T.C.
İSTANBUL ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

DOKTORA TEZİ

MOBİL UYGULAMALARIN GÜVENLİK RİSKİNİ ANALİZ EDEN YAZILIM TABANLI SERVİS TASARIMI

Asım Sinan YÜKSEL

Bilgisayar Mühendisliği Anabilim Dalı

Bilgisayar Mühendisliği Programı

Danışman

Prof. Dr. Ahmet SERTBAŞ

Haziran, 2015

İSTANBUL

Bu çalışma, 10/06/2015 tarihinde aşağıdaki jüri tarafından Bilgisayar Mühendisliği Anabilim Dalı Bilgisayar Mühendisliği programında Doktora Tezi olarak kabul edilmiştir.

Tez Jürisi:

Prof. Dr. Ahmet SERTBAŞ (Danışman)
İstanbul Üniversitesi
Mühendislik Fakültesi

Prof. Dr. Abdul Halim ZAİM
İstanbul Ticaret Üniversitesi
Mühendislik ve Tasarım Fakültesi

Prof. Dr. Selim AKYOKUŞ
Doğuş Üniversitesi
Mühendislik Fakültesi

Doç. Dr. Hacı Ali MANTAR
Gebze Teknik Üniversitesi
Mühendislik Fakültesi

Yrd. Doç. Dr. Muhammed Ali AYDIN
İstanbul Üniversitesi
Mühendislik Fakültesi

ÖNSÖZ

Her şeyden önce; benim bu günlere kadar gelmemi sağlayan, bana maddi ve manevi her konuda destek olup yol gösteren canım annem Günser YÜKSEL'e, canım babam Bekir YÜKSEL'e, çalışmalarımda fikir ve görüşlerinden yararlandığım abim Mehmet Erkan YÜKSEL'e sonsuz minnet ve sevgilerimi sunuyorum. Onlarla, kendimi dünyanın en şanslı insanı sayıyorum.

Tez çalışmalarımın yürütülmesi sürecinde; bana destek olan danışman hocam Sayın Prof. Dr. Ahmet SERTBAŞ'a, her türlü ilgi ve desteği gösteren, moral veren, değerli fikirleri, yorumları, önerileri ve katkıları ile çalışmalarımı zenginleştiren tez izleme jüri üyesi hocalarım; Sayın Prof. Dr. Abdul Halim ZAİM'e ve Sayın Yrd. Doç. Dr. Muhammed Ali AYDIN'a en içten teşekkürlerimi arz ediyorum.

Haziran, 2015 **Asım Sinan YÜKSEL**

İÇİNDEKİLER

Sayfa No

ŞEKİL LİSTESİ

Sayfa No

TABLO LİSTESİ

Sayfa No

ALGORİTMA LİSTESİ

Sayfa No

SİMGE VE KISALTMA LİSTESİ

Simgeler **Açıklama**

D	: Riskin düşük olma olasılığı
F	: Fonksiyon
$İ_n$	: n nolu izin
k	: TR, İSR, İR, NR kategorilerinden birini temsil eder.
$max(o)$	: Bir iznin kullanılabileceği maksimum miktar
N	: Riskin normal olma olasılığı
o_i	: İznin tekrarlanma sıklığı
O	: Riskin orta olma olasılığı
P_i	: 0-100 aralığında ölçeklenmiş risk olasılık değeri
R	: Risk
T	: Riskin tehlikeli olma olasığı

Kısaltmalar **Açıklama**

API	: Application Programming Interface
APK	: Android Application Package
App	: Application
CoG	: Center of Gravity
CPU	: Central Processing Unit
CSS	: Cascading Style Sheet
DED	: Dalvik Executable Decompiler
DEX	: Dalvik Executable
EC2	: Elastic Computing Cloud
FIS	: Fuzzy Inference System
GPS	: Global Position System
GSM	: Global System for Mobile Communication
HTML	: Hypertext Markup Language
ISO	: International Organization for Standardardization
iOS	: iPhone Operating System
İR	: İmza Riski kategorisi
İRP	: İmza Riski kategorisindeki izinlerin toplam puanı
$İ_n$RŞ	: $İ_n$ izninin risk şiddeti
İSR	: İmza yada Sistem Riski kategorisi
İSRP	: İmza yada Sistem Riski kategorisindeki izinlerin toplam puanı
JSON	: Javascript Object Notation
MMS	: Multimedia Message
NFC	: Near Field Communication
NR	: Normal Risk kategorisi
NRP	: Normal Risk kategorisindeki izinlerin toplam puanı
OHA	: Open Handset Alliance

OpenGL/ES : Open Graphics Library for Embedded Systems
$RED_k(i_n)$: $İ_n$ izninin k kategorisindeki risk etki değeri
$ROD_{RT}(i_n)$: $İ_n$ izninin RT isimli risk türünün risk olasılık değeri
RT : Tehlikeli, Orta, Normal, Düşük risk türlerinden birini temsil eder.
SaaS : Security as a Service
SELinux : Security Enhanced Linux
SGL : Skia Graphics Library
SSL : Open Secure Socket Layer
SVM : Support Vector Machine
TCG : Trusted Computing Group
TPM : Trusted Platform Module
TR : Tehlikeli Risk kategorisi
TRP : Tehlikeli Risk kategorisindeki izinlerin toplam puanı
XML : Extensible Markup Language
Wi-Fi : Wireless Fidelity
WLAN : Wireless Local Area Network

ÖZET

DOKTORA TEZİ

MOBİL UYGULAMALARIN GÜVENLİK RİSKİNİ ANALİZ EDEN YAZILIM TABANLI SERVİS TASARIMI

Asım Sinan YÜKSEL

İstanbul Üniversitesi

Fen Bilimleri Enstitüsü

Bilgisayar Mühendisliği Anabilim Dalı

Danışman : Prof. Dr. Ahmet SERTBAŞ

Akıllı telefonların popülerleşmesi ve kullanımlarının yaygınlaşması ile birlikte, akıllı telefon kullanıcıları tüm işlemlerini bu cihazlarla yapar hale gelmişlerdir. Kullanıcılar, telefonlarına yükledikleri interaktif uygulamalar aracılığı ile alışveriş yapma, borç ödeme, bilgi paylaşma, çevrimiçi bankacılık gibi her türlü işlevi yerine getirebilmektedirler. Yüklenen uygulamalar; kullanıcıya ait telefon rehberi, konum bilgisi, kişisel fotoğraflar, videolar ve notlar, takvim, telefon numarası, e-posta, SMS mesajları, pil durumuna ait istatistikler gibi hassas bilgilere de erişebilmektedir. Ancak; uygulamaların güvenilir olup olmadıkları, zararlı davranış gösterip göstermedikleri, kullanıcının hangi bilgilerine erişilip erişilmediği kontrol edilmemektedir. Dolayısıyla, uygulamalar, kullanıcı isteği ile hiçbir sınırlama ve uyarı olmaksızın akıllı telefonlara yüklenebilmektedir. Bunun sonucunda, kullanıcılar kötü amaçlı uygulamaların hedefi haline gelmekte, kişisel güvenlik ve gizlilik tehlike altına girmektedir.

Akıllı telefonlar üzerinde çalışan çok sayıda işletim sistemi vardır. Bunlar arasında yer alan Android işletim sistemi, her geçen gün popülerliğini ve akıllı telefon pazarındaki payını artırmaktadır. Android platformunun açık kaynak kodlu olması, uygulamaların kolay bir şekilde geliştirilebilmesi ve Google Play Uygulama Mağazası'na yüklenebilmesi bu platformu cazibe merkezi haline getirmiştir. İsteyen herkes, Android tabanlı mobil uygulamalar geliştirip uygulama mağazasına ekleyerek son kullanıcının hizmetine sunabilmektedir. Android tabanlı akıllı telefon kullanıcılarının ve uygulama geliştiricilerinin sayısının artmasına paralel olarak, bu platformdaki güvenlik riskleri ve

tehditleri de diğer platformlara oranla daha fazla artmıştır ve artmaya da devam etmektedir.

Çalışmamız, literatürde yapılan çalışmaların eksiklerini gidermek, Android platformundaki güvenlik ve gizlilik problemlerini kullanıcı odaklı olacak şekilde çözmek için; herhangi bir uygulamanın kullanıcı açısından ne kadar risk taşıdığını inceleyen, akıllı telefonda erişilen özellikleri tespit eden, hangi tür kişisel bilgilere erişildiğini ve kişisel gizliliğinin ne şekilde riske atıldığını ortaya koyan yazılım odaklı bir servis tasarımını ve prototipinin geliştirilmesini amaçlamaktadır.

Çalışmamızda gerçekleştirdiğimiz sistem sayesinde akıllı telefonlara yüklenmek istenen mobil uygulamalar otomatik olarak analiz edilmekte, uygulamaların kullanılması halinde ortaya çıkacak güvenlik riskleri tespit edilmekte, uygulamalar için risk puanlaması yapılmakta ve risk raporları oluşturulmakta, İnternet ortamında da bu raporlar kullanıcıların bilgisine sunulmaktadır. Sisteme herhangi bir web tarayıcısı ile erişen kullanıcılar, İnternet üzerinden tüm uygulamaların risk puanlarını ve hangi risklere neden olduklarını içeren kritik bilgileri görebilecek, bu bilgileri referans alarak uygulamaları yükleyip yüklememe noktasında karar verebileceklerdir. Sonuç olarak, çalışmamızda; kullanıcıların, uygulamaları telefonlarına yüklemeden önce bilgi sahibi olmalarına imkan veren, güvenlik ve gizliliklerine karşı oluşacak tehditlerle ilgili detaylı ve anlaşılır bilgiler sunan, bulanık mantık tabanlı risk puanlama tekniği kullanan, risk analizi yaparak karar-destek mekanizması şeklinde çalışan bir servis tasarladık ve gerçekleştirdik.

Haziran 2015, 109 sayfa.

Anahtar kelimeler: Mobil güvenlik, mobil risk analizi, güvenlik risk analizi, bulanık sistemler, karar-destek sistemleri.

SUMMARY

Ph.D. THESIS

DESIGNING A SOFTWARE BASED SERVICE TO ANALYZE THE SECURITY RISKS OF MOBILE APPLICATIONS

Asım Sinan YÜKSEL

İstanbul University

Institute of Graduate Studies in Science and Engineering

Department of Computer Engineering

Supervisor : Prof. Dr. Ahmet SERTBAŞ

The increasing popularity of the smart devices have led users to complete all of their daily work with these devices. Users are able to perform any kind of task such as shopping online, sharing information, online banking with the interactive applications that they install on their smart devices. Installed applications gain access to various sensitive information such as the user's contact list, location information, personal photos, personal videos, calendar, phone number, SMS messages, battery status. However, there is no control mechanism in place that checks whether these applications are safe to install, show malicious behavior or which user information they access to. Therefore, applications are installed according to the users' decisions, without any limitations or warnings. As a result, users become the target of malicious applications, and the personal security and privacy are compromised.

There is a large number of operating systems that are used for mobile devices. Through these operating systems, Android continuously increases its popularity and market share. The open-source nature of the Android platform, the ease of application development and the submission of applications to Google Play Store have made this platform more attractive. Anyone who wants to develop Android-based mobile applications is able to submit his/her application that he/she developed to Google's application store for the end users without any problems. However, security risks and threats have increased and continue to increase more than other mobile platforms in parallel to the increase of Android based mobile phone users and developers.

In order to remedy the shortcomings of current studies in the literature and resolve the security and privacy problems in Android platform with a user-focused approach, our study aims to design and develop a prototype of a software based service that analyzes how risky an application is from the user's perspective, identifies the smart device features that are used, reveals what kind of personal information is accessed and how the personal privacy is compromised.

In our study, mobile applications that are desired to be installed on Android based smart devices are automatically analyzed, security and privacy risks that can be caused by installing these applications are identified, the risk scores are calculated, the risk reports are produced and presented online for the attention of users. Any user who accesses the system via a web browser can see the risk scores of the applications, the risks that can be caused by these applications and can make decisions about whether to install the applications or not by taking the risk information into account. As a result, we have designed and implemented a service that allows users to have knowledge about the applications before they install them to their smart devices, presents the users with detailed and easy to understand reports related to the threats against their privacy and security, does risk analysis and fuzzy-logic based risk scoring and will serve as a decision support mechanism.

June 2015, 109 pages.

Keywords: Mobile security, mobile risk analysis, security risk analysis, fuzzy systems, decision support systems.

1. GİRİŞ

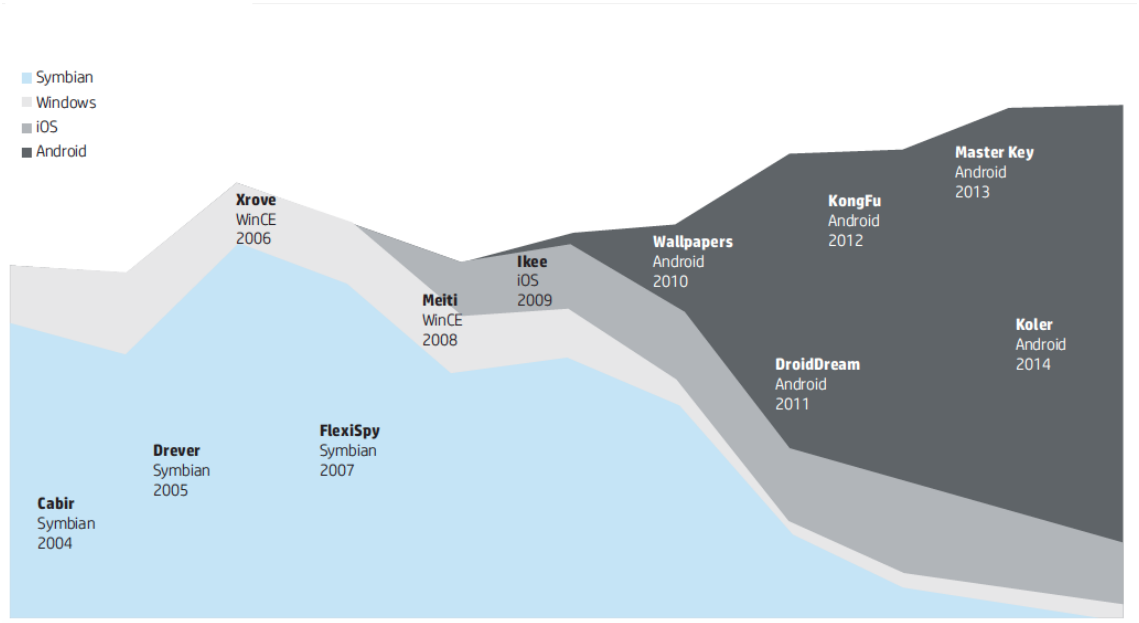
Akıllı telefonların ortaya çıkması ve yaygınlaşması ile birlikte, kullanıcılar bilerek ya da bilmeyerek telefonlarında daha fazla özel bilgiler taşımaya başlamışlardır. Bunlar arasında; konum bilgisi, mesajlar, videolar, fotoğraflar, takvimler, e-posta, kişisel notlar, etkinlikler, hatırlatıcılar ve hatta banka bilgileri gibi kişilere ait önemli bilgiler yer almaktadır. Mobil cihazlara yapılan saldırılar genellikle telefona mesaj yollayarak ya da yüksek ücretli telefon numaralarını arayarak saldırgana hızlı bir şekilde para kazandırma eğilimindedirler. Bu saldırılar, artık cihazlardaki özel bilgilerin çalınmasını da hedeflemektedirler [1-3].

Google firmasının yayınladığı 2014 yılı Mayıs ayı verilerine göre, 900 milyon adet Android cihaz kullanıma açılmıştır [4]. Temmuz 2014 itibari ile de, Google Uygulama Mağazası'ndan 25 milyondan fazla uygulama indirilmiştir. Bu bilgiler göz önüne alındığında, Android işletim sisteminin pazar payının büyük bir kısmını elinde bulundurduğu ve bu payı sürekli artırdığı görülmektedir. Sonuçta, Android işletim sistemi; akıllı cihazlardaki özel bilgileri elde etmek isteyen suçluların ilgisini çekmeye başlamış, saldırıların ana hedefi haline gelmiş, bunlara bağlı olarak Android işletim sistemine yönelik güvenlik riskleri ve tehditleri de diğer platformlara oranla daha fazla artmıştır.

Kötücül yazılımlar her platform için ortak bir problemdir. 2010 yılında ilk kötücül Android uygulamasının ortaya çıkmasından bu yana, kötücül yazılımların sayısı sürekli artmıştır [5]. Çeşitli kötücül Android uygulamalarının detaylarına bakıldığında, bilgi çalma yönünde bir eğilim olduğu görülmektedir. Buradan hareketle, kötücül yazılımların çok büyük bir bölümünün kullanıcıların özel bilgilerini çalmayı hedeflediği ortaya çıkmaktadır [3, 6-8].

Mobil cihazları hedef alan ilk kötücül yazılım olan "Cabir" [9], kendini Symbian işletim sistemine sahip cihazlara bulaştırmış ve bluetooth bağlantısı aracılığıyla diğer cihazlara yayılmayı amaçlamıştır. Öte yandan, "DroidKungFu" [10] ve "GinMaster" [11] gibi yeni nesil mobil kötücül yazılımlar cihazlardaki özel bilgileri çalmayı hedeflemektedir.

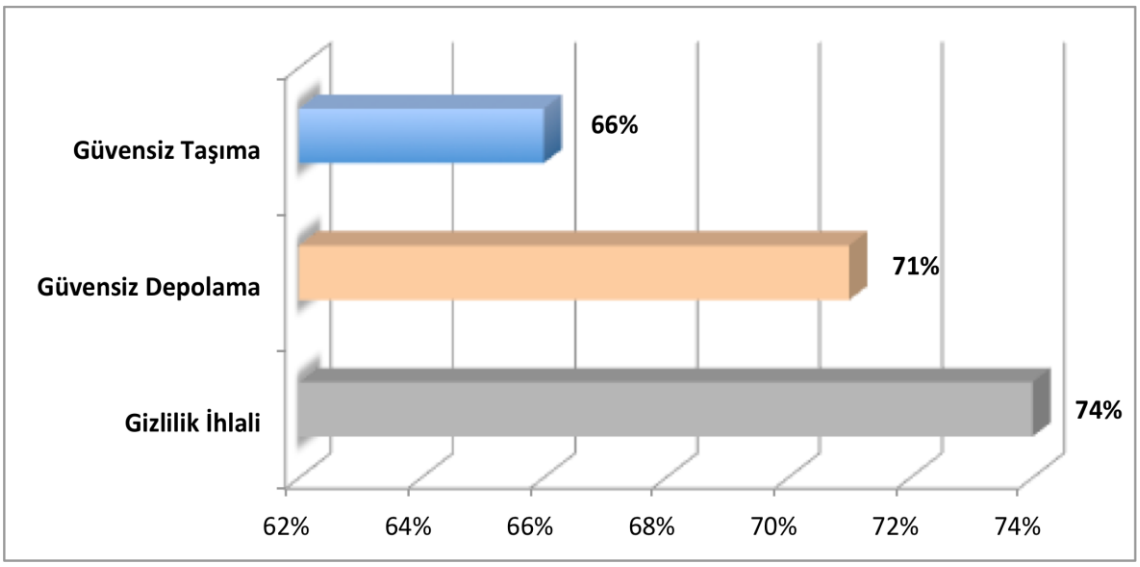
Diğer kötücül yazılımlar ise mobil cihazları birer bota dönüştürmeyi, kötücül yazılım geliştirenlere para kazandırmayı amaçlamaktadırlar. Android platformunu hedef alan bu tür yazılımlar, birer kavram olmanın ötesinde yıkıcı bir aşamaya geçmişlerdir. HP Güvenlik Araştırma Merkezi tarafından yayınlanan "HP 2015 Siber Risk Raporu[1]" incelendiğinde, Android tabanlı kötücül uygulamaların sayısının diğer platformlara oranla daha fazla arttığı gözlemlenmektedir. Şekil 1.1, bu raporda yer alan kötücül yazılımların son 10 yılda geldiği noktayı göstermektedir.



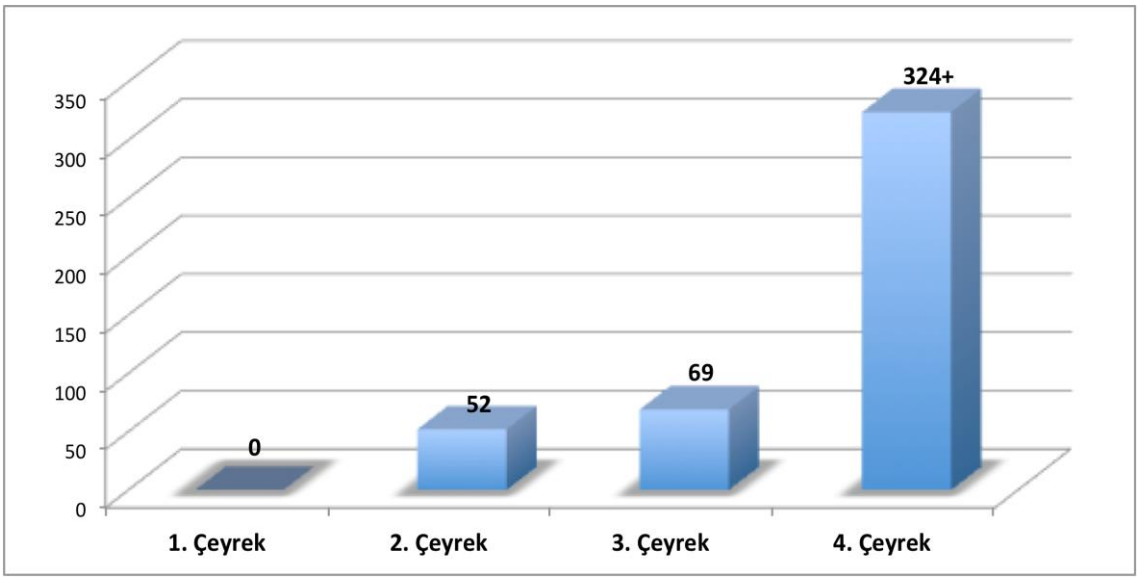
Şekil 1.1: Kötücül yazılımların son 10 yılı.

Aynı raporda, kötücül uygulamalar tarafından en çok suistimal edilen güvenlik açıklarına bakıldığında, gizlilik ihlali %74 oranla ilk sırada gelmektedir. Sonuçlar, kötücül uygulamaların bilgi çalma yönündeki eğilimini destekler niteliktedir. Şekil 1.2, kötücül uygulamalar tarafından en çok ihlal edilen bu güvenlik açıklarını belirtmektedir. Şekil 1.3 ise; kötücül yazılımcılara para kazandırmayı hedefleyen, kullanıcı bilgilerini ele geçiren ve şifreleyen, şifrenin açılması için de fidye talep eden uygulamaların (ransomware) 2014 yılı çeyreklerindeki miktarlarını göstermektedir. Şekil 1.3'de görüldüğü üzere, bu tarz uygulamaların miktarında sürekli bir artış gözlemlenmektedir.

[1] Hp Güvenlik Araştırma Merkezi 2015 Yılı Siber Risk Raporu, http://images.info.arcsight.com/Web/ArcSight/%7bfe8772f5-04dc-42f2-900c-68907be21770%7d_Cyber_Risk_Report_2015_EN_final.PDF, [Erişim tarihi: 12.06.2015]



Şekil 1.2: En çok suistimal edilen güvenlik açıkları.



Şekil 1.3: 2014 yılı çeyreklerindeki fidyeci uygulama miktarları.

Android güvenlik çözümleri alanında bu kötücül yazılımların tespiti ve önlenmesi yolunda araçlar geliştiren birçok aktör vardır. Buna rağmen, mobil dünyadaki kötücül yazılımların miktarı ve gelişimi konusunda güvenilir veriler bulmak oldukça zordur. Gerçek değerlere ulaşmak zor olsa da, Temmuz 2011 tarihinden bugüne Android tabanlı kötücül yazılım miktarı %472 artmıştır [12]. Google ekibinin bu tür uygulamaları Google Play Uygulama Mağazası'ndan anında kaldırabilmesine rağmen, uygulamalar kullanıcıların büyük bir kısmı tarafından yüklendiğinde, bu uygulamaların zararlı olup olmadıkları tespit edilebilmektedir. Mobil anti-virüs uygulamaları ve yakın zamanda Google tarafından geliştirilen Google Bouncer [13] hizmeti kısmen boşluğu doldursa

da, bu tarz uygulamalar kötücül davranış içeren kodların önceden analiz edilmesine ihtiyaç duymaktadırlar. Dolayısıyla, kötücül yazılımların uygulama mağazalarına düşmeden önce belirlenmesini sağlayan bir yöntem gerekmektedir.

Literatüre bakıldığında; Android işletim sistemini daha güvenli hale getirmek, kullanıcıların güvenliği ve gizliliğili ile ilgili sorunlarına çözüm bulmak amacıyla birçok çalışma yapılmıştır. Araştırmaların bir bölümü Android işletim sisteminin çalışma yapısına müdahale ederek güvenliği sağlama yoluna giderken [14-18], bir bölümü de sadece Android izin sistemine dayalı çözümler sunmuştur. Android izin sistemine dayanan çalışmalarda, Android izin sistemini genişleterek güvenlik sağlanmaya çalışılmıştır [19-24]. Bunların yanında, uygulamaların yeniden oluşturulmasına dayanan çözümler de mevcuttur [25-28]. Bu tip çalışmalar, belirli kurallar çerçevesinde uygulamaların bayt koduna inip, uygulamaların çalışma şeklinde değişiklik yaparak güvenliği sağlamaktadır.

Literatürde yapılan çalışmaların eksiklerinin giderilebilmesi, Android platformundaki güvenlik ve gizlilik problemlerinin kullanıcı odaklı olacak şekilde çözülebilmesi, mobil cihazdan bağımsız bir servis geliştirilmesi artık bir ihtiyaç haline gelmiştir [29-31]. Bu tarz bir yaklaşım; akıllı cihazlara yüklenmek istenen mobil uygulamaları daha cihaza yüklenmeden otomatik olarak analiz edebilecek, uygulamanın kullanılması halinde ortaya çıkacak güvenlik risklerini tespit edebilecek ve kullanıcıyı anlaşılır raporlarla anında bilgilendirebilecek düzeyde olmalıdır.

2. GENEL KISIMLAR

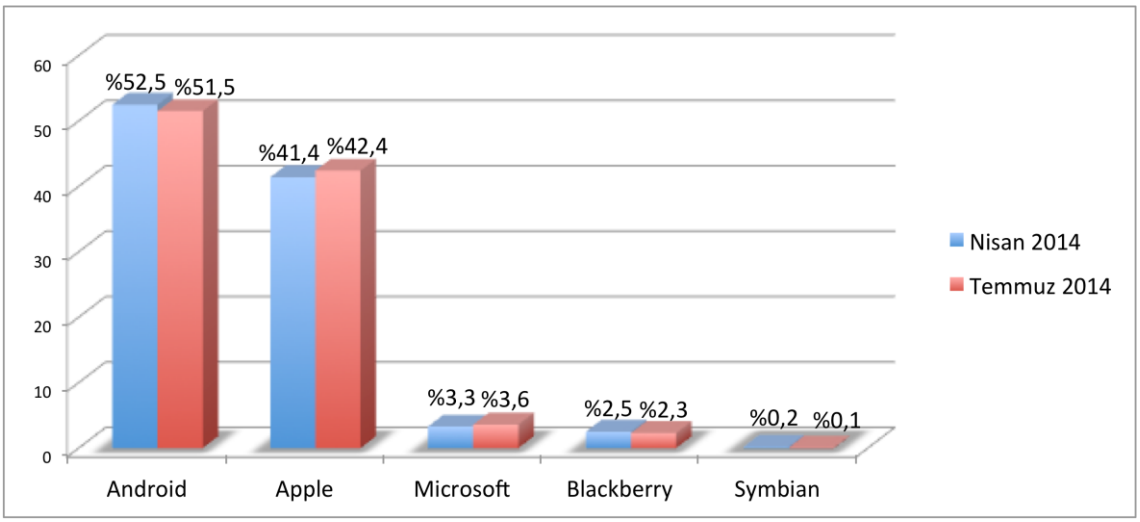
2.1. ANDROID İŞLETİM SİSTEMİ

Android işletim sistemi Open Handset Alliance (OHA) [32] ve Google firması öncülüğünde geliştirilen açık kaynak kodlu, Linux tabanlı, mobil cihazlar için geliştirilmiş bir işletim sistemidir. Platform önceleri Silikon Vadisi tabanlı bir şirket olan Android Inc. tarafından geliştirilmeye başlanmış, 2005 yılında ise Google firması tarafından satın alınarak hızla büyüyen ve gelişen bir platform haline gelmiştir. Üç yıllık bir gelişme sürecinin sonucunda 2008 yılı Kasım ayında ilk Android tabanlı cihaz satışa sunulmuştur. Tablo 2.1, Android platformunun kilometre taşlarını göstermektedir [33].

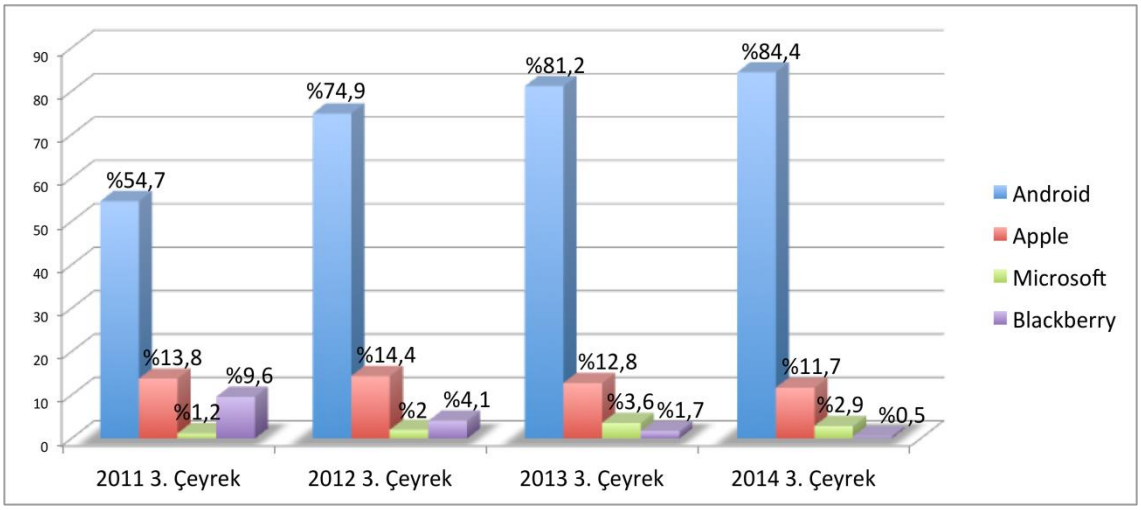
Tablo 2.1: Android platformu ve kilometre taşları.

Tarih	Olay
1 Temmuz 2005	Google Android Inc. firmasını satın aldı.
12 Kasım 2007	Android piyasaya sürüldü.
28 Ağustos 2008	Android Market duyuruldu.
23 Eylül 2008	Android 1.0 platformu yayınlandı.
21 Kasım 2008	Android açık kaynak kodlu yazılım olarak sunuldu.
13 Şubat 2009	Amerika'da Android Market ücretli uygulama alımına başladı.
2009-2015	Android platformu güncellenerek yeni sürümleri yayınlanmaya devam edildi. Son sürüm 5.0 Lollipop.

Android işletim sistemi akıllı cihaz pazarında büyük bir etki yaratmıştır. İlk Android tabanlı cihazın piyasaya çıkmasından 2 yıl sonra Android 65 milyonun üzerinde kullanıcı ve %26'lık pazar payıyla 2. büyük mobil platform iken, bugün Amerika'da 2014 yılında %52 pazar payı ve 100 milyonun üzerinde kullanıcısı ile en büyük mobil platform haline gelmiştir [34, 35]. Dünya çapında ise %84'lük bir pazar payına sahiptir. Şekil 2.1, Android platformunun 2014 Temmuz ayında Amerika'daki pazar payını, Şekil 2.2 ise dünyadaki pazar payını göstermektedir.



Şekil 2.1: Amerika'daki Android pazar payı.



Şekil 2.2: Dünyadaki Android pazar payı.

Android tabanlı mobil cihazların sayısı arttıkça, cihazlarda kullanılan veri miktarı da eşit miktarda artacaktır. Cihazlar çok miktarda kişisel veri içermesinden dolayı, siber suçluların saldırı yapması için cazip bir ortam sunmaktadırlar. Android platformundaki saldırılara karşı önlem alıp, çözüm sunabilmek için bu platformun mimarisinin, bileşenlerinin ve çalışma prensibinin Android geliştiricileri, firmalar ve uygulama geliştiricileri tarafından iyi anlaşılması gerekmektedir.

2.2. ANDROID TEMEL ÖZELLİKLERİ

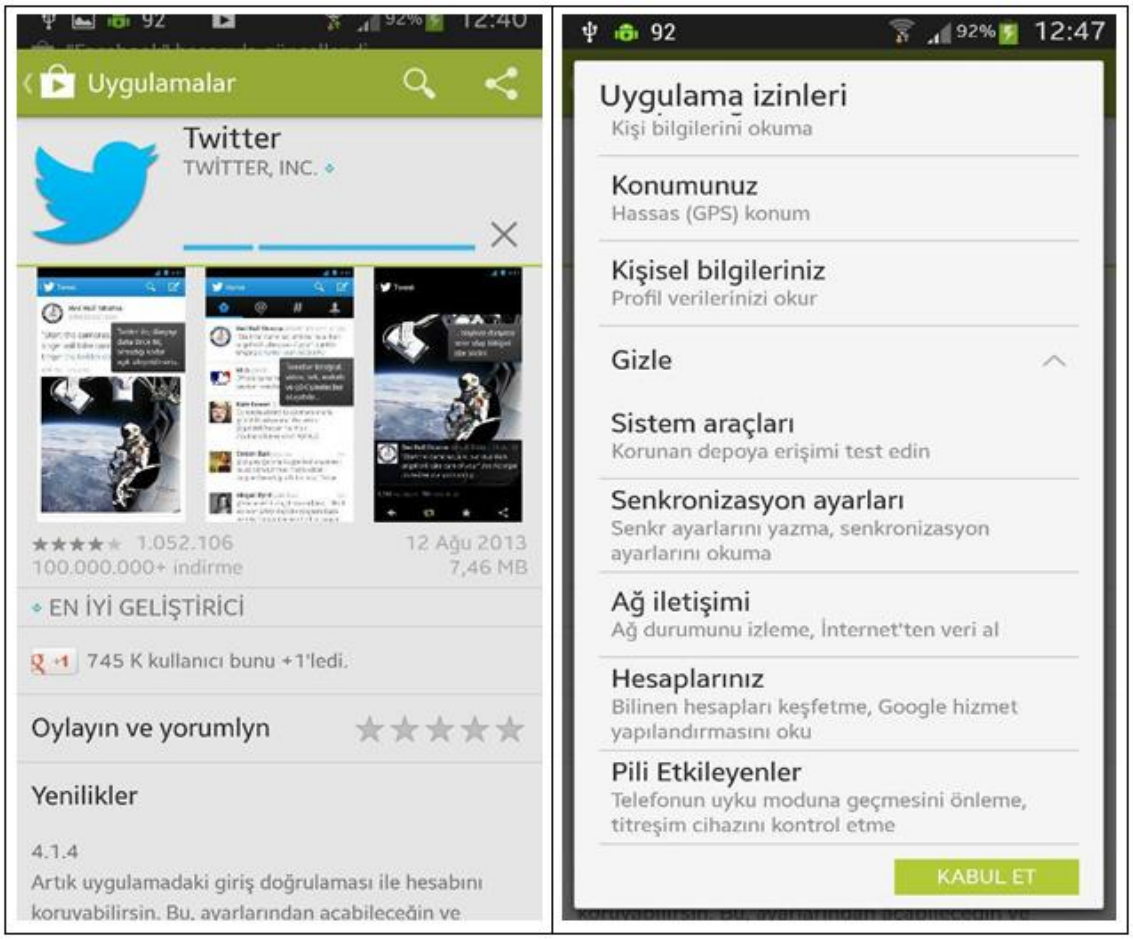
Android'in ilk temel özelliği herhangi bir anda çevrimiçi olabilme özelliğidir. Bu nedenle de hem GSM (3G, 4G) ağı hem de kablosuz ağ (Wi-Fi) desteği vardır. Bunun yanında mesaj gönderme ve alma, arama yapma, aramaya cevap verme gibi diğer GSM

tabanlı özellikler de yer almaktadır. İkinci temel özelliği ise Android Google Play Uygulama Mağazası'ndan uygulama indirme ve yükleyebilme özelliğidir. Bu özellik, hem kullanıcıların cihazlarını daha verimli kullanmalarını sağlayan hem de güvenlik ile ilgilenen araştırmacılara araştırma imkanı açısından zengin bilgi kaynağı sunan bir özelliktir. Üçüncü ve son temel özelliği ise kullanıcılara verilerini dahili ve harici bellekler aracılığı ile depolama imkanı sunmasıdır.

Android çok güçlü bir mobil platform olması yanında tamamen ücretsizdir. Ücretsiz olmasının sebebi ise Google firmasının stratejisinden kaynaklanır. Google firması arama motoru üzerinde araştırma ve geliştirme yapan bir firma olmasından dolayı, gelirinin temeli arama üzerine dayalıdır. Ücretsiz olan bu platform aracılığı ile insanların daha fazla çevrimiçi olması, daha fazla arama yapması sağlanmakta ve bu da Google'a kazanç olarak yansımaktadır [36].

Android platformunun yenilikleri desteklemesini sağlayan en önemli yolu sağladığı çekirdek yazılımlar sayesinde üçüncü parti uygulamaların geliştirilmesi ve dağıtımıdır. Şubat 2015 itibariyle Android Market'te bulunan uygulama miktarı 1.500.383'tür ve sadece Amerika'da aylık ortalama 500 milyondan fazla uygulama indirilmektedir [37]. İkinci en büyük mobil platform olan Apple iOS'de de aynı strateji mevcuttur. Fakat uygulama dağıtımı noktasında temel farklılıklar mevcuttur. Apple firması App Store'a gönderilen uygulamaları gizli bir değerlendirme sürecine tabi tutmaktadır. Uygulamalar zaman alıcı bir değerlendirme ve kabul sürecinden geçmektedir [38, 39]. Android tarafında ise neredeyse yok denebilecek düzeyde bir değerlendirme süreci mevcuttur. Uygulama geliştirmek isteyen herhangi bir kişi 25 dolar ücret ödeyerek ve kendilerine atanan benzersiz bir kimlik numarası ile markette geliştirici olarak yer alabilmektedir [40]. Google kötü amaçlı geliştiricileri güvenlik nedeniyle engelleyebilmekte ve uygulamaları cihazlardan ve marketten uzaktan silebilmektedir [41]. Buna rağmen Google bu tip durumlarda çok fazla işlem yapmamaktadır. Bunun yerine Google, Android işletim sisteminin güvenliğini sağlamak için ileride detaylıca anlatılacak olan ve temel bir bileşen olan izin sistemini (permissions) kullanmaktadır [42]. Bu sistemde bir uygulamayı yüklemeye karar veren kullanıcıya uygulamanın istekte bulunduğu tüm izinlerin listesini içeren bir pencere sunulur. İzin sistemi ya hep ya hiç prensibine dayanmaktadır [43]. Kullanıcının bu listedeki izinlerden herhangi birini seçme imkanı

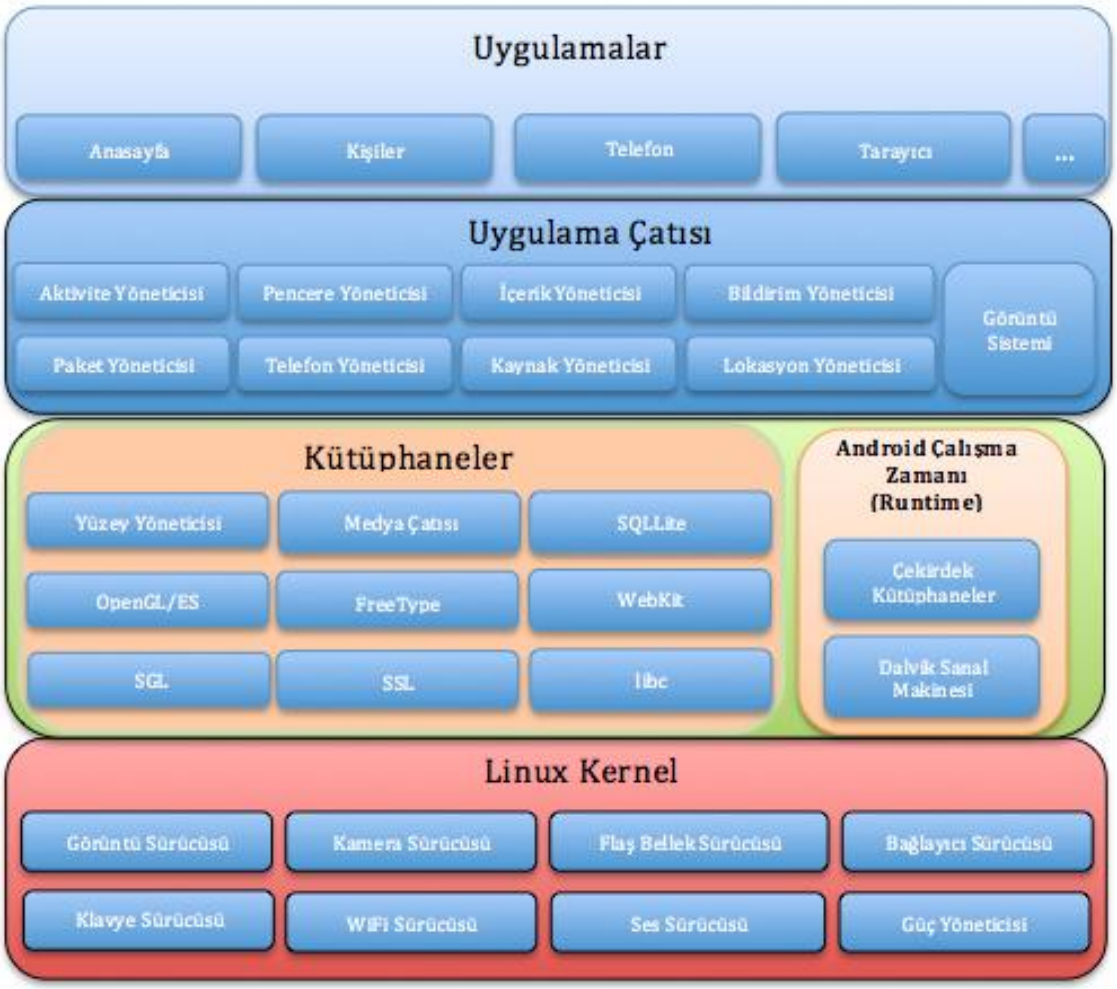
yoktur. Ya tüm izinler kabul edilir ve uygulama yüklenir ya da hiçbiri kabul edilmeyerek uygulama yüklenmez [44]. Şekil 2.3, uygulama yükleme ekranını ve izin listesini göstermektedir.



Şekil 2.3: Android uygulama yükleme ekranı ve izin listesi.

2.3. ANDROID MİMARİSİ

Güvenlik analizi yapabilmek, güvenlik yazılımları ya da servisleri geliştirebilmek için Android işletim sistemi mimarisinin iyi anlaşılması gerekir. Bu bölümde Android mimarisi ve katmanları detaylı bir şekilde anlatılmaktadır. Android mimarisi 5 temel katmandan oluşmaktadır. Şekil 2.4, mimarinin tüm katmanlarını göstermektedir [45].



Şekil 2.4: Android mimarisi ve katmanları.

Bu katmanlar:

- *Linux Kernel Katmanı*
- *Kütüphane Katmanı*
- *Çalışma Zamanı Katmanı*
- *Uygulama Çatısı Katmanı*
- *Uygulama Katmanı*

Linux Kernel Katmanı: Mimarinin en altta bulunan katmanıdır. Geliştirici ve kullanıcılarla doğrudan bir bağı olmamasına rağmen, tüm sistemin kalbidir. Android sisteminde aşağıdaki fonksiyonları sunar [46]:

- *Donanım soyutlaması*
- *Hafıza yönetimi*
- *Güvenlik*
- *Güç yönetimi*
- *Donanım sürücüleri*
- *Paylaşılan kütüphane desteği*
- *Ağ bağlantısı*
- *İş parçacıklarının iletişimi için bağlayıcı çatı (binder framework)*

Kütüphane Katmanı: Linux kernel katmanının bir üstünde yer alan bu katman çeşitli kütüphaneleri barındırır. Kütüphaneler, çok çeşitli verilerin nasıl ele alınacağına yardımcı olan fonksiyonlar içermektedir. Örneğin; çeşitli video ve ses dosyalarının oynatılması ya da kayıt edilmesini Medya Çatısı yönetir. Bu katmanda bulunan açık kaynak kodlu diğer kütüphaneler [47]:

- *Yüzey yöneticisi :* Ekrandaki pencereleri yönetir.
- *SGL :* 2D grafik kütüphanesidir.
- *OpenGL/ES :* 3D grafik kütüphanesidir.
- *Freetype :* Font kütüphanesidir.
- *Webkit :* Tarayıcı motorudur.
- *Libc :* Sistem C kütüphanesidir.
- *SqlLite :* SQL veritabanı motorudur.
- *Open SSL :* Güvenlik kütüphanesidir.
- *Medya çatısı:* Ses, video oynatma/kaydetme, resim görüntüleme gibi özellikler sunar.

Çalışma Zamanı Katmanı: Kütüphane katmanıyla aynı düzeyde bulunan bu katmanda kullanıcıların uygulamalarını geliştirmelerini sağlayan Java kütüphaneleri ve Dalvik sanal makinesi yer almaktadır. Dalvik sanal makinesi uygulamaların Android cihazlarda çalışmasını sağlar. Kayıtçı (Register) tabanlıdır ve düşük bellek gereksinimleri için optimize edilmiştir. DEX uzantılı Java sınıf dosyalarının Dalvik sanal makinesine uyarlanmış hali olan derlenmiş uygulama kodları üzerinde çalışır [48].

Uygulama Çatısı Katmanı: Bu katman geliştirilen uygulamaların doğrudan etkileşim içinde olduğu katmandır. Bu katmanda yer alan uygulamalar telefonun kaynak yönetimi, ses ve çağrı yönetimi gibi temel fonksiyonlarını yönetirler [49]. Bazı önemli yönetim uygulamaları şunlardır:

- *Aktivite Yöneticisi :* Uygulamaların aktivite yaşam döngülerini yönetir.
- *İçerik Sağlayıcı :* Uygulamalar arasındaki veri paylaşımını yönetir.
- *Telefon Yöneticisi :* Tüm sesli aramaları yönetir.
- *Kaynak Yöneticisi :* Uygulamalarda kullanılan kaynakların yönetimini sağlar.
- *Lokasyon Yöneticisi :* GPS ve baz istasyonu kullanarak lokasyon yönetimini gerçekleştirir.

Uygulama Katmanı: Android mimarisindeki en üst katmandır. Kullanıcıların en fazla etkileşimde bulunduğu (arama yapma, aramaya cevap verme, tarayıcı ile İnternette dolaşma vs.), standart programların yer aldığı katmandır. Geliştiricilerin, programcıların en çok erişim yaptığı katmanlar bu katmanın altındaki katmanlardır [50].

2.4. ANDROID GÜVENLİK MODELİ

Android işletim sistemi kullanıcıların, verilerin, uygulamaların, cihazın ve ağın güvenliğini sağlamak için bir güvenlik mimarisine sahiptir [51]. Mimari çok katmanlı bir güvenlik yapısı sunarken hem kullanıcı güvenliğini hem de açık kaynak kodlu olması nedeniyle esnekliği dikkate alarak tasarlanmıştır [52]. Android güvenlik mimarisi, en kullanışlı ve en güvenli mobil işletim sistemi olmak için kullanıcı verilerini ve sistem kaynaklarını korumayı hedefleyerek geliştirilmiştir. Bu hedefe ulaşmak için şu önemli güvenlik özelliklerini sağlamaktadır [53]:

- İşletim sistemi düzeyinde Linux çekirdeğine dayalı güçlü güvenlik mekanizması
- Tüm uygulamalar için şart olan uygulama izolasyonu (sandbox)
- Güvenlik iş parçacığı iletişimi
- Uygulama imzalama
- Uygulama tanımlı ve kullanıcı onaylı izinler

Çalışmamız Android uygulama güvenliğine dayalı bir çalışma olması nedeniyle Android uygulama güvenliği ile ilgili (özellikle izin modeli üzerinde) güvenlik mekanizması üzerinde durulmuştur.

2.4.1. Uygulama Güvenliği ve İzin Modeli

Android uygulamaları genellikle Java programlama dili kullanılarak yazılır ve Dalvik sanal makinesi üzerinde çalışırlar [54]. Bunun yanında, C/C++ dilleri kullanılarak da kod yazılabilmektedir. Uygulamalar ".apk" uzantılı tek bir dosya aracılığı ile yüklenmektedirler. Bir Android uygulamasının temel yapısı şu şekildedir [55]:

- *Android Manifesto Dosyası:* "AndroidManifest.xml" olarak adlandırılmış bu dosya, sisteme tüm üst düzey bileşenler ile (aktiviteler, servisler, içerik sağlayıcılar, yayın

alıcılar) ne yapılması gerektiğini söyleyen bir kontrol dosyasıdır. Buna ek olarak hangi izinlerin gerektiğini tanımlar.

- *Aktiviteler:* Bir aktivite tek bir göreve odaklanmış kod parçasıdır. Genelde bir kullanıcı arayüzü (User Interface) içerir ve aktivitelerden birisi uygulamanın başlama noktasıdır.
- *Servisler:* Arka planda çalışan kod parçacıklarıdır. Kendi iş parçacığı ya da başka bir uygulamanın iş parçacığı içinde çalışabilirler. Diğer bileşenler bu servislere bağlanır ve uzak metot çağrımı yöntemiyle metot çağrımı yaparlar.
- *Yayın Alıcılar (Broadcast Receivers):* İşletim sistemi ya da diğer uygulamalar tarafından "Intent" olarak adlandırılan iş parçacıkları oluşturulduğunda harekete geçen nesnelerdir. Uygulamalar bu yayın alıcılara kayıt olurlar ve gelen bilgiye göre davranışlarını değiştirirler.

Android İzin Modeli: Android'de tüm uygulamalar "Sandbox" adı verilen, izolasyonu sağlayan bir güvenlik kutusu içerisinde çalışırlar. Varsayılan olarak uygulamaların sistem kaynaklarına kısıtlı erişimi vardır. Sistem, Android uygulamalarının kaynaklara erişimini yönetir; uygulamarın doğru bir şekilde erişimde bulunup bulunulmadığını, zararlı davranışlar sergileyip sergilemediğini kontrol eder [56].

Kısıtlamalar değişik yöntemler kullanılarak geliştirilmiştir. Bazı durumlarda depolama izolasyonu yoluna gidilerek, bazı durumlarda ise hassas uygulama programlama arayüzleri (Application Programming Interface-API), "Permission" olarak adlandırılan izin modeline dayalı mekanizmayla korunarak kısıtlama sağlanmıştır. Korumaya alınmış API'lerden bazıları şunlardır [26]:

- *Kamera fonksiyonları*
- *Lokasyon verisi (GPS)*
- *Bluetooth fonksiyonları*
- *Telefon fonksiyonları*
- *SMS/MMS fonksiyonları*
- *Ağ/Veri ve NFC bağlantıları*

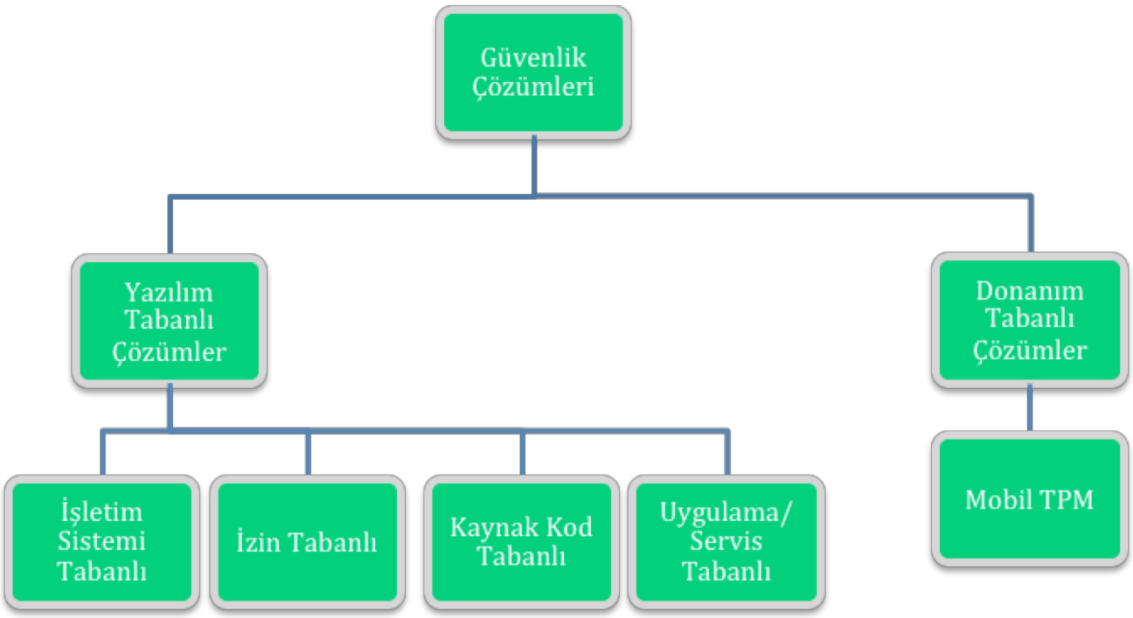
Bu API'lere sadece işletim sistemi aracılığı ile erişilebilir. SMS/MMS, telefon, Ağ/Veri ve NFC API'leri zararlı uygulamalar tarafından kullanıldığında kullanıcıya maliyet açısından zarar verecek fonksiyonlar içerdiğinden daha büyük öneme sahiptirler. Korunan API'lerin kullanılabilmesi için her uygulama kullanacağı fonksiyona ait izini kendi manifesto dosyasında belirtmek zorundadır. Uygulamanın yüklenmesi esnasında

sistem kullanıcıya bir diyalog ekranı aracılığı ile izinlerin listesini göstererek uygulamanın yüklenmesine devam edilip edilmeyeceğini sorar. Bu sistem ya hep ya hiç prensibine dayanmaktadır. Yani kullanıcı ya tüm izinleri kabul eder ve uygulama yüklenir ya da hiçbirini kabul etmez ve uygulama yüklenmez. Uygulama yüklü olduğu müddetçe izinler geçerliliğini korur, izin penceresi bir daha gösterilmez; silindiğinde ise izinler de silinmiş olur. Manifesto dosyasında belirtilmeyen bir izin kullanıldığında işletim sistemi güvenlik hatası fırlatır ve uygulamanın çalışması durdurulur. Android platformunda kullanılması halinde manifesto dosyasında tanımlanması gereken 151 adet izin mevcuttur. İlaveten, uygulamalar da kendi izinlerini tanımlayabilmektedirler [58].

2.5. İLGİLİ ÇALIŞMALAR

Mobil güvenlik alanında yapılan çalışmaları ve sunulan çözümleri organize bir şekilde anlatmak için bir sınıflandırma oluşturulmuştur. Bu sınıflandırma Android için önerilen güvenlik çözümleri temel alınarak elde edilmiştir. Android güvenliği temel alan olarak ele alınmış ve analiz edilmiştir. Sınıflandırma yapılmadan önce, mobil güvenlik bilgisine sahip olmak ön şarttır. Bu nedenle Android tabanlı güvenlik çözümlerine ulaşmak, araştırmacıların bu alanda yapmış olduğu çalışmaları incelemek gerekmektedir. Ancak literatürde yer alan tüm çalışmaların teker teker incelenmesi neredeyse imkansızdır.

İlgili çalışmaları incelerken ve sınıflandırma yaparken kullandığımız strateji, 2008 yılından (Android Uygulama mağazasının duyurulduğu yıl esas alınmıştır.) günümüze kadar olan yıllar arasında güvenlikle ilgili dergi ve konferansların hedef ve kapsamları incelenerek, mobil güvenlik, mobil gizlilik, bilgi güvenliği, Android güvenliği kelimeleri anahtar kelime olarak kullanılarak makale ve bildirilerin toplanmasına dayanmaktadır. Bu çalışma sonucunda Android güvenliği ile ilgili çalışmalar iki ana başlık altında incelenmiştir. Bunlar “Yazılım Tabanlı” ve “Donanım Tabanlı” çözümlerdir. Şekil 2.5, Android güvenlik çözümleri için geliştirdiğimiz sınıflandırmayı göstermektedir.



Şekil 2.5: Android güvenlik çözümlerinin sınıflandırılması.

2.5.1. Yazılım Tabanlı Çözümler

Yazılım tabanlı çözümler dört grupta sınıflandırılmıştır. Bu sınıf işletim sistemi, izin, kaynak kod ve uygulama/servis tabanlı çözümleri içerir.

2.5.1.1. İşletim Sistemi Tabanlı Çözümler

Bu tarz çözümler işletim sisteminde değişiklik yapılarak elde edilen çözümlerdir. Nauman ve diğ. [59] tarafından yapılan çalışmada, araştırmacılar kullanıcıya esneklik sunan, seçmeli izinlere, kısıt belirtmeye olanak veren, kaynakların kullanımını kısıtlamaya izin veren APEX adında bir sistem geliştirmişlerdir. Geliştirilen ek bir arayüz sayesinde de uygulama yüklenirken kullanıcı kullanmak istediği servislere izin verebilmekte ya da servisleri reddedebilmektedir. Bu sistem Android işletim sisteminin kodunda, yani mimaride değişiklik yaparak elde edilmiştir. Yapılan çalışma Android izin sistemine güçlü özellikler eklemesine, kullanıcıya izin seçme hakkı vermesine rağmen, teknik bilgisi olmayan kullanıcıların bilinçsiz bir şekilde, bu izinlerin ne işe yaradığını ve kabul etmeleri durumunda ne gibi risklerle karşılaşacağını bilmeden bu sistemi kullanmaları, güvenlik ve gizlilik açısından bir katkı sağlamamaktadır. Buna ek olarak kullanıcıların bu sistemi kullanabilmeleri için araştırmacıların değiştirmiş olduğu işletim sisteminin kullanıcıların cihazlarına yüklenmesi gerekmektedir. Teknik bilgisi olmayan normal kullanıcı için bunu yapmak mümkün değildir ve bu tip bir işlemin

yapılması da kullanıcının akıllı cihazını garanti kapmasına düşürecektir. Bunun yanında, geliştirilen yazılım APEX kullanıcıların veya geliştiricilerin kullanımına sunulmamıştır.

Bugiel ve diğ. [60] tarafından yapılan çalışmada, araştırmacılar iş dünyasına yönelik mobil cihazların güvenliğini Android sisteminin her katmanında ayrı ayrı sağlayacak TrustDroid adında bir yazılım çatısı geliştirmişlerdir. Kernel düzeyinde işlemlerin birbirleriyle haberleşmelerini, dosya sistemini kontrol eden Kernel Erişim Yöneticisi (Mandatory Access Control) yöneticisi, orta katmanda Kural Yöneticisi (Policy Manager), Güvenlik Duvarı Yöneticisi, en üst katmanda ise özelleştirilmiş Paket Yöneticisi yer almaktadır. Güvenlik sağlanırken 'Uygulama Renklendirme' yöntemi kullanılmıştır. Bu yöntemle her bir uygulamaya bir alana (domain) ait olacak şekilde renk atanır ve sadece aynı renkte yer alan uygulamalar birbirleriyle iletişim kurabilir, veri alıp gönderebilir. Dolayısıyla güvensiz olduğu belirtilen bir renkle işaretlenen uygulamalar hiçbir zaman güvenli olarak belirlenen uygulamalarla iletişime geçemez. Bu çalışmadaki sistem Android işletim sisteminde değişiklikler, sisteme ekler yapılarak oluşturulmuştur. Kullanıcıların bu sistemi kullanabilmeleri için özelleştirilmiş bu işletim sistemini yüklemeleri gerekmektedir.

SELinux (Güvenliği Artırılmış Linux), Linux'a güvenlik açısından ekstra özelliklerin eklenerek geliştirilmiş güvenlik özellikleri ve zorunlu giriş kontrolleri içeren bir çekirdek yapısıdır. Shabtai ve diğ. [61] tarafından yapılan çalışmada, araştırmacılar bu çekirdek yapısını Android işletim sistemine uyarlayarak daha güvenli bir sistem elde etme yoluna gitmişlerdir. Bunu yapmak için de Android işletim sistemini SELinux ile birlikte kaynağından yeniden derlemişlerdir. Yapılan çalışmada işletim sistemine müdahalede bulunulmuştur. Bu sistemin kullanıcılara güvenlik sağlayabilmesi için kullanıcı cihazındaki işletim sistemi kaldırılarak bu sistemin yüklenmesi gerekmektedir.

Ongtang ve diğ. [62] tarafından yapılan çalışmada araştırmacılar Android cihazlarda güvenliği sağlamak amacıyla SAINT adında bir yazılım çatısı geliştirmişlerdir. Geliştirilen çatı Android işletim sisteminin orta katmanında (middleware) değişiklikler yapılarak elde edilmiştir. Mimari 5 modülden oluşmaktadır. Birinci modül Saint Yükleyici Programıdır ve mevcut Android Yükleyici Programı'nda değişiklikler yapılarak oluşturulmuştur. Bu modül programların yüklenmesi sırasında aktif olarak, program içindeki izinleri inceler ve bu izinleri AppPolicy olarak adlandırılan 2. modülü

(Uygulama Kural Yöneticisi) aracılığı ile mevcut tanımlanmış kurallarla karşılaştırır. Çelişkili durumlar varsa uygulama yüklenmez. 3. Modül Saint Arabulucu (Mediator) modüldür. Bu modül çalışma zamanında aktif olarak arka planda gerçekleşen işlemlerin belirtilen kurallara göre çalışıp çalışmadığını denetler. 4. modül ise Yazılım Çatısı Kural Yöneticisi (Framework Policy Manager) modülüdür. Bu modül bir uygulama niteliğinde olup kullanıcıya var olan kuralların değiştirilmesi imkanını verir. Kurallar sadece bu modül aracılığı ile değiştirilebilmektedir. 5. modül ise uygulama geliştiricilerin kendi belirledikleri kuralları ekleyerek Saint'e entegre etmelerini sağlayacak bir modüldür.

2.5.1.2.İzin Tabanlı Çözümler

İzin tabanlı güvenlik çözümleri genelde pratik ve deneysel analize dayalı çözümler sunmaktadırlar. Deneysel analiz çalışmaları genellikle izin mekanizmasının nasıl kullanıldığını ya da nasıl kötüye kullanıldığını, eksik ya da fazla izin kullanıp kullanmadığını araştıran ve istatistiki raporlar sunan çalışmalardır. Diğer tarafta, pratik çözümler izin tabanlı filtreleme yapmayı ya da zararlı sayılabilecek izinlerin kaldırılmasını hedefler.

Barrera ve diğ. [63] tarafından yapılan çalışma, Android işletim sistemindeki izin modelini deneysel olarak analiz etmektedir. Çalışmanın temel amacı Android' deki izin tabanlı sistemin pratikte nasıl kullanıldığını (tasarımdaki beklentilerin gerçek hayattaki özellikleri karşılayıp karşılamadığı), bu sistemin güçlü ve zayıf yanlarını araştırmaktır. Yazarlar Android tabanlı 1100 uygulamayı öz düzenleyici harita (Self organizing maps) algoritmasını kullanarak analiz etmişlerdir ve geliştiricilerin Android izin modelini nasıl kullandıklarını belirlemeye çalışmışlardır. Belirtilen sonuçlara göre uygulamaların %60'ı İNTERNET iznini mutlaka kullanmaktadır. Uygulama İnternet iznine ihtiyaç duymasa bile, reklamların İnternet aracılığı ile yayınlanması geliştiricilerin bu izini eklemesine neden olmaktadır. İnternet izninin yanında SMS okuma, SMS yazma gibi izinler de çoğu uygulamalar tarafından kullanılmaktadır. Bu tip izinler kullanıcının bilgisi dışında kullanılabilmekte ve istenmeyen yüksek faturalara neden olabilmektedir. Sonuç olarak çalışma Android izin sisteminin geliştirilmesini önermekte, izin tabanlı sistemlerde iyi tanımlanmış izinlerin kullanıcılara daha detaylı izin hakimiyeti sağlayacağını not etmektedir. Aşırı derecede izin kullanımı kullanıcıyı olumsuz yönde

etkilemektedir. Kullanıcılar uygulamanın bu istekleri neden istediği konusunda hiçbir bilgi sahibi olmadan cihazlarının ekranında gösterilen izin isteklerini ya kabul etmekte ya da reddetmektedirler. Çalışma uygulamaların izin sistemini nasıl kullanıldığı, hangi kategorilerdeki uygulamaların hangi izinleri kullandığını, uygulamalarda ne kadar izin kullanıldığını ortaya koyan bir çalışma olması nedeniyle herhangi bir risk analizi, çözüm önerisi sunmamaktadır. Kullanıcı düzeyinde risk bilgilendirmesi yapacak, uygulamaların hangi izinleri ne şekilde kullandığı konusunda açıklayıcı, anlaşılır bilgilendirme yapacak olan bir sistem geliştirmiş olmamız bizim çalışmamızı bu çalışmadan farklı kılmaktadır.

Android sisteminde her bir API çağrımı manifesto (izinleri içeren XML dosyası) dosyasındaki bir izine karşılık gelmektedir. Kullanıcı uygulamayı yüklerken bu izinleri içeren bir liste sunulmaktadır. İzin verildiği zaman API çağrımları etkin hale gelmektedir. Ancak kullanıcın bu izinleri seçme hakkı yoktur. Gereğinden fazla izin verilmesi de güvenlik, gizlilik problemlerine yol açmaktadır. Johnson ve diğ. [64] tarafından yapılan çalışmada, araştırmacılar Android Market'teki uygulamaları otomatik olarak tarayarak indirmiş, bayt kod üzerinde statik analiz yaparak Android API' sindeki metot çağrımlarını manifesto dosyası içerisindeki izinlerle eşleştirme yapmışlardır. Bu sayede hangi uygulamaların gereğinden fazla ya da az izin kullandığı ortaya çıkarılmıştır. Bu makalede ortaya çıkarılan sistem izin listesi üzerinde çalışması nedeniyle çalışmamıza benzemesine rağmen, bizim çalışmamızda yapılmış olan sistemin sadece bir bölümünü (bayt kod analizi ve izin karşılaştırma) içermektedir. Buna ek olarak araştırma sadece aşırı izin ya da yetersiz izin kullanımının var olup olmadığını incelemekte, kullanıcı odaklı herhangi bir işlem içermemektedir.

Diğer çalışmalara benzer olarak Felt ve diğ. [65] tarafından yapılan çalışma da izin listesi üzerinde çalışmıştır ve bizim çalışmamıza da ışık tutmuştur. Geliştirilen Stowaway yazılım aracı ile izin dosyası ve kaynak kod incelenmiş, manifesto dosyasında kullanılacak API çağrımlarına ilişkin gerekli izin olmamasına rağmen kod düzeyinde yine de bu API çağrımlarını gerçekleştiren uygulamalar (gereğinden fazla izin kullanan uygulamalar) ortaya çıkarılmıştır. Bununla beraber, bir uygulamanın ihtiyaç duyabileceği maksimum izin miktarı belirlenerek, uygulamanın hali hazırda manifesto dosyasına eklemiş olduğu izin miktarıyla karşılaştırılmış ve uygulamaların

%35'inin gereksiz izin kullandığı ortaya çıkarılmıştır. Uygulamaların bu şekilde bir işlem yapmalarının nedenleri, şüpheli davranışlar/metot çağrımları, gereksiz izin kullanımları araştırılmıştır. Buna ek olarak izin haritası çıkarılmış, API çağrımlarıyla, izinler eşleştirilmiştir ve %85 oranında bir başarı elde edilmiştir. Yapılan çalışma izinlerin incelenmesi fikri açısından bizim çalışmamıza benzemekte, izin haritalaması yapması açısından farklılaşmaktadır. Tasarlamış olduğumuz serviste de izinler incelenmektedir ve gereksiz izin kullanımları tespit edilmektedir. Ancak kod analizi yoluyla haritalama işlemi yerine, kullanılan izinlerin ne kadar riskli olduğu, kullanıcıya ne kadar zarar verebileceği analiz edilerek bu bilgi görsel ve anlaşılır bir şekilde kullanıcı ile paylaşılmaktadır.

Stevens ve diğ. [66] tarafından yapılan çalışmada, veri madenciliği teknikleri kullanılarak Android Marketteki 10.000 uygulama incelenmiş, Android sistemindeki hangi izinlerin (manifesto dosyasındaki) daha sık kullanıldığı, hangi izinlerin popüler olduğu, izinlerin ne kadar kullanıldığı ve etkileri araştırılmıştır. Bunu yaparken Stackoverflow sitesi (yazılım geliştiriciler için çevrimiçi bilgi paylaşım sitesi) üzerindeki tartışmaların bahsedilenler (mentions) kısmında yazılan bilgiler kullanılmıştır. Analiz sonuçlarına göre incelenen uygulamaların %40'ı gereksiz yere izin kullanmakta, izinlerin popülerliği (Stackoverflow'daki konuşulma miktarına göre bulunmuş) ile izinlerin kötüye kullanımı arasında bir paralellik bulunmaktadır. Yapılan çalışma sadece bir analiz çalışması olup herhangi bir probleme çözüm üretmemekte, gerçekleştirmiş olduğumuz servise benzer herhangi bir araç ya da servis sunamamaktadır.

Zhou ve diğ. [67] tarafından yapılan çalışmada, araştırmacılar 5 popüler Android markette (resmi ve resmi olmayan) yer alan 204.040 uygulamayı incelemiş, izin tabanlı davranışsal kimlikleme (footprint) ile zararlı yazılımların tespitini yapan, filtreleme yolu ile de gereksiz ve zarara neden olan izinleri kaldıran DroidRanger adında bir sistem geliştirmişlerdir. Sistem 204.040 uygulamadan 211 tanesini zararlı ve bulaşıcı olarak tanımlamıştır. Bu tip zararlı yazılımlar Google tarafından marketten kaldırılmadan 48 saat içinde 260.000 kullanıcıya bulaşmıştır. Araştırmacılar davranışsal kimlikleme işlemini yaparken bilinen zararlı uygulamaları ve kullandıkları izinleri incelemişler ve bu uygulamalardan çıkan kimliği, taranan uygulamalardaki kimlikle karşılaştırarak

eşleme yapmışlardır. Kimliği bilinemeyen uygulamalar için de sezgisel filtreme yoluna gitmişlerdir. Sezgisel filtreleme işlemi, zararlı uygulamalardaki şüpheli davranışları belirleyerek ve bu bilgiyi kullanarak diğer bilinmeyen zararlı uygulamaların tespitinden ibarettir. Verilen sonuçlara göre geliştirilen sistem popüler anti-virüs yazılımlarının başarısız kaldığı durumlarda bile başarılı sonuçlar vermiştir. Ancak, oluşturulan sezgisel yaklaşım sadece belli başlı davranışlar için geliştirilmiştir ve yeni davranış geliştiren uygulamaları tespit edemeyecektir. Çalışma, zararlı yazılımların tespitine yönelik ve kullanıcıyı dahil etmeyen bir çalışma iken, bizim çalışmamız uygulamaların risk analizi ve puanlamasına yönelik, kullanıcı odaklı, web tabanlı bir servis niteliğindedir.

Jeon ve diğ. [68] tarafından yapılan çalışmada, araştırmacılar Android platformundaki güvenlik, gizlilik problemlerini önlemeye yönelik 2 ana modülden oluşan ve mobil cihaz üzerinde çalışan bir uygulama geliştirmişlerdir. Uygulamanın ilk modülü Mr. Hide olarak adlandırılmıştır. Bu modül manifesto dosyasındaki izinleri daha özelleştirerek değiştirmektedir. Örneğin; çok tehlikeli ve genel olarak görülen İnternet iznini, belli bir alan adına kısıtlayan (İnternet-URL(domain)) bir izin yapısıyla değiştirmektedir. Bu da izinler üzerinde daha fazla kontrol imkanı sunmaktadır. 2. Modül ise Dr. Android olarak adlandırılmıştır. Bu modül, 1. Modülde değiştirilen izinleri dikkate alarak uygulamanın bayt kodunda değişiklik yapmakta ve uygulamayı yeni izinlere göre yeniden oluşturmaktadır. Bu işlemler yapılırken hiçbir şekilde cihazda çalışan işletim sistemine bir müdahale, sistemde kod düzeyinde bir değişiklik yapılmamaktadır. 1. Modül cihaza %10-50 arasında ekstra yük bindirirken, 2. Modülün kodu yeniden oluşturması ortalama 1 dakika almaktadır. Çalışma marketteki çeşitli kategorilerde 19 popüler ücretsiz uygulama üzerinde ve araştırmacılar tarafından tehlikeli olarak tanımlanmış 7 adet izin üzerinde denenmiştir. Elde edilen sonuçlara göre geliştirilen sistem başarılı bir şekilde uygulamalarda istenen değişiklikleri yapmış, yeniden oluşturulan uygulamalar sorunsuz olarak çalışmıştır. Ancak sistem cihazda arka planda çalışan bir servis niteliğinde olduğundan ve kodun yeniden oluşturulması cihaz üzerinde gerçekleştiğinden dolayı sisteme yük binmekte, uygulamalar da orijinallerine oranla daha geç açılmaktadır. Buna ek olarak geliştirilen sistem kullanıcıların hizmetine sunulmadığı için bir fayda sağlayamamaktadır. Çalışma sadece 19 popüler ücretsiz uygulama üzerinde uygulanmasına karşın, bizim çalışmamız daha geniş ve kapsamlı (zararlı yazılımları da içerecek şekilde) bir veri seti kullanmaktadır.

Enck ve diğ. [21] tarafından yapılan çalışmada, araştırmacılar Android izin sistemini genişleterek güvenliği sağlamaya yönelik Kirin adında bir uygulama yükleyici yazılım geliştirmişlerdir. Yazılım, Android uygulama yükleyicisi yerine kullanılmakta ve uygulamalardaki manifesto dosyasındaki izinleri yükleme anında okuyup önceden tanımlanmış güvenlik kurallarıyla karşılaştırmaktadır. Eğer uygulama içindeki izinler belirlenen kurallara uyum sağlarsa yüklenmekte, sağlamazsa yüklenmemektedir. Bu çalışma sadece izin sistemi üzerinde işlem yapmaktadır. Risk analizi, puanlama ve kullanıcı tabanlı servis sunma gibi özelliklerden yoksundur.

Sarma ve diğ. [69] tarafından yapılan çalışmada, araştırmacılar Android platformundaki uygulamaların içinde yer alan izin listesine göre risk sinyali oluşturan bir sistem geliştirmişlerdir. Risk sinyalleri oluşturulurken Android Market'teki 150.000 civarında zararsız uygulama ve 121 adet zararlı uygulama incelenmiştir. 122 Android izninden ise 26 kritik izin seçilip bu izinler üzerinde işlem yapılmıştır. Zararlı ve zararsız uygulamaları birbirinden ayırmak için de ağırlıklandırılmış Destek Vektör Makinesi (SVM) kullanılmıştır. Risk sinyalleri oluşturulurken uygulamaların sağladıkları faydayı gösteren kategori bilgilerinden de faydalanılmıştır. Kategori bilgisinin alınmasındaki amaç ise uygulamanın kendisinden beklenen işlevi yerine getirip getirmediğini anlamaktır. Bu çalışma risk değerlendirme açısından bizim çalışmamıza ışık tutan bir çalışma olmasına rağmen herhangi bir puanlama yapmamaktadır.

2.5.1.3. Kaynak Kod Tabanlı Çözümler

Bu kategorideki çözümler uygulamaların bayt kodlarını işleyerek çözüm sağlarlar. Bu çözümlerden bazıları statik ve dinamik analiz teknikleri kullanarak karar verirken, bazıları da uygulamaların bayt kodlarında değişiklik yaparak uygulamaları yeniden inşa ederler.

Enck ve diğ. [20] tarafından yapılan çalışmada, araştırmacılar Android Market'teki 1100 popüler ve ücretsiz uygulamayı incelemişlerdir. Kendi geliştirdikleri ve açık kaynak kodlu olarak sunulan DED kod dönüştürücü (decompiler, çalıştırılabilir kodu kaynak koda dönüştürücü) aracılığı ile Android uygulama imajından kaynak koduna erişerek yaklaşık 21 Milyon satır kod üzerinde statik analiz uygulamışlardır. Yapılan analizde konum bilgisi, kişiyi tanımlayıcı bilgiler gibi kişisel gizliliği tehlikeye atacak bilgilerin yanlış kullanımı ortaya çıkarılmıştır. Bu çalışmada oluşturulan DED kod

dönüştürücü şu ana kadar oluşturulan ilk ve en başarılı (%94 oranında başarı sağlayan) kaynak kod oluşturma aracı olduğundan çalışmamızda da Android tabanlı uygulamalarda kullanılan izinleri kaynak koda inerek incelemek amacıyla kullanılmıştır. Bu çalışmanın temeli detaylı bir şekilde kaynak kod analizi yaparak Android sisteminin güvenliğini incelemekte, son teknoloji ürünü yeni bir DED kod dönüştürücü aracını tanıtmakta ve bu aracın başarısını sergilemektedir.

Gibler ve diğ. [70] tarafından yapılan çalışmada, araştırmacılar uygulamaların yüklenmesi durumunda telefon numarası, kişi listesi, konum, Wi-Fi verisi, mikrofonla kaydedilmiş ses gibi hassas bilgilerin sızıp sızmadığını kod düzeyinde statik analiz yöntemiyle inceleyerek otomatik olarak bulan bir yazılım çatısı geliştirmişlerdir. 24.350 adet uygulama incelenmiş, bunların arasında 7414 uygulamada toplam 57.299 adet potansiyel gizlilik sızıntısı bulunmuştur. Bu sızıntıların çoğunun da reklam kütüphaneleri aracılığı ile ortaya çıktığı belirlenmiştir. Uygulama, sızıntıları rapor halinde sunma özelliğine sahip olmasına rağmen, bir güvenlik uzmanının bu raporu inceleyerek gerçekten de kişisel gizliliği tehlikeye atıp atmadığı konusunda doğrulamasına ihtiyaç duymaktadır. Bu nedenle geliştirilen araç güvenlik uzmanlarının zararlı yazılımları inceleme zamanını azaltabilecek bir sistem niteliğindedir. Güvenlik konusunda hiçbir bilgisi olmayan normal kullanıcılara hitap edecek bir sistem olmaması nedeniyle çalışmamızdan farklılık göstermektedir.

Zhou ve diğ. [71] tarafından yapılan çalışmada, araştırmacılar bir yıllık çaba sonucunda Android uygulama mağazalarından topladıkları 1260 zararlı yazılımı (malware) sistematik bir şekilde inceleyip analiz etmişlerdir. Analizi yapmak için herhangi bir yazılım geliştirilmemiş, uygulamalar manuel olarak incelenmiştir. Zararlı yazılımlar 3 ana grup altında kategorize edilmiştir. Marketteki popüler uygulamaları değiştirip, içlerine zararlı kod parçacıkları yerleştirerek yeniden paketleyen zararlı taklit uygulamalar birinci grubu oluşturmaktadır. Zararlı yazılımların %86'sı bu işlemi yapmaktadır. İkinci grup ise Android platformunun açıklarından yararlanarak kernel düzeyinde zararlı kod çalıştıran zararlı yazılımlardır. Yine zararlı yazılımların %36'sı bu açığı kullanmaktadır. Üçüncü grup mobil cihazları bot haline dönüştürüp, uzak sunuculardaki komuta ve kontrol merkezlerinden emir alarak zararlı faaliyetlerde bulunan uygulamalardır. Zararlı yazılımların %90'ı bu özelliğe sahiptir. Araştırmacılar

bu sonuçların yanında, mobil anti-virüs alanında önemli yerlere sahip olan AVG Anti-virüs, Trend Micro, Norton firmalarının zararlı yazılımların tespiti konusundaki başarılarına dair sonuçları da yayınlamışlardır. Varılan sonuca göre, bu yazılımlar en iyi senaryoda %79, en kötü senaryo da ise %20 oranında başarı sağlamışlardır. Bu çalışma, veri toplama ve analiz etme işlemlerine dayanan başarılı ve geniş kapsamlı bir çalışmadır. Ancak herhangi bir çözüm üretme yerine bilgilendirme amacı güden bir çalışmadır. Bu çalışmada elde edilen veri seti bir yıllık bir çabanın ürünü ve en kapsamlı veri setlerinden biri olması nedeniyle bizim çalışmamızdaki kötücül yazılım veri setinin de temelini oluşturmaktadır.

Android uygulamalarına yükleme zamanında bir defa izin verildiğinde izinleri sonradan iptal etme imkanı yoktur. Ayrıca dinamik izin atama gibi bir özellik de desteklenmemektedir. Uygulamaya izin verildikten sonra, cihazdaki verilerin nasıl ve nerelerde kullanıldığını, ne gibi etkilere yol açacağını kullanıcı hiçbir şekilde bilemez. Backes ve diğ. [72] tarafından yapılan çalışmada, araştırmacılar AppGuard isimli Android cihazlarda çalışan bir güvenlik uygulaması geliştirmişlerdir. Bu uygulamanın 3 temel özelliği vardır. Birinci özelliği güvenlik politikaları oluşturabilmesidir. Kişisel bilgiye erişim, ağ erişimi için soket oluşturma, GPS ve kameraya erişimi kısıtlayıcı ya da bunlara izin veren politikalar oluşturacak metotlar sunmaktadır. İkinci özelliği uygulamaların bayt kodunu yeniden yazmasıdır. Üçüncü özelliği ise sunduğu arayüz ile kullanıcıların uygulama bazlı bireysel politikalar oluşturmasıdır. Uygulama kritik metotları izlerken Java Yansıma (Reflection) API' sini kullanmaktadır. Bu API herhangi bir sınıf, metot ya da değişken isimleri hakkında bilgi sahibi olmadan bu yapıları çalışma zamanında incelemeyi ve onlara müdahale etmeyi sağlamaktadır. Kritik metotlar çağrıldığında, oluşturulan politika metotlarının yansıma yardımıyla parametreleri incelenir. Güvenlikle ilgili bir metot bulunduğunda, monitör arayüzü buna karşılık gelen koruma metodunu çağırır. Geliştirilen uygulama hali hazırda cihazlara yüklenmiş uygulamalar için herhangi bir işlem yapamamaktadır. Bu tip durumlarda sistemin çalışabilmesi için uygulamanın silinerek yeniden yüklenmesi gerekmektedir. Araştırmacılar Android marketteki 13 uygulama üzerinde sistemi denemişlerdir ve test sonuçlarına göre uygulamalar cihazlarda yeniden oluşturulurken sisteme aşırı yük binmektedir. Örneğin popüler Angry Birds oyununu mobil cihazda yeniden oluşturmak yaklaşık 45 saniye, Instagram uygulamasını yeniden oluşturmak 66

saniye, WhatsApp içinse 58 saniye sürmektedir. Uygulanan güvenlik politikaları nedeniyle fonksiyon çağrımlarına binen yük ise %5 civarındadır. Araştırmacıların geliştirmiş olduğu uygulama cihazdan silinmesi durumda, uygulamalar tekrar eski çalışma durumuna dönmektedir. Bu da geliştirilen sistemin bir zayıflığıdır. Google firması bu uygulamayı Android uygulama mağazasından kaldırmıştır ve daha önce bu uygulamayı marketten yükleyenler için sadece firmanın sitesinden indirilmesine izin verilmektedir.

Xu ve diğ. [23] tarafından yapılan çalışmada, araştırmacılar Android cihazlarda güvenliği ve gizliliği sağlamaya yönelik Aurasium adında bir araç geliştirmişlerdir. Araç 2 bileşenden oluşmaktadır. İlk bileşen uygulamaya yönetim kodu ekleyen yeniden paketleme mekanizmasıdır. İkinci bileşen ise işletim sistemiyle iletişime geçen uygulamaların önüne geçerek onları izleyen mekanizmadır. Uygulamanın içine yerleştirilen izleme mekanizması kişinin güvenliğini, gizliliğini tehlikeye düşürecek herhangi bir metot çağrımında kullanıcıya uyarı göstermekte ve kullanıcıya bu metodun çağrımını engelleyip engellememesine dair soru sormaktadır. Kullanıcının cevabı kaydedilerek bu cevaba göre işlem yapılmaktadır.

Davis ve diğ. [26] tarafından yapılan çalışmada, araştırmacılar mobil uygulamaların güvenliğini sağlamak için uygulamaları kullanıcının isteğine göre bayt kod düzeyinde yeniden yazan bir yazılım çatısı geliştirmişlerdir. Sistem kullanıcıdan aldığı veriler dahilinde uygulamada statik analiz tekniğini uygulayarak değiştirilmesi gereken metotları bulur. Bulunan metotlar yerine kullanıcının istediği davranışı sergileyecek metotlar bayt kodun içine yerleştirilir ve bu metotların çağrılması sağlanır. Çalışmada Android uygulama mağazasındaki 100 popüler uygulama arasından rastgele 30 uygulama seçilmiş, “Math.sqrt”, “url.OpenStream”, “StringBuilder.append”, “java.lang.String” gibi belli başlı metotların çağırımında araya girilerek, bunları özel metotlarla değiştirme işlemi yapılmıştır. Buna ek olarak bu metotlara günlük özelliği eklenmiş ve çalışma anında kendi metotlarının çağırılıp çağrılmadığı test edilmiştir. Elde edilen sonuçlara göre yeniden yazılan uygulamalarda hiçbir sorun çıkmamış ve cihazlarda sorunsuz çalışmıştır. Ancak uygulamaların yeniden yazılabilmesi için kullanıcının hangi metotların yeniden yazılması gerektiğini bilmesi gerekmektedir ve kullanıcı geliştirilen sisteme bu metotların tam listesini, metotların imzalarını (dönüş

tipi ve parametre), bulundukları paketleri sunmalıdır. Aksi taktirde sistem başarısız olacaktır. Ek olarak kullanıcı uygulama değiştirildikten sonra beklediği davranışı da belirtmesi gerekmektedir. Bunu da Java kodu yazarak gerçekleştirmektedir. Bu özellikler göz önünde bulundurulduğunda sistemi normal kullanıcıların kullanması mümkün değildir. Hangi metotların güvenlik tehlikesi arz ettiğini, değiştirilmesi gerektiğini çoğu kullanıcı bilemez. Ayrıca beklenen davranış kod yazılarak belirlenmektedir ki yine çoğu kullanıcıdan kod yazmasını beklemek iyi bir yaklaşım değildir.

2.5.1.4. Uygulama/Servis Tabanlı Çözümler

Uygulama/Servis tabanlı çözümleri iki grupta sınıflandırılmıştır. İlk gruptaki çözümler arka planda çalışan bir iş parçacığı olarak akıllı cihazlara yüklenir ve sistemi inceler ya da masaüstü bilgisayarlara yüklenen bir uygulama aracılığı ile analiz gerçekleştirir. Mobil anti-virüs yazılımları ve veri analiz araçları bu grupta yer alır. İkinci grupta yer alan çözümler ise güvenlik kontrol ve analiz işlemlerini bulutta gerçekleştiren, Servis olarak Güvenlik (Security as a Service, SaaS) sağlayan çözümlerdir. Bu tarz çözümlerde veriler akıllı cihazlardan toplanır, buluta gönderilir, bulutta yer alan sunucularda analiz edilir ve güvenlik raporları üretilir.

Anti-virüs yazılımları işletim sistemini sürekli olarak tarayan, analiz eden bir yapıya sahiptir. Bu nedenle sistemi yorar ve yavaşlatırlar. Bunun yanında farklı anti-virüs yazılımları farklı teknikler kullanarak kötücül yazılımları tanırlar ve farklı raporlar üretirler. Bir kötücül yazılım bir anti-virüs programı tarafından tanınırken, diğer bir anti-virüs programı tarafından tanınamayabilir. Oberheide ve diğ. [73] tarafından yapılan çalışmada, araştırmacılar bu farklılıkları ortadan kaldırmak, sisteme binen yükü azaltmak için "Servis olarak Anti-virüs" fikrini ortaya atmışlardır. Bu fikre göre taranması gereken uygulamalar ya da dosyalar bulut ağına gönderilmekte, bulutta sanal makinelerde yer alan birden fazla anti-virüs programı da bu dosyaları paralel olarak incelemekte ve raporlayarak kullanıcıya sunmaktadır. Paralel olarak çalıştırılan anti-virüs programlarından gelen sonuçların da ortalaması alınarak başarı oranı çıkarılmıştır. Elde edilen sonuçlara göre uygulanan sistem tek anti-virüs programının başarısından daha iyidir ve kullanıcıların bilgisayarlarındaki yükü azaltmaktadır. Ancak birden fazla anti-virüs yazılımının kullanılması hem lisanslama problemlerine hem de mali

problemlere yol açacaktır. Bunun yanında şirketlerin tek bir anti-virüs firması ile anlaşması ve politika gereği başka anti-virüs programı kullanamamaları bu çözümü mümkün kılmamaktadır. Bu çalışma temelde kişisel bilgisayarlara yönelik bir çalışmadır. Mobil ortamlar için bir çözüm sunmamaktadır. Bu nedenle bizim çalışmamızdan farklılık göstermektedir. Ancak cihazdan bağımsızlaşarak bulut ortamında analiz yapılması, sonuç üretilmesi ve kullanıcıya sonuçların gösterilmesi yönüyle geliştirdiğimiz sisteme ışık tutmuştur.

Shabtai ve diğ. [74] tarafından yapılan çalışmada, Android tabanlı mobil cihazlardaki kötü amaçlı yazılımları önlemek için bir yazılım çatışı geliştirilmiştir. Yazılım mobil cihazları sürekli izleyip, aldığı verileri Makine Öğrenmesi teknikleri kullanarak zararlı ya da zararsız olarak kategorize etmektedir. Çalışmada üretilen yazılım Android cihazlara yüklenerek CPU kullanımı, Wi-Fi üzerinden gönderilen paket miktarı, arka planda çalışan iş parçacıkları sayısı, pil durumu gibi veriler gerçek zamanlı olarak toplanmaktadır. Toplanan veriler analiz edilerek tehdit değerlendirilmesi yapılmaktadır. Yapılan tehdit değerlendirilmesi sonrasında kullanıcıya uyarı gösterilmektedir. Kullanıcıya uyarıyla birlikte uygulamayı silme, arka planda çalışmayı durdurma, cihazı kilitleme gibi seçenekler de sunulmaktadır. Bu çalışmada araştırmacılar geliştirdikleri yazılımın çalışması için mobil bir cihaza ihtiyaç duymaktadırlar ve geliştirilen yazılım arka planda sürekli veri toplayarak, analiz yapmaktadır. Yapılacak analiz için de bir eğitim süreci gerekeceği için bu da sistem kaynaklarının aşırı tüketilmesine neden olmakta ve mobil cihazı yormaktadır. Ayrıca zararlı yazılımın tespit edilmesi için yazılımın mobil cihaza yüklenmiş olması gerekmektedir. Gerçekleştirmiş olduğumuz sistemde ise cihaza herhangi bir yazılım yüklenmediği için sistem cihazdan bağımsız çalışmaktadır. Bu nedenle de kullanıcı cihazını kullanırken herhangi bir performans kaybına uğramayacaktır.

Mobil tabanlı anti-virüs programları belli ölçüde mobil cihazları korusa da sınırlı oranda işlem ve hafıza kapasitesi olan bu cihazlara aşırı yük bindirmekte, cihazları yavaşlatmakta ve cihazların şarj seviyesini hızlıca düşürmektedir. Portokaladis ve diğ. [75] tarafından yapılan çalışmada araştırmacılar bu durumları göz önünde bulundurarak Paranoid Droid isminde güvenlik kontrolünü buluta kaydıran bir sistem geliştirmişlerdir. Sistem mobil cihazların çalışması sırasındaki izleri (çalışan işlemler

düzeyinde) minimal düzeyde kaydederek uzak bilgisayarlardaki sanal makinelere şifreli bir şekilde gönderip çoklu güvenlik denetimi yapabilmektedir. Geliştirilen sistem prototip aşamasında olup 2 tip güvenlik denetimi yapmaktadır. 1. tip denetim dinamik kod analizi yaparak kod enjeksiyonu ve bellek aşırı yükleme (buffer overflow) denetimidir. 2. tip denetim ise dosya denetimidir. Bu denetimde açık kaynak kodlu bir anti-virüs programı kullanılarak dosyalar önce taranmakta ve daha sonra kullanıma sunulmaktadır. Bu sistem belli bir düzeyde güvenlik sağlasa da mobil cihazlarda oluşturulan izlerin şifrelenerek uzak bilgisayara gönderilmesiyle gelen yük fazladır. Bu da %30 oranında şarjı azaltmaktadır. Bunun yanında verilerin 3G bağlantısı ile gönderilmesi kullanıcıya maliyet açısından da yük bindirecektir ve verilerin uzak bilgisayar gönderilmesi için Wi-Fi bağlantısının aktif olması beklenecektir.

Tesfay ve diğ. [76] tarafından yapılan çalışmada, araştırmacılar itibar (reputation) tabanlı bir güvenlik mekanizması geliştirmişlerdir. Kullanıcıya uygulamayı yüklemeden önce uygulamanın bulut ortamında hesaplanmış itibar puanı gösterilir. İtibar puanı düşükse kullanıcı uyarılır, değilse uygulama yüklenir. İtibar puanı hesaplanırken de aynı uygulamayı kullanan ya da kullanmış kullanıcıların geri beslemesinden yararlanılmaktadır. Bulut ortamında çalışan sistem tüm kullanıcılardan aldığı geri besleme değerlerine göre itibar puanını hesaplamaktadır.

Oberheide ve diğ. [77] tarafından yapılan bu çalışma, bir önceki çalışmalarının [73] geliştirilmiş ve mobil ortamlara uyarlanmış halidir. Cihaza bağımlı olarak çalışan anti-virüs programlarındaki virüs tarama ve bulma mantığını buluta kaydırmışlardır. Sistem 2 modülden oluşmaktadır. 1. modül bulut ortamında sanallaştırılmış makinelerde çalışan anti-virüs programlarından oluşmaktadır. 2. modül ise mobil cihazlarda çalışan ve virüs ya da zararlı yazılımların taranması için gerekli dosyaları buluta gönderen modüldür. Araştırmacılar bu şekilde bir yaklaşımla çok fazla CPU ve bellek gerektiren işlemleri mobil cihazlardan ayırarak daha performanslı bir virüs tarama ve bulma mekanizması önermişlerdir.

2.5.2. Donanım Tabanlı Çözümler

Donanım tabanlı çözümler geliştirilme aşamasında olup Güvenilir Platform Modülü (TPM) [78] kullanımını hedef almaktadır. TPM dizüstü ve masaüstü bilgisayarlarda sistemin bütünlüğünü, zararlı yazılımların sistemde değişiklik yapıp yapmadığını

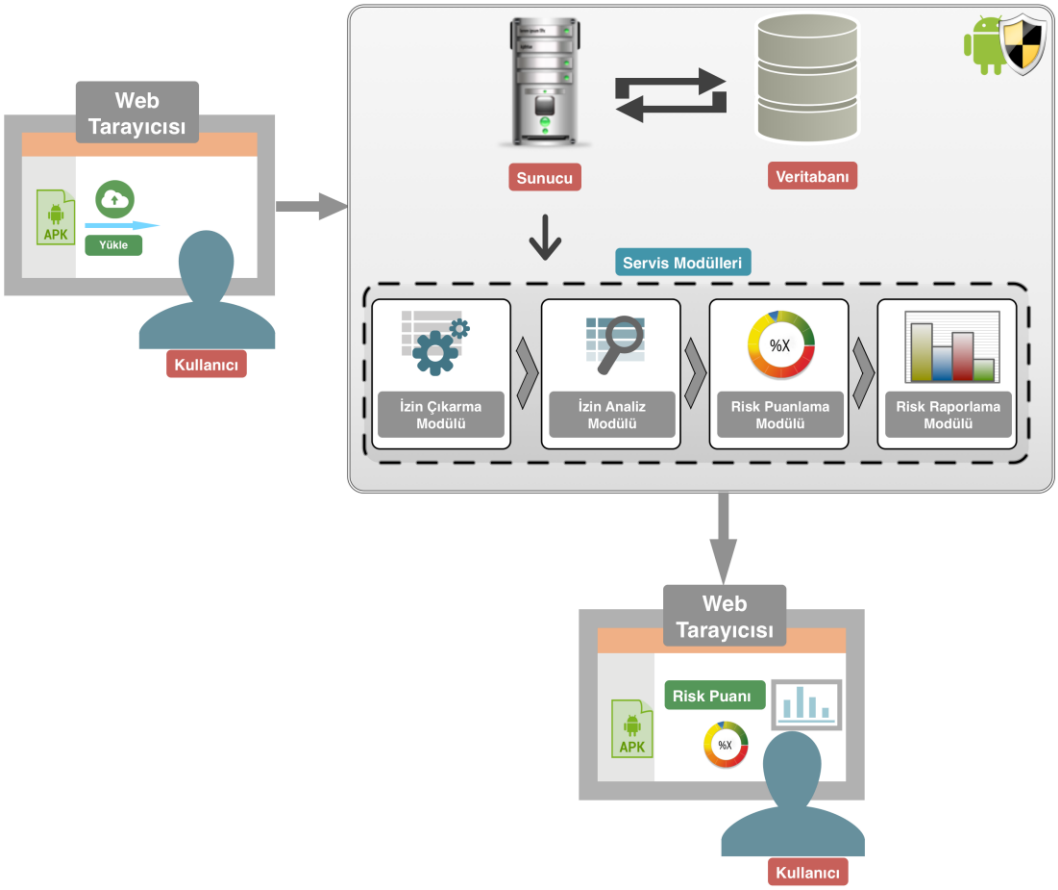
kontrol etmek amacıyla yıllardır kullanılan bir teknolojidir. Ancak bu sistem henüz mobil cihazlarda mevcut değildir. Güvenilir İşletim Grubu (Trusted Computing Group, TCG) bu sistemin mobil cihazlarda kullanılması için standartları belirlemeye çalışmaktadır [79]. Şu an için donanımsal olarak bir çip bulunmamasına rağmen, TPM çipinin yaptığı işi yazılım düzeyinde yapan hem PC tabanlı hem de mobil tabanlı öykünücüler (emulator) mevcuttur. Nauman ve diğ. [80] tarafından yapılan çalışmada Android cihazlarda hem performanslı çalışacak hem de Android sisteminin bütünlüğünü ölçecek bir öykünücü ve bütünlüğünü onaylayacak bir sistem geliştirilmiştir. Geliştirilen sistem hem platformun hem de yüklü uygulamaların bütünlüğünü ölçebilecek düzeydedir. Uygulamaların bütünlüğü ölçülürken, uygulamanın içinde yer alan tüm sınıfların bütünlüğü de ayrı ayrı ölçülmektedir. Uygulamaların ve sistemin bütünlüğü ölçüldükten sonra uzakta çalışan sistem cihazdan aldığı bilgilere göre onaylama işlemi yapmaktadır. Bu çalışma TPM cihazlarının mobil ortamlara nasıl entegre edileceğini ve mobil cihazların güvenliğinin bu tip bir sistemle nasıl korunacağını gösteren ilk çalışma olması nedeniyle önem taşımaktadır.

3. MALZEME VE YÖNTEM

Bir yazılım sistem mimarisi; sistemin performansını, dayanıklılığını, dağıtılabilirliğini ve bakım kolaylığını doğrudan etkilemektedir. Dolayısıyla, yazılım geliştirme aşamasına geçilmeden önce sistemi oluşturan alt modüller ve bu modüller arasındaki akışın iyi tanımlanması gerekmektedir.

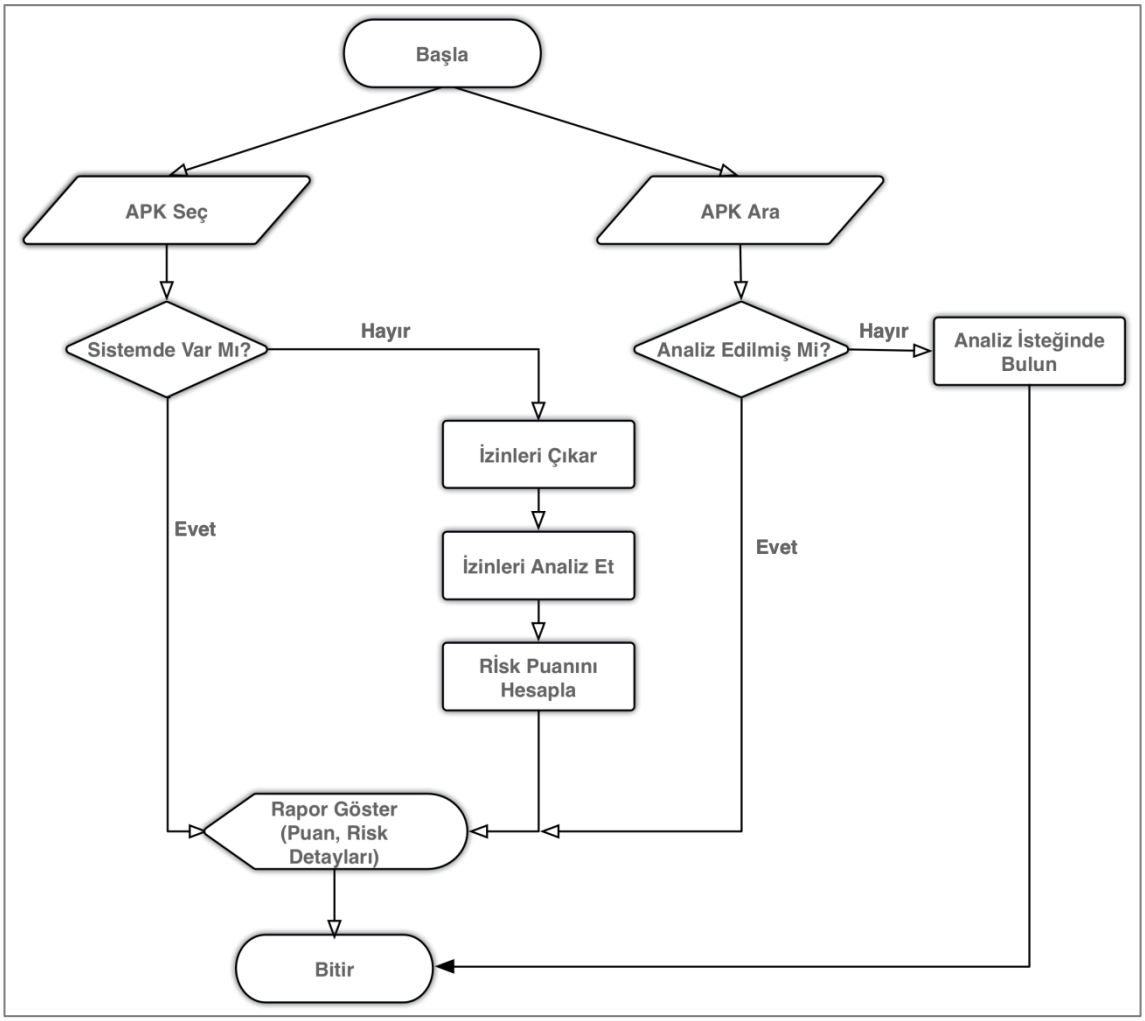
3.1. SİSTEM MİMARİSİ

Mobil uygulamaların güvenlik risklerini analiz etmek amacıyla geliştirdiğimiz servisin; kullanıcıların gereksinimlerini ne şekilde karşıladığını, sistemin işleyişini ve içerdiği modüllerin birbirleriyle olan ilişkilerini açıklamak için bir sistem mimarisi tasarladık. Şekil 3.1, bu mimariyi göstermektedir.



Şekil 3.1: Mobil uygulama güvenlik risk analiz servisine ait sistem mimarisi.

Şekil 3.2, sistemimizin nasıl işlediğini açıklayan akış diyagramını göstermektedir. Tasarlanan mobil güvenlik risk analiz servisi 4 ana modülden oluşmaktadır. Bunlar; *izin çıkarma, izin analiz, risk hesaplama* ve *risk raporlama* modülleridir. Bu modüller bir sonraki bölümde detaylı olarak açıklanmıştır.



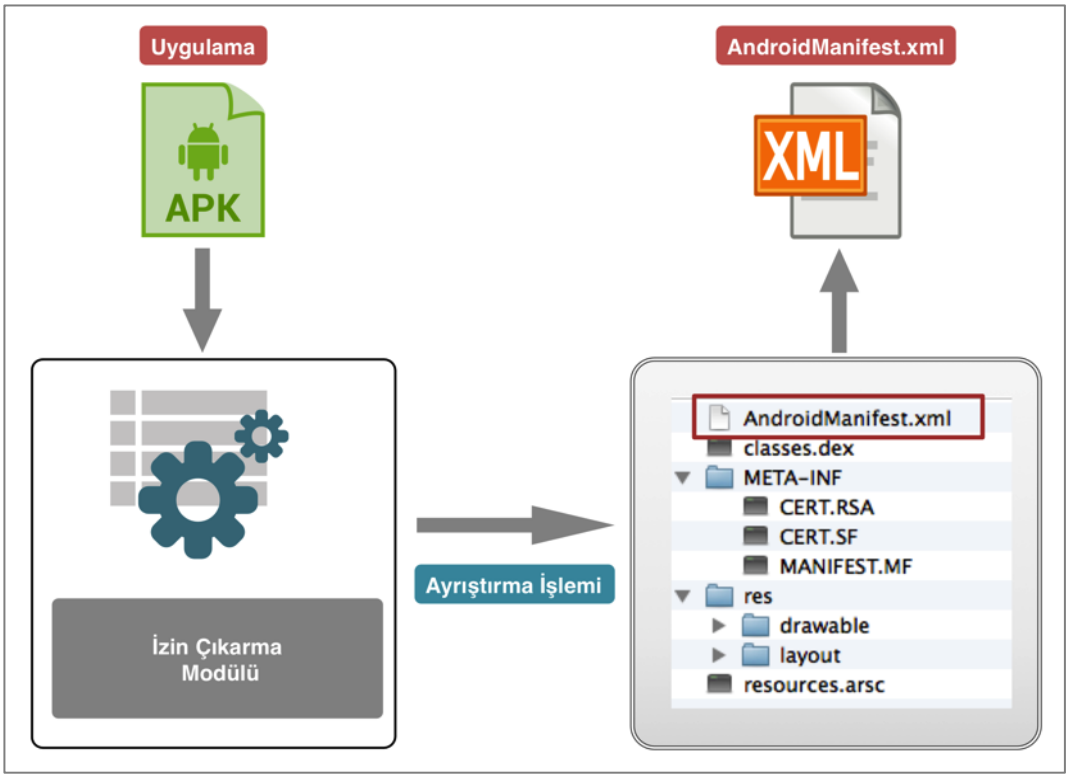
Şekil 3.2: Sistem akış diyagramı.

3.2. SİSTEM MODÜLLERİNİN TASARIMI

3.2.1. İzin Çıkarma Modülü

Android uygulamaları genellikle Java programlama dili kullanılarak yazılır ve Dalvik sanal makinesi üzerinde çalışırlar. Uygulamalar ".apk" uzantılı tek bir dosya aracılığı ile yüklenirler. APK dosyalarının içlerinde izinlerin bulunduğu "AndroidManifest.xml" olarak adlandırılmış bir manifesto dosyası yer almaktadır. Bu dosya sisteme tüm üst düzey bileşenler ile (aktiviteler, servisler, içerik sağlayıcılar, yayın alıcılar) ne yapılması

gerektiğini söyleyen bir kontrol dosyasıdır. Bu modülün amacı risk puanlamasının yapılabilmesi için gerekli olan ve derlenmiş uygulama dosyasında yer alan izinleri uygulamadan ayrıştırmaktır. Kullanıcının tarayıcı yardımıyla sisteme yükleyeceği ya da arama modülü aracılığı ile aratacağı ".apk" uzantılı uygulama dosyası bu modüle girdi olarak verilir. Sisteme başarıyla yüklenen ya da arama sonucunda getirilen uygulama dosyasındaki izin listesi bu modülün çıktısıdır. Sonuç olarak ayrıştırılan izin listesi izin analiz modülünün işleyebilmesi için hazır hale getirilir ve izin analiz modülüne girdi olarak sunulur. Sistemde aynı uygulamanın aynı sürümünün tekrar yüklenmesini engellemek için her bir uygulamaya bir kimlik atanır, Uygulama-İzinler ilişkisi oluşturularak veritabanında saklanır. Şekil 3.3 bu modülün işleyişini göstermektedir.



Şekil 3.3: İzin çıkarma modülü işleyişi.

3.2.2. İzin Analiz Modülü

İzin çıkarma modülünde elde edilen APK dosyasındaki AndroidManifest.xml dosyası Şekil 3.4'deki gibi ikili (binary) formatta, içeriği anlaşılamayan XML formatında bir dosyadır. Bu dosyanın okunabilir, klasik XML dosyasına çevrilmesi gerekmektedir.

AndroidManifest.xml

```
0300 0800 1409 0000 0100 1c00 cc04 0000
1d00 0000 0000 0000 0000 0000 9000 0000
0000 0000 0000 0000 1a00 0000 3400 0000
4000 0000 5a00 0000 6600 0000 7400 0000
8200 0000 9a00 0000 ac00 0000 0401 0000
0801 0000 1a01 0000 2e01 0000 6601 0000
7001 0000 9201 0000 d601 0000 2802 0000
6202 0000 b802 0000 fe02 0000 1803 0000
2c03 0000 7e03 0000 9c03 0000 ac03 0000
e403 0000 f803 0000 0b00 7600 6500 7200
7300 6900 6f00 6e00 4300 6f00 6400 6500
0000 0b00 7600 6500 7200 7300 6900 6f00
6e00 4e00 6100 6d00 6500 0000 0400 6e00
6100 6d00 6500 0000 0b00 6100 6c00 6c00
6f00 7700 4200 6100 6300 6b00 7500 7000
0000 0400 6900 6300 6f00 6e00 0000 0500
6c00 6100 6200 6500 6c00 0000 0500 7400
6800 6500 6d00 6500 0000 0a00 6400 6500
6200 7500 6700 6700 6100 6200 6c00 6500
0000 0700 6100 6e00 6400 7200 6f00 6900
6400 0000 2a00 6800 7400 7400 7000 3a00
2f00 2f00 7300 6300 6800 6500 6d00 6100
7300 2e00 6100 6e00 6400 7200 6f00 6900
6400 2e00 6300 6f00 6d00 2f00 6100 7000
```

Şekil 3.4: İkili formatta AndroidManifest.xml dosyası.

Bu modül iki aşamada çalışmaktadır. Birinci aşamada ikili formattaki XML dosyası klasik okunabilir bir XML dosyasına dönüştürülür. Şekil 3.5 klasik yapıya çevrilmiş AndroidManifest.xml dosyasını göstermektedir.

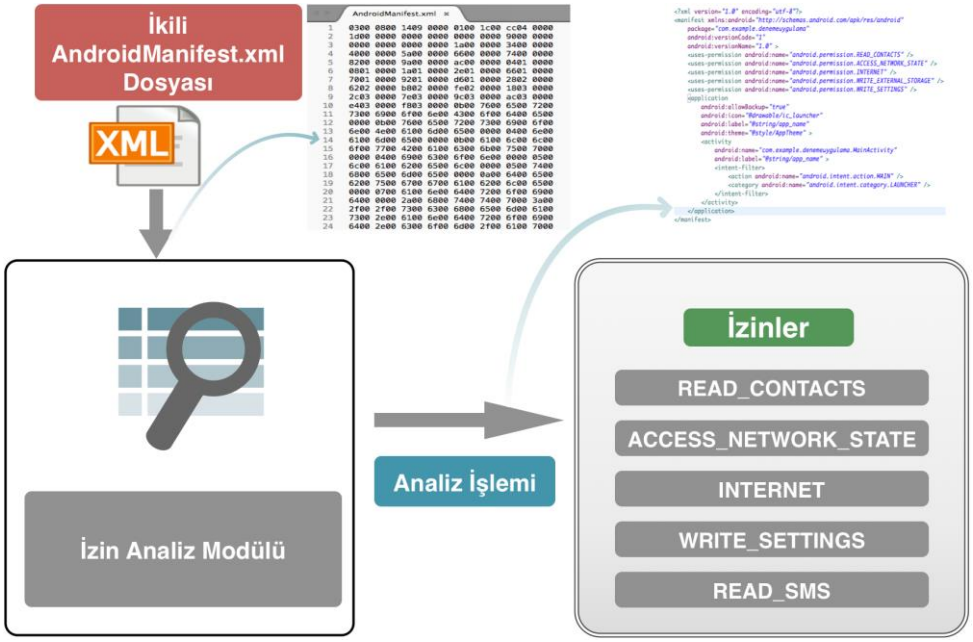
```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="com.example.denemeuygulama"
    android:versionCode="1"
    android:versionName="1.0" >
    <uses-permission android:name="android.permission.READ_CONTACTS" />
    <uses-permission 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.WRITE_SETTINGS" />
    <application
        android:allowBackup="true"
        android:icon="@drawable/ic_launcher"
        android:label="@string/app_name"
        android:theme="@style/AppTheme" >
        <activity
            android:name="com.example.denemeuygulama.MainActivity"
            android:label="@string/app_name" >
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />
                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>
</manifest>
```

Şekil 3.5: Klasik, okunabilir AndroidManifest.xml dosyası.

Şekil 3.5'de görüldüğü üzere kullanılan tüm izinler "uses-permission" etiketi ile belirtilmiştir. İkinci aşamada ise bu etiket bilgisi kullanılarak izinler dosyadan ayrıştırılır. Bu modül sonunda çıktı olarak ayrıştırılan bilgi;

(READ_CONTACTS,ACCESS_NETWORK_STATE,İNTERNET,WRITE_EXTERNAL_STORAGE, WRITE_SETTINGS)

risk puanlama modülünün girdisi olacaktır. Şekil 3.6, bu modülün işleyişini göstermektedir.



Şekil 3.6: İzin anliz modülü işleyişi.

3.2.3. Risk Hesaplama Modülü

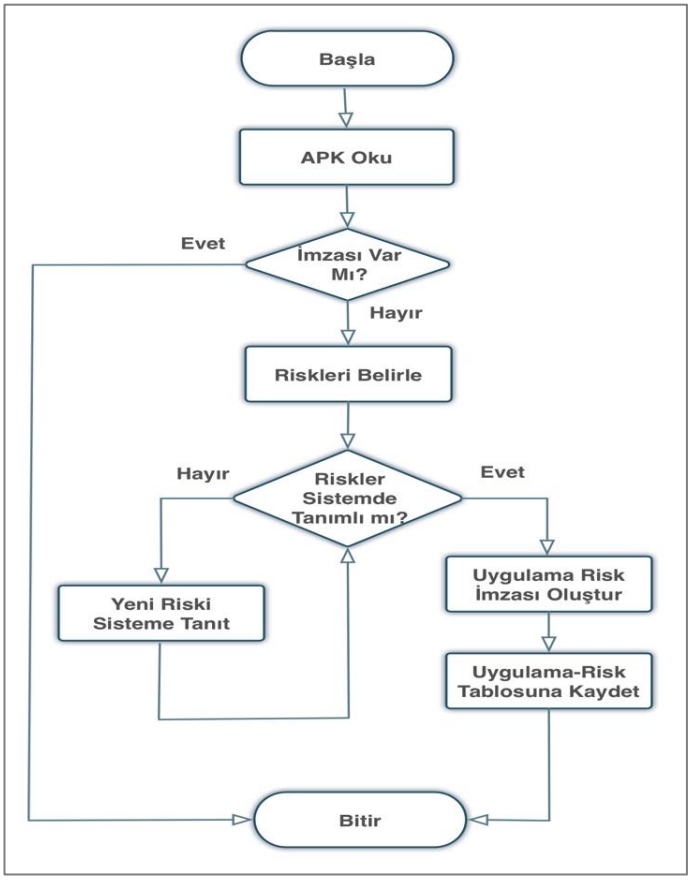
Risk puanı belirlenmeden önce yapılacak ilk adım, risk analizi yapmaktır. Bu tür analizler ISO27001 bilgi güvenliği standarlarında da belirlenmiştir ancak belirli bir risk analiz yöntemi önerilmemiştir. Bu standartlarda risk analiz sürecinin zorunlu olduğu ve "risk değerlendirmesine yönelik sistematik bir yaklaşım" tanımlanması gerektiği belirtilmiştir [81].

Risk, bilinen ve sabit olan somut bir değer değil, değişebilen bir olasılık değeridir. Dolayısıyla risk analizi olasılığa dayanan bir hesap türüdür [82]. Konu bilgi ve iletişim teknolojileri olduğunda risk analizi sürecinin karmaşıklığı artmaktadır. Bilgi ve iletişim teknolojileri açısından risk sadece basit bir olasılık değeri değildir. Risk mevcut olan bir açıklığın bir tehdit tarafından kullanılıp kullanılamama olasılığıdır. Sonuç olarak risk üç temel girdiye bağımlıdır. Bunlar varlık, açıklık ve tehdittir. Bir başka ifadeyle risk bu üç temel girdinin bir fonksiyonudur. Çıktı ise risk değeridir ve Denklem 3.1'deki gibi tanımlanır [83].

$$R=F(varlık,\ açıklık,\ tehdit) \tag{3.1}$$

Bu model mobil uygulamalardaki risk analizine uyarlandığında; "varlık" kullanıcıların mobil cihazlarını, "açıklık" izin sistemini, "tehdit" ise kişisel güvenlik ve gizliliği temsil etmektedir.

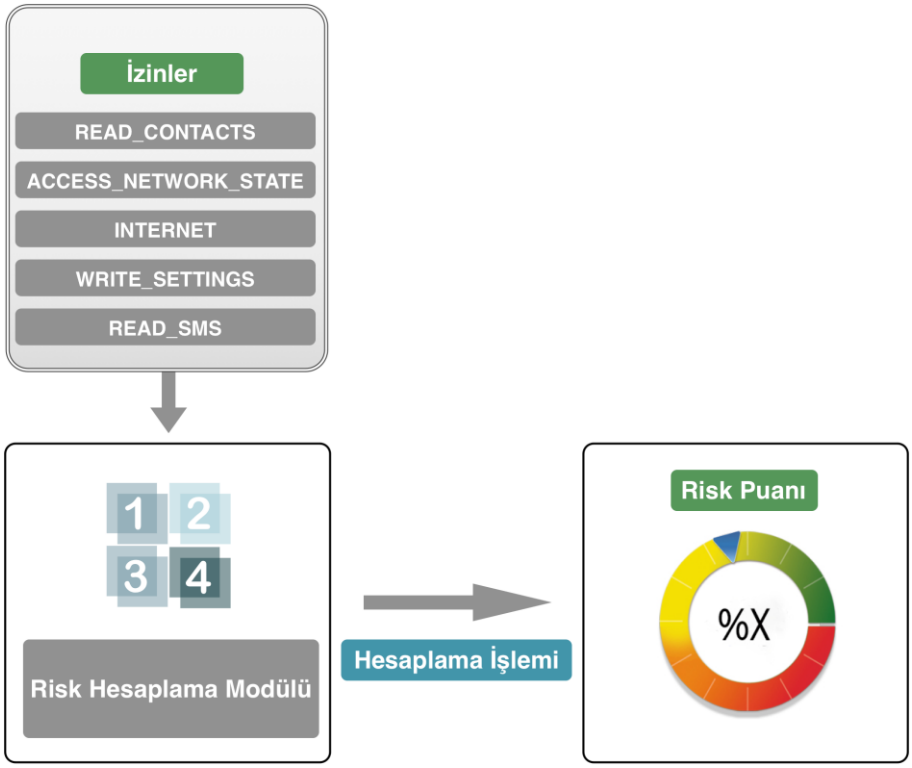
Geliştirdiğimiz risk analiz algoritmamız bu üç temel girdi üzerine inşa edilerek modellenmiştir ve risk puanlamasını yapacak algoritma için girdi üreterek ön-işleme sürecini başlatacaktır. Algoritma sisteme yüklenecek ya da mağazada aranacak ".apk" uzantılı uygulama dosyaları üzerinde çalışmaktadır. Okunan dosyanın imzasının veri tabanında yer alması sistemde zaten var olduğunu gösterir. İmzası bulunamayan uygulamalar için tüm riskler (uygulama izinleri) belirlenerek sistemimizde tanımlı olan risklerle karşılaştırılır. Sistemdeki risklerden farklı bir riskle karşılaşıldığında, bu risk yeni bir risk (uygulama izini) olarak sistemimize eklenir. Eğer tüm riskler sistemde tanımlı ise uygulama için benzersiz bir risk imzası (kimlik) oluşturulur. İmzası oluşan her uygulama Uygulama-Risk tablosuna kaydedilerek risk puanı hesabı için hazır hale getirilir. Şekil 3.7, risk analiz işlemine ait akış diyagramını göstermektedir.



Şekil 3.7: Risk analiz işlemi.

Risk, bir olayın istenmeyen biçimde sonuçlanma ihtimalidir. Bir riskin ortaya çıkabilmesi için bir tehlike olmalıdır ve değer verilen bir şey bu tehlikeye maruz kalmalıdır. Risk analizi ve risk hesaplama aşamasında öncelikle sistemdeki tehlikeler belirlenmelidir. Bizim ele aldığımız durumda tehlike, kullanıcıların zararlı yazılımları

mobil cihazlara yüklemesiyle ortaya çıkacak kişisel güvenliğe ve gizliliğe yönelik tehlikelerdir. Bu bağlamda değer verilen şeyler mobil cihazlar, kişisel güvenlik ve gizliliktir. Bu tehlikenin asıl kaynağı ise yüklenecek uygulamalardaki izinlerin bilinçsizce kabul edilip kullanılmasıdır. Bu modülde bir önceki modülde elde edilen izinler kullanılarak her bir izinin olasılık ve etkilerinin değerlendirilmesiyle risk şiddeti ortaya çıkarılır. Elde edilen risk puanlarına göre de sisteme yüklenen uygulama "% X" riskli şeklinde sınıflandırılır. Burada uygulama risk değeri 100'e ne kadar yakın ise uygulamanın riski de o kadar fazla olacaktır. Her yüklenen uygulamaya atanacak benzersiz bir kimlik yardımı ile de Uygulama-Risk Puanı bilgisi veri tabanında kaydedilir. Şekil 3.8 bu modülün işleyişini göstermektedir.



Şekil 3.8: Risk hesaplama modülü işleyişi.

3.2.3.1.Risk Puanlama İşlemi

Yapılan literatür taramasında, risk analizi konusunda bulanık mantık tabanlı matematiksel yöntemlerin çok sık kullanıldığı ve etkili sonuçlar elde edildiği görülmüştür. Fu ve diğ. [84] tarafından yapılan çalışmada bulanık mantık kullanılarak risk değerlendirmesi yapan bir metot geliştirilmiştir. Risk analiz aşamasında anlatmış olduğumuz risk girdilerinin benzeri bu çalışmada da kullanılmıştır ve risk 3 temel girdiye dayandırılmıştır. Bunlar varlık, tehdit, sistemin dayanma gücüdür. Geliştirilen

algoritma bir örnek senaryo üzerinde uygulanmış, kullanılan güvenlik risk değerlendirme tekniğinin de etkili ve geçerli bir yöntem olduğunu ispatlamışlardır.

Shameli-Sendi ve diğ. [85] tarafından yapılan çalışmada bilgi güvenliği risk değerlendirmesi için bulanık mantık tabanlı bir karar verme mekanizması geliştirilmiştir. Önerilen modelin etkinliğini doğrulamak için model bir tedarik zinciri yönetim firmasının bilgi işlem bölümüne uygulanmıştır. 9 farklı tehlike alanı ve 91 adet tehdit belirlenmiştir. Bunun yanında tehditlerin derecelendirilmesi için (riskli, çok riskli vs.) uzman görüşlerinden de yararlanılmıştır.

McGill ve diğ. [86] ise yapmış oldukları çalışmada yine risk 3 girdili bir fonksiyon olarak tanımlanmıştır. Çok miktarda kriter içeren güvenlik sistemlerinin performansını değerlendiren bulanık mantık tabanlı bir model geliştirilmiştir. Önerilen sistem sadece öneri aşamasında kalmış olup gerçekleştirilmemiştir.

Liu ve diğ. [87] yapmış oldukları çalışmada kablosuz ağlardaki (WLAN) güvenliği değerlendiren SAEW adında bulanık mantık tabanlı bir sistem geliştirmişlerdir. Geliştirilen sistem altı bölümden oluşmaktadır: WLAN risk indeks girdileri, bulanık kural tabanı, bulanık çıkarım motoru, bulanıklaştırıcı, durulaştırıcı ve WLAN güvenlik değerlendirme raporları. Elde edilen sonuçlara göre WLAN'ın güvenliğinde gelişme sağlanmıştır. Benzer bir şekilde Sendi ve diğ. [88] çalışmalarında bilgi güvenliği yönetimi alanında bir organizasyondaki risklerin değerlendirmesinde FEMRA adında bulanık mantık tabanlı uzman bir sistem geliştirmiştir. Geliştirilen sistem risk için numerik bir değer oluşturmakta, bir organizasyon içinde yüksek risklerin ortaya çıkarılması, risklerin karşılaştırılması ve azaltılması konusunda yöneticilerin uygun stratejiler geliştirmesine yardımcı olmaktadır.

Bulanık mantık tekniği Bilgisayar Bilimleri alanı dışında Çevre Mühendisliği, İnşaat Mühendisliği, Enerji Mühendisliği gibi alanlarda da çeşitli risklerin değerlendirilmesinde etkili bir şekilde kullanılmaktadır. Jamshidi ve diğ. [89] çalışmalarında gaz boru hatlarının risk değerlendirmesinde kullanılmak üzere bulanık mantık tabanlı bir çıkarım sistemi geliştirmişlerdir. Gerçek hayattan alınan verilerden risk girdileri, uzman görüşlerine dayanarak da bulanık kurallar oluşturulmuştur. Araştırmacılar önerdikleri sistemden gaz boru hatlarının risklerine ait kesin ve doğru

sonuçlar elde etmişlerdir. Benzer bir çalışmada Carr ve diğ. [90] elektrik şebekelerinin güvenlik risklerini gerçek zamanlı olarak değerlendiren bulanık mantık tabanlı bir sistem geliştirmişlerdir. Sistem güç kontrol yönetimi odasındaki kararlılığın artırılmasında, sistemin güvenliği konusunda önemli riskle karşılaşıldığında kontrol operatörünün uyarılmasında kullanılmıştır.

Tüm bu çalışmalar ışığında risk puanlama işlemi için bulanık mantık tekniğinin uygun, etkili ve kullanışlı olduğu, mobil güvenlik değerlendirmesinde de kullanılabileceği öngörülmüştür. Bu nedenle güvenlik risklerinin puanlanmasında kullanacağımız algoritma bulanık mantık tabanlı bir çözüm sunacak şekilde tasarlanmıştır.

3.2.3.2. Bulanık Mantık Temelleri

Bulanık mantık öznel (kişinin bakış açısına bağlı) bir bilgiyi daha tarafsız, nesnel bir bilgiye dönüştürmeyi sağlayan bir tekniktir. Çok geniş bir uygulama alanı olan başarılı bir yöntemdir. Aşağıda bulanık mantığın bazı avantajlarına yer verilmiştir.

- Risk analiz süreci insanların yaklaşık değerlere ulaşmasını içeren bir süreçtir. Bulanık mantık bu yaklaşık değerlerin sezgisel olarak tanımlanmasını kolaylaştırır. Sezgisel değerleri (çok riskli, az riskli vs.) gerçek değerlere dönüştürür [91].
- Bulanık mantık risk analizinde var olan öznelliğin hesaplanmasında kullanılır ve risk düzeyinin kabul edilebilir olup olmadığını belirler [94]. Bulanık mantık öznelliği daha nesnel, tarafsız yapar.
- İnsanlar genelde nümerik değerler yerine metine dayalı ifadeler (düşük, orta, yüksek) kullanmayı tercih ederler. Bulanık mantık insanların düşünme şeklinin modellenmesini sağlar [95].

Bulanık mantığın 3 temel taşı vardır. Bunlar:

- *Bulanık kümeler*
- *Üyelik fonksiyonları*
- *Bulanık kurallar*

Bulanık kümeler: Bir bulanık küme Denklem 3.2'deki formüldeki gibi tanımlanır. Üretilen değer 0-1 arasında olup, A bulanık kümeyi, X evrensel kümeyi ve x ise 0-1 arasında bir değere dönüştürülecek parametreyi belirler.

$$\mu_A(x): X \rightarrow [0,1] \tag{3.2}$$

Bulanık bir kümeye dahil olan bir elaman kısmi olarak o kümeye üye sayılır ve kümeye aitliğinin bir derecesi vardır. Üyelik fonksiyonunun değeri bire eşitse x elemanı bulanık kümeye tam olarak ait olmuş olur. Bu değerin sıfır olması durumunda, x bulanık kümeye ait değildir. Eğer ortaya çıkan değer sıfır ile bir arasında ise x kısmi üyedir.

Üyelik fonksiyonları: Klasik bir kümenin kesin sınırları vardır. Örnek verecek olursak:

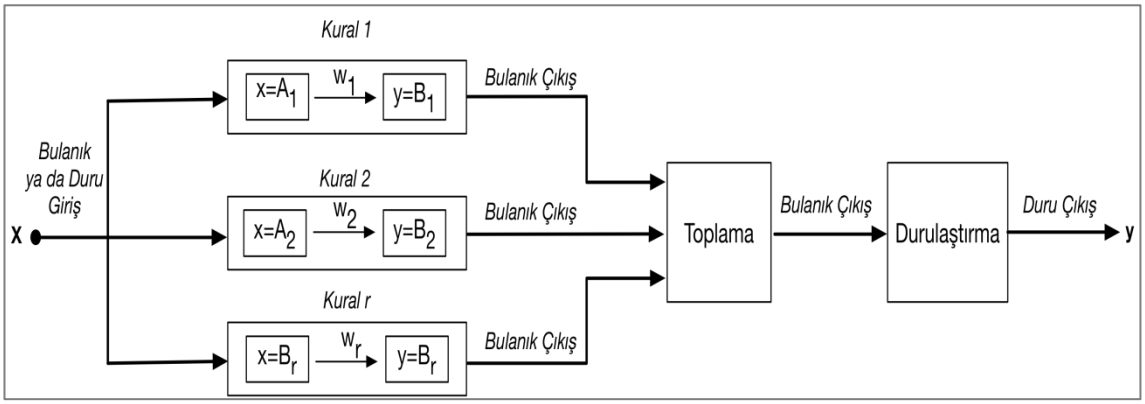
$$B = \{ x \mid 0 < \mathrm{x} < 10 \} \tag{3.3}$$

Denklem 3.3'de görüldüğü üzere sınırlar 0 ve 10'dur. Eğer x değeri sıfırdan büyük ve ondan küçükse x değeri B kümesine aittir. Bulanık kümelerin ise kesin bir sınırı yoktur. Ait olma durumu yerine, ne kadar ait olacağı durumu mevcuttur. Buna ek olarak kümeler "düşük risk " veya "yüksek risk" gibi tanımlar ile modellenerek esneklik sağlanır. Hangi dilsel tanıma ait olduğunu belirleyen bu fonksiyonlara üyelik fonksiyonları denir.

Bulanık kurallar: Bir riskin kullanıcıların güvenlik ve gizliliğini ne düzeyde etkileyeceğini bir şekilde kestirmek gerekmektedir. Bu amaç için bulanık mantık riskin etki düzeyini sezgiye dayalı metinsel kurallar ile hesaplamayı sağlar. Bu kurallara bulanık kurallar denir ve aşağıdaki örnekteki gibi tanımlanırlar.

Örnek: EĞER Kural1 ve Kural2 ve .... İSE uygulama riski X'dir.

Bulanık çıkarım sistemleri, yukarıda açıklanan üç temel üzerine oturtulmuş sistemlerdir ve bulanık küme teorisini, bulanık "eğer-ise" kurallarını kullanarak hesaplamalar yapmaktadırlar. Bu tarz sistemler, gerek bulanık gerekse duru girişleri kabul eder. Eğer girişler duru ise bulanıklaştırılır. Çıkışlar genelde bulanıktır ve gerçek dünyada kullanılabilmesi için durulaştırılırlar. Şekil 3.9, bir bulanık çıkarım sistemine ait genel yapıyı göstermektedir.



Şekil 3.9: Bir bulanık çıkarım sisteminin yapısı.

3.2.3.3.Risk Puanlama Süreci

Risk puanlama sistemi 6 adımdan oluşan bir süreç olarak tasarlanmıştır. Bu adımlar:

i. **Problemin belirlenmesi:** Önceki bölümlerde bahsedildiği üzere, riske neden olan problem (yani tehdit) izinlerdir. Kullanıcıların bilinçsiz bir şekilde uygulamalara izin vermesi güvenlik ve gizlilik risklerine neden olmaktadır. Dolayısıyla, riske neden olan izinlerin kabul edilmesi ana problemdir.

ii. **İzinlerin listelenmesi ve etkilerinin belirlenmesi:** Bu aşamada zararlı yazılımlar tarafından en çok kullanılan izinler bulunmuş ve etki değerleri hesaplanmıştır. Bunun için en kapsamlı veri seti olarak, Zhou ve diğ. [71], Arp ve diğ. [92], Spreitzenbarth ve diğ. [93] tarafından yapılan çalışmalarda kullanılan zararlı mobil yazılımlar veri seti kullanılmıştır. Bu veri setinde 179 farklı kategoride toplam 6000 adet zararlı yazılım mevcuttur. Yaptığımız analiz sonucunda, zararlı yazılımlar tarafından kullanılan 117 (Android işletim sisteminde tanımlı izinlerin 80%'i) izin tespit edilmiştir. Tablo 3.1; 117 izin arasından en çok kullanılan 30 izni, izinlerin kullanım miktarlarını ve risk olasılık değerlerini göstermektedir. İzin isimleri Android geliştirici sitesinden faydalanarak elde edilmiştir [57]. Risk olasılık değerlerinin hesaplanıp ölçeklendirilmesinde aşağıdaki denklem kullanılmıştır.

$$P_i = \frac{o_i}{\max(o)} * 100 \qquad (3.4)$$

P_i : 0-100 arasında ölçeklendirilmiş risk olasılık değeri.

o_i : i. İzinin tekrarlanma/kullanılma sıklığı.

$\max(o)$: Bir iznin kullanılabileceği maksimum miktar (Bizim durumumuzda bu değer maksimum izin miktarıdır ve 6000'dir).

Burada P_i değerinin 100'e daha yakın olması bu izinin riske neden olma olasılığının daha fazla olduğu anlamına gelmektedir. Tablo 3.1'den de görüldüğü gibi kullanılması halinde çok büyük olasılıkla riske neden olacak izin "İNTERNET", yani İnternetin kullanımını sağlayan izindir. İkinci sırada ise telefon durumunu almayı sağlayan "READ_PHONE_STATE" iznidir. İzin risk olasılıkları ve bunlara karşılık gelen dilsel değerler ise Tablo 3.2'de gösterilmiştir.

Tablo 3.1: Zararlı yazılımlar tarafından kullanılan çeşitli izinler.

İzinler	Kullanım Miktarları	Risk Olasılığı
İNTERNET	1219	97
READ_PHONE_STATE	1166	93
ACCESS_NETWORK_STATE	1022	81
WRITE_EXTERNAL_STORAGE	834	66
ACCESS_WIFI_STATE	804	64
READ_SMS	790	63
RECEIVE_BOOT_COMPLETED	688	55
WRITE_SMS	658	52
SEND_SMS	540	43
RECEIVE_SMS	486	39
VIBRATE	482	38
ACCESS_COARSE_LOCATION	480	38
READ_CONTACTS	457	36
ACCESS_FINE_LOCATION	432	34
WAKE_LOCK	425	34
CALL_PHONE	424	34
CHANGE_WIFI_STATE	398	32
WRITE_CONTACTS	374	30
WRITE_APN_SETTINGS	349	28
RESTART_PACKAGES	333	26
INSTALL_PACKAGES	251	20
DISABLE_KEYGUARD	247	20
READ_LOGS	240	19
GET_TASKS	217	17
ACCESS_LOCATION_EXTRA_COMMANDS	178	14
READ_HISTORY_BOOKMARKS	176	14
READ_EXTERNAL_STORAGE	174	14
SET_WALLPAPER	157	12
WRITE_HISTORY_BOOKMARKS	134	11
MOUNT_UNMOUNT_FILESYSTEMS	124	10

Tablo 3.2: İzin olasılık değerleri.

Olasılık Aralığı (%)	Dilsel Değer	Nümerik Değer
75-100	Tehlikeli (T)	4
50-75	Orta (O)	3
25-50	Normal (N)	2
0-25	Düşük (D)	1

iii. **Dilsel risk değerlerinin ve kategorilerinin belirlenmesi:** Bu adımda riske neden olan izinlerin dilsel ve sayısal olarak risk değerleri, kategorileri belirlenmiştir. Android sistemindeki izinlerin açıklamaları dikkate alınarak 4 adet risk kategorisi ve her bir kategori için dilsel ve numerik etki değerleri belirlenmiştir. Bunlar Tablo 3.3'de listelenmiştir. Bu dört risk kategorisi Android işletim sisteminde izinlere atanmış koruma düzeyleri dikkate alınarak oluşturulmuştur [57].

Tablo 3.3: Risk kategorileri ve etki değerleri.

Risk ID	Risk Kategorisi	Dilsel Değer	Etki Değeri
1	Tehlikeli Risk (TR)	Yüksek	25
2	İmza ya da Sistem Riski (İSR)	Orta	15
3	İmza Riski (İR)	Düşük	10
4	Normal Risk (NR)	Çok Düşük	5

Tablo 3.1'de verilen her bir izin bu kategorilerin birinde yer almaktadır. Tablo 3.2 ve Tablo 3.3'deki değerler kullanılarak Tablo 3.4'deki Risk-Etki-Olasılık Matrisi elde edilmiştir.

Tablo 3.4: Risk-Etki-Olasılık tablosu.

Olasılık	Etki			
	NR (5)	**İR (10)**	**İSR (15)**	**TR (25)**
T (4)	20	40	60	100
O (3)	15	30	45	75
N (2)	10	20	30	50
D (1)	5	10	15	25

Tablo 3.4, bulanık mantık tabanlı risk puanlama işleminde izinlerin risk değerlerinin belirlenmesinde kullanılacaktır. Her bir izinin etki ve olasılık değerleri çarpılarak "Risk Şiddeti" elde edilmiştir. Riskler aşağıdaki şekilde kategorize edilmiştir:

- **Yüksek Şiddetteki Riskler:** Kırmızı renk ile boyanmış alanlardır. Risk şiddet değeri 60 ile 100 arasında olan izinler bu kategoridir.

- **Orta Şiddetteki Riskler:** Mavi renk ile boyanmış alanlardır. Risk şiddet değeri 30 ile 60 arasında olan izinler bu kategoridedir.
- **Düşük Şiddetteki Riskler:** Yeşil renk ile boyanmış alanlardır. Risk şiddet değeri 0 ile 30 arasında olan izinler bu kategoridedir.

Tablo 3.4 aracılığıyla elde edilen değerlere göre Tablo 3.5'de zararlı yazılımlar tarafından en çok kullanılan 10 izin ve risk şiddeti değerleri gösterilmektedir. Tablo 3.5'den de görüldüğü gibi, kullanılması durumunda en çok riske neden olan izin "İNTERNET" izinidir.

Tablo 3.5: En çok kullanılan 10 izin ve risk şiddetleri.

İzinler	Risk Şiddeti
İNTERNET	Yüksek
WRITE_EXTERNAL_STORAGE	Yüksek
READ_SMS	Yüksek
READ_PHONE_STATE	Yüksek
WRITE_SMS	Orta
SEND_SMS	Orta
RECEIVE_SMS	Düşük
ACCESS_COARSE_LOCATION	Düşük
READ_CONTACTS	Düşük
ACCESS_FINE_LOCATION	Düşük

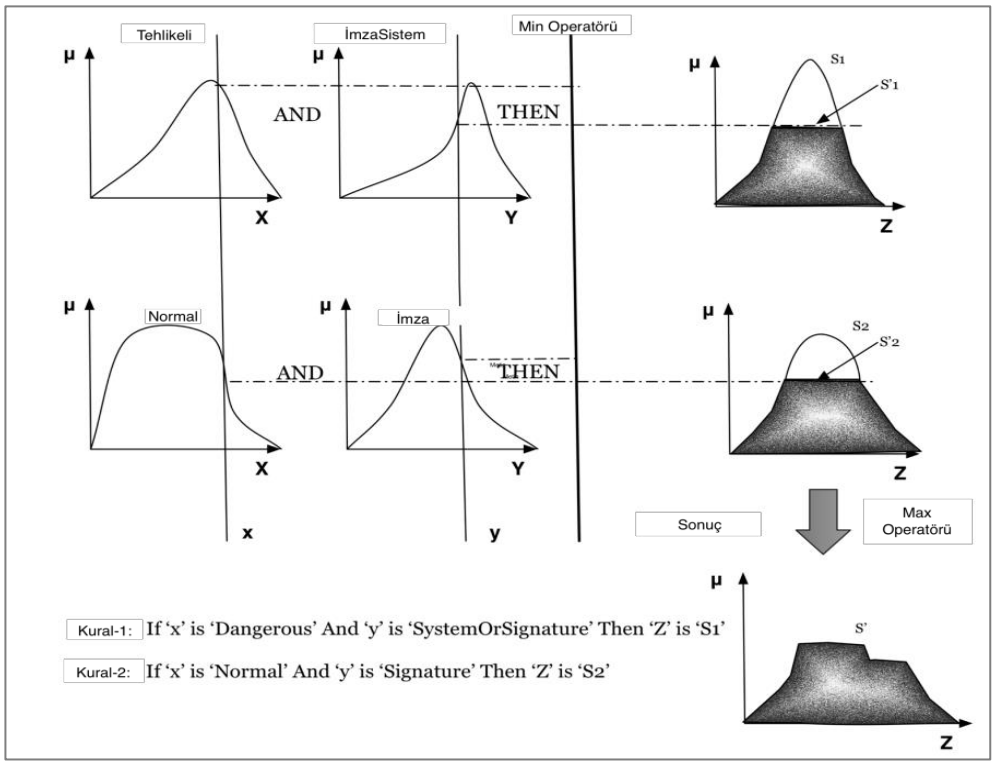
iv. **Kuralların Belirlenmesi:** Bu aşamada dilsel ve sayısal olarak belirlenen değerler ve risk kategorileri de dikkate alınarak bulanık kural tablosu oluşturulur. Kural tablosu bulanık kümeleri ve işlemleri koşullu ifadeler şeklinde tanımlar [94]. Kurallar yardımı ile tüm izinler incelenerek tek bir risk değeri çıkarılır.

Örnek kurallar:

- Eğer Tehlikeli İzin Düşük ve Sistem İzni Yüksek ve .... İSE risk X
- Eğer Normal İzin Yüksek ve İmza İzni Yüksek.... İSE risk Y

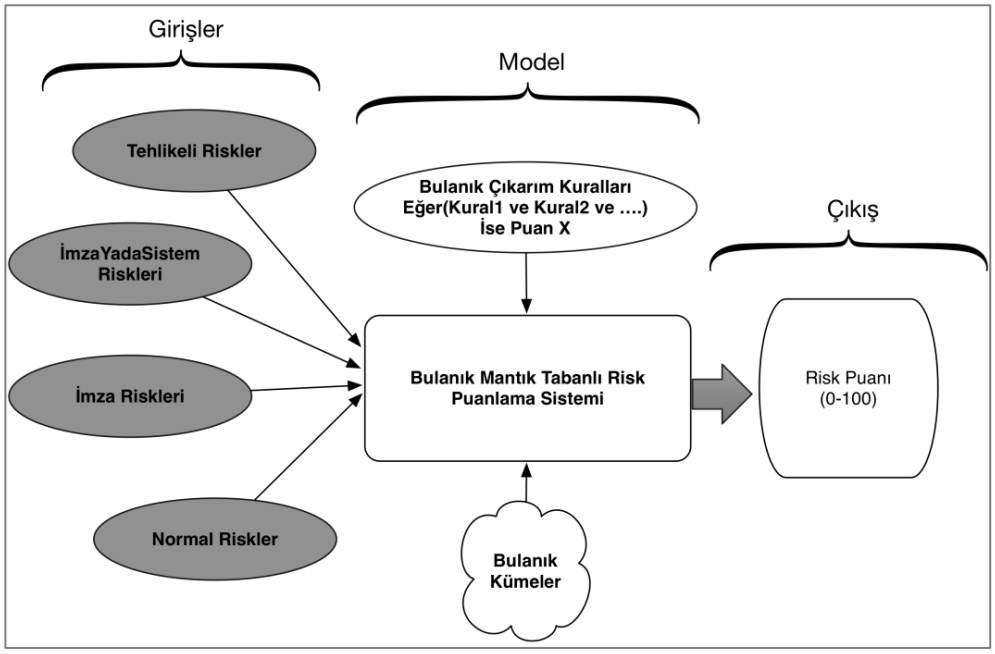
v. **Bulanık Çıkarım Yapılması:** Çıkarım yapmak için en çok kullanılan ve bilinen teknik olan Mamdani bulanık modeli kullanılmıştır. Şekil 3.10'da tasarlamış olduğumuz bulanık sistemin iki kuraldan oluşan bir örneği, giriş parametrelerine göre çıkışları verilmiştir [95]. Şekil iki girişten oluşan ve iki kuralı olan bir bulanık sistemi göstermektedir. Sistem şu şekilde işlemektedir: Önce, x ve y girişlerinin her

hangi bir andaki değerlerine ve tanımlanan üyelik fonksiyonuna göre girişlerin üyelik dereceleri belirlenir. Bu dereceler ***min*** operatöründen geçirilir. Bu işlem her bir kural için işletildiğinde kural sayısı kadar çıkış bulanık kümeleri elde edilir. Bu çıkış bulanık kümeler de ***max*** operatöründen geçirilir. Kesin değere ulaşmak için de çıkış bulanık kümesi durulandırma işleminden geçirilmelidir.



Şekil 3.10: Mamdani bulanık girişimli sistemi.

vi. Durulaştırma: Bu aşamada 5. adımda oluşturulan bulanık kümeden kesin değerler oluşturma işlemi yapılır ve risk puanı elde edilir. Bulanık mantık tabanlı risk puanlama algoritmasını içeren sistemimizin modeli Şekil 3.11'de gösterilmiştir. Bulanık sistemin girdilerini 3. adımda tanımladığımız risk kategorileri oluşturmaktadır. Bulanık Mantık Tabanlı Risk Puanlama Sistemi, yazılan bulanık çıkarım kuralları, oluşturulan bulanık kümeleri ve girdileri değerlendirerek 0 ile 100 arasında bir risk puanı üretir. 100'e yakın değerler yüksek riskli uygulamaları temsil etmektedir.



Şekil 3.11: Bulanık mantık tabanlı risk puanlama sistemi modeli.

3.2.3.4.Risk Puanlama Sisteminin Matematiksel Modeli

Bulanık mantık tabanlı risk değerlendirme sistemimizin matematiksel modelinde Denklem 3.5-3.16 arası kullanılacak parametreler şu şekilde tanımlanmıştır:

- TRP : kullanılacak TehlikeliRisk kategorisindeki izinlerin toplam puanı.
- $İSRP$: İmzaYadaSistemRiski kategorisindeki izinlerin toplam puanı.
- $İRP$: İmzaRiski kategorisindeki izinlerin toplam puanı.
- NRP : NormalRisk kategorisindeki izinlerin toplam puanı.
- k : TehlikeliRisk (TR), İmzaYadaSistemRiski (İSR), İmzaRiski (İR), NormalRisk (NR) risk kategorilerinden birini temsil eder.
- RT : Tehlikeli, Orta, Normal, Düşük risk türlerinden birini temsil eder.
- $İ_n$: n nolu izin.
- $RED_k(İ_n)$: $İ_n$ izninin k kategorisindeki Tablo 3.4'de yer alan risk etki değeri.
- $ROD_{RT}(İ_n)$: $İ_n$ izninin RT isimli risk türünün Tablo 3.4'de bulunan risk olasılık değeri.
- $İ_nRŞ$: n nolu iznin $RED_k(İ_n)$ ve $ROD_{RT}(İ_n)$ değerlerinin çarpımından elde edilen risk şiddeti.

Modelde ilk aşama olarak her bir risk kategorisinin risk puanı hesaplanır.

- $k = TR$ kategorisindeki izinlerin TRP'si:

$$\dot{I}_n RŞ = RED_{TR}(\dot{I}_n) \times ROD_{RT}(\dot{I}_n) \tag{3.5}$$

$$TRP = \sum_{0}^{n} \dot{I}_n \cdot RŞ \tag{3.6}$$

- $k = \dot{I}SR$ kategorisindeki izinlerin $\dot{I}SRP$'si:

$$\dot{I}_n RŞ = RED_{\dot{I}SR}(\dot{I}_n) \times ROD_{RT}(\dot{I}_n) \tag{3.7}$$

$$\dot{I}SRP = \sum_{0}^{n} \dot{I}_n \cdot RŞ \tag{3.8}$$

- $k = \dot{I}R$ kategorisindeki izinlerin $\dot{I}RP$'si:

$$\dot{I}_n RŞ = RED_{\dot{I}R}(\dot{I}_n) \times ROD_{RT}(\dot{I}_n) \tag{3.9}$$

$$\dot{I}RP = \sum_{0}^{n} \dot{I}_n \cdot RŞ \tag{3.10}$$

- $k = NR$ kategorisindeki izinlerin NRP'si:

$$\dot{I}_n RŞ = RED_{NR}(\dot{I}_n) \times ROD_{RT}(\dot{I}_n) \tag{3.11}$$

$$NRP = \sum_{0}^{n} \dot{I}_n \cdot RŞ \tag{3.12}$$

İkinci aşamada; TRP, $\dot{I}SRP$, $\dot{I}RP$ ve NRP, "Mamdani" bulanık sistemine girdi olarak verilir. Girdiler, Denklem 3.13'te sunulan üçgen üyelik fonksiyonu ile bulanıklaştırılır.

$$\mu_F(x;\ a,\ b,\ c) = \begin{Bmatrix} 0 & , & x < a \\ \dfrac{x-a}{b-a} & , & a \le x \le b \\ \dfrac{c-x}{c-b} & , & b \le x \le c \\ 0 & , & x > \mathrm{c} \end{Bmatrix} \tag{3.13}$$

Üçüncü aşamada, "Eğer-İse" bulanık operatörleri ile oluşturulan kurallar çalıştırılır. Kuralların çıkartılmasında, Denklem 3.14 ve Denklem 3.15'de gösterilen MIN-MAX kompozisyonu kullanılmıştır.

$$\mu_{A\cap B}(x) = MIN(\mu_A(x), \mu_B(x)) \tag{3.14}$$

$$\mu_{A\cup B}(x) = MAX(\mu_A(x), \mu_B(x)) \tag{3.15}$$

K_1 ve K_2; $X \times Y$ ve $Y \times Z$ 'de tanımlı iki bulanık ilişki olsun. K_1 ve K_2 'nin MIN - MAX kompozisyonu bir bulanık kümedir ve Denklem 3.16'da ifade edilmiştir.

$$K_1 O K_2 = \left\{ \left[(x,z), MINMAX\left(\mu_{K_1}(x,y), \mu_{K_2}(y,z)\right) \right] \mid x \in X, y \in Y, z \in Z \right\} \tag{3.16}$$

Oluşturulan kurallar ise aşağıdaki biçimde formüle dökülebilir.

- K_i : Bulanık operatörler kullanılarak elde edilen giriş parametreleri ve bu paremetrelerin bulanık kümelerinin ilişkisi.
- u_j : Çıkış.
- A_{jk} : j çıkış değerinin bulanık kümesi

olmak üzere oluşturulan kural aşağıda gösterildiği şekilde ifade edilir:

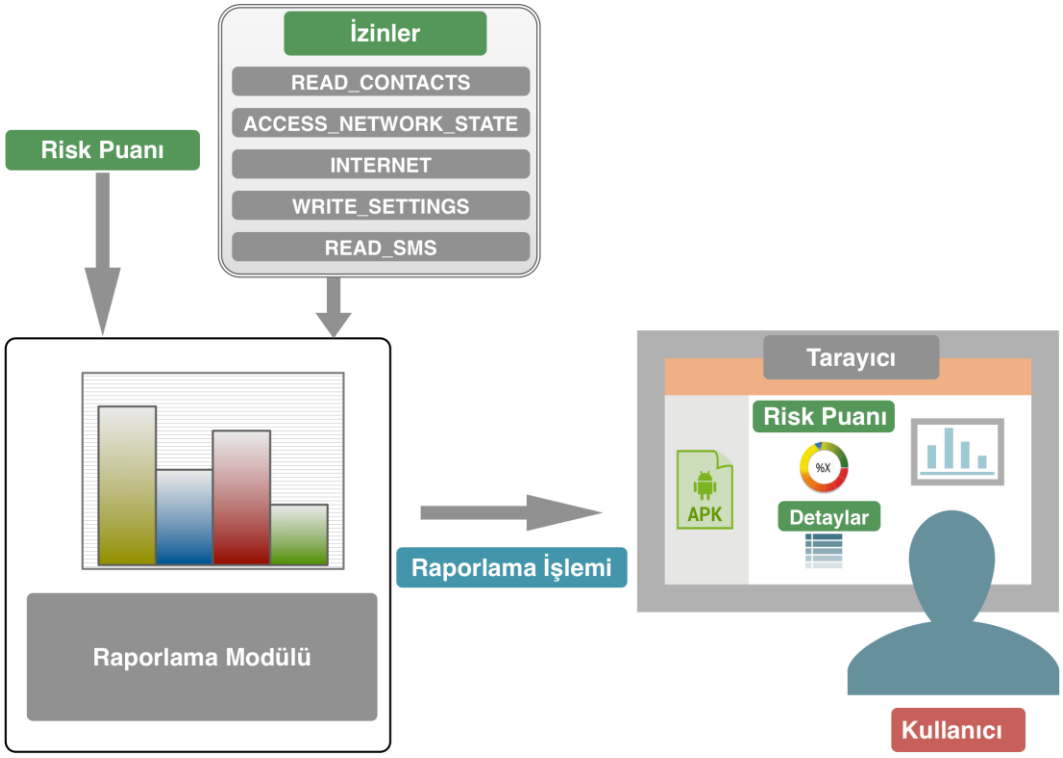
$$IF \quad K_i \quad THEN \quad u_j = A_{jk}$$

Her bulanık kural, girişlerin bulanık kümesinin, çıkışların her bulanık kümesi üzerindeki etkisini tanımlar. Bu etkiyi numerik olarak tanımlamayı sağlayan prosedüre bulanık çıkarım denir. Son aşamada çıkış bulanık kümelerine tüm kurallar uygulanarak üretilen etkiler birleştirilerek tek bir çıkış değeri üretilir. Elde edilen bulanık değer durulaştırılarak numerik bir çıkış değeri üretilir. Çıkış değeri en çok kullanılan durulaştırma yöntemi olan Ağırlık Merkezi yöntemi (CoG) kullanılarak elde edilir. Bu yöntemde üyelik fonksiyonunun çevrelediği alanın ağırlık merkezi ya da üyelik fonksiyonunun ağırlıklı ortalaması bulanık değerin en saf değeri olarak hesaplanır. Örneğin; Riskin çok yüksek olması sonucu Denklem 3.17'de şu şekilde ifade edilebilir:

$$CoG(Yüksek\_Risk) = \frac{\sum_x \mu_{YüksekRisk}(x) * x}{\sum_x \mu_{YüksekRisk}(x)} \tag{3.17}$$

3.2.4. Risk Raporlama Modülü

Bu modülün amacı kullanıcıya yükleyeceği uygulama hakkında anlaşılır, detaylı bilgiler sunmaktır. Önceki modüllerde elde edilen risk puanı, çıkarılan izinler grafiksel olarak gösterilir. Bunun yanında uygulamanın hangi izinleri kullandığı, kullanılan her izinin ne işe yaradığı, uygulamanın yüklenmesi durumunda ne gibi tehlikelerin ortaya çıkacağı bilgisi sunularak ekranda yer alır. Şekil 3.12 bu modülün işleyişini göstermektedir.



Şekil 3.12: Risk raporlama modülü işleyişi.

3.3. SİSTEM MODÜLLERİNİN GERÇEKLEŞTİRİLMESİ

Bu aşamada, oluşturduğumuz tasarım dikkate alınarak hedeflediğimiz sistem gerçekleştirilmiştir. Sistemin daha iyi anlaşılması için sistem iki ana modüle ayrılmıştır. Birinci modül sistemin temelini oluşturan ve arka planda sürekli olarak çalışan Sistem Servis modülüdür. Tüm hesaplama işlemleri bu modülde yapılmaktadır. Bu modül Python [96] programlama dili kullanarak bir servis sağlayacak şekilde geliştirilmiştir. İkinci modül kullanıcının gördüğü kısmı oluşturan Web Servis modülüdür. Kullanıcının uygulama yüklemesini, aramasını ve analiz sonuçlarını görebilmesini sağlayacak arayüzü içerir. PHP [97], HTML [98], CSS [98], JQuery [99], Javascript [100] teknolojileri kullanarak geliştirilmiştir.

3.3.1. Sistem Servis Modülleri

3.3.1.1. İzin Çıkarma Modülü

İzin çıkarma işlemi; *"apk"* uzantılı Android uygulamalarının içinde bulunan ve izinlerin yer aldığı ikili formatta olan "AndroidManifest.xml" [101] dosyası üzerinde yapılır. Bu aşamada; ilk önce *"apk"* dosyası bir arşiv dosyası olması nedeniyle *"zip"* formatında okunur, doğrulanır, "AndroidManifest.xml" dosyasına ulaşılır. Bir sonraki aşama olarak da ikili formattaki XML dosyası klasik, okunabilir XML dosyasına çevrilir ve belleğe kaydedilir. Bu işlemin işleyişi Algoritma 3.1'de sunulmuştur.

Giriş: APK dosyası
Çıkış: İzin dosyası

```
1:  izinCikar(apkDosyasi):
2:    zipDosyasi=zipOku(apkDosyasi)
3:       FOR EACH i IN zipDosyasi.icindekiler():
4:         IF i = "AndroidManifest.xml" THEN
5:            androidXML[i]=XMLCevirici(zipDosyasi.oku(i))
6:         ENDIF
7:       ENDFOR
8:  klasikXML[i] = xmlAyristir(androidXML[i].bellektenAl())
9:  RETURN klasikXML
10: XMLCevirici(veri):
11:   androidxml =AndroidXMLParser(veri)
12:   bellek= u"
13:   WHILE True AND androidxml.gecerli():
14:     tur = androidxml.next()
15:     IF  tur = XML_BASLANGICI THEN
16:       bellek+= u'<?xml version="1.0" encoding="utf-8"?>\n'
17:     ELSE IF tur = BASLANGIC_ETIKETI THEN
18:       bellek += u'<' + getOnek(androidxml.getOnek()) + androidxml.getAd() + u'\n'
19:      bellek+= androidxml.getXMLNameSpace()
20:       FOR EACH i IN kapsam(0,  axml.getAttributeCount()):
21:          bellek+= "%s%s=\"%s\"\n" % ( getOnek(androidxml.getOzellikOnek(i)),
22:           axml.getOzellikAd(i),  getOzellikDeger(i))
23:       ENDFOR
24:      bellek += u'>\n'
25:     ELSE IF tur = BITIS_ETIKETI THEN
26:       bellek+= "</%s%s>\n" % ( getOnek( axml.getOnek()), androidxml.getAd())
27:     ELSE IF  tur = METIN THEN
28:       bellek+= "%s\n" % androidxml.getMetin()
29:     ELSE IF  tur = XML_BITISI THEN
30:       break
31:   ENDWHILE
32: RETURN bellek
```

Algoritma 3.1: İzin çıkarma işlemi.

3.3.1.2. İzin Analiz Modülü

Bu aşamada, bir önceki aşamadan elde edilen okunabilir "AndroidManifest.xml" dosyası okunur. Bu dosyada kullanılan tüm izinler "uses-permission" etiketi ile

belirtilmiştir. Bu etiket bilgisi kullanılarak izinler dosyadan ayrıştırılıp kaydedilir. İşleyiş, Algoritma 3.2'de gösterilmiştir.

```
Giriş: İzin dosyası
Çıkış: İzinler

1: izinAnalizEt(manifestDosyasi):
2:    manifestXML = XMLOku(manifestDosyasi)
3:    FOR EACH i IN manifestXML.etiketler:
4:       i.ozellikBul('uses-permission')
5:       androidizinler.ekle (i)
6:    ENDFOR
7: RETURN androidizinler
```

Algoritma 3.2: İzin analiz işlemi.

İzinler elde edildikten sonra risk analiz işlemine geçilir. Bunun için riske neden olabilecek tüm izinlerin listesinin mevcut olması gerekmektedir. Analiz aşamasına geçmeden önce tüm izinlerin ve açıklamalarının yer aldığı bir veri seti oluşturulmuştur. Bu listede Android işletim sisteminde tanımlı 151 izin [67] için her bir izinin ismi, tehlike düzeyi, grubu ve açıklaması yer almaktadır. Bunun yanında risk puanlama modülünde risk hesabı yapabilmemiz için her bir izin bir risk kategorisine atanmıştır. Risk kategorileri izinler listesinde tanımlanan tehlike düzeylerine [73] göre oluşturulmuştur ve toplam 4 adettir. Bunlar:

- *TEHLIKELI_RISK:* Kullanılması durumunda en çok riske neden olacak risk kategorisidir. Ağırlığı en yüksektir. Android'de koruma düzeyi "dangerous" (tehlikeli) olarak adlandırılmıştır.
- *IMZASISTEM_RISK:* Riske etkisi bakımından ağırlığı 2. sırada olan risk kategorisidir. Android'de koruma düzeyi "signature or system" (imza ya da sistem) olarak adlandırılmıştır.
- *IMZA_RISK:* Riske etkisi bakımından 3. sıradadır. Android'de koruma düzeyi "signature" (imza) olarak adlandırılmıştır.
- *NORMAL_RISK:* En düşük risk etkisine sahip risk kategorisidir. Android'de koruma düzeyi "normal" olarak adlandırılmıştır.

Uygulamanın kullandığı izinlere bakarak, bu izinlerin hangi kategorilerde yer aldığını, detaylı bilgilerini alabilmek için Algoritma 3.3'de belirtilen izin_detaylarini_getir metodu oluşturulmuştur.

```
Giriş: İzin bilgilerini içeren dosya
Çıkış: Ayrıştırılmış izinler ve detayları

1:  izin_detaylarini_getir():
2:  liste = {}
3:    FOR EACH i IN androidizinler :
4:      izin = i
5:      konum = i.rfind(".")//android.permission.SEND_SMS son noktadan sonraki izin ismini çek
6:      IF  konum != -1 THENizin = i[pos+1:] ENDIF
7:      TRY : liste[ i ] = TUM_IZINLER["MANIFESTO_IZINLER"][ izin ]
9:      EXCEPT Hata
11: ENDFOR
12: RETURN liste
```

Algoritma 3.3: İzin bilgilerini getirme.

İzinlerin ve detaylı bilgilerinin elde edilmesinden sonra risk puanlama işlemi için hazırlanması gerekmektedir. Tasarım aşamasında oluşturduğumuz matematiksel model kullanılarak risk girdileri hazırlanır ve bir sonraki aşama olan bulanık mantığa dayalı risk puanı hesaplama aşamasında girdi olarak kullanılır. Algoritma 3.4 bu aşamayı yerine getiren izin_risklerini_değerlendir metodunu göstermektedir.

```
Giriş: İzinler
Çıkış: Risk bilgisi

1: izin_risklerini_degerlendir(izinler, riskBilgisi) :
2:  FOR EACH i IN izinler :
3:      izin = i
4:      IF izin.find(".") != -1  THEN izin = izin.split(".")[-1] ENDIF
5:      IF izin IN TUM_IZINLER["MANIFESTO_IZINLER"] THEN
6:         izinkategorisi=TUM_IZINLER["MANIFESTO_IZINLER"][izin][1]
7:         aciklama=TUM_IZINLER["MANIFESTO_IZINLER"][izin][3]
8:         riskBilgisi=RiskBilgisi()
9:          riskBilgisi.izinAdi=izin
10:        riskBilgisi.izinKategori=izinkategorisi
11:        riskBilgisi.izinAciklamasi=aciklama
12:        risk_turu = ANDROID_KORUMA_DUZEYI_RISK[ izinler[ i ][0] ]
13 :    ENDIF
14:     IF risk_turu == 1 THEN
15:        riskPuan[ TEHLIKELI_RISK ] += riskEtki [ 1 ]*riskO[ izinler[ i ][0]]
16:     ELSE IF risk_turu == 2 THEN
17:        riskPuan [ IMZA_SISTEM_RISK ]+= riskEtki [ 2 ]*riskO[ izinler[ i ][0]]
18:     ELSE IF risk_turu == 3  THEN
19:        riskPuan [ IMZA_RISK ] += riskEtki [ 3 ]*riskO [ izinler[ i ][0]]
20:     ELSE IF risk_turu==4 THEN
21:        riskPuan [ NORMAL_RISK ] += riskEtki [ 4 ]*riskO[ izinler[ i ][0]]
22:  ENDFOR
23:    riskBilgisi.riskPuan=riskPuan
24: RETURN riskBilgisi
```

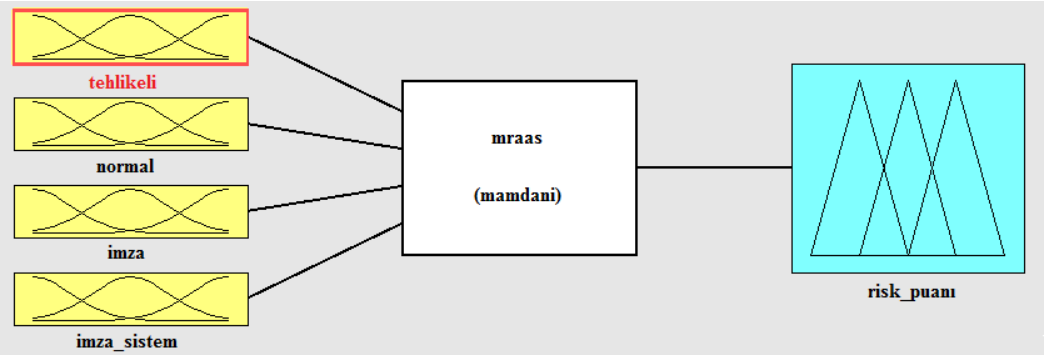
Algoritma 3.4: İzin risklerinin değerlendirilmesi.

İzin riskleri değerlendirildikten sonra oluşturduğumuz bulanık mantık tabanlı puanlama algoritması çalıştırılarak riskler değerlendirilir ve risk puanı üretilir.

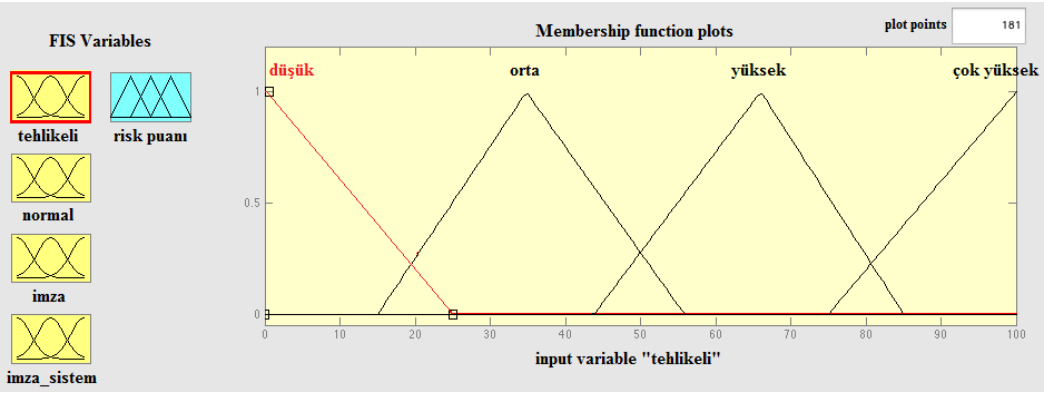
3.3.1.3.Risk Hesaplama Modülü

Bulanık mantık yapısını tasarlamada MATLAB Bulanık Mantık Kütüphanesi ve gerçekleştirimde de Python için geliştirilmiş "pyfuzzy"[2] bulanık mantık kütüphanesi kullanılmıştır.

Girdilerin ve üyelik fonksiyonlarının tanımlanması: Daha önceki aşamada dört adet risk kategorisi oluşturmuştuk. Bu aşamada oluşturulan bu dört risk kategorisi kullanılarak dört adet risk girdisi oluşturulmuştur. Her bir risk düzeyi için "dusuk, orta, yuksek, cokyuksek" olmak üzere dört adet tehlike seviyesi oluşturulmuştur. Ek olarak her bir risk girdisi için de üçgen üyelik fonksiyonu oluşturulmuştur. Şekil 3.13 Matlab FIS Editörü aracılığı ile oluşturduğumuz risk puanlaması için gerekli bulanık mantık girişlerini, Şekil 3.14 ise tehlikeli izin kategorisi üyelik fonksiyonunu göstermektedir.



Şekil 3.13: Risk puanlama bulanık mantık girişleri.



Şekil 3.14: Tehlikeli izin kategorisi üyelik fonksiyonu.

[2] Pyfuzzy Python Bunalık Mantık Kütüphanesi, http://pyfuzzy.sourceforge.net, [Erişim tarihi: 12.06.2015]

Risk girdilerinin Python tarafında oluşturulmasını sağlayan metot Algoritma 3.5'de, girdilerin üyelik fonksiyonlarını oluşturan metot Algoritma 3.6'da gösterilmiştir.

Giriş: Yok
Çıkış: Risk girdileri

```
1:  risk_girdisi_olustur() :
2:      bulanik_sistem = BulanikSistem()
3:      giris_tehlikeli_risk = GirisDegiskeni()
4:      giris_normal_risk = GirisDegiskeni()
5:      giris_imza_risk = GirisDegiskeni()
6:      giris_imzasistem_risk =InputDegiskeni()
```

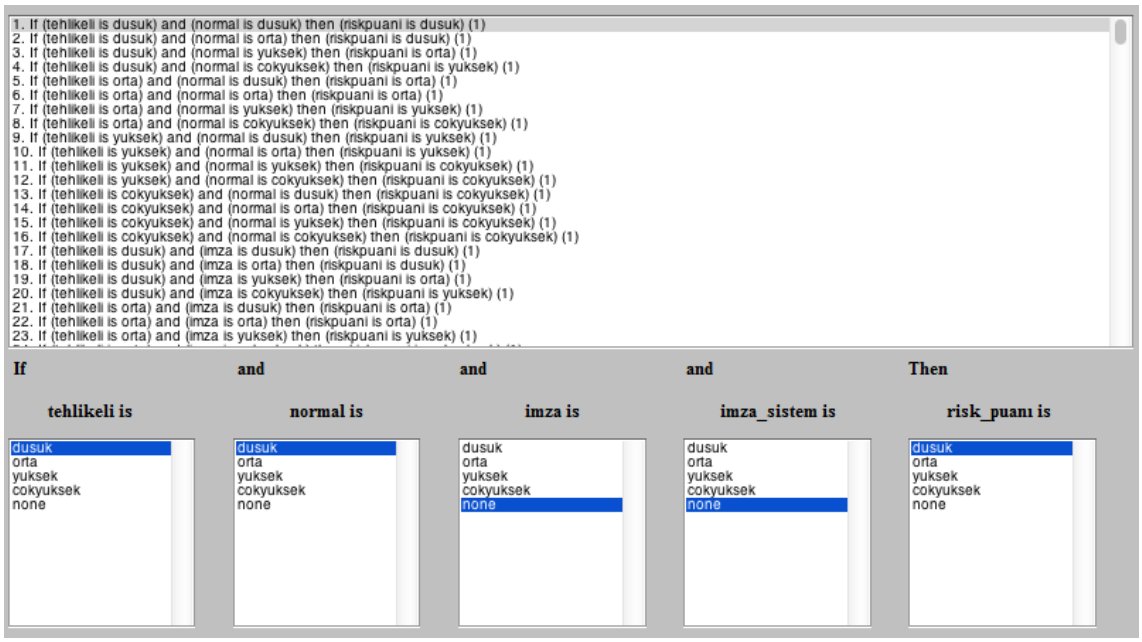
Algoritma 3.5: Risk girdilerinin oluşturulması.

Giriş: Risk girdileri
Çıkış: Üyelik fonksiyonları

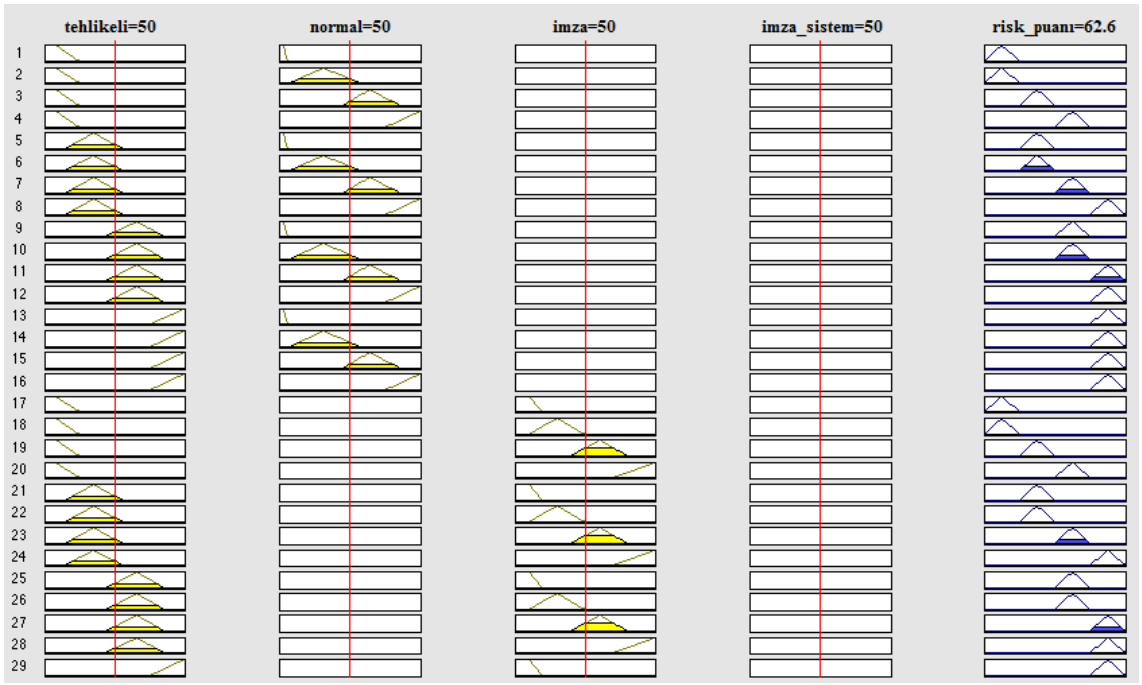
```
1: uyelik_fonksiyonlarini_olustur:
2:      bulanik_sistem.degisken["tehlikeli_risk"] = giris_tehlikeli_risk
3:      giris_tehlikeli_risk.uyelik[DUSUK_RISK] = UcgenMF("Düşük Risk Aralığı")
4:      giris_tehlikeli_risk.uyelik[ORTA_RISK] = UcgenMF("Orta Risk Aralığı")
5:      giris_tehlikeli_risk.uyelik[YUKSEK_RISK] = UcgenMF("Yüksek Risk Aralığı")
6:      giris_tehlikeli_risk.uyelik[COK_YUKSEK_RISK] = UcgenMF("Çok Yüksek Risk Aralığı")
7:      bulanik_sistem.degisken["normal_risk"] = giris_normal_risk
8:      giris_normal_risk.uyelik[DUSUK_RISK] = UcgenUF("Düşük Risk Aralığı")
9:      giris_normal_risk.uyelik[ORTA_RISK] = UcgenUF("Orta Risk Aralığı")
10:     giris_normal_risk.uyelik[YUKSEK_RISK] = UcgenUF("Yüksek Risk Aralığı")
11:     giris_normal_risk.uyelik[COK_YUKSEK_RISK] = UcgenUF("Çok Yüksek Risk Aralığı")
12:     bulanik_sistem.degisken["imza_risk"] = giris_imza_risk
13:     giris_imza_risk.uyelik[DUSUK_RISK] = UcgenUF("Düşük Risk Aralığı")
14:     giris_imza_risk.uyelik[ORTA_RISK] = UcgenUF("Orta Risk Aralığı")
15:     giris_imza_risk.uyelik[YUKSEK_RISK] = UcgenUF("Yüksek Risk Aralığı")
16:     giris_imza_risk.uyelik[COK_YUKSEK_RISK] = UcgenUF("Çok Yüksek Risk Aralığı")
17:     bulanik_sistem.degisken["imzasistem_risk"] = giris_ imzasistem_risk
18:     giriş_imzasistem_risk.uyelik[DUSUK_RISK] = UcgenUF("Düşük Risk Aralığı")
19:     giris_imzasistem_risk.uyelik[ORTA_RISK] = UcgenUF("Orta Risk Aralığı")
20:     giris_imzasistem_risk.uyelik[YUKSEK_RISK] = UcgenUF("Yüksek Risk Aralığı")
21:     giris_imzasistem_risk.uyelik[COK_YUKSEK_RISK] = UcgenUF("Çok Yüksek Risk Aralığı")
```

Algoritma 3.6: Üyelik fonksiyonlarının oluşturulması.

Kuralların Tanımlanması: Risk hesaplamada kullanılacak kuralların tanımlanması için Matlab Kural editörü kullanılmıştır. Kurallar, uygulamalarda yer alabilecek izinlerin risk kategorileri dikkate alınarak oluşturulmuştur. Toplam 4 adet risk kategorisi mevcut olduğundan, bir uygulama minimum sıfır, maksimum dört kategoride izin kullanabilir. Bu nedenle izin kategorilerinin kombinasyonları üretilerek çeşitli kurallar oluşturulmuştur. Şekil 3.15 örnek kural listesini, Şekil 3.16 da risk hesaplamaya yönelik kural izleyiciyi göstermektedir.



Şekil 3.15: Örnek kurallar.



Şekil 3.16: Risk hesaplamaya yönelik kural izleyici.

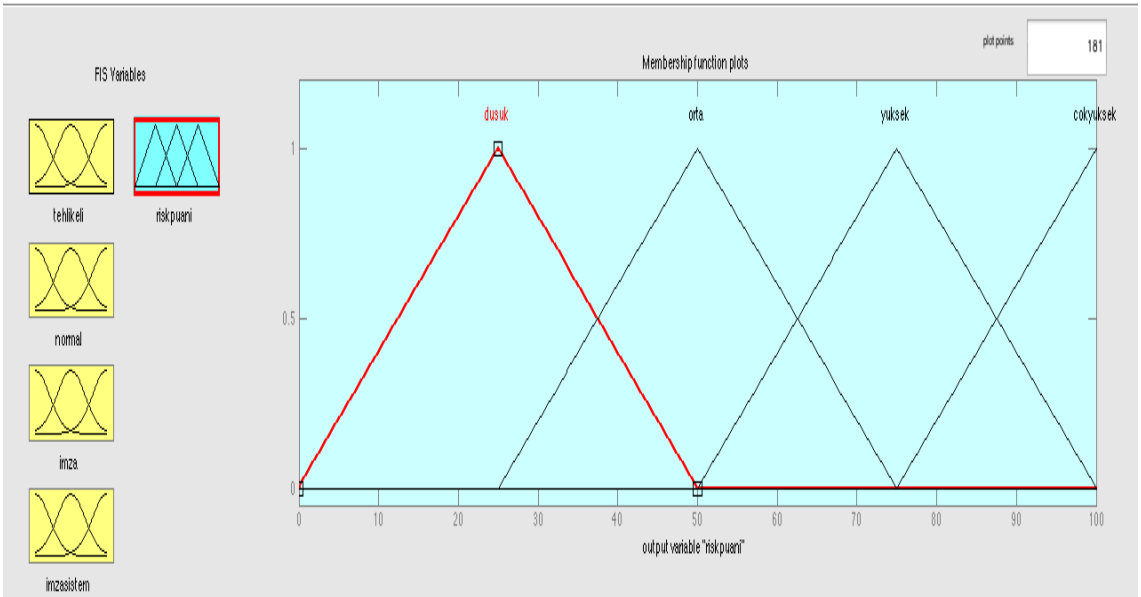
Algoritma 3.7, kuralların Python tarafında nasıl oluşturulduğunu göstermektedir.

Giriş: Kural adı, bulanık sistem
Çıkış: Kural

```
1: kural_ekle(bulaniksistem, kural_adi, kural) :
2:     bulaniksistem.rules[ kural_adi ] = kural
3:     kural_ekle(bulanik_sistem, "kuralX", Kural(
4:     uyelik=[bulanik_sistem.degisken["total_risk"].uyelik[COK_YUKSEK]],
5:     operator=BirlesikOperator(
6:     norm.Min.Min(),
7:     Giris( bulanik_sistem.degisken["tehlikeli_risk"].uyelik[COK_YUKSEK_RISK] ),
8:     Giris( bulanik_sistem.degisken["imza_risk"].uyelik[ORTA_RISK] ),
9:     Giris( bulanik_sistem.degisken["imzasistem_risk"].uyelik[YUKSEK_RISK] ) ) ) )
```

Algoritma 3.7: Kural oluşturma işlemi.

Çıkışın Üretilmesi: Bu aşamada kurallar dahilinde üyelik fonksiyonları çalıştırılarak çıkış, yani risk puanı üretilir. Şekil 3.17 risk puanı için çıkışın ve çıkışın üyelik fonksiyonunun oluşturulmasını, Algoritma 3.8 ise çıkışın Python tarafında oluşturulmasını sağlayan metodu göstermektedir.



Şekil 3.17: Risk puanı için çıkışın oluşturulması.

Giriş: Bulanıklaştırma yöntemi, minimum maksimum değerler, üyelik değerleri
Çıkış: Bulanık çıkış

```
1: cikis_olustur:
2:   total_risk = CikisDegiskeni(
3:                   bulaniklastirmaYontemi=COGS.COGS(),
4:                   aciklama="Toplam Risk Puani",
5:                   minimum=0.0,maksimum=100.0,)
6:   total_risk.uyelik[DUSUK] = UyelikDeger("Düşük Değer")
7:   total_risk.uyelik [ORTA] = UyelikDeger("Orta Değer")
8:   total_risk.uyelik[YUKSEK] = UyelikDeger("Yüksek Değer")
9:   total_risk.uyelik[COK_YUKSEK] = UyelikDeger("Çok Yüksek Değer"))
10:  bulanik_sistem.degisken["total_risk"] = total_risk
```

Algoritma 3.8: Çıkışların oluşturulması.

Bulanık sistem yapısı oluşturulduktan sonra puanlama işlemi yapılmaktadır. Puanlama işleminin işleyişi Algoritma 3.9'da gösterilmiştir. Algoritma 3.9'da yer alan "bulanikHesap(input=girdi, output = sonuc)" bulanık hesaplamayı gerçekleştiren metottur. Bu metot 4 aşamada çalışmaktadır ve işleyişi Algoritma 3.9'da gösterilmiştir. Bu aşamalar şunlardır:

- *Sistemin sıfırlanması*
- *Girdinin bulanıklaştırılması*
- *Çıkarımın gerçekleştirilmesi*
- *Sonucun durulaştırılması*

Giriş: Riskler, dosya adı, izin listesi
Çıkış: Risk sonuçları

```
1:  tum_riskleri_degerlendir(self, riskler,dosyaadi,izin_listesi) :
2:      sonuc = {"total_risk" : 0.0}
3:      girdi["tehlikeli_risk"] = riskler[ TEHLIKELI_RISK ]
4:      girdi["normal_risk"] = riskler[ NORMAL_RISK ]
5:      girdi["imza_risk"] = riskler[ IMZA_RISK ]
6:      girdi["imzasistem_risk"] = riskler[ IMZA_SISTEM_RISK ]
7:      bulanikHesap (input=girdi, output = sonuc)
8:      risk_degeri = sonuc[ "total_risk" ]
9:      sonuc=RiskSonuc()
10:     sonuc.riskDuzeyi=risk_seviyesi
11:     sonuc.riskPuani=risk_degeri
12:     sonuc.zararliMi=zararliyazilimbul(hashBul(dosyaadi))
13:     sonuc.izinler=izin_listesi
14:  RETURN sonuc
15: bulanikHesap(giris,cikis):
16:     sifirla() # 1. adım
17:     bulaniklastir(giris) # 2. adım
18:     bulanikCikarimYap() 3. adım
19:     cikis=durulastir() # 4. adım
20: RETURN cikis
```

Algoritma 3.9: Bulanık mantık ile puan hesaplama işlemi.

3.3.1.4. Raporlama Modülü

Raporlama modülü istemci-sunucu modeli kullanılarak geliştirilmiştir. Analiz edilecek uygulama dosyası istemci aracılığı ile alınarak sunucuya bildirir. Sunucu parametre olarak aldığı dosya ismine göre uygulamayı işleyerek sonuçları geri döndürür. Algoritma 3.10 istemcinin çalışma prensibini göstermektedir.

```
Giriş: Dosya adı
Çıkış: Risk raporu

1: raporOlustur(dosyaAdi)
2:      http_istemci = HTTPIstemcisiOlustur( adres = "http://localhost:8080")
3:      risk=http_istemci.analizet(dosya_adi)
4:      yazdır risk
```

Algoritma 3.10: Raporlama istemcisi.

Raporlama istemcisi oluşturduğumuz web servisi aracılığı ile çağrılmaktadır. Dosya adı istemci modülüne parametre olarak geçirilmektedir. Raporlama sunucumuz "http://localhost:8080" adresinde çalışmaktadır ve 'analizet' metodu ile uzaktan metot çağrımına izin veren bir hizmet sağlamaktadır. Algoritma 3.11 raporlama sunucumuzun işleyişini göstermektedir.

```
Giriş: Dosya adı
Çıkış: Risk sonucu

1:  risk = RiskIslemleri()
2:  risk.risk_analizi_ekle(IzinDetay())
3:  risk.risk_analizi_ekle(FuzzyRisk())
4:  class IstekIsle(HttpIstekIsleyici):
5:  analizet(dosyaadi):
6:     dosyaadi, dosyauzantisi = dosyaAdiveUzantiyiBul(dosyaadi)
7:     IF dosyauzantisi==".apk" THEN
8:          apkdosyasi = APKOlarakOku(dosyaadi)
9:          fuzzyRiskSonuc=risk.apk_analiz(apkdosyasi,dosyaadi)["Fuzzy Risk Degeri"]
10:         izinRiskSonuc=risk.apk_analiz(apkdosyasi,dosyaadi)["Izin Kullanim Detaylari"]
11:         fuzzyRiskSonuc.uygulamaAdi=apkdosyasi.uygulamaAdiDondur()
12:         fuzzyRiskSonuc.tehlikeliIzinSayisi=izinRiskSonuc["IZINLER"]["TEHLIKELI"]
13:         fuzzyRiskSonuc.imzaIzinSayisi=izinRiskSonuc["IZINLER"]["IMZA"]
14:         fuzzyRiskSonuc.imzaSistemIzinSayisi=izinRiskSonuc["IZINLER"]["IMZASISTEM"]
15:         fuzzyRiskSonuc.normalIzinSayisi=izinRiskSonuc["IZINLER"]["NORMAL"]
16:    ENDIF
17: RETURN fuzzyRiskSonuc.to_JSONYap()
18: http_sunucusu = YeniHTTPSunucusu(sunucuAdresi= ('localhost', port=8080), IstekIsleyiciSinif = IstekIsle)
19: yazdir "HTTP Sunucusu Başlatılıyor ..."
20: yazdır "URL: http://localhost:8080"
21: sunucuyuCalistir()
```

Algoritma 3.11: Raporlama sunucusu.

Raporlama sunucumuz Ubuntu Linux üzerinde arka planda sürekli olarak çalışan bir servistir. Web servisi aracılığı ile gelen her isteğe cevap vererek JSON [97] formatında sonuç döndürmektedir.

3.3.2. Web Servis Modüllerin Gerçekleştirilmesi

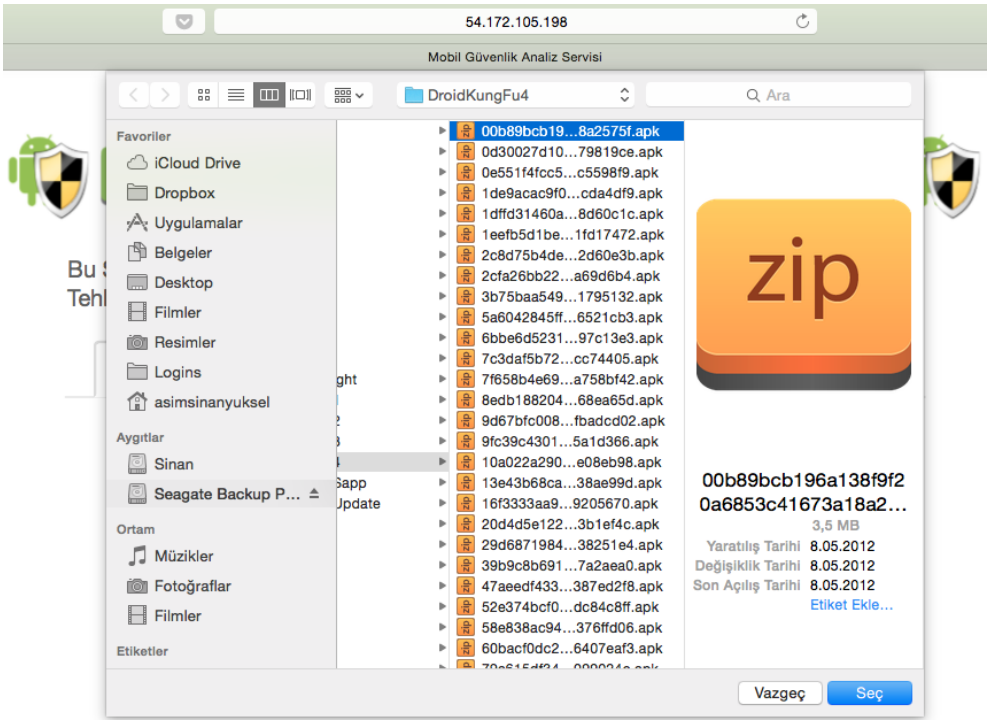
Web servis modülü 4 alt modülden oluşmaktadır. İlk modül araştırmacıların, Android tabanlı cihaz kullanıcılarının ellerinde mevcut olan uygulamaları yüklemelerini sağlayan Uygulama Yükleme Modülü'dür. İkinci modül Uygulama Arama Modülü'dür. Kullanıcıların Google Uygulama Mağazasında yer alan uygulamaları aramalarını sağlar. Üçüncü modül Analiz Modülü'dür. Yüklenen uygulamaların analiz edilmesini sağlar. Son modül ise Raporlama Modülü'dür. Analiz sonuçlarını kullanıcının anlayabileceği şekilde web ortamında gösterir.

3.3.2.1. Uygulama Yükleme Modülü

Android tabanlı cihaz kullanıcılarının ya da mobil güvenlik alanında çalışmalar yapan araştırmacılar için ellerinde mevcut olan uygulamaları yüklemelerini sağlayan Uygulama Yükleme Modülü'dür. Sadece '.apk' uzantılı ve boyut olarak 100 MB'ı geçmeyen Android uygulama dosyalarının yüklenmesine izin verir. Kullanıcılar aynı anda sadece 1 adet uygulama yükleyip analiz edebilirler. Uygulama yükleme işlemi tamamlandığında yüklenen uygulama bulut ortamında kaydedilir ve uygulamaya ait bilgiler veritabanımızda saklanır. Yüklemenin başarılı olması durumunda Analiz Modülü'ne yönlendirme yapılır. Şekil 3.18, Uygulama Yükleme Modülü'nün arayüzünü göstermektedir.

Şekil 3.18: Uygulama yükleme modülü.

Kullanıcı ‘Yükle ve Analiz Et’ tuşuna bastığında Şekil 3.19’da görülen ve uygulamayı seçip yüklemesine yardımcı olacak pencere açılır. Uygulama seçildikten sonra yükleme işlemi başlamaktadır.



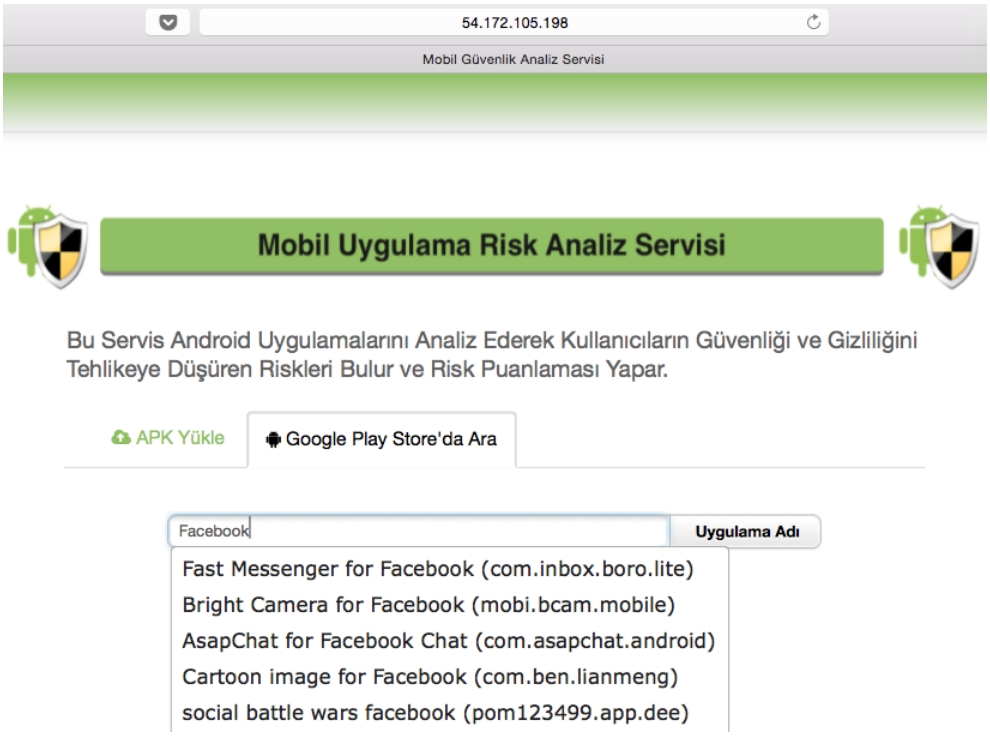
Şekil 3.19: Uygulama seçme ekranı.

Ek olarak yükleme işleminin ilerleme durumu yüzde cinsinden Şekil 3.20’da olduğu gibi gösterilmektedir.

Şekil 3.20: Yükleme ilerleme durumu.

3.3.2.2. Uygulama Arama Modülü

Arama modülü kullanıcıların herhangi bir dosya yüklemeden Google Play Uygulama Mağazası'nda [36] yer alan uygulamaları aramasına imkan verir. Girilen anahtar kelimeye göre arama yapıldığında Şekil 3.21'de görüldüğü gibi sonuçlar listelenir. Sonuç listesinden istenen uygulama seçilip "Analiz Et" tuşuna basıldığında uygulama analiz modülüne yönlendirilir.



Şekil 3.21: Arama modülü.

3.3.2.3. Analiz Modülü

Bu modül yüklenen uygulamaları ya da Google Play Uygulama Mağazasında aranan uygulamaları analiz eder. 2 alt modülden oluşmaktadır. Bunlar yüklenen uygulamaların analizini yapan 'APK Analiz Modülü' ve yapılan aramalara göre sonuç döndüren 'Google Play' modülüdür.

APK Analiz Modülü

Yükleme modülü aracılığı ile yüklenen uygulamalar bulutta yer alan dosya sistemine kaydedilmektedir. 'analiz.et.py' skripti dosya yolu ve yüklenen uygulamanın ismini parametre olarak alır ve 'analizservis.py' skriptine uzak metot çağırımında bulunur. Servisten dönen sonuç JSON formatına dönüştürülerek 'SESSION' dediğimiz oturumlarda saklanır. Bu sayede saklanan bilgiler sayfadan sayfaya (AnalizEt.php

sayfasından AnalizSonuc.php sayfasına) taşınabilir. Bu modülün işleyişi Algoritma 3.12'de gösterilmiştir.

Giriş: APK dosyası
Çıkış: Analiz sonuçları

```
1:  oturumuBaslat()
2:  dosyaAdi=dosyaAdiniAl()
3:  oturumaEkle(dosyaAdi)
4:  tamDosyaYolu = "Analiz edilecek dosyaların tutulacağı klasör"
5:  birlestir(tamDosyaYolu, dosyaAdi)
6:  komut = "python  analizet.py "
7:  birlestir(komut, tamDosyaYolu)
8:  komutSonuc = komutCalistir(komut)
9:  WHILE EOF(komutSonuc)
10:   birlestir(sonuç,sonuc.oku(komutSonuc))
11: END WHILE
12: jsonSonuc=jsonaDonustur(sonuç)
13: paketAdi=jsonSonuc->{'uygulamaAdi'}
14: oturumaEkle(paketAdi)
15: zararliMi=jsonSonuc->{'zararliMi'}
16: toplamIzinMiktari= sonSonuc->{'izinMiktari'}
17: oturumaEkle(toplamIzinMiktari)
18: riskPuani= sonSonuc->{'riskPuani'}
19: oturumaEkle(riskPuani)
20: kullanilanIzinler= sonSonuc->{'izinler'}
21: oturumaEkle(kullanilanIzinler)
22: yonlendir("Analiz Sonuç Sayfası"))
```

Algoritma 3.12: APK analiz işlemi.

Google Play Modülü

Bu modül kullanıcıların hiçbir uygulama yüklemeden risk analizi yapabilmelerini sağlar. Geliştirdiğimiz web örümceği sayesinde Google Play Uygulama Mağazasından 27 kategoride kullanıcılar tarafından en çok indirilen 100 uygulama bulutta yer alan sistemimize yüklenmiş (toplam 2700 uygulama), analiz edilmiş ve analiz sonuçları veritabanımıza kaydedilmiştir. Modülün işleyici Algoritma 3.13, kullandığımız uygulama kategorileri Tablo 3.6'da gösterilmiştir. Kategoriler Google Play Uygulama Mağazasından alınmıştır.

```
Giriş: Aranacak uygulama adı
Çıkış: Risk analiz sonucu

1:  oturumuBaslat()
2:  aramaKelimesiAyarla(aranacakKelime,girilenKelime)
3:  uygulamaBul(aranacakuygulama, bulunan);
4:  paketadi=bulunan[1];
5:  oturumaEkle(paketAdi)
6:  veritabaninaBaglan(kullaniciAdi,sifre,veritabani)
7:  IF !bosMu(paketAdi) THEN
8:          sonuc=uygulamaGetir(paketadi)
9:          WHILE EOF(sonuc)
10:             dosyaAdi=birlestir(sonuc['uygulamaAdi'], '.apk')
11:             oturumaEkle(dosyaadi)
12:             zararliMi=sonuc['zararliMi']
13:             oturumaEkle(zararliMi)
14:             riskPuani=sonuc['riskPuani']
15:             oturumaEkle(riskPuani)
16:             kullanilanIzinler=sonuc['izinler']
17:             oturumaEkle(kullanilanIzinler)
18:         END WHILE
19: END IF
20: yonlendir("Analiz Sonuç Sayfası")
```

Algoritma 3.13: Google Play modülü işleyişi.

Tablo 3.6: Uygulama kategorileri.

Kategori ID	Kategori Adı	Kategori ID	Kategori Adı
1	Alışveriş	15	Kitaplıklar ve Kısa Sunum
2	Araçlar	16	Medya ve Video
3	Canlı Duvar Kağıdı	17	Müzik ve Ses
4	Eğitim	18	Sağlık ve Fitness
5	Eğlence	19	Seyahat ve Yerel
6	Finans	20	Sosyal
7	Fotoğrafçılık	21	Spor
8	Haberler ve Dergiler	22	Tıp
9	Haberleşme	23	Ulaşım
10	Hava Durumu	24	Verimlilik
11	İş	25	Widgetler
12	Karikatür	26	Yaşam Tarzı
13	Kişiselleştirme	27	Oyunlar
14	Kitaplar ve Referans		

3.3.2.4.Raporlama Modülü

Bu modül sistem çıktılarının kullanıcının anlayabileceği bir şekilde sunulmasını sağlayan modüldür. Kullanıcının bir uygulamayı yüklemesi durumunda karşılaşacağı riskler, uygulama bilgileri ve risk puanı bu modül aracılığı ile gösterilir. Risk Bilgilerinin yer aldığı 'AnalizSonuc' sayfası 3 temel kısımdan oluşmaktadır. İlk kısım 'Uygulama Bilgisi' kısmıdır. Analiz edilen uygulamanın ismini, Google Play Mağaza

adresini, zararlı yazılımlar veritabanında yer alıp almadığını, ne kadar izin kullandığını ve risk puanını göstermektedir. Şekil 3.22 ‘AnalizSonuc’ sayfamızın ‘Uygulama Bilgisi’ kısmını göstermektedir. Şekilde de görüldüğü üzere yüklenen yazılımın risk puanı 100 üzerinde 75 olarak atanmıştır. Risk puanını gösteren çemberde risk puanı 100 değerine yaklaştıkça renk kırmızılaşmaktadır. Risk puanı azaldıkça puan çemberi turuncu, sarı ve yeşil renklere dönüşmektedir. İncelenen uygulamanın imzası oluşturmuş olduğumuz zararlı yazılımlar veritabanında yer aldığı için ‘Bilinen Zararlı Yazılım Mı?’ kısmı ‘Evet’ olarak gösterilmektedir. Ek olarak uygulamanın kullandığı izin sayısı ve mağaza adresi gibi detaylı bilgiler de görülebilir.

Şekil 3.22: Uygulama bilgisi bölümü.

Sonuç sayfasının ikinci kısmında uygulamanın kullandığı tüm izinler, izinlerin risk kategorileri ve risk tehlike düzeyleri listelenmektedir. Toplam 26 adet risk kategorisi ve 4 adet (Yüksek, Orta, Normal, Düşük) risk düzeyi oluşturulmuştur. Buradaki risk kategorileri ve tehlike düzeyleri Android işletim sistemine ait açıklamalarda yer alan bilgilere dayanılarak oluşturulmuştur. Tablo 3.7 risk kategorilerini ve Şekil 3.23 izin detaylarını göstermektedir.

Tablo 3.7: Risk kategorileri ve açıklamaları.

Risk Kategorisi	Risk Açıklaması
Mesaj	Uygulama metin mesajı, sesli ya da görüntülü mesaj gönderme, alma, mesajlara erişim izni istemiştir.
Kişisel Bilgi	Uygulama takvim, kişi listesi, arama geçmişi, İnternet geçmişi vs. gibi bilgilere erişim izni istemiştir.
Konum Bilgisi	Uygulama GPS ya da baz istasyonu aracılığı ile aldığı konum bilgisine erişim izni istemiştir.
Ağ	Uygulama ağa bağlanma, bağlantı durumu, ağ durumunu değiştirme gibi işlemlere erişim izni istemiştir.
Bluetooth	Uygulama bluetooth bağlantısı kurma, bluetooth aracılığı ile haberleşme işlemleri için izin istemiştir.
Sistem	Uygulama sistemi yeniden başlatma, fabrika ayarlarına döndürme, sistem bilgisi alma vs. gibi tehlikeli sayılabilecek işlemlere erişim izni istemektedir.
Pil	Uygulama pil ile ilgili istatistiki bilgilere ya da pil düzeyini etkileyebilecek titreşim, flaş ışığı gibi özelliklere erişmek için izin istemiştir.
Ses	Uygulama cihazın ses ayarlarını değiştirebilmek için izin istemiştir.
Donanım	Uygulama bazı donanımlara (usb, mtp, seri port) erişim izni istemiştir.
Mikrofon	Uygulama mikrofon ile ilgili işlemlere (ses kayıt etme gibi) erişim izni istemiştir.
Kamera	Uygulama cihazın kamerası kullanılarak yapılacak işlemler (fotoğraf çekme) için izin istemiştir.
Arama	Uygulama kişi arama, gelen aramalara cevap verme ya da aramanın durumu (bağlandı, bekliyor) gibi özelliklere erişmek için izin istemiştir.
Depolama	Uygulama cihazın dahili ya da harici hafızasında işlem yapmak (silme, yazma gibi) için izin istemiştir.
Ekran	Uygulama ekran koyucu işlemleri, mesaj penceresi gösterme, ekran oryantasyonunu değiştirme gibi işlemler için izin istemiştir.
Uygulama	Uygulama sistemde var olan ya da arka planda çalışan uygulamalarla ilgili bilgilere erişmek için izin istemiştir.
Saat	Uygulama saat, bölge ayarlama, alarm kurma, silme gibi işlemler için izin istemiştir. ile ilgili işlemleri yapmak için izin istemiştir.
Senkronizasyon	Uygulama bilgilerin senkronizasyonu ile ilgili ayarlara erişmek, ayarları değiştirmek gibi işlemleri yapabilmek için izin istemiştir.
Geliştirici	Uygulama sadece geliştiricilerin kullanabileceği özelliklere erişmek için izin istemiştir.
Klavye	Uygulama klavye ayarlarını değiştirme, yazılan karakterleri izleme gibi işlemler için izin istemiştir.
Bulut	Uygulama uzak bir sunucuya bağlantı kurma, bulut aracılığı ile haberleşme ya da bildirim gönderme gibi işlemler için izin istemiştir.
Reklam	Uygulama reklam servislerine erişebilme, uygulama içi ürün yerleştirme gibi işlemler için izin istemiştir.
Kısayol	Uygulama kısayol oluşturabilme, silebilme gibi işlemler için izin istemiştir.

Analiz Sonuçları | Detaylı Rapor

Kullanılan İzin	Risk Kategorisi	Risk Seviyesi
ACCESS_FINE_LOCATION	Konum Bilgisi	Çok Yüksek
INTERNET	Ağ	Çok Yüksek
WRITE_SMS	Mesaj	Çok Yüksek
ACCESS_NETWORK_STATE	Ağ	Normal
ACCESS_COARSE_LOCATION	Konum Bilgisi	Çok Yüksek
WAKE_LOCK	Pil	Normal
READ_PHONE_STATE	Arama	Çok Yüksek
READ_SMS	Mesaj	Çok Yüksek

Şekil 3.23: İzin detayları.

Sonuç sayfasının son kısmında detaylı raporlara yer verilmiştir. Bu kısımda kullanılan her bir izinin detaylı açıklamasına ve kullanılmaları durumunda neden olabilecekleri risklere yer verilmiştir. Kullanıldığında daha fazla zarara neden olabilecek izinler '**Lütfen Dikkat!**' ibaresi kullanılarak koyu bir font ile yazılmış ve kullanıcılar uyarılmıştır. Şekil 3.24 detaylı rapora ait ekran görüntüsünü göstermektedir.

Analiz Sonuçları | Detaylı Rapor

Aşağıda bu uygulamanın kullandığı izinler, nasıl kullanıldıkları ve uygulamanın kullanılması durumunda karşılaşılabilecek riskler listelenmiştir.

! İZİN AÇIKLAMALARI VE MUHTEMEL RİSKLER

ACCESS_FINE_LOCATION : Uydudan aldığı konum bilgisine erişmeye izin verir. **Lütfen Dikkat!** Bu uygulama o an uydudan alınan nerede olduğunuz bilgisini görebilir ve o anki konumunuzu istenmeyen kişilerle paylaşabilir. Pil ömrünüzü azaltabilir.

INTERNET : İnternete erişmeyi sağlar.

WRITE_SMS : Uygulamanın sim kartta ya da telefonda depolanan SMS mesajlarını değiştirmesine izin verir. **Lütfen Dikkat!** Bu uygulama kişisel mesajlarınızı sizden habersiz silebilir.

ACCESS_NETWORK_STATE : Herhangi bir ağa bağlı olup olmadığınız bilgisine erişmeye izin verir.

ACCESS_COARSE_LOCATION : Mobil ağ veritabanına erişerek yaklaşık konumunuzu bulmaya izin verir.**Lütfen Dikkat!** Bu uygulama yaklaşık konumunuzu görebilir ve istenmeyen kişilerle paylaşabilir.

WAKE_LOCK : Uyku durumunu önlemeye izin verir.**Lütfen Dikkat!** Bu uygulama uyku durumuna geçişi engelleyerek pil kullanımını artırabilir.

READ_PHONE_STATE : Cihazın telefon özelliklerine erişime izin verir. **Lütfen Dikkat!**Bu uygulama isteğiniz dışında arayan, aranılan kişinin telefon numarasına erişebilir.

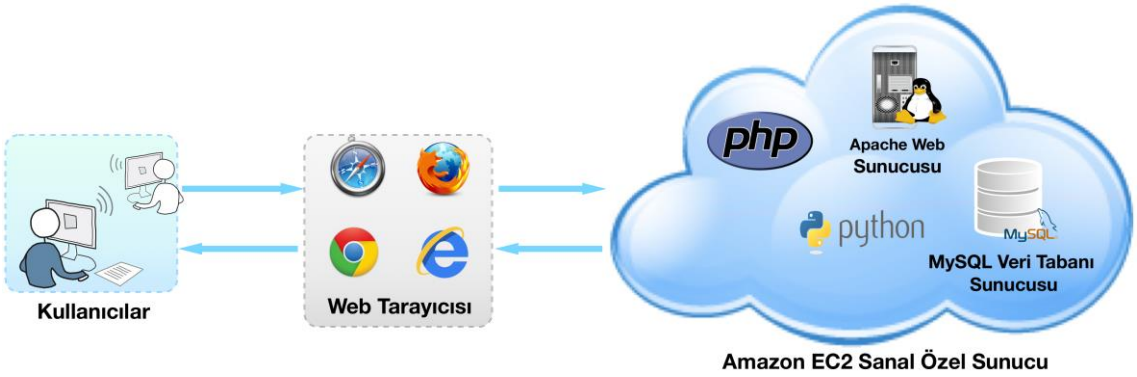
READ_SMS : Uygulamanın sim kartta ya da telefonda depolanan SMS mesajlarını okumasına izin verir. **Lütfen Dikkat!** Bu uygulama kişisel mesajlarınızı okuyabilir, gizliliğinizi tehlikeye düşürebilir.

Şekil 3.24: Detaylı rapor.

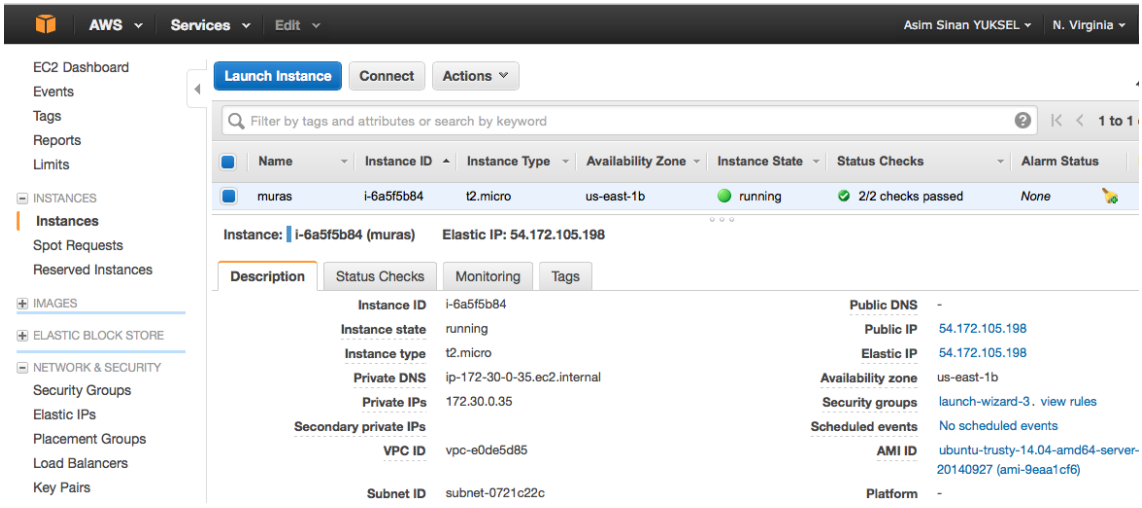
Geliştirmiş olduğumuz sistem Amazon Elastik İşlem Bulutu (Amazon Elastic Compute Cloud-EC2)[3] üzerinde çalışmaktadır. EC2, bulut ortamında genişleyebilen işlem kapasitesi sunan bir web servisi olarak tasarlanmıştır. Basit, sade bir arayüz vasıtasıyla konfigürasyon yapmayı ve Amazon'un gelişmiş ortamında kaynakları, sunucuları etkin,

[3] Amazon EC2, http://aws.amazon.com/ec2/, [Erişim tarihi: 12.06.2015]

tam kontrol sağlayacak şekilde yönetmeyi sağlar. Kurulan sanal sunucular sadece bir kaç dakika içinde başlatılabilir ve kapatılabilir. Kurulan sistemlere ait detaylı bilgiler de sağlanan arayüz aracılığı ile anlık olarak görülebilir. Şekil 3.25 sistemin Amazon ekosistemindeki bulut mimarisini göstermektedir ve Şekil 3.26'de EC2 üzerinde çalışan sistemimize ait ekran görüntüsü yer almaktadır.



Şekil 3.25: Sistemin bulut mimarisi.



Şekil 3.26: Amazon EC2 üzerinde çalışan sistem.

Şekil 3.26'de de görüldüğü üzere Amazon kurulan sanal sunucumuza tam yetkiyle erişmemizi, değişiklik yapmamızı sağlayan bir arayüz sunmaktadır. Bu sayede sunucumuza IP atayabilir, güvenlik ilkeleri oluşturabilir, sistemin sağlıklı çalışıp çalışmadığını kontrol edebiliriz. Kurmuş olduğumuz sistem AMD tabanlı 64 bit işlemciye sahip olup, Ubuntu Trusty 14.04 üzerinde, 54.172.105.198 IP'si ile hizmet sunmaktadır. Arka planda Apache sunucu [102], MySQL sunucusu [103] hizmet vermektedir. Web tarafının çalışması için PHP altyapısı, analiz servisinin çalışması için

Python altyapısı kurulmuştur. Kullanıcılar sistemimize Türkçe arayüzle[4], ya da İngilizce arayüzle[5] ulaşabilirler, risk analizi yapabilirler.

[4] Türkçe arayüz, http://54.172.105.198/muras/AnaSayfa.php, [Erişim tarihi: 12.06.2015]
[5] İngilizce arayüz, http://54.172.105.198/mraas/Home.php, [Erişim tarihi: 12.06.2015]

4. BULGULAR

Mobil güvenlik alanında çok çeşitli çalışmaların yapıldığı görülmektedir. Geliştirilen çoğu çözüm kullanıcı odaklı değildir ya da kullanıcıya ulaşmamıştır. Kimi çözümler antivirüs yazılımı şeklinde geliştirilmiştir ve kullanıcıların cihazlarında aşırı yüke neden olmaktadır. Kimi çözümler ise Android işletim sisteminde değişiklik yaparak güvenliği sağlamaya çalışmıştır. Ancak, bu tarz çözümler de cihazdan bağımsız çalışamamaktadır ve resmi olarak kullanıcıların hizmetine sunulmamıştır.

Geliştirdiğimiz bulanık mantık tabanlı güvenlik risk analiz sistemi, mobil güvenlik konusunda yeni bir yaklaşım sunmaktadır. Sistemizin diğer çalışmalardan farkı ve üstünlüğü, cihazdan bağımsız bir şekilde çalışmasıdır ve kullanılan bulanık mantık modeli sayesinde hangi uygulamaların yüksek riskli olduğunu tespit edebilmesidir. Geliştirdiğimiz servis İnternet ortamında kullanıcıların hizmetine sunulmuştur. Sisteme web tarayıcısı vasıtasıyla erişen kullanıcılar; ister Google Play Uygulama Mağazası'nda arama yaparak isterse indirmiş oldukları uygulamanın ".apk" uzantılı dosyasını sistemimize yükleyerek güvenlik risk analizi yapabilirler, cep telefonlarına ya da diğer mobil cihazlarına uygulamaları kurmadan uygulamalar hakkında anlaşılır, detaylı bilgiler alabilirler. Daha uygulama cihaza kurulmadan kullanıcının ne kadar riske maruz kalacağı gösterilir ve kullanıcı bilgilendirilir. Bu sayede kullanıcı, kişisel güvenlik ve gizliliğini tehlikeye atacak riskleri görerek uygulamayı yükleyip yüklememe konusunda sağlıklı karar verebilmektedir.

Ek olarak, literatürde yapılan Android tabanlı güvenlik çözümleri kendi çalışmamızda gerçekleştirmiş olduğumuz servis tabanlı yaklaşımla karşılaştırılmıştır. Karşılaştırmada kullandığımız kriterler şunlardır:

- *Cihaz Yükü* *:* Önerilen çözüm kullanıcının akıllı cihazında ek bir yüke (bellek ya da işlemci yükü) neden oluyor mu?
- *Kullanılabilirlik* *:* Önerilen çözüm son kullanıcılara hitap ediyor mu?
- *Cihaza Bağımlılık* *:* Önerilen çözümün akıllı cihazlara bağımlılığı var mı?

- *Erişilebilirlik :* Önerilen çözüm erişilebilir mi? Son kullanıcının hizmetine sunulmuş mu?

Yukarıdaki kriterler göz önünde bulundurulduğunda, işletim sistemi tabanlı çözümlerin doğrudan sistemde değişiklik yapması ve değişiklik yapılan sistemin akıllı cihazlara yüklenmesi nedeniyle bu çözümler herhangi bir ek yüke neden olmamaktadırlar. Ancak, kullanıcıların bu tarz çözümlerden fayda sağlayabilmeleri için değişiklik yapılmış işletim sisteminin kullanıcıların akıllı cihazlarına yüklenmesi gerekmektedir. Dolayısıyla, işletim sistemi tabanlı çözümler cihazdan bağımsız çalışamamaktadırlar. Ayrıca, bu çözümlerin resmi olarak yayınlanmaması erişilebilirliklerini engellemiştir.

İzin tabanlı çözümler, Android uygulamalarda yer alan izin listesinde değişikliğe giden ve cihaza bağımlı çalışan çözümlerdir. İzinlerde değişiklik yapılması hem bellek hem de işlemci açısından ek yüklere neden olmaktadır. Bu çözümler kullanılabilir olmalarına rağmen, son kullanıcının hizmetine sunulmamıştır.

Kaynak kod tabanlı çözümler, akıllı cihazlarda çalışırlar ve uygulamaların kaynak kodlarında değişikliğe sebebiyet verirler. Kodların cihaz üzerinde değiştirilmesi hem bellek hem de işlemci bakımından ek yüklere neden olmaktadır. Bu tarz çözümler araştırmacılar için erişilebilir olsa da, son kullanıcının hizmetine sunulmamıştır.

Uygulama tabanlı çözümlerin doğrudan cihaz üzerinde çalışması, bu çözümleri cihaz bağımlı yapmaktadır. Arka planda sürekli çalışmalarından dolayı da bellek ve işlemci açısından ek yüklere sebep olmaktadırlar. Servis tabanlı çözümler ise diğer çözümlerden daha etkilidirler. Çünkü, servis tabanlı çözümler; güvenliğin buluta kaydırılmasıyla cihazdan bağımsız çalışabilirler, İnternet ortamında son kullanıcının hizmetine sunulmalarıyla da erişilebilirlik imkanı sağlarlar. Tablo 4.1, güvenlik çözümlerine ait değerlendirmemizi göstermektedir.

Tablo 4.1: Güvenlik çözümlerinin değerlendirilmesi.

Güvenlik Çözümleri	Cihaz Yükü	Kullanılabilirlik	Cihaza Bağımlılık	Erişilebilirlik
İşletim Sistemi Tabanlı	Yok	Var	Var	Var
İzin Tabanlı	Var	Var	Var	Yok
Kaynak Kod Tabanlı	Var	Yok	Var	Yok
Uygulama Tabanlı	Var	Var	Var	Var
Servis Tabanlı	Yok	Var	Yok	Var

Tablo 4.1'de görüldüğü üzere, servis tabanlı çözümler; cihazdan bağımsız çalışmaları, kullanılabilir ve erişilebilir olmaları, cihazda ek yüke yol açmamaları nedenleriyle daha etkilidir. Çalışmamızda gerçekleştirdiğimiz sistem; uygulamaları akıllı cihazlardan bağımsız bir şekilde incelemesi, uygulamaların kaynak koduna inmesi, ilgili izinleri analiz ederek servis tabanlı bir yaklaşım sunması nedenleriyle hibrit bir çözümdür. Geliştirdiğimiz sistemin yenilikçi yönlerini ve sağladığı avantajları özetleyecek olursak:

- ✓ Bulanık mantık tabanlı risk puanlama yöntemi kullanan özgün bir yapı
- ✓ Güvenlik ve gizliliğe yönelik risk analizi yapma
- ✓ Cihazdan bağımsız çalışma
- ✓ Kullanıcı odaklı çözüm sunma
- ✓ Riske maruz kalmadan kullanıcıyı bilgilendirme
- ✓ Anlaşılır raporlar sunma
- ✓ Bulut tabanlı ve ücretsiz olma
- ✓ Genişletilebilir ve yeni yaklaşımlara açık olma

4.1. RİSK ANALİZ SONUÇLARI

Bir uygulamada zararlı yazılım tespit edilmemesi, o uygulamanın riskli olmadığı anlamına gelmez. Uygulamalar zararlı kod parçacıkları içermeden, kişisel bilgileri izinler aracılığıyla elde ederek de kullanıcılara zarar verebilir. Buradan hareketle, geliştirdiğimiz servis; kötücül uygulamaları tanıyan bir antivirüs yazılımı olmamasına rağmen, kötücül uygulamaları risk oluşturma ve ilgili izinler açısından incelediğinde bu uygulamalara yüksek risk puanları atayan sonuçlar üretmektedir.

Servisimizin ne kadar etkili çalıştığını görebilmek için; 179 farklı kategoride 6000 zararlı uygulama ve geliştirdiğimiz web örümceği aracılığıyla Google Uygulama Mağazası'ndan indirdiğimiz 27 kategoride popüler (en çok indirilen) 100 uygulama ayrı ayrı incelenmiştir. Her bir kategori kendi içinde değerlendirilmiştir. Kategori bazında ortalama risk puanları oluşturulmuştur. Tablo 4.2, Google Uygulama Mağazası'ndan indirdiğimiz uygulamalar için kategori bazında ürettiğimiz ortalama risk puanlarını göstermektedir. Tablo 4.3 ise, zararlı uygulamalar için elde edilen kategori bazında risk puanlarını belirtmektedir.

Tablo 4.2: Mağaza uygulamalarının kategori bazında ortalama risk puanları.

Kategori	Ortalama Risk Puanı	Kategori	Ortalama Risk Puanı
Küçük Uygulamalar	48	Karikatür	37
Haberleşme	47	Hava Durumu	37
Fotoğrafçılık	45	Medya Ve Video	36
Verimlilik	44	Kişiselleştirme	36
Seyahat Ve Yerel	43	Ulaşım	36
Haberler Ve Dergiler	42	Eğlence	35
Sosyal	42	Finans	34
Araçlar	41	Oyun	33
İş	40	Alışveriş	38
Müzik Ve Ses	38	**Genel Ortalama**	**39,57**

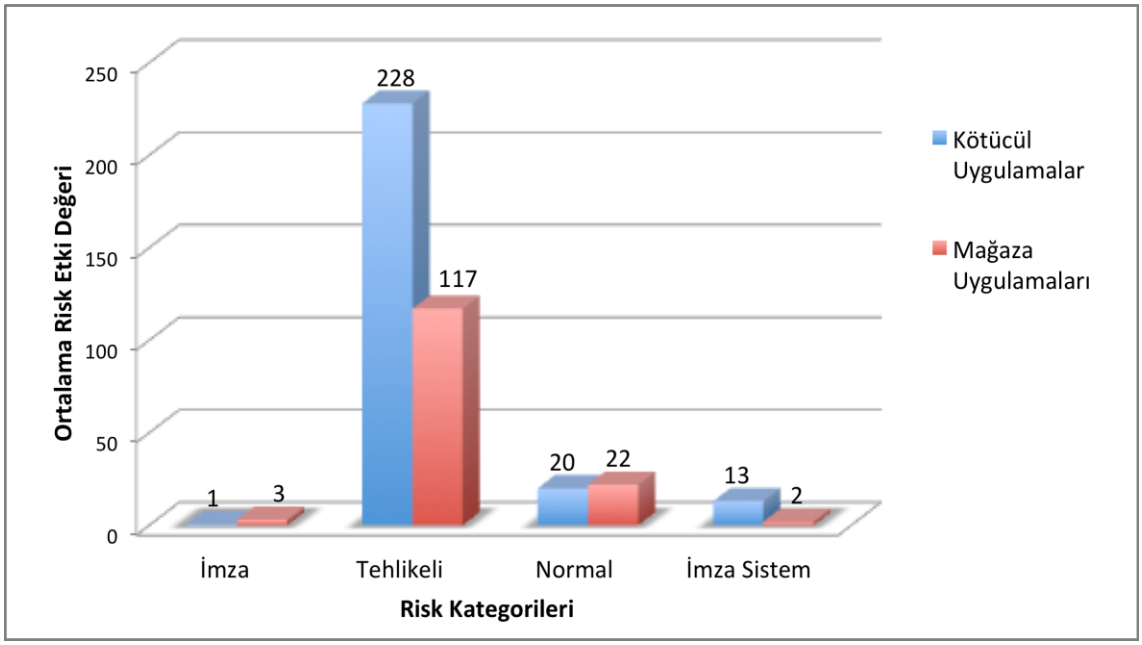
Tablo 4.3: Zararlı uygulamaların kategori bazında ortalama risk puanları.

Kategori	Ortalama Risk Puanı	Kategori	Ortalama Risk Puanı
YZHC	94	ADRD	70
Plankton	93	DroidKungFu3	68
Geinimi	92	jSMSHider	65
KMin	92	BaseBridge	62
Pjapps	83	DroidDream	62
AnserverBot	82	DroidKungFu2	60
zHash	82	Zsone	60
Beanbot	81	DroidKungFu4	55
GoldDream	78	DroidDreamLight	55
DroidKungFu1	74	**Genel Ortalama**	**74,1**

Tablo 4.2 ve 4.3'deki sonuçlara bakıldığında, zararlı uygulamaların ortalama risk puanı ile Google Uygulama Mağazası'ndan indirilen uygulamaların risk puanı arasında büyük bir fark olduğu görülmektedir. Servisimiz, zararlı uygulamalara daha yüksek risk puanları atamıştır. Mağaza uygulamaları içerisinde "Küçük Uygulamalar (Widgets)" kategorisindeki uygulamaların maksimum ortalama risk puanı %48 iken, zararlı uygulamalar içerisinde "DroidDreamLight" kategorisindeki uygulamaların minimum ortalama risk puanı %55'tir.

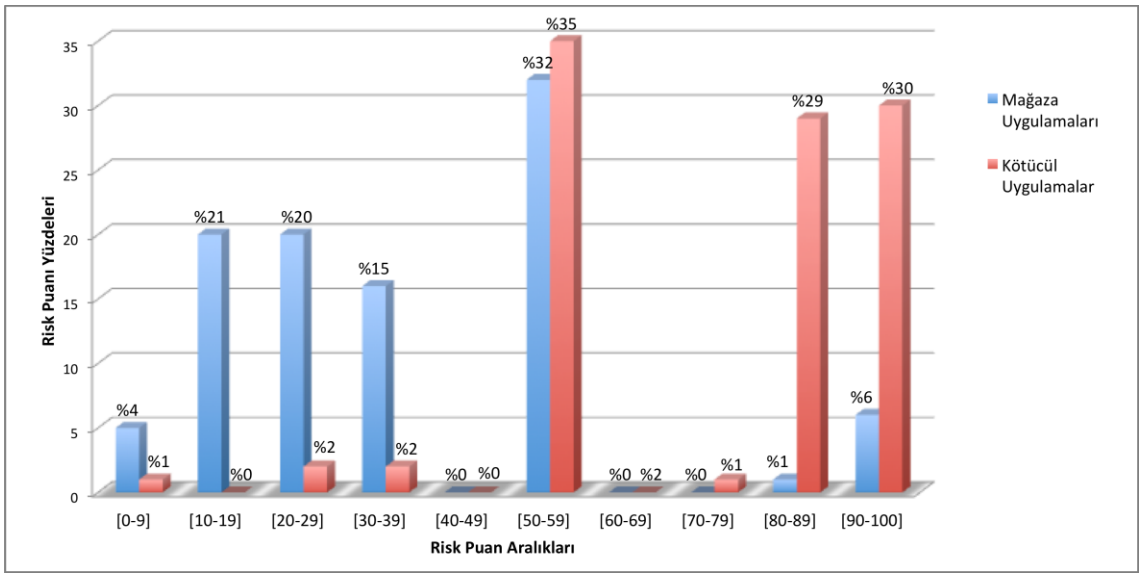
Mağaza uygulamalarının ve kötücül uygulamaların hangi tür izinleri (hangi risk kategorileri) daha çok kullandıklarını tespit etmek ve kullanılan izinlerin risk puanına etkilerini görmek için ayrı ayrı analizler yapılmıştır. Risk puanına etki, aynı zamanda bulanık mantık tabanlı risk hesaplama algoritmamızın girişlerini oluşturmaktadır. Durulaştırma aşamasında sonuç olarak üretilen risk puanı 0-100 aralığında iken, risk etki değerleri yüzün üzerine çıkabilmektedir. Üyelik fonksiyonlarının çalıştırılmasıyla

bu değerler 0-1 aralığında bir sayıya karşılık gelecektir. Etki değeri yüzün üzerine ne kadar çıkarsa çıksın, üyelik fonksiyonunun sınırı yüz olduğu için bu sınırın üzerindeki tüm değerler üyelik fonksiyonu tarafından 1 sayısına eşlenecektir. Şekil 4.1, oluşturduğumuz dört risk kategorisini ve her bir risk kategorisinin ortalama risk etki değerini; Şekil 4.2 ise, risk puanlarının mağaza uygulamaları ve kötücül uygulamalar için 0-100 arasındaki dağılımını göstermektedir.



Şekil 4.1: Kullanılan izinlerin risk kategorileri ve ortalama risk etki değerleri.

Şekil 4.1'den görüldüğü gibi, kötücül uygulamalar ve mağaza uygulamaları "Tehlikeli" risk kategorisinde yer alan izinleri çok fazla kullanarak risk etki değerlerini yükseltmiştirler. "İmza" ve "Normal" risk kategorisindeki ortalama risk etki değerlerine bakıldığında bir benzerlik görülsede, "İmza Sistem" ve "Tehlikeli" risk kategorisindeki ortalama risk etki değerlerinde çok net görülebilen farklılıklar vardır. Mağaza uygulamaları "Tehlikeli" risk kategorisinde ortalama 117 puan üretirken, kötücül uygulamalar 228 (yaklaşık 2 katı) puan üretmiştir. "İmza Sistem" kategorisinde ise bu fark daha da belirgin olup, kötücül uygulamalar mağaza uygulamalarının ürettiği puanın yaklaşık 6 katına eş değer puan üretmiştirler. Android geliştirici sitesindeki açıklamalara bakıldığında, "İmza Sistem" kategorisindeki izinlerin sadece Android işletim sistemi imajındaki uygulamalar tarafından kullanılması ve geliştiricilerin bu izinleri kullanmaması gerektiği belirtilirken, kötücül yazılımlar ısrarla bu izinleri kullanmaya çalışmıştır.

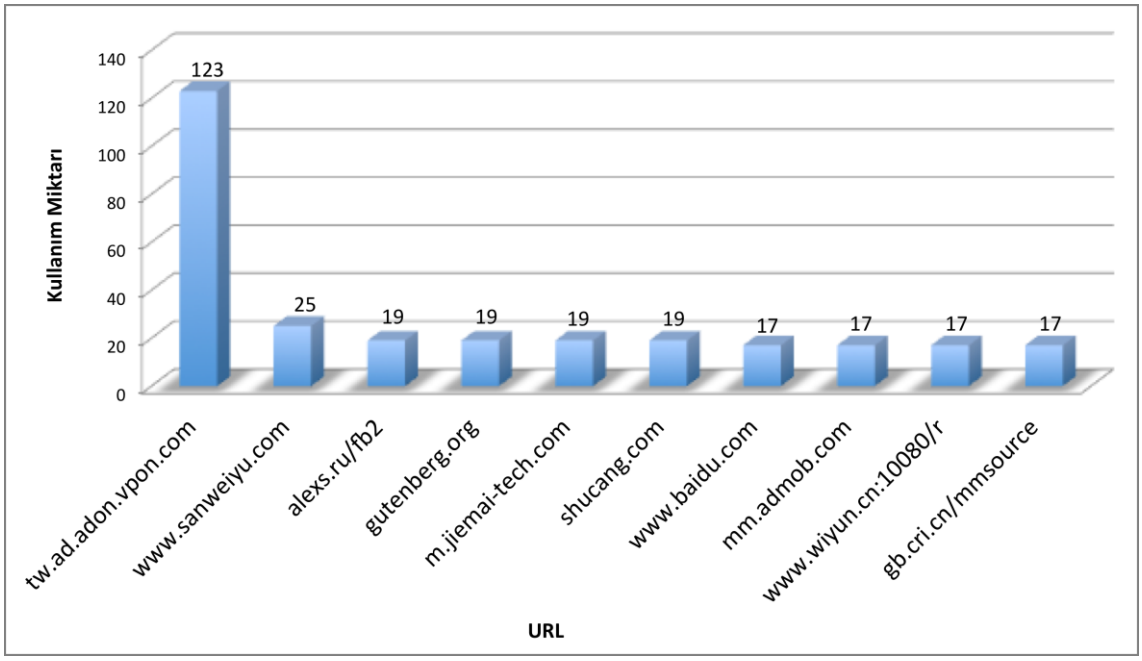


Şekil 4.2: Uygulamaların Risk dağılımı.

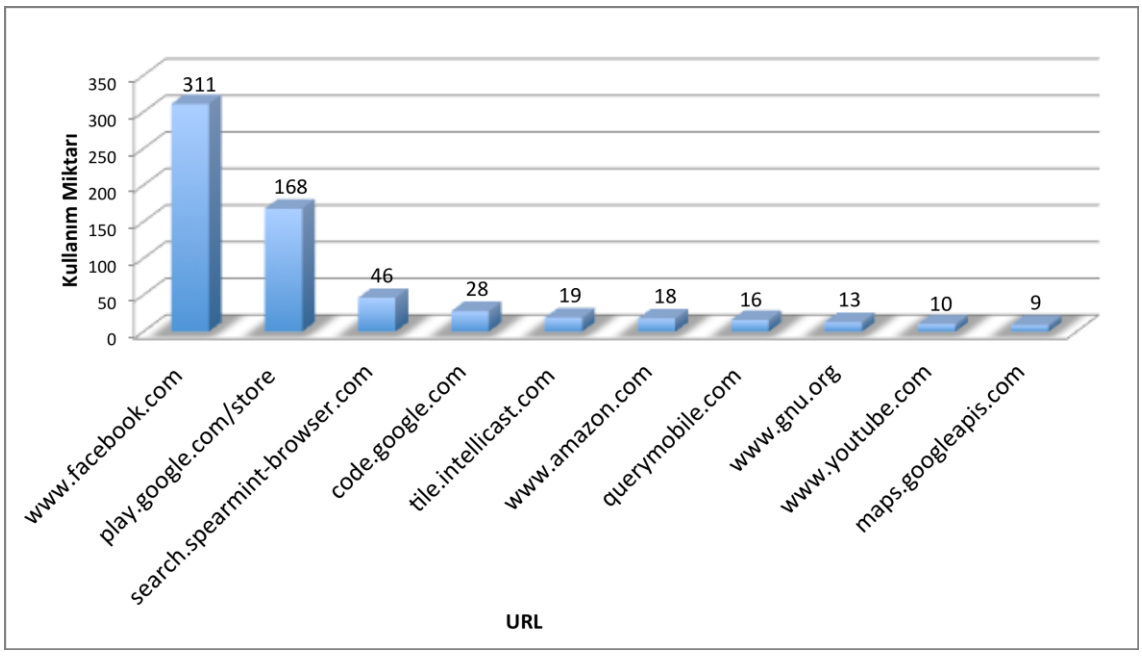
Şekil 4.2'deki risk dağılımı dikkatle incelendiğinde, mağaza uygulamaları ve kötücül uygulamalar arasında çok net farklar bulunduğu görülmektedir. Mağaza uygulamaları daha az riske neden olmaktadırlar. Bunun aksine, kötücül uygulamalar daha fazla riske sebebiyet vermektedirler. Mağaza uygulamalarının risk puanları 0-60 aralığında dağılım gösterirken, kötücül uygulamalar 50-100 arasında risk puanı dağılımı göstermektedir. Mağaza uygulamalarının %32'si ve kötücül uygulamaların %35'i 50-59 arasında risk puanları üretmiştir. En farklı risk puanı dağılımı ise 80-100 aralığındadır. Mağaza uygulamalarının sadece %7'si, kötücül uygulamaların %59'u (mağaza uygulamalarının yaklaşık 8.5 katı) bu aralıkta risk puanına sahiptir.

Yukarıdaki analizlere ek olarak, kötücül uygulamaların ve mağaza uygulamalarının davranışlarındaki farklılıkları gözlemlemek için iki farklı analiz gerçekleştirilmiştir. Bu analizlerden ilki; uygulamalara gömülmüş web adreslerini (URL), ilginç sayılabilecek sayıları, güvenlik riskine neden olabilecek Linux kabuk komutlarını (Linux shell commands) inceler. İkinci analiz ise; uygulamaların, izin dosyasında tanımladıkları izinleri gerçekten kullanıp kullanmadıklarını, eğer kullanmamışlarsa da bu izinlerin ne kadarını kullanmadıklarını (gereksiz izin kullanımı) inceler. Böyle bir analiz yapmak için uygulamaların kaynak koduna inilmiştir ve [64]'ki çalışmada çıkarılan izin haritası kullanılarak izin dosyasındaki tanımlı izinlere yönelik API düzeyinde çağırım yapılıp yapılmadığı araştırılmıştır. Manifesto dosyasında tanımlandığı halde API düzeyinde kullanılmamış izin sayısı, manifesto dosyasında tanımlı toplam izin sayısına bölünerek

bir oran hesaplanmıştır. Bulunan oran daha sonra "%" olarak tanımlanmıştır. Şekil 4.3, kötücül yazılımların kullandıkları URL'leri belirtmektedir. Şekil 4.4 ise, mağaza uygulamalarının kullandıkları URL'leri sunmaktadır. Tablo 4.4, kötücül uygulamaların kullandıkları Linux kabuk komutlarını ifade etmektedir. Şekil 4.5 ve Şekil 4.6 sırasıyla, kötücül uygulamaların ve mağaza uygulamalarının kendi kategorileri bazında ne kadar gereksiz izin kullandıklarını göstermektedir.



Şekil 4.3: Kötücül uygulamaların kullandığı URL'ler ve kullanım miktarları.



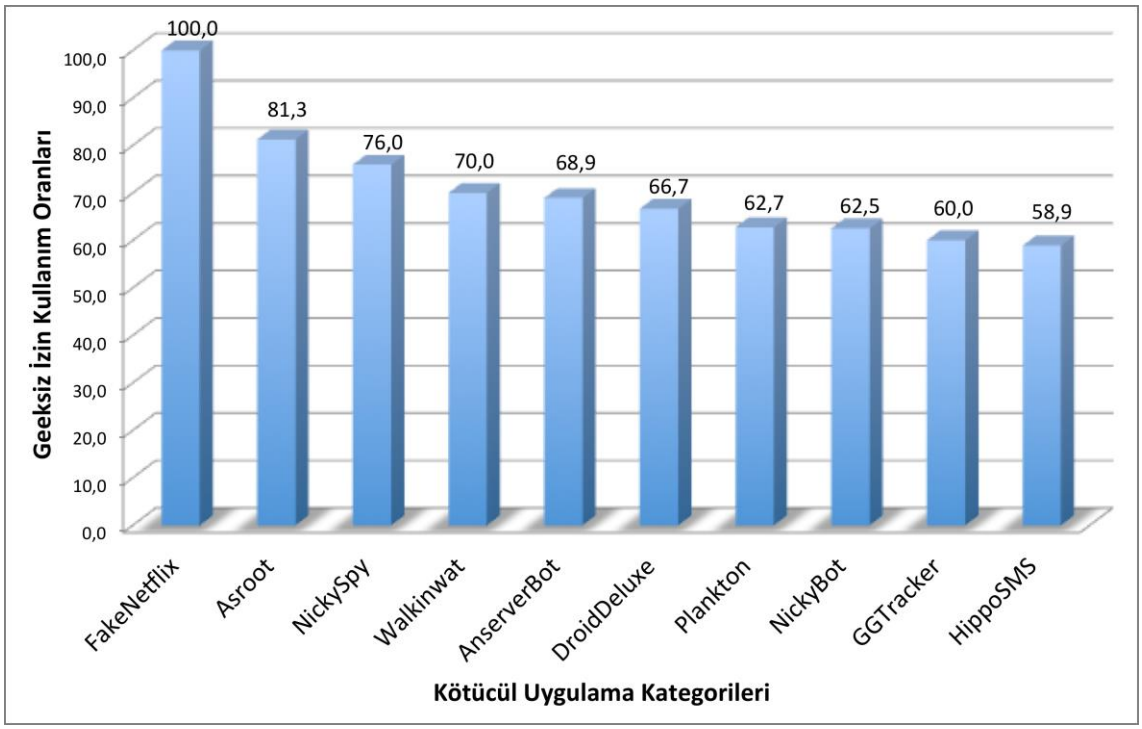
Şekil 4.4: Mağaza uygulamarının kullandığı URL'ler ve kullanım miktarları.

Şekil 4.3 ve Şekil 4.4 dikkatli bir şekilde incelendiğinde, kötücül uygulamaların ve mağaza uygulamalarının kullandıkları URL'lerin tamamen farklı olduğu görülmektedir. Buradan hareketle; mağaza uygulamaları Facebook, Google Play Uygulama Mağazası, Amazon, Youtube gibi bilinen URL'leri kullanmakta iken, kötücül uygulamaların ise bilinmeyen, İnternet üzerinde var olmayan, şüpheli, bilgi sızdırabilecek ve uzak doğu menşei URL'leri kullandıkları gözlemlenmiştir.

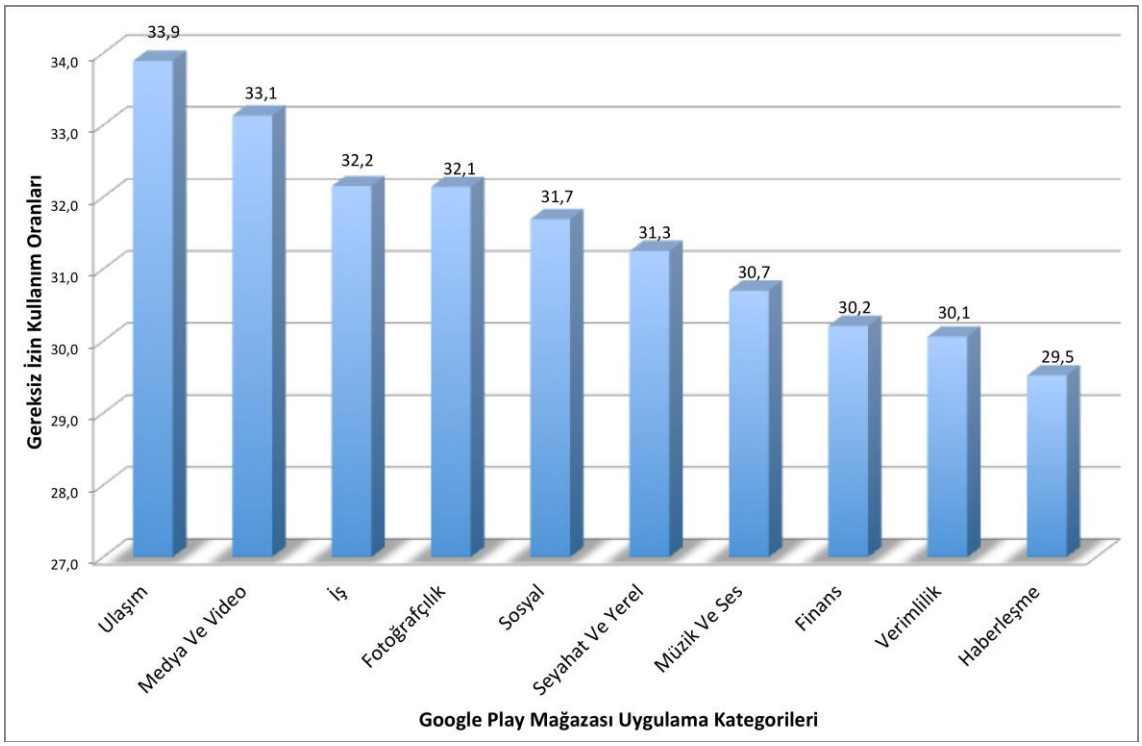
Tablo 4.4: Kötücül uygulamalar tarafından kullanılan Linux kabuk komutları.

Komut	**Açıklama**
chmod	Sistemdeki dosya ya da klasörlere olan erişim izinlerini değiştirir.
Sh	Dosyadan, komut satırından, standart girişlerlerden girilen komutların çalıştırılmasını sağlar.
Su	Kullanıcı değiştirmeyi sağlar. Varsayılan olarak super kullanıcı (tam yetki) olmayı sağlar.
mount	Bir dosya sistemini başka bir dosya sistemine bağlamayı sağlar.
insmod	Linux çekirdeğine bir modül eklemeyi sağlar.
chown	Dosya ya da dosya gruplarının sahiplerini değiştirmeyi sağlar
stop vold	Android sisteminde arka planda çalışan program parçalarını durdurmayı sağlar.
start vold	Android sisteminde arka planda çalışan program parçalarını başlatmayı sağlar.
Cat	Dosyadan bilgi okuyup içeriğini yazdırmayı sağlar.
remount	Bir dosya sistemini ya da depolama birimini başka bir dosya sistemine bağlar.
setprop	Konfigürasyon dosyalarında değişiklik yapmayı sağlar.
mkdir	Klasör oluşturmayı sağlar.
Cp	Dosya ya da klasör kopyalamayı sağlar.
Mv	Dosya ya da klasör taşımayı sağlar.
exec	Linux sisteminde bir iş parçacığının farklı bir program olarak çalışmasını sağlar.

Yaptığımız analizlerde, mağaza uygulamalarının herhangi bir Linux kabuk komutu kullandığı tespit edilmemiştir. Tablo 4.4'den de görüldüğü üzere, kötücül uygulamaların çok çeşitli Linux komutları kullandıkları anlaşılmaktadır. Bu komutlar; işletildiğinde kullanıcıya zarar verebilecek, kişisel gizliliği ve güvenliği tehlikeye düşürebilecek riskli komutlardır. Ayrıca, kötücül uygulamaların kullandığı ilginç sayılara bakıldığında, "48734154" sayısının çok sık kullanıldığı saptanmıştır. Bu kötücül uygulamalar daha detaylı incelendiğinde, "48734154" sayısının şifreleme amaçlı olduğu belirlenmiştir. Sonuçta, kötücül uygulamalar kişisel bilgileri elde ettikten sonra, kontrol sunucusuna bu bilgileri şifreleyerek göndermektedir.



Şekil 4.5: Kötücül uygulamaların gereksiz izin kullanım oranları.



Şekil 4.6: Google Play Mağazası'ndaki uygulamaların gereksiz izin kullanım oranları.

Şekil 4.5 ve Şekil 4.6, en yüksek gereksiz izin kullanım oranlarına sahip kategorilerden ilk 10 tanesini göstermektedir. Tüm kategoriler incelendiğinde; kötücül uygulamaların

ortalama %41'i ve mağaza uygulamalarının da %27'si gereksiz izin kullanmaktadır. Sonuç olarak kötücül uygulamaların daha fazla gereksiz izin kullandığı tespit edilmiştir.

4.2. RİSK KATEGORİ SAYISININ RİSK PUANINA ETKİSİ

Risk puanının hesaplanmasında kullanılan 4 adet risk kategorisinin Android geliştirici sitesindeki izin koruma düzeyleri dikkate alınarak elde edildiğine, bulanık mantık tabanlı risk puanlama sistemimizin girişlerinin de bu risk kategorileri aracılığıyla oluşturulduğuna daha önce değinilmişti. Risk kategorisi (diğer bir ifadeyle risk girdileri) sayısının artmasının risk puanlamasına etkisini incelemek için iki adet yeni risk kategorisi oluşturulmuştur. İlk kategori, "Parasal Risk" kategorisidir. Bu kategoride; mesaj gönderme, arama yapma, İnternet bağlantısını kullanma gibi yollarla kullanıcıya maddi açıdan zarar verebilecek çeşitli izinler bulunmaktadır. Tablo 4.5, parasal risk kategorisinde yer alan izinleri göstermektedir.

Tablo 4.5: Parasal riske neden olan izinler.

İzin Adı	İzin Bilgisi
SEND_SMS	Uygulamanın mesaj göndermesine izin verir.
SEND_SMS_NO_CONFIRMATION	Kullanıcı girişi ya da izni olmadan mesajlaşma uygulaması aracılığıyla mesaj gönderimine izin verir.
WRITE_SMS	Uygulamanın SIM kartta ya da telefonda depolanan mesajlara yazmasına izin verir.
CALL_PHONE	Uygulamanın numara çevirme ekranı olmadan arama yapmasına izin verir.
CALL_PRIVILEGED	Uygulamanın acil numaralar da dahil olmak üzere herhangi bir numarayı aramasına izin verir.

İkinci kategori, "Gizlilik Riski" kategorisidir. Bu kategori; konum bilgisi, kişi listesi, telefon rehberi, takvim, yapılacaklar, notlar, e-posta gibi kişisel bilgileri kullanarak kullanıcıların gizliliğini tehlikeye düşürebilecek izinleri kapsar. Tablo 4.6, gizlilik riski kategorisindeki izinleri belirtmektedir.

Tablo 4.6: Gizlilik riskine neden olan izinler.

İzin Adı	İzin Bilgisi
READ_PHONE STATE	Uygulamanın telefon özelliklerine erişmesini sağlar.
READ_SMS	Uygulamanın telefonda ya da SIM kartta depolanan mesajlara erişmesini sağlar.
RECEIVE_SMS	Uygulamanın gelen mesajları okumasını ve işlemesini sağlar.
ACCESS_COARSE_LOCATION	Uygulamanın mobil ağ veritabanını kullanarak kullanıcının yaklaşık olarak yerini bulmasına izin verir.
READ_CONTACTS	Uygulamanın telefonda depolanan tüm kişi listesine (adresler de dahil) erişmesine izin verir.
ACCESS_FINE_LOCATION	Uygulamanın GPS bilgisini kullanarak kullanıcının yerini bulmasına izin verir.
READ_LOGS	Uygulamanın telefondaki günlüklere erişimini sağlar. Günlükler telefonda ne yapıldığı ile ilgili bilgileri barındırır.
READ_HISTORY_BOOKMARKS	Uygulamanın kişinin İnternet geçmişini okumasına izin verir.
ACCESS_LOCATION_EXTRA_ COMMANDS	Uygulamanın konum bilgisi sağlayan ek özelliklerine erişimini sağlar.
READ_CALENDAR	Uygulamanın kullanıcının kişisel takvimini okumasına olanak sağlar.
READ_USER_DICTIONARY	Uygulamanın kullanıcının sözlüğüne kaydettiği özel kelimelere, isimlere, cümlelere erişimine izin verir.
READ_CALL_LOG	Uygulamanın kişinin arama geçmişini görmesine izin verir.

Yeni kategoriler dikkate alındığında, toplam 6 adet bulanık mantık girişi elde edilmiştir. Bu kategorilerde yer alan izinlerin hepsi aynı zamanda Android işletim sisteminde "Tehlikeli" olarak işaretlenmiştir. Diğer bir ifadeyle, yeni kategorilerde bulunan izinler "Tehlikeli Risk" kategorisine de dahil olan izinlerdir. Dolayısıyla, ilaveten oluşturulan yeni risk kategorilerinin de etki değerlerine aynı değerler verilmiştir. Oluşturulan yeni kategoriler bir nevi cezalandırma mekanizması şeklinde görev yapmaktadır. Parasal ve gizliliğe neden olan izinlere ek puanlar eklenerek bu izinlerin kullanımları cezalandırılmaktadır. Tablo 4.7, yeni risk kategorileri ve etki değerlerini göstermektedir.

Tablo 4.7: Yeni risk kategorileri ve etki değerleri.

Risk ID	Risk Kategorisi	Dilsel Değer	Etki Değeri
1	Tehlikeli Risk (TR)	Yüksek	25
2	İmza Ya da Sistem Riski (ISR)	Orta	15
3	İmza Riski (IR)	Düşük	10
4	Normal Risk (NR)	Çok Düşük	5
5	Parasal Risk (PR)	Yüksek	25
6	Gizlilik Riski (GR)	Yüksek	25

Çalışmamızda geliştirdiğimiz sistem, oluşturulan yeni risk kategorileri ile birlikte tekrar yapılandırılmıştır. Buna göre, sistemimiz yeni risk puanları hesaplamıştır. Tablo 4.8;

yeni ortalama risk puanları ikinci sütunda, eski ortalama risk puanları da birinci sütunda olmak üzere, Google Play Mağazası uygulamaları için sonuçları listelemektedir.

Tablo 4.8: Mağaza uygulamalarının eski ve yeni ortalama risk puanları.

Kategori	Eski Ortalama Risk Puanı	Yeni Ortalama Risk Puanı
Küçük Uygulamalar	48	60
Haberleşme	47	56
Fotoğrafçılık	45	55
Verimlilik	44	52
Seyahat Ve Yerel	43	51
Haberler Ve Dergiler	42	52
Sosyal	42	52
Araçlar	41	51
İş	40	49
Müzik Ve Ses	38	50
Karikatür	37	51
Hava Durumu	37	45
Medya Ve Video	36	50
Kişiselleştirme	36	49
Ulaşım	36	47
Eğlence	35	47
Finans	34	47
Oyun	33	49
Alışveriş	38	47
Genel Ortalama	**39,57**	**48,4**

Sonuçlar analiz edildiğinde, Google Play Mağazası'ndan indirilen uygulamalara ait yeni risk puanlarının ortalama 10 puan arttığı gözlemlenmiştir. Dolayısıyla, yeni eklenen risk kategorilerine bağlı olarak geliştirdiğimiz sistemimiz, Google Play Mağazası'ndan indirilen uygulamalarda parasal ve gizlilik riski taşıyan yeni tehditleri algılamaktadır.

Ek olarak, yeniden yapılandırdığımız bu sistemin kötücül uygulamalara yönelik ürettiği eski, yeni ve genel ortalama risk puanları Tablo 4.9'da gösterilmiştir. Sistemimizde; kötücül uygulamalarda parasal ve gizlilik bakımından kullanıcıya zarar verecek yeni risklerin algılandığı, genelde yeni risklerin ortaya çıkmadığı, buna rağmen bazı kategorilerde ortalama risk puanlarının düştüğü (örneğin; YZHC, Plankton, Geinimi, KMin, Pjapps, DroidKungFu3, DroidKungFu2, Zsone), bazı kategorilerde sabit kaldığı, ADRD ve jSMSHider kategorilerindeki ortalama risk puanlarının da 15 puan arttığı saptanmıştır. ADRD grubundaki zararlı uygulamalar kişisel bilgileri çalmaya yönelik yazılımlardır ve bu nedenle kişisel gizliliği tehlikeye düşürmektedir. jSMSHider

kategorisindeki zararlı uygulamalar ise metin mesajlarını okuma ve mesaj gönderme gibi parasal risklere sebep olan izinleri kullanmaktadırlar. Tüm kötücül uygulama kategorilerinin ortalama risk puanlarında bir yükselme beklenirken, sadece iki kötücül uygulama kategorisindeki (ADRD ve jSMSHider) puanların yükselmesi geliştirdiğimiz sistemde istenmeyen bir durumdur. Bunun sebebi, diğer kategorilerdeki kötücül uygulamaların parasal ve gizlilik riskleri dışında farklı riskler de oluşturmasıdır. Sistemin yeni risk kategorileriyle birlikte çalıştırılması bazen çok sağlıklı sonuçlar üretmeyebilir. Dolayısıyla, istenmedik durumlar kötücül uygulamaları detaylıca inceleyerek ve daha kapsamlı risk kategorileri oluşturarak çözülmelidir.

Tablo 4.9: Kötücül uygulamaların eski ve yeni ortalama risk puanları.

Kategori	Eski Ortalama Risk Puanı	Yeni Ortalama Risk Puanı
YZHC	94	82
Plankton	93	80
Geinimi	92	82
KMin	92	84
Pjapps	83	78
AnserverBot	82	82
zHash	82	82
Beanbot	81	81
GoldDream	78	78
DroidKungFu1	74	74
ADRD	70	85
DroidKungFu3	68	64
jSMSHider	65	81
BaseBridge	62	62
DroidDream	62	62
DroidKungFu2	60	55
Zsone	60	50
DroidKungFu4	55	55
DroidDreamLight	55	55
Genel Ortalama	**74,1**	**72,2**

4.3. BENZER SERVİS TABANLI ÇÖZÜMLER İLE KARŞILAŞTIRMA

Çalışmamıza başladığımızdan bugüne kadar geçen süre içinde geliştirmiş olduğumuz servise benzer çözümlerin ortaya çıktığı görülmüştür. Bu çalışmalardan biri olan

Dexter[6], Android uygulama analizi yapabilen bir yazılım çatısı olarak geliştirilmiştir ve bir web arayüzüne sahip servis niteliğindedir. Geliştirilen araç, normal ve kötücül uygulamalardan çıkarabildiği kadar bilgiyi çıkarıp farklı görünümlerle web ortamında sunmaktadır. Dexter, Python programlama dili kullanılarak bir web uygulaması şeklinde geliştirilmiş, performans kaygısından dolayı da XML önbellekleme yapan SQL veri tabanı kullanmıştır.

Aracın ön plana çıkan özellikleri şu şekilde sıralanabilir:

- APK ayrıştırma
- Manifesto dosyasını gösterme ve analiz etme
- Aktiviteler, yayın alıcılar, izinler hakkında bilgi verme
- Sınıf hiyerarşisi oluşturma
- Derlenmiş dosyadan kaynak kod oluşturma

Diğer bir araç ise Andrubis[7]'tir. Andrubis sistem ve Dalvik sanal makinesi düzeyinde statik ve dinamik analiz yöntemlerini birleştiren tam otomatik çalışan Android uygulama analiz aracıdır. Herkesin ulaşabileceği bir web arayüzü vasıtasıyla mobil uygulamalar yüklenebilmektedir. Günde 3500 adet örnek analiz edebilecek kapasitededir. Şu ana kadar ise toplamda 1.000.000 Android uygulaması analiz edilmiştir.

AVC Undroid[8] Android uygulamalarını statik analiz tekniği ile inceleyen ücretsiz, web tabanlı bir servistir. Kayıtlı kullanıcılar daha büyük boyutlu dosyaları analiz edebilir, analiz edilen uygulamalara yorum yapabilir, analiz için daha yüksek öncelik elde edebilirler. Kullanıcıların yükledikleri uygulamalar dışında, arka planda Android uygulaması toplayabilmektedir. Analiz sonunda gösterilen bilgiler çok çeşitli güvenlik firmalarının araçları kullanılarak elde edilmiştir. Sonuçlar sadece bilgi amaçlı olup dosya hakkında herhangi bir karar vermeye yardımcı olmamaktadır. Dolayısıyla, uygulamaların zararlı ya da kötücül olup olmadığını garanti etmemektedir.

---

[6] Dexter, http://dexter.dexlabs.org, [Erişim tarihi: 12.06.2015]
[7] Anubis, https://anubis.iseclab.org, [Erişim tarihi: 12.06.2015]
[8] AVC Undroid, http://undroid.av-comparatives.info/index.php, [Erişim tarihi: 12.06.2015]

Android Sanbox[9] elde mevcut olan ya da mobil cihazda yüklü olan Android uygulamalarını ücretsiz olarak analiz edebilmektedir. Analiz sonucunda hangi web adresleri ile bağlantı kurulduğu, hangi izinlerin kullanıldığı, uygulamaya ait kaynak kodlar ve ekran görüntüleri gibi bilgiler sunabilmektedir.

Mobile-Sanbox[10] Andrubis'e benzer bir şekilde statik ve dinamik analiz tekniklerini kullanarak web tabanlı Android uygulama analizi yapabilmekte, uygulamalardaki metot çağırımlarını izleyebilmektedir. Statik analiz yöntemi ile uygulamalar hakkında bilgi toplamasının yanında, çok çeşitli anti-virüs yazılımları kullanarak da ek bilgiler elde etmektedir. Şüpheli kod tespiti için de manifesto dosyasını ayrıştırma, derlenmiş uygulama dosyasından kaynak kod oluşturma işlemlerini de yapabilmektedir.

CopperDroid[11] ise sadece dinamik analiz yöntemiyle analiz yapabilmektedir. Geliştirilen araç sanal makine tabanlıdır. Sistem cağrı analizi ve simülasyon yöntemleri kullanarak kötücül uygulamaların analizini yapar ve davranışlarını yeniden oluşturur. Herkese açık olan web arayüzü ile de kullanıcılar APK dosyalarını yükleyerek analiz yapabilirler. Analiz sonuçları çok çeşitli formatlarda gösterilebilir ve sonuçlar ağ trafiği, yeniden oluşturulan normal ve çalıştırılabilir dosyalar, sistem çağrı izleri gibi bilgileri içermektedir.

Web tabanlı diğer bir analiz aracı ise App360Scan[12]'dir. Bu araç diğerlerine göre özellik bakımından en zayıf araçtır ve sadece uygulama aramaya izin vermektedir. Girilen anahtar kelimeye göre uygulamalar aranır, o uygulama daha önce sistem tarafından analiz edilmişse, uygulamanın kullandığı izinler, uygulamaya ait ekran görüntüleri kullanıcıya web ortamında sunulur. Buna ek olarak kullanıclar yapılan analizlere yorum yapabilmektedir. Ancak geliştirilen sistem herhangi bir analiz yöntemi kullanmamaktadır.

Android uygulamalarını analiz etmek için geliştirilen ve aktif olarak çalışan bu web tabanlı servisler inceleyerek sahip oldukları özellikler bakımından geliştirdiğimiz Mobil Uygulama Risk Analiz Servisi (Muras) ile karşılaştırıldı. Çalışmamızda kullandığımız

---

[9] Android Sandbox, http://www.androidsandbox.net, [Erişim tarihi: 12.06.2015]
[10] Mobile-Sandbox, http://mobilesandbox.org, [Erişim tarihi: 12.06.2015]
[11] CopperDroid, http://copperdroid.isg.rhul.ac.uk, [Erişim tarihi: 12.06.2015]
[12] App360Scan, http://www.app360scan.com, [Erişim tarihi: 12.06.2015]

mağaza uygulamaları ve kötücül uygulamalar veri setleri ile her bir servise erişim sağlanarak bu servisler test edildi ve sonuçlar belli kritlerler dahilinde ayrıntılı olarak incelendi. Karşılaştırmadalarımızda kriter olarak kullandığımız özellikler ve bu özelliklere ait açıklalamar şunlardır:

- ***Risk Puanlama:*** Risk puanlama sistemi mevcut mu?
- ***İzin Risk Seviyeleri:*** Kullanılan izinlerin risk seviyeleri gösteriliyor mu?
- ***Rapor:*** Risklere ait detayları gösteren raporlar üretebiliyor mu?
- ***APK Yükleme:*** APK dosyası yüklemeye izin veriyor mu?
- ***Arama:*** Google Play Uygulama Mağazasında uygulama arama yolu ile analiz yapılabiliyor mu?
- ***Gereksiz İzin Tespiti:*** Tanımlandığı halde kullanılmamış izinlerin tespitini yapabiliyor mu?
- ***Komut Tespiti****:* Çalıştırıldığında riske neden olabilecek Linux kabuk komutlarının tespitini yapabiliyor mu?

Tablo 4.10 yukardaki kriterler göz önünde bulundurularak elde edilen karşılaştırma sonuçlarını göstermektedir.

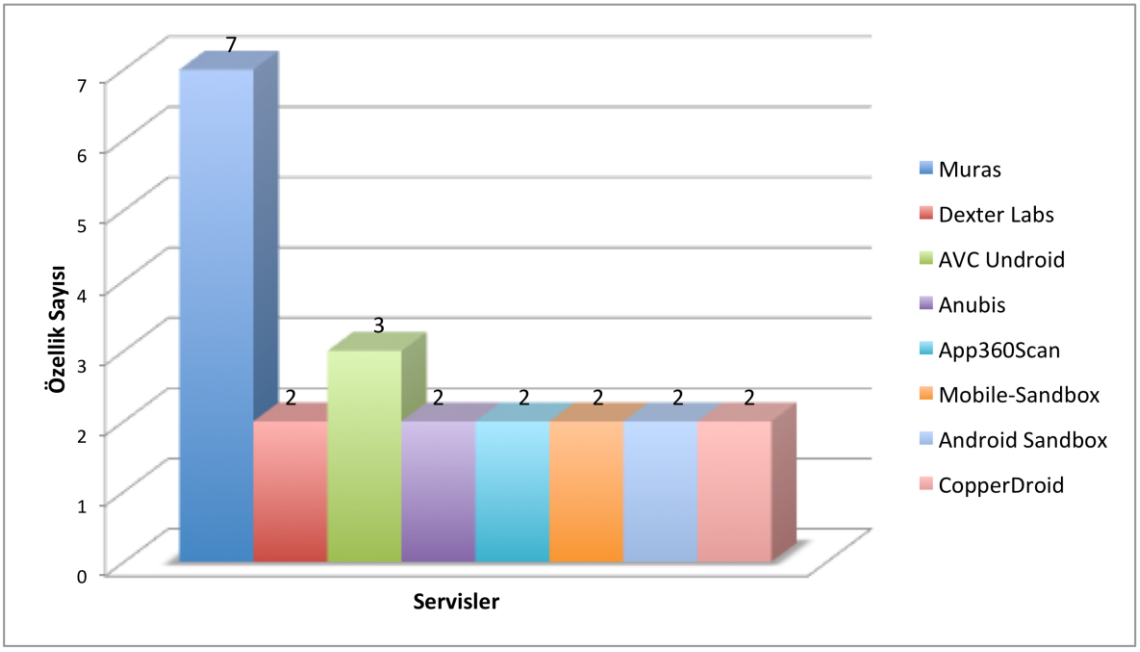
Tablo 4.10: Karşılaştırma sonuçları.

Analiz Servisi	Risk Puanlama	İzin Risk Seviyeleri	Rapor	APK Yükleme	Arama	Gereksiz İzin Tespiti	Komut Tespiti
Muras	Var	Var	Var	Var	Var	Var	Var
Dexter	Yok	Yok	Var	Var	Yok	Yok	Yok
AVC Undroid	Yok	Var	Var	Var	Yok	Yok	Yok
Andrubis	Yok	Yok	Var	Var	Yok	Yok	Yok
App360Scan	Yok	Yok	Var	Yok	Var	Yok	Yok
Mobile-Sandbox	Yok	Yok	Var	Var	Yok	Yok	Yok
Android Sandbox	Yok	Yok	Var	Var	Yok	Yok	Yok
CopperDroid	Yok	Yok	Var	Var	Yok	Yok	Yok

Dexter, Anubis, Mobile-Sandbox, Android Sandbox ve CopperDroid yüklenen APK dosyalarını analiz edebilmekte, raporlama yapabilmektedir. Ancak risk paunlama, izin risk düzeylerini gösterme, Google Uygulama Mağazası'ndaki bir uygulamanın

aratılarak analiz sonuçlarının görülmesi gibi özelliklerden yoksundurlar. Buna ek olarak oluşturulan raporlar teknik düzeyde bilgisi olmayan kullanıcıların anlayacağı düzeyde değildir. AVC Undroid diğerlerinden farklı olarak izinlerin risk seviyelerini göstermektedir. Risk puanlaması yaparak numerik bir risk değeri göstermese de uygulamalara puan olarak çeşitli renkler atamaktadır. Buna rağmen bu renklerin nasıl atandığına, nasıl bir algoritma kullanıldığına yer verilmemiştir. Son olarak, geliştirdiğimiz olduğumuz sistem CopperDroid, Mobile-Sandbox ve Andrubis araçlarının dinamik analiz yöntemi ile sanal makinelerde uygulamaların analizi işlemini içermese de bu tarz bir yöntem sistemimizle kolaylıkla bütünleştirilebilir.

Tablo 4.10'da gösterilen sonuçlar özellik sayısı bakımından incelendiğinde Şekil 4.7'deki Servis-Özellik Sayısı grafiği elde edilmiştir. Şekilden de görüldüğü üzere geliştirmiş olduğumuz sistem diğer sistemlere oranla daha fazla özellik sunmaktadır. Buna ek olarak uygulamanın kullandığı tüm izinlerin elde edilmesi noktasında da yüksek başarı göstermektedir. Diğer servislerin kullanılmasıyla elde edilen izinler ve miktarları, sistemimiz tarafından çıkarılan izinler ve miktarlarıyla birebir örtüşmektedir.



Şekil 4.7: Servis-Özellik sayısı grafiği.

5. TARTIŞMA VE SONUÇ

Bulanık mantık tabanlı sistemler; Bilgisayar Bilimleri ve diğer çoğu alanda verinin az, bilginin yetersiz olduğu zamanlarda risklerin analiz edilmesi ve değerlendirilmesinde sıklıkla kullanılmaktadırlar. İyi tasarlanmış bir bulanık mantık sistemi risklerin anlaşılmasında kolaylık sağlayacaktır. Bu sayede, hangi risklere ne derecede maruz kalındığı ortaya çıkarılabilir ve riskler yönetilebilir. Mobil cihazların popülerliğinin ve kullanımlarının yaygınlaşmasını göz önünde bulundursak, bulanık mantık tabanlı bir sistemin mobil güvenlik alanında da risklerin analiz edilmesi ve değerlendirilmesinde kolaylıkla kullanılabileceği geliştirdiğimiz sistem aracılığıyla doğrulanmıştır.

Bu çalışmada, bulanık mantık tabanlı risk puanlaması yapan mobil uygulama risk analiz servisinin prototipi tasarlanmış ve gerçekleştirilmiştir. Cihazdan bağımsız ve kullanıcı odaklı olan servis, uygulamaların potansiyel risklerini ve risk puanlarını göstermektedir. Geliştirilen sistem; kullanıcıların, bir uygulamayı yükleyip yüklememe noktasında referans alacakları, güvenlik ve gizliliklerini tehdit edecek riskleri daha uygulamayı yüklemeden görebilecekleri bir karar destek mekanizmasıdır.

Gelecek çalışmamızda, sistem performansının arttırılması ve risk puanlama algoritmasının iyileştirilmesi üç aşamada sağlanacaktır. Bulgularda elde ettiğimiz analiz sonuçlarında; risk kategori sayısının arttırılması durumunda sistemimizin yeni riskler yakaladığını ancak yeni risk kategorilerinin kötücül uygulamaların risk puanlarında düşüşe neden olduğunu, zararlı uygulamaların mağaza uygulamalarından farklı olarak kabuk komutları kullandığını, İmza Sistem kategorisindeki izinlerin ise zararlı yazılımlar tarafından daha fazla kullanıldığını tespit etmiştik. Birinci aşamada; bu bulgular ışığında hareket edilecek ve risk puanlama algoritmasına daha kapsamlı sonuçlar çıkarmamızı sağlayacak yeni risk kategorileri eklenecek, İmza Sistem risk kategorisinin risk puanına etki oranı arttırılacaktır. Bunun yanında, risk puanlama sistemini besleyecek bir ceza puanı sistemi geliştirilerek gereksiz izin kullanımı ve riskli kabuk komutu kullanımı cezalandırılacaktır. Bu işlem sonucunda elde edilecek puanlar, risk puanına doğrudan eklenerek daha sağlıklı risk puanları üretilecektir.

İkinci aşamada, çalışmamızda gerçekleştirdiğimiz sistem mevcut antivirüs yazılımlarıyla birlikte çalışacaktır. Bir geri besleme mekanizması oluşturulacak ve antivirüs yazılımlarından gelen bilgiler analiz edilerek risk puanlama sistemi beslenecektir. Buna ek olarak, geliştirdiğimiz sistemin veritabanında 6000 adet kötücül uygulamanın imzası yer almaktadır. Analiz edilen uygulamaların bu veritabanındaki varlığı tespit edilebilmektedir. Dolayısıyla, bu bilgiler kullanılarak oluşturulacak geri besleme mekanizmasının performansı arttırılacaktır. Antiviürs yazılımlarına ilaveten, antivirüs yazılımları tarafından tespit edilememiş olası zararlı uygulamaları yakalamak için mobil cihazlar üzerinde çalışacak ve elde ettiği bilgileri buluta göndererek analiz edecek bir saldırı tespit ve önleme (IDS/IPS-Intrusion Detection System/Intrusion Prevention System) yazılımı geliştirilecektir. Yazılım; IDS/IPS özellikleri göstermesi yanında, çalışmamızda gerçekleştirdiğimiz sisteme bir geri besleme sağlayarak uygulamalar hakkında uygulamalar yüklenmeden önce kullanıcılara daha tutarlı bilgiler sunacaktır. Normal IDS/IPS sistemlerinden farklı olarak, Android uygulamalarının çalışma performansının ölçümünde; kullanılan işlemci, bellek ve bant genişliği kapasitesinin yanında enerji (pil) tüketim miktarını da inceleyen, mobil cihazlarda çalıştırılması uygun bir tasarım hayata geçirilecektir. Bu tarz bir çözüm bulut tabanlı IDS/IPS çözümlerine benzemesine rağmen, çalışma prensibi açısından tamamen farklı tasarlanacaktır. Bulut tabanlı risk analiz ve hesaplama sistemimiz; uygulama hakkında kullanıcıyı uygulamayı yüklemeden bilinçlendirirken, geliştirilecek mobil cihazdaki uygulama vasıtası ile de mobil uygulamaların dinamik kötücül davranışları hakkında geri besleme alabilir bir yapıda olacaktır. Dolayısıyla, sistemin çalışması için sürekli bir veri bağlantısı kurulmayacak ve sadece belli aralıklarla buluta bağlantı yapılacaktır.

Son aşamada ise, kullanılan bulanık mantık yapısı yapay sinir ağları ile birleştirilerek kendi kendine öğrenebilen, adaptif nöro-bulanık bir sistem tasarlanacaktır. Böyle bir sistemin gerçekleştirilmesi için, MATLAB içerisinde yer alan ANFIS (Adaptive Neuro Fuzzy Inference System) denetçisi kullanılacaktır. Gerçekleştirdiğimiz bulanık mantık tabanlı risk puanlama sistemi, tepeden tırnağa giriş ve çıkış verileri kullanılarak tasarlanmıştır. Sistemde yer alan 4 girişin üyelik fonksiyonları elle tasarlanmıştır. Belirli girişlere karşılık gelen tepki biliniyorsa (yani elimizde yeterli ve sağlıklı veriler varsa), ANFIS sistemi kullanılarak tüm sistemin programlanması otomatikleştirilebilir.

Girişleri ve bunlara karşılık gelen çıkışları matematiksel olarak modellediğimizden dolayı ANFIS eğitimi için gerekli yapılar zaten mevcuttur.

Adaptif bir sistemin programlanması için gerekli veri yapısı, eğitim ve kontrol veri çiftleri kullanılarak kolayca elde edilebilir. ANFIS'te bulunan sinirsel ağ sistemi, giriş ve çıkış verileri arasında bir bağ kurmamızı sağlar. Bu bağ aracılığıyla, giriş-çıkış üyelik fonksiyonları ve kural tabanı otomatik bir şekilde oluşturulur. Ancak, adaptif sistem tasarlanırken şu noktalara dikkat edilmelidir.

- ANFIS sistemi sadece Sugeno tipinde denetçileri desteklemektedir. Dolayısıyla, sistemimizde kullandığımız Mamdani denetçisinin Sugeno tipine çevrilmesi gerekmektedir.
- Girdiler ve bunlara karşılık gelen çıktılar, sistemi iyi bir şekilde tanımlamalıdır. Çünkü, bu tanımlama başarıyı doğrudan etkilemektedir.
- ANFIS, sadece MATLAB'da ön tanımlı bulunan üyelik fonksiyonları ile çalışabilmektedir.

Sistemimizde tüm kurallar işletildiğinde tek bir sonuç üretilmektedir. Ancak, ANFIS'te her bir kural tek bir çıkış üyeliği üzerinde çalışabilecek şekilde tasarlanmalıdır. Yeni sistem tasarlanırken aşağıdaki adımların izlenmesi planlanmaktadır.

- Giriş ve çıkış verileri toplanarak, bu verilerin ANFIS'in kullanabileceği bir duruma getirilmesi.
- ANFIS sisteminin anlayabileceği bir "fismat" yapısının oluşturulması.
- Eğitim hatası sıfır olacak şekilde bir sinir ağının oluşturulması.
- Eğitim ve kontrol verileri oluşturarak sistemin çalışmasının denetlenmesi.

Bu üç aşama başarılı bir biçimde uygulandığında; daha iyi risk puanlaması yapan, kendi kendini eğiten, antivirüs yazılımları ve saldırı tespit sistemleri ile entegre çalışan, kapsamlı ve güncel kötücül uygulama veritabanına sahip bir sistem gerçekleştirilecektir.

KAYNAKLAR

[1]. Bhattacharya, P., Yang, L., Guo, M., Qian, K., Yang, M., 2014, Learning mobile security with Labware, *Security & Privacy*, 12 (1), 69-72.

[2]. Fang, Z., Han, W., Li, Y., 2014, Permission based Android security: Issues and countermeasures, *Computers & Security*, 43, 205-218.

[3]. Seo, S. H., Gupta, A., Sallam, A. M., Bertino, E., Yim, K., 2014, Detecting mobile malware threats to homeland security through static analysis, *Journal of Network and Computer Applications*, 38, 43-53.

[4]. Hatwar, M. S., Shelke, C., 2014, An assess Android antimalware that detects malicious dynamic code in apps, *International Journal of Computer Science and Mobile Computing*, 3 (3), 263-268.

[5]. Yerima, S. Y., Sezer, S., McWilliams, G., 2014, Analysis of Bayesian classification-based approaches for Android malware detection, *IET Information Security*, 8 (1), 25-36.

[6]. Zhang, Z., Wang, Y., Jing, J., Wang, Q., Lei, L., 2014, January, Once root always a threat: analyzing the security threats of android permission system, *Information Security and Privacy*, 8544, pp. 354-369.

[7]. Jeong, Y. S., Lee, H. T., Cho, S. J., Han, S., Park, M., 2014, A kernel-based monitoring approach for analyzing malicious behavior on Android, *29th Annual ACM Symposium on Applied Computing*, 24-28 March 2014 Gyeongju, Korea, ACM, 1737-1738.

[8]. Hsiao, S. W., Hung, S. H., Chien, R., Yeh, C. W., 2014, PasDroid: Real-Time Security Enhancement for Android, *Eighth International Conference on Innovative Mobile and İnternet Services in Ubiqutious Computing*, 2-4 July 2014 Birmingham, England, IEEE, ISBN: 978-1-4799-4333-3, 229-235

[9]. Mohite, S., 2014, A Survey on mobile malware: war without end, *International Journal of Computer Science and Business Informatics*, 9 (1), 273-280.

[10]. Wu, F., Narang, H., Clarke, D., 2014, An overview of mobile malware and solutions, *Journal of Computer and Communications*, 2 (12), 8-16.

[11]. Suarez-Tangil, G., Tapiador, J. E., Peris-Lopez, P., Ribagorda, A., 2014, Evolution, detection and analysis of malware for smart devices, *Communications Surveys & Tutorials*, 16 (2), 961-987.

[12]. Kelley, P. G., Consolvo, S., Cranor, L. F., Jung, J., Sadeh, N., Wetherall, D., 2012, A conundrum of permissions: installing applications on an android

smartphone, *16th International Conference on Financial Cryptography and Data Security*, 27 February-2 March 2012 Berlin, Germany, Springer, ISBN: 978-3-642-34637-8, 68-79.

[13]. Poeplau, S., Fratantonio, Y., Bianchi, A., Kruegel, C., Vigna, G., 2014, Execute this! analyzing unsafe and malicious dynamic code loading in android applications, *20th Annual Network & Distributed System Security Symposium*, 24-27 February 2013 San Diego, USA, The İnternet Society, 1-16.

[14]. Beresford, A. R., Rice, A., Skehin, N., Sohan, R., 2011, Mock-Droid:trading privacy for application functionality on smart-phones, *12th Workshop on Mobile Computing Systems and Applications*, 1-3 March 2011 Phoenix,USA, ACM, 49-54.

[15]. Wang, X., Sun, K., Wang, Y., Jing, J., 2015, DeepDroid: Dynamically Enforcing Enterprise Policy on Android Devices, *Network and Distributed System Security Symposium*, 8-11 February 2015 San Diego, CA, İnternet Society, 1-15.

[16]. Cozzette, A., Lingel, K., Matsumoto, S., Ortlieb, O., Alexander, J., Betser, J., Reiher, P., 2013, Improving the security of android inter-component communication, *International Symposium on Integrated Network Management*, 27-31 May 2013 Belgium, IEEE, 808-811.

[17]. Isohara, T., Takemori, K., Kubota, A., 2011, Kernel-based behavior analysis for android malware detection. *Seventh International Conference on Computational Intelligence and Security*, 3-4 December 2011 Hainan, China, IEEE, 1011-1015.

[18]. Tang, Y., Ames, P., Bhamidipati, S., Bijlani, A., Geambasu, R., Sarda, N., 2012, CleanOS: Limiting Mobile Data Exposure with Idle Eviction, *10th Usenix Conference on Operating Systems Design and Implementation*, 8-10 October 2012 Hollywood, CA ,USA, Usenix, 77-91.

[19]. Vilwock, W., Madiraju, P., Ahamed, S. I., 2013, A System implementation of interruption management for mobile devices, *16th International Conference on Computational Science and Engineering*, 3-5 December 2013 Sydney, Australia, IEEE, 181-187.

[20]. Enck, W., Octeau, D., McDaniel, P., Chaudhuri, S., 2011, A study of Android application security, *USENIX '11 Security Symposium*, 8-12 August 2011 San Francisco, USA, Usenix Association, 315-330.

[21]. Enck, W., Ongtang, M., McDaniel, P., 2009, On lightweight mobile phone application certification, *16th ACM conference on Computer and communications security*, 13-19 November 2013 Chicago, IL, USA, ACM, 235-245.

[22]. Enck, W., Ongtang, M., McDaniel, P., 2008, Mitigating Android software misuse before it happens, *Pennsylvania State University Department of Computer Science and Engineering Network and Security Research Center Technical Report*, NAS-TR-0094-2008.

[23]. Xu, R., Saidi, H., Anderson, R., 2012, Aurasium: practical policy enforcement for Android applications, *USENIX '12 Security Symposium*, 8-10 August 2012 Washington,USA, Usenix Association, 539-552.

[24]. Kaur, A., Upadhyay, D., 2014, PeMo: Modifying application's permissions and preventing information stealing on smartphones, *5th International Conference-Confluence The Next Generation Information Technology Summit*, 25-26 September 2014 Noida, India, IEEE, 905-910.

[25]. Gilbert, P., Chun, B. G., Cox, L. P., Jung, J., 2011, Vision: automated security validation of mobile apps at app markets, *2nd International Workshop on Mobile Cloud Computing and Services*, 28-31 June 2011 Washington, USA, ACM, 21-26.

[26]. Davis, B., Sanders, B., Khodaverdian, A., Chen, H., 2012, I-arm-droid: a rewriting framework for in-app reference monitors for android applications, *Workshop on Mobile Security Technologies*, 24 May 2012 San Francisco, USA, IEEE, 1-9.

[27]. Bruneton E., Lenglet R., Coupaye T., 2002, ASM: a code manipulation tool to implement adaptable systems, *International Conference on Adaptable and Extensible Component Systems*, 17-18 October 2002 Grenoble, France, RSTI-TSI, 30.

[28]. Hornyack, P., Han, S., Jung, J., Schechter S., Wetherall, D., 2011, These aren't the droids you're looking for: retrofitting android to protect data from imperious applications, *18th ACM conference on Computer and Communications Security*, 17-21 October 2011 New York, NY, USA, ACM, 639–652.

[29]. Mahmood, R., Esfahani, N., Kacem, T., Mirzaei, N., Malek, S., Stavrou, A., 2012, A whitebox approach for automated security testing of Android applications on the cloud, *7th International Workshop on Automation of Software Test*, 2-9 June 2012 Zurich, Switzerland, IEEE, 22-28.

[30]. Zonouz, S., Houmansadr, A., Berthier, R., Borisov, N., Sanders, W., 2013, Secloud: A cloud-based comprehensive and lightweight security solution for smartphones, *Computers & Security*, 37, 215-227.

[31]. Pandian, V. A., Kumar, T. G., 2014, A Novel Cloud Based NIDPS for Smartphones, *Second International Conference on Recent Trends in Computer Networks and Distributed Systems Security*, 13-15 March 2014 Trivandrum, India, ACM, ISBN: 978-3-642-54524-5, 473-484.

[32]. Hall, S. P., Anderson, E., 2009, Operating systems for mobile computing, *Journal of Computing Sciences in Colleges*, 25 (2), 64-71.

[33]. Yuksel, A. S., Zaim, A. H., & Aydin, M. A., 2014, A Comprehensive Analysis of Android Security and Proposed Solutions, *International Journal of Computer Network and Information Security*, 6 (12), 9.

[34]. Yue, M., Robinson, W. H., Watkins, L., Corbett, C., 2014, Constructing timing-based covert channels in mobile networks by adjusting cpu frequency, *Third Workshop on Hardware and Architectural Support for Security and Privacy*, 15 June 2014 New York, USA, ACM, 2-2.

[35]. Zheng, M., Sun, M., Lui, J., 2014, DroidRay: a security evaluation system for customized android firmwares, *9th ACM Symposium on Information, Computer and Communications Security*, 4-6 June 2014 Kyoto, Japan, ACM, 471-482.

[36]. George, S., 2014, Google Inc.: Not just a search engine, but an engine of strategic product diversification and excellence in corporate strategy, *International Journal of Advanced Research in Management and Social Sciences*, 3 (2), 62-81.

[37]. Kaplan, D., Kedmi, S., Hay, R., Dayan, A., 2014, Attacking the Linux PRNG on android: weaknesses in seeding of entropic pools and low boot-time entropy, *8th USENIX conference on Offensive Technologies*, 27 June 2014 San Diego, CA, USA, USENIX Association, 14-14.

[38]. Egele, M., Kruegel, C., Kirda, E.,Vigna, G., 2011, PiOS: Detecting Privacy Leaks in iOS Applications, *18th Annual Network & Distributed System Security Symposium*, 6-9 February 2011 San Diego, CA, USA, İnternet Society.

[39]. Barrera, D.,Van Oorschot, P., 2011, Secure software installation on smartphones, *Security & Privacy*, 9 (3), 42-48.

[40]. Urra, O., Ilarri, S., Trillo, R., Mena, E., 2009, Mobile Agents and Mobile Devices: Friendship or Difficult Relationship?, *Journal of Physical Agents*, 3(2), 27-37.

[41]. Ahmad, M. S., Musa, N. E., Nadarajah, R., Hassan, R., Othman, N. E., 2013, Comparison between android and iOS Operating System in terms of security, *8th International Conference on Information Technology in Asia*, 1-4 July 2013 Kuching, Malasia, IEEE, 1-4.

[42]. Felt, A. P., Ha, E., Egelman, S., Haney, A., Chin, E.,Wagner, D., 2012, Android permissions: User attention, comprehension, and behavior, *8th Symposium on Usable Privacy and Security*, 11-13 July 2012 Washington DC, USA, ACM, 3.

[43]. Vidas, T., Votipka, D., Christin, N., 2011, All Your Droid Are Belong to Us: A Survey of Current Android Attacks, *5th USENIX conference on Offensive Technologies*, 8-12 August 2011 San Francisco, USA, Usenix, 81-90.

[44]. Pearce, P., Felt, A. P., Nunez, G.,Wagner, D., 2012, Addroid: Privilege separation for applications and advertisers in android, *7th Symposium on Information, Computer and Communications Security*, 2-4 May 2012 Seoul,Korea, ACM, 71-72.

[45]. Song, M., Xiong, W., Fu, X., 2010, Research on architecture of multimedia and its design based on android, *International Conference on İnternet Technology and Applications*, 21-23 August 2010 Wuhan, China, IEEE, 1-4.

[46]. Bing, H., 2012, Analysis and research of system security based on android, Fifth *International Conference on Intelligent Computation Technology and Automation*,12-14 January 2011 Hunan, China, IEEE, 581-584.

[47]. Shewale, H., Patil, S., Deshmukh, V., Singh, P., 2014, Analysis of Android vulnerabilities and modern exploitation techniques, *Journal on Communication Technology*, 5 (1), 863-867.

[48]. Armando, A., Merlo, A., Verderame, L., 2013, An empirical evaluation of the Android security framework, *11th International Conference on Security and Privacy Protection in Information Processing Systems*, July 8-10 2013 Auckland, New Zealand, Springer, 176-189.

[49]. Liu, J., Yu, J., 2011, Research on development of android applications, Fourth *International Conference on Intelligent Networks and Intelligent Systems*, 1-3 November 2011 Kunming, China, IEEE, 69-72.

[50]. Meier, R., 2012, *Professional Android 4 application development*, John Wiley & Sons, USA, ISBN: 978-1118102275.

[51]. Shin, W., Kiyomoto, S., Fukushima, K.,Tanaka, T., 2009, Towards formal analysis of the permission-based security model for android, *Fifth International Conference on Wireless and Mobile Communications*, 23-29 August 2009 Cannes, France, IEEE, 87-92.

[52]. Butler, M., 2011, Android: Changing the mobile landscape, *Pervasive Computing*, 10 (1), 4-7.

[53]. Lettner, M., Tschernuth, M., Mayrhofer, R., 2012, Mobile platform architecture review: Android, iPhone, Qt, *13th International Conference on Computer Aided Systems Theory*, 6-11 February 2012 Las Palmas de Gran Canaria, Spain, Springer, 544-551.

[54]. Oh, H. S., Kim, B. J., Choi, H. K., Moon, S. M., 2012, Evaluation of Android Dalvik virtual machine, 10th International Workshop on Java Technologies for Real-time and Embedded Systems, 24-26 October 2012, Copenhagen, Denmark, ACM, 115-124.

[55]. Barrera, D., Clark, J., McCarney, D., Van Oorschot, P. C., 2012, Understanding and improving app installation security mechanisms through empirical analysis of android, *2nd workshop on security and privacy in smartphones and mobile devices*, 16-18 October 2012 Raleigh, NC, USA., ACM, 81-92.

[56]. Shin, W., Kiyomoto, S., Fukushima, K., Tanaka, T., 2010, August, A formal model to analyze the permission authorization and enforcement in the android framework, *Second International Conference on Social Computing*, 20-22 August 2010 Minneapolis, USA, IEEE, 944-951.

[57]. Gunasekera, S., 2012, *Android Security Architecture, Android Apps Security*, Apress, NY, USA, ISBN: 978-1-4302-4062-4.

[58]. Six, J., 2011, *Application Security for the Android Platform: Processes, Permissions, and Other Safeguards*, O'Reilly Media, Inc, Sebastopol, CA, USA, ISBN: 978-1449315078.

[59]. Nauman, M., Khan, S., Zhang, X., 2010, Apex: extending android permission model and enforcement with user-defined runtime constraints, *5th ACM Symposium on Information, Computer and Communications Security*, 13-16 April 2010 Beijing, China, ACM, 328-332.

[60]. Bugiel, S., Davi, L., Dmitrienko, A., Heuser, S., Sadeghi, A. R., Shastry, B., 2011, Practical and lightweight domain isolation on android, *1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices*, 17-21 October 2011 Chicago,IL, USA, ACM, 51-62.

[61]. Shabtai, A., Fledel, Y., Elovici, Y., 2010, Securing Android-powered mobile devices using SELinux, *IEEE Security & Privacy*, 8 (3), 36-44.

[62]. Ongtang, M., McLaughlin, S., Enck, W., McDaniel, P., 2012, Semantically rich application-centric security in Android, *Security and Communication Networks*, 5 (6), 658-673.

[63]. Barrera, D., Kayacik, H. G., van Oorschot, P. C., Somayaji, A., 2010, A methodology for empirical analysis of permission-based security models and its application to android, *17th ACM conference on Computer and Communications Security*, 4-8 October 2010 New York, USA, ACM, 73-84.

[64]. Johnson, R., Wang, Z., Gagnon, C., Stavrou, A., 2012, Analysis of android applications' permissions, *Sixth International Conference on Software Security and Reliability Companion*, 20-22 June 2012 Washington, USA, IEEE, 45-46.

[65]. Felt, A.P., Chin, E., Hanna, S., Song, D., Wagner, D., 2011, Android permissions demystified, *18th ACM conference on Computer and Communications Security*, 17-21 October 2011 Chicago, IL, USA, ACM, 627-638.

[66]. Stevens, R., Ganz, J., Filkov, V., Devanbu, P., Chen, H., 2013, Asking for (and about) permissions used by android apps, *10th Working Conference on Mining Software Repositories*,18-19 May 2013 San Francisco, USA, IEEE Press, 31-40.

[67]. Zhou, Y., Wang, Z., Zhou, W., Jiang, X., 2012, Hey, you, get off of my market: detecting malicious apps in official and alternative Android markets, *19th Annual Network and Distributed System Security Symposium*, 6-8 February 2012 San Diego, USA, The İnternet Society, 1-13.

[68]. Jeon, J., Micinski, K. K., Vaughan, J. A., Fogel, A., Reddy, N., Foster, J. S., Millstein, T., 2012, Dr. android and mr. hide: fine-grained permissions in android applications, *2nd ACM workshop on Security and Privacy in Smartphones and Mobile Devices*, 16-18 October 2012 North Carolina, USA, ACM, 3-14.

[69]. Sarma, B. P., Li, N., Gates, C., Potharaju, R., Nita-Rotaru, C., Molloy, I., 2012, Android permissions: a perspective combining risks and benefits, *17th ACM*

Symposium on Access Control Models and Technologies, 20-22 June 2012 Newark, USA, ACM, 13-22.

[70]. Gibler, C., Crussell, J., Erickson, J., Chen, H., 2012, AndroidLeaks: Automatically detecting potential privacy leaks in Android applications on a large scale, *5th International Conference on Trust and Trustworthy Computing*, 13-15 June 2012 Vienna, Austuria, Springer, ISBN: 978-3-642-30920-5, 291-207.

[71]. Zhou, Y., Jiang, X., 2012, Dissecting android malware: characterization and evolution, *IEEE Symposium On Security and Privacy*, 20-23 May 2012 San Francisco, USA, IEEE, 95-109.

[72]. Backes, M., Gerling, S., Hammer, C., Maffei, M., Von Styp-Rekowsky, P., 2013, AppGuard–enforcing user requirements on Android apps, *19th International Conference onTools and Algorithms for the Construction and Analysis of Systems*, 16-24 March 2013 Rome, Italy, Springer, ISBN: 978-3-540-41865-8, 543-548.

[73]. Oberheide, J., Cooke, E., Jahanian, F., 2008, CloudAV: n-version antivirus in the network cloud, *17th USENIX Security Symposium*, 28 July-1 August 2008 California, USA, Usenix Association, 91-106.

[74]. Shabtai, A., Kanonov, U., Elovici, Y., Glezer, C., Weiss, Y., 2012, "Andromaly": a behavioral malware detection framework for android devices, *Journal of Intelligent Information Systems*, 38 (1), 161-190.

[75]. Portokalidis, G., Homburg, P., Anagnostakis, K., Bos, H., 2010, Paranoid android: versatile protection for smartphones, *26th Annual Computer Security Applications Conference*, 6-10 December 2010 Austin, Texas, USA, 347-356.

[76]. Tesfay, W. B., Booth, T., Andersson, K., 2012, Reputation based security model for android applications, *11th International Conference on Trust, Security and Privacy in Computing and Communications*, 25-27 June 2012, Liverpool, UK, IEEE, 896-901.

[77]. Oberheide, J., Veeraraghavan, K., Cooke, E., Flinn, J., Jahanian, F., 2008, Virtualized in-cloud security services for mobile devices, *1st Workshop on Virtualization in Mobile Computing*, 17-20 June 2008 Breckenridge, CO, USA, ACM, 31-35.

[78]. Kinney, S. L., 2006, *Trusted platform module basics: using TPM in embedded systems*, Newnes, Burlington, USA, ISBN: 978-0-7506-7960-2.

[79]. Dietrich, K., Winter, J., 2009, Implementation aspects of mobile and embedded trusted computing, *2nd International Conference on Trusted Computing*, 6-8 April 2009 Villach, Austuria, Springer, ISBN: 978-3-642-00586-2, 29-44.

[80]. Nauman, M., Khan, S., Zhang, X., Seifert, J. P., 2010, Beyond kernel-level integrity measurement: enabling remote attestation for the android platform, *3rd International Conference on Trust and Trustworthy Computing*, 21-23 June 2010 Berlin, Germany, ISBN: 978-3-642-13868-3, 1-15.

[81]. Fenz, S., Goluch, G., Ekelhart, A., Riedl, B., Weippl, E., 2007, Information security fortification by ontological mapping of the ISO/IEC 27001 standard, *13th Pacific Rim International on Dependable Computing*, 17-19 December 2007 Melbourne, Victoria, Australia, IEEE, 381-388.

[82]. Bilbao, A., 1992, TUAR-a model of risk analysis in the security field, *International Carnahan Conference on Security Technology: Crime Countermeasure*, 14-16 October 1992 Madrid, Spain, IEEE, 65-71.

[83]. De Ru, W. G., Eloff, J. H., 1996, Risk analysis modeling with the use of fuzzy logic, *Computers & Security*, 15 (3), 239-248.

[84]. Fu, Y., Qin, Y., Wu, X., 2008, A method of information security risk assessment using fuzzy number operations, *4th International Conference Wireless Communications, Networking and Mobile Computing*, 12-14 October 2008 Dalian, China, IEEE, 1-4.

[85]. Shameli-Sendi, A., Shajari, M., Hassanabadi, M., Jabbarifar, M., Dagenais, M., 2012, Fuzzy multi-criteria decision-making for information security risk assessment, *The Open Cybernetics and Systemics Journal*, 6, 26-37.

[86]. McGill, W. L., Ayyub, B. M., 2007, Multicriteria security system performance assessment using fuzzy logic, *The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology*, 4 (4), 356-376.

[87]. Liu, Y. L., Jin, Z. G., 2015, SAEW: a security assessment and enhancement system of wireless local area networks (WLANs), *Wireless Personal Communications*, 82 (1), 1-19.

[88]. Sendi, A. S., Jabbarifar, M., Shajari, M., Dagenais, M., 2010, FEMRA: Fuzzy expert model for risk assessment, *Fifth International Conference on İnternet Monitoring and Protection*, 9-15 May 2010 Barcelona, Spain, IEEE, 48-53.

[89]. Jamshidi, A., Yazdani-Chamzini, A., Yakhchali, S. H., Khaleghi, S., 2013, Developing a new fuzzy inference system for pipeline risk assessment, J*ournal of loss Prevention in the Process Industries*, 26 (1), 197-208.

[90]. Carr, V., Tah, J. H. M., 2001, A fuzzy approach to construction project risk assessment and analysis: construction project risk management system, *Advances in Engineering Software*, 32 (10), 847-857.

[91]. Gallego, L. E., Duarte, O., Torres, H., Vargas, M., Montaña, J., Pérez, E., Younes, C., 2004, Lightning risk assessment using fuzzy logic, *Journal of Electrostatics*, 60 (2), 233-239.

[92]. Arp, D., Spreitzenbarth, M., Huebner, M., Gascon, H., Rieck K., 2014, Drebin: Efficient and Explainable Detection of Android Malware in Your Pocket, *21th Annual Network and Distributed System Security Symposium*, 23-26 February 2014 San Diego, California, İnternet Society, 1-15.

[93]. Spreitzenbarth, M., Echtler, F., Schreck, T., Freling, F.C., Hoffmann J., 2013, MobileSandbox: Looking Deeper into Android Applications, *28th International ACM Symposium on Applied Computing*, 18-22 March 2013 Coimbra, Portugal, ACM, ISBN: 978-1-4503-1656-9, 1808-1815,

[94]. Earl, C., 1994, *Fuzzy systems handbook. A practitioner's guide to building, using, and maintaining fuzzy systems*, AP Professional, San Diego, CA, USA, ISBN: 978-0121942700.

[95]. Mamdani, E. H., Assilian, S., 1999, An experiment in linguistic synthesis with a fuzzy logic controller, *International journal of man-machine studies*, 7 (1), 1-13.

[96]. Zelle, J. M., 2004, *Python programming: an introduction to computer science*, Franklin, Beedle & Associates, Inc., Oregon, USA, ISBN: 978-1887902991.

[97]. Atkinson, L., Suraski, Z., 2004, *Core PHP programming*, Prentice Hall Professional, New Jersey, USA, ISBN: 978-0130463463.

[98]. Goodman, D., 2002, *Dynamic HTML: The Definitive Reference: A Comprehensive Resource for HTML, CSS, DOM & JavaScript*, O'Reilly Media, Inc., Sebastopol, CA, USA, ISBN: 978-0596003166.

[99]. Chaffer, J., 2009, *Learning JQuery 1.3: Better Interaction and Web Development with Simple JavaScript Techniques*, Packt Publishing Ltd., Birmingham, USA, ISBN: 1847196705.

[100]. Flanagan, D., 2002, *JavaScript: the definitive guide*, O'Reilly Media, Inc., Sebaptopol, CA, USA, ISBN: 978-0596000486.

[101]. Enck, W., Ongtang, M., McDaniel, P., 2009, Understanding android security, *Security & Privacy*, 7 (1), 50-57.

[102]. Bloom, R. B., Foreword By-Behlendorf, B., 2002, *Apache Server 2.0: The Complete Reference*, Osborne/McGraw-Hill, Berkeley, CA, USA, ISBN: 0072223448.

[103]. Welling, L., Thomson, L., 2003, *PHP and MySQL Web development*, Sams Publishing, USA, ISBN: 978-0-672-32525-0.

ÖZGEÇMİŞ

Kişisel Bilgiler

Adı Soyadı	Asım Sinan YÜKSEL
Uyruğu	T.C
Doğum tarihi, Yeri	29.09.1983, Zile
E-mail	asimyuksel@sdu.edu.tr

Eğitim

Derece	**Kurum/Anabilim Dalı/Programı**	**Yılı**
Doktora	İ.Ü. Fen Bilimleri Enstitüsü / Bilgisayar Mühendisliği / Bilgisayar Mühendisliği Programı	2015
Yüksek Lisans	Indiana University / School of Informatics and Computing / Computer Science	2010
Lisans	Ege Üniversitesi / Mühendislik Fakültesi / Bilgisayar Mühendisiliği Bölümü	2006
Lise	Zile Anadolu Öğretmen Lisesi	2002

Makaleler / Bildiriler

[1]. Yuksel, A.S., Cankaya, I.A., Yuksel, M.E., 2015, An Analysis of RDF Storage Models and Query Optimization Techniques, *International Journal of Information Engineering and Electronic Business*, 7 (2), 20-26.

[2]. Cankaya, I.A., Yuksel, A.S., Koyun, A., Yigit, T., 2015, A Dynamic Content Generation Tool for OOP Course, *International Journal of Computer Theory and Engineering*, 7 (1), 66-69.

[3]. Yuksel, A. S., Zaim, A. H., Aydin, M. A., 2014, A Comprehensive Analysis of Android Security and Proposed Solutions, *International Journal of Computer Network and Information Security*, 6 (12), 9-20.

[4]. Yigit, T., Koyun, A., Yuksel, A. S., Cankaya, I. A., 2014, Evaluation of Blended Learning Approach in Computer Engineering Education. *Procedia-Social and Behavioral Sciences*, 141, 807-812.

[5]. Yigit, T., Koyun, A., Yuksel, A.S., Cankaya, I.A., Kose, U., 2014, *An Example Application of Artificial Intelligence Supported Blended Learning Education Program in Computer Engineering*, Artificial Intelligence Applications in Distance Education, In: Kose, U. (ed.), Koc, D. (ed.), Chapter 12, IGI-Global, Pennsylvania, USA, ISBN: 9781466662766, 1-329.

[6]. Yigit, T., Cakar, M.A., Yuksel A.S., 2013, The Experience of NoSQL Database in Telecommunication Enterprise, *International Conference on Information and Communication Technologies*, 23-25 October 2013 Baku, Azerbaijan, IEEE, 1-4.

[7]. Küçüksille, E.U., Öztürk, N., Çankaya, İ.A., Yüksel, 2013, Test Güdümlü Yazılım Geliştirme Süreci Ve Kullanılan Çerçeveler, *7. Ulusal Yazılım Mühendisliği Sempozyumu*, 25-28 Eylül 2013 İzmir, Türkiye CEUR, 1-8.

[8]. Yüksel, M.E., Yüksel A.S., Zaim, A.H., 2013, A reputation-based privacy management system for social networking sites, *Turk. J. Elec. Eng. & Comp. Sci.*, 21 (3), 766-784.

[9]. Yüksel, M.E., Yüksel, A.S., 2011, A System Design for The Integration of RFID Systems with Wireless Network Technologies, *7th International Conference on Wireless and Mobile Communications*, 19-24 June 2011 Luxembourg, IARIA, 112-117.

[10]. Yüksel, A.S., Yüksel, M.E., Zaim, A.H., 2010, An Approach for Protecting Privacy on Social Networks, *The 5th International Conference on Systems and Networks Communications*, 22-27 August 2010 Nice, France, IARIA, 154-159.

[11]. Yüksel, M. E., Yüksel, A. S., 2010, An Application for Protecting Personal Information on Social Networking Websites, The 4th International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies, October 2010, Florence, Italy.

[12]. Yüksel, M. E., Zaim, A. H., Yüksel, A. S., 2010, Impacts of RFID in Business Systems and Supply Chain Management, *2nd International Symposium on Sustainable Development*, 8 June 2010 Sarajevo, Bosnia and Herzegovina,International Burch University, 59.

[13]. Shue, C. A., Gupta, M., Lubia, J. J., Kong, C. H., Yuksel, A.S., 2009, Spamology: A study of spam origins, *6th Conference on Email and Anti-Spam*, 16-17 July 2009 California, USA, CEAS, 18-26.