Doğruluk = TP+TN/TP+FP+FN+TN= 0,8939 olarak elde edilmektedir.

Analiz sonuçları, Tablo 4'te görüldüğü gibi Knn sınıflandırıcısının kötücül uygulamaları Naïve Bayes sınıflandırıcısına göre daha yüksek bir oranda tespit ettiğini ancak Naïve Bayes sınıflandırıcısının ise iyicil uygulamaları daha yüksek bir oranda tespit ettiğini göstermektedir.

Tablo 4. Naïve Bayes ve Knn sınıflandırıcısı için sınıflandırma sonuçları
(Classification results for Naïve Bayes and Knn classifier)

Sınıflandırıcı	TP	FP	FN	TN
Naïve Bayes	1080	51	30	178
Knn	1085	117	25	112

Tespit edilen kötücül uygulama sayısının, tüm kötücül uygulama sayısına olan oranı ile edilen duyarlılık parametresinin Knn sınıflandırıcısı için daha yüksek bir değere sahip olduğu Tablo 5'te görülmektedir.

Tablo 5. Naïve Bayes ve Knn sınıflandırıcısı için değerlendirme sonuçları
(Evaluation results for Naïve Bayes and Knn classifier)

Sınıflandırıcı	Hassasiyet	Duyarlılık	Doğruluk
Naïve Bayes	0,954	0,972	0,939
Knn	0,902	0,977	0,8939

Analiz sonuçları Naïve Bayes sınıflandırıcısının 1100 adet kötücül uygulamanın 1080 adedini doğru bir şekilde tespit ederek %97,29 oranında kötücül yazılım tespitinde başarılı olduğunu, Knn sınıflandırıcısının ise 1100 kötücül uygulamadan 1085'ini doğru bir şekilde tespit ederek %97,74 oranında kötücül yazılım tespitinde başarılı olduğunu göstermiştir.

5. SONUÇLAR VE TARTIŞMALAR (RESULTS AND DISCUSSIONS)

Birden çok bağlantı ve sensör gibi özellikler ile donatılmış mobil cihazların sayısındaki artış ile birlikte mobil kötücül yazılım sayısında da artış yaşanmıştır. Bilgisayar ortamından farklı olarak kötücül kodların bulaşmasını ve yayılmasını önlemek için kullanılacak çözümlerde birçok faktör göz önünde bulundurulmaktadır. Mevcut sınırlı kaynaklar, güç ve işlem üniteleri gibi çok sayıda özellik, saldırganlar tarafından istismar edilebilmektedir. Bu çalışma kapsamında günümüzdeki mobil kötücül yazılım senaryoları tartışılmış, mobil cihazlara yönelik uygulama seviyesindeki bilinen kötücül yazılımlar kategorize edilerek izin tabanlı bir kötücül yazılım tespit sistemi geliştirilmiştir. Bu çalışma kapsamında geliştirilen sistem ile test verisinde bulunan kötücül yazılımların %97'den fazlası doğru bir şekilde tespit edilmiştir. Bu çalışma kapsamında, Android işletim sistemini kullanan mobil cihazlar için izin tabanlı bir kötücül yazılım tespit sistemi geliştirilmiştir. Literatürde, mobil kötücül yazılım üzerine odaklanmış çalışmalar içinde, izin tabanlı tespit sistemlerinde makine öğrenmesi yöntemlerinin kullanılmasına yönelik ilk olma niteliği gösteren bu çalışma ile yaklaşık olarak %97 başarı oranı ile kötücül yazılım tespiti gerçekleştirilmiştir. Gelecek çalışmalarda veri seti genişletilerek ve daha farklı makine öğrenmesi yöntemleri kullanılarak sistem başarısının iyileştirilmesi hedeflenmektedir.

KAYNAKLAR (REFERENCES)

1. He D., Chan S., Guizani M., Mobile application security: malware threats and defenses, Wireless Communications, 22 (1), 138-144, 2015.
2. Tekerek A., Gemci C., Bay Ö.F., Design and implementation of a web-based intrusion prevention system: a new hybrid model, Journal of the Faculty of Engineering and Architecture of Gazi University, 31 (3), 645-653, 2016.
3. Wang X., Yang Y., Zeng Y., Tang C., Shi J., Xu K., A Novel Hybrid Mobile Malware Detection System Integrating Anomaly Detection With Misuse Detection, In Proceedings of the 6th International Workshop on Mobile Cloud Computing and Services, 15-22, 2015.
4. Kabakuş A.T., Doğru İ.A., Çetin A., Android kötücül yazılım tespit ve koruma sistemleri, Erciyes Üniversitesi Fen Bilimleri Enstitüsü Dergisi, 31 (1), 9-16, 2015.
5. Talha K.A., Alper D.I., Aydin C., APK Auditor: Permission-based Android malware detection system, Digital Investigation, 13, 1-14, 2015.
6. Burguera I., Zurutuza U., Nadjm-Tehranı S., Crowdroid: behavior-based malware detection system for android, Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices, 15-26, 2011.
7. Shabtai A., Kanonov U., Elovici Y., Glezer C., Weiss Y., Andromaly: A behavioral malware detection

framework for android devices, Journal of Intelligent Information Systems, 38 (1), 161-190, 2012.

8. Wu D.J., Mao C.H., Wei T.E., Lee H.M., Wu K.P., Droidmat: Android malware detection through manifest and api calls tracing, Information Security (Asia JCIS), 2012 Seventh Asia Joint Conference**,** 62-69, 2012.

9. Arp D., Spreitzenbarth M., Hübner M., Gascon H., Rieck K., Drebin: Effective and explainable detection of android malware in your pocket, Proceedings of the Annual Symposium on Network and Distributed System Security (NDSS)*,* 2014.

10. La Polla M., Martinelli F., Sgandurra D., A survey on security for mobile devices, Communications Surveys & Tutorials IEEE, 15 (1), 446-471, 2013.

11. Dua L., Bansal D., Taxonomy: Mobile Malware Threats and Detection Techniques, Computer Science & Information Technology (CS & IT), 213-221, 2014.

12. Wang X., Yang Y., Zeng,Y., Accurate mobile malware detection and classification in the cloud, SpringerPlus*,* 2015.

13. Shen Y.C., Chien R., Hung S.H., Toward Efficient Dynamic Analysis and Testing for Android Malware, IT CoNvergence PRActice (INPRA)*,* 2 (3), 14-23, 2014.

14. Tchakount´e F., Permission-based Malware Detection Mechanisms on Android: Analysis and Perspectives, Journal of Computer Science and Sofware Application, 1 (2), 63-77, 2014.

15. Aung, Z., Zaw W., Permission-Based Android Malware Detection, International Journal Of Scientific & Technology Research, 2 (3), 228-234, 2013.

16. Ren J., Lee S.D., Chen X., Kao B., Cheng R., Cheung D., Naive bayes classification of uncertain data, Ninth IEEE International Conference on Data Mining, 944-949, 2009.

17. Dragomir E.G., Air quality index prediction using K-nearest neighbor technique, Bulletin of PG University of Ploiesti, Series Mathematics, Informatics, Physics, LXII, 1, 2010.