IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 20, 2021, accepted July 7, 2021, date of publication July 14, 2021, date of current version July 23, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3097152

ARANAC: A Bring-Your-Own-Permissions Network Access Control Methodology for Android Devices

J. A. GÓMEZ-HERNÁNDEZ, J. CAMACHO, J. A. HOLGADO-TERRIZA, (Member, IEEE), P. GARCÍA-TEODORO, AND G. MACIÁ-FERNÁNDEZ

Network Engineering and Security Group, School of Computer Science and Telecommunications Engineering, University of Granada, 18071 Granada, Spain

Corresponding author: J. A. Gómez-Hernández (jagomez@ugr.es)

This work was supported in part by the Spanish Government-Ministerio de Economía y Competitividad (MINECO), and in part by the European Regional Development Fund (ERDF) under Project TIN2017-83494-R.

ABSTRACT In this paper, we introduce a new methodology for network access control for Android devices based on app risk assessment. Named ARANAC (which stands for *Application Risk Assessment based Network Access Control*), this methodology is specially tailored for scenarios using the Bring-Your-Own-Device (BYOD) policy, where the adoption of some solutions can lead to problems in security and privacy for both the employees and the business organization. ARANAC mainly relies on the analysis of an aggregate of permissions declared in the manifests of installed applications on users' devices. The access control scheme combines three operational modules: *i)* a device monitoring tool, *ii)* a novel permission-based risk model, and *iii)* an anomaly-based detection machine learning module based on a methodology (called MSNM, from *Multivariate Statistical Network Monitoring*) that provides both detection and diagnostic capabilities. ARANAC's novelty is in the combination of four features. Firstly, it is privacy-aware, and thus, it does not require detailed information about installed applications but only an aggregate of permissions. Secondly, it builds a normality model by combining expert knowledge with data, capturing the behavior of a complete population of mobile devices. Thirdly, it is dynamic, as permissions are updated in real time, allowing the network to re-assess access control on a continuous basis. Finally, its diagnostic capabilities allow for giving recommendations to final users so that they are capable of mitigating their risks when accessing networks. We evaluated the approach with more than 80 Android devices at a university campus network and obtained interesting results regarding security risks in the usual deployment of device apps.

INDEX TERMS Android permissions, bring-your-own-device, mobile security, network access control, risk assessment.

I. INTRODUCTION

Mobile devices such as smartphones and tablets are the most globally widespread platforms among users nowadays. According to GSMA, there are around 5.2 billion people subscribed to mobile services (around 67% of the global population). These subscriptions generated $4.1 trillion in economic added value globally in the last year [1], and by the year 2024 these figures will approach 5.8 billion people and $5 trillion, respectively. In accordance with the ever-increasing relevance of mobile environments, a Cisco study reveals that mobile traffic constitutes around 40 Exabytes of traffic per month at present and around 80 Exabytes per month in the coming years [2].[1]

As the adoption of mobile devices and services increases, security incidents in such environments have experienced a huge increase in the last years [3], in the form of different types of malware, *e.g.*, adware, SMS trojans, ransomware, dropper and banking trojans. Moreover, given the fact that around 85% of the mobile market corresponds to Android devices [4], this operating system is exposed to a wide number and variety of risks and attacks [5], [6].

The associate editor coordinating the review of this manuscript and approving it for publication was Adnan M. Abu-Mahfouz.

[1] See also https://newsroom.cisco.com/press-release-content?type=webcontent&articleId=2055169

As a consequence of all the above, mobile security constitutes a primary challenge for our society. Primary security services like authentication and authorization are usually implemented within *access control systems* (ACSs). Frequently, ACSs are only used as an entrance barrier to a system, but others adopt a continuous monitoring approach, where some kind of detection procedures are used to both determine the occurrence of undesired activities or behaviors [7], [8] during the lifetime operation of devices as well as to carry out some kind of countermeasures.

In this paper, we describe the fundamentals of ARANAC (*Application Risk Assessment based Network Access Control*), a continuous monitoring access control proposal for mobile devices to access resources in the context of an organization, using the Bring-Your-Own-Device (BYOD) policy. ARANAC is based on the monitoring of app permissions [9]. From them, we follow a Risk-Based Access Control model [10] to assign a risk value to each device permission set from expert knowledge. Then we build a normality model for those permissions by considering not only local information (a single device), but also global knowledge of permissions in a sample of devices belonging to the same organization (*e.g.* devices from workers of a company in a BYOD scenario). To build this model we apply the *Multivariate Statistical Network Monitoring* (MSNM) detection methodology [11], for two main reasons: *i)* it allows for building an aggregated model to capture normality for the whole population of devices, and *ii)* it is capable of providing not only anomaly detection capabilities, but also diagnosis, *i.e.*, identifying those permissions that have caused an anomaly in a device, which is essential for sending recommendations for final users to reduce risks. Thus, ARANAC's detection capabilities will allow for granting and restricting access to the devices on the network environment. Moreover, the diagnostic information is provided to the final devices/users to solve the detected risks in order to reconsider their access to the network.

For the evaluation of the system, we have built a prototype that includes a monitoring tool for Android devices developed by the authors to collect information over time about communications, apps, resource consumption and interfaces, among others. We evaluate ARANAC with more than 80 devices in a university campus network. The results obtained show the promising performance of the access control proposal. Both the monitoring tool and the anonymized data collected for experimentation are made available in a public repository.

In summary, the main features and contributions of ARANAC to address the security and privacy issues in BYOD scenarios are:

- Firstly, it follows a Risk-Based Access Control approach, which allows the system to obtain a certain risk score for the different permission set on the devices in the network so that policies can be applied according to risk levels. The process to evaluate the risk of a mobile device is based on the continuous monitoring of the Android permissions declared by the installed applications. From this monitoring, we estimate the risks involved in the target system.
- As a main novelty, the system evaluates the risk of a single device by comparing the data collected on it with the global information provided by all the different devices in the network in a crowdsourcing basis. The system relies on the use of exploratory techniques based on the *Multivariate Statistical Network Monitoring* (MSNM) approach, which decides whether to allow or deny access to the device.
- The system respects users' privacy, since the data employed by the monitoring system (permissions used by applications) are conveniently aggregated to allow the detection process without divulging private information.
- The system is able to provide recommendation services. Indeed, the diagnosis capabilities provided by the MSNM technique allow final users to receive tips/advice about how to reduce their risk level, if necessary, and how to (re)gain network access.

The organization of the rest of the paper is as follows: Section II reviews the main proposals in the field of mobile security in the specialized literature, with special emphasis on access control. Section III describes the overall operation of the permission-based access control system introduced in this work: ARANAC. In particular, Section III-A presents the overall architecture of the proposal, while each of its component modules is subsequently described: Section III-B for the monitoring tool used in the prototype developed, Section III-C for the permission risk analysis developed, and Section III-D for the use of the MSNM detection approach to detect compromised devices in the network, and thus, for considering when to grant access or not. This last section also explains how to diagnose elevated risk levels in order to recommend users how to mitigate said risks. Thereafter, Section IV presents the experimentation carried out to evaluate the overall performance of ARANAC and then discusses the results obtained. Finally, Section V draws the main conclusions and offers/proposes some future lines of work.

II. BACKGROUND ON ACCESS CONTROL FOR MOBILE PLATFORMS

As the social adoption of mobile platforms (namely smartphones, tablets and even IoT related devices) has become wider, the number of threats and security incidents for these kinds of devices has also increased [1], [12]–[14]. In this way, the topic of mobile security has consequently received significant attention by the research community in the last years [15]–[21], with a number of specific proposals being developed and specially focused on Android platforms.

Like in other environments, security solutions for mobile devices are varied and range from prevention to reaction related approaches, either regarding authentication, confidentiality, integrity, privilege escalation, information leakage, fraud avoidance, etc. As an example of this variety of proposals, authors in [22] present *Secand*, an application designed

to locate stolen mobile devices with turned on status. When the user of a smartphone notices that their phone is stolen and sends an SMS to the stolen phone from another phone, *Secand* would detect the password within the SMS and send current GPS coordinates back to the phone, from which an SMS is sent. The app is also able to switch the front camera of smartphone on in order to take and send pictures to the counter party as a multimedia message system and/or to lock the phone. Another example of a security-oriented app is that of [23], wherein the authors introduce a face recognition system to access Android devices, both to unlock the screen and to use installed apps.

Beyond the existence of specific protection related works like [22] and [23] mentioned before, one principal R&D line in the field of mobile security regards malware detection [8], [24]–[28], intended to determine the existence of potentially malicious apps on the device. Some specific well-known proposals developed for that are TaintDroid [29], CrowDroid [30], DREBIN [31], BehaviorDroid [32], EnDroid [33], BDfinder [34], in which activity and variables like API calls, system calls, permissions, addresses, system logs, and processes, among others, are monitored to determine harmful behavior.

Another key direction to secure mobile environments is that of access control, which is aimed at restricting the access and use of system resources. In the next subsection we describe several proposals developed in the literature on this topic.

A. ACCESS CONTROL

Access control (AC) refers to the selective restriction of access to a place or resource. The permission to access is called *authorization* and constitutes one of the principal security measures in informated-related environments. As an example, the Android security system is built upon a permission-based framework [9], [35], [36] which restricts the access of third-party apps to sensitive resources such as external storage, contacts, emails, and even credit card numbers (see Section III for some further details about this). Once an app is installed on the device, the user must grant the requested permissions at runtime in order to allow access to specific resources such as GPS or SMS.

As discussed in papers like [37], [38], various formal access control models are presented in the literature, such as: Discretionary Access Control (DAC), Mandatory Access Control (MAC), Role-Based Access Control (RBAC), Attribute-Based Access Control (ABAC), etc. These models are used to implement organizational policies that prevent the unauthorized disclosure of sensitive data and for enabling secure access to data and resources. Each AC model has its own methods for making AC decisions and policy enforcement.

In the context above, authors in [39] propose a modified version of the Android OS supporting context-based access control policies. These policies restrict applications from accessing specific data and/or resources based on the user context. The restrictions specified in a policy are automatically applied as soon as the user device matches the pre-defined context associated with the policy. In a similar line, authors in [40] present CoDRA, an access control system for Android that offers context-based dynamically configurable restrictions as well as fine granular policy and the ability to enforce various policy configurations at different levels of system operation. CoDRA employs both static and dynamic restrictions to improve the overall security of the device.

In [41], authors propose a lightweight, fine-grained, and flexible access control scheme for file storage in mobile cloud computing, known F2AC, this access control schema can not only achieve iterative authorization, authentication with tailored policies, and access control for dynamically changing accessing groups, but also can provide transition and revocation of access privileges. A new access control model called ''directed tree with linked leaf model'' is proposed for further implementations in data structures and algorithms.

Oglaza *et al.* present in [42] Kapuer, an IBAC (Identity Based Access Control) related permission management system for Android devices that: *i)* learns users' privacy preferences with a novel learning algorithm, *ii)* proposes abstract authorization rules, and *iii)* provides advanced features to manage these high-level rules.

Authors in [43] introduce a context-aware role-based access control model that can provide dynamic granting and revoking permissions while keeping the number of policies as small as possible. In the model, Android applications are assigned roles which contain a set of permissions and contexts associated with said permissions. These permissions are activated and deactivated for the containing role based on the associated contexts. The approach is unique in that the system associates contexts with permissions as opposed to existing similar works that associate contexts with roles.

Related with ABAC, Baseri *et al.* [44] investigate providing Location-Based Services (LBSs) for attribute-based access control in mobile clouds. More specifically, the authors propose a multi-authority attribute-based access control scheme to support coexistence of authorities, to provide anonymity of users and to protect their identity against malicious authorities. The proposed scheme uses the dynamic location of mobile users as contextual information about those users, employs location range constraints as a policy in attribute-based encryption and authorizes users with dynamic locations which satisfy access policies.

Authors in [45] design an Android lightweight kernel layer mandatory access control framework, analyze and discuss the necessity of terminal kernel layer security protection and propose and finalize verifiable the lightweight kernel layer access control model. Similarly, authors in [46] present a security system called collaborative policy-based security scheme (CSS) that permits users to customize the access permissions of Android applications during runtime. They therefore present a collaboration-based security model for discovering m-apps that misuse user/system permissions to

violate the predefined security policies. The proposed CSS model strengthens the MAC model of AOS to protect the underlying computing environment from the execution of malicious m-apps on the device.

More recently, Dutta *et al.* propose in [47] the creation of the PALS system that builds upon existing work in attribute based access control models, captures physical context collected from sensed data (attributes), and performs dynamic reasoning over these attributes and context driven policies using Semantic Web technologies to execute access control decisions. Based on reasoning about user context, details collected by cloud service providers and device type, this mechanism generates corresponding access control decisions.

1) RISK-BASED ACCESS CONTROL

An interesting access control scheme is the so-called *risk-based access control model* which estimates the security risk value related to the access request in order to dynamically determine the access decision. Authors in [48] perform a review of risk-based access control works.

The risk-based access control model comprises some key modules (see Figure 1). The *risk estimation* module is the main one, which gets access requests from users, analyzes them, collects the required information of *risk factors*, and estimates the security risk value related to each access request. Then the estimated risk value is checked against *access policies* to make the *access decision*, *i.e.*, whether to grant or deny access.

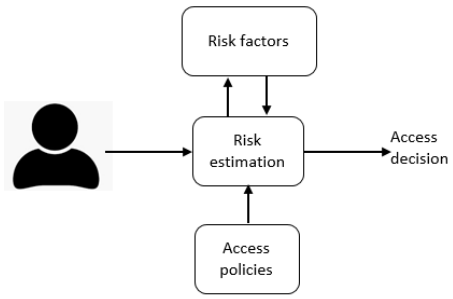

FIGURE 1. Risk-based access control model (from [48]).

One example of this is [49], in which Choi *et al* offer an approach and framework for context-sensitive risk-based access control suitable for medical information systems. This approach categorizes context information, estimating and applying risk through context-based and treatment-based permission profiling and specifications by expanding the eXtensible Access Control Markup Language (XACML) to apply risk.

Additionally, Hintze *et al.* present in [50] CORMORANT, an extensible framework for risk-aware multimodal authentication on mobile devices. By continuously assessing the risk of unauthorized access while evaluating the user's identity using various biometrics, this framework facilitates both convenient and more user-friendly security and can also be configured to achieve a higher level of overall security.

Meanwhile, authors in [10] propose a risk-based access control model for IoT technology that takes into account real-time data information requests for IoT devices and gives dynamic feedback. The proposed model uses IoT environment features to estimate the security risk associated with each access request by using user context, resource sensitivity, action severity and risk history as inputs for a security risk estimation algorithm that is responsible for access decision. Then the proposed model uses smart contracts to provide adaptive features in which the user behavior is monitored to detect any abnormal actions from authorized users. A further variant of that paper can be found in [51].

In [52], Tyche is presented, a secure development methodology that leverages the risk-asymmetry in physical device operations to limit the risk that apps pose to smart home users, without increasing the user's decision overhead. Tyche introduces the notion of risk-based permissions for IoT systems. When using risk-based permissions, device operations are grouped into units of similar risk, and users grant apps access to devices at that risk-based granularity. Starting from a set of permissions derived from the popular Samsung SmartThings platform, they conduct a user study involving domain-experts and Mechanical Turk users to compute a relative ranking of risks associated with device operations.

III. ARANAC: APPLICATION RISK ASSESSMENT BASED NETWORK ACCESS CONTROL

ARANAC is a novel approach to grant or restrict the access of mobile devices to a network within a corporate BYOD environment. Unlike other risk-based access control approaches, risk estimation is performed in ARANAC by taking into account the risk of the permissions of all the applications running on a mobile device in comparison with other devices. After the analysis of the overall risk of all the devices in the environment, we are able to differentiate between anomalous devices and 'normal' devices, thus ensuring that user's privacy is always preserved. In this way, ARANAC decides whether a device can access the network infrastructure or not.

As stated, ARANAC uses permission data to determine access control to the network, specifically focused on Android devices. The permission model adopted by Android is a central part of the Android security model that governs the access of applications (apps in the context of mobile devices) to device resources, system components, and sensitive user data [9]. The granting of these permissions by the user is essential to gain a limited access to specific device resources and capabilities of the device, but the permissions also expose the user/device to security and privacy threats. Thus, the analysis of the risks involved in permissions granted by the user on all applications installed on a specific mobile can provide useful information for access control.

There is a great wealth of studies which have analyzed the Android security mechanism based on permissions and other sets of features of the Android ecosystem that can

compromise the security of the device or expose the user's privacy [53], [54].

Android groups the permissions into three protection levels in increasing order of severity: normal, dangerous or signature. *Normal* permissions do not pose a risk to the user's privacy or the device's operation and are automatically granted. Some of these permissions are related to network access, Bluetooth, WiFi, or alarm settings. *Dangerous* permissions are those that may pose a risk and should therefore be explicitly granted by the user. At this level, such permissions include access to the camera, contacts and location access, microphone usage, sensors operation, SMS, and external storage. *Signature* permissions are automatically granted by the system if the requesting app is signed with the same certificate as the application that declared the permission.

Such permission-based mechanisms have been widely criticized for several reasons. Firstly, in many cases, apps request more permissions than are strictly necessary; in fact, the latest Android version (API 29) includes 158 permissions.[2] Therefore, the requirements/consequences of accepting so many permissions with the installation of any app coupled with, in many cases, a lack of information about the risk level of each permission, can severely confuse developers and end-users. Consequently, the current permission system is not able to help users to make the correct decision about the security/privacy risk level that is associated with the installation of a specific app [55]. Therefore, malicious third-party apps may pose a severe risk to the device's security and may become a source of user privacy leakage.

Fang *et al.* [9] explore some of the risks that can impact security and user privacy in Android mobile devices, such as the coarse granularity of permissions, incompetent permission administration, insufficient permission documentation, overclaiming of permissions, permission escalation attacks, and TOCTOU (Time of Check to Time of Use) attacks.

In the same direction, Alepis *et al.* [56] examine some security flaws that can be caused by the recent modifications to permission models on Android devices. In fact, a new dynamic permission management was introduced in latest releases of Android that allows the revocation or granting of app permissions at runtime. Furthermore, these permissions may be checked and revoked by users transparently and at anytime. However, in order to do so, detailed information must be provided on the risk level involved in each permission.

To determine the risks, an analysis of the attack vectors and existing attacks that can be produced on the Android ecosystem is carried out in different studies [57], [58]. In the analysis, malicious attacks as well as not-fully malicious attacks (*e.g.*, collecting sensitive user' information) were considered [59]. In this way, Sadeghi *et al.* present in [60] a taxonomy which systematically analyzes some of the most relevant security risk assessment approaches recently proposed for Android devices.

[2]https://developer.android.com/reference/android/Manifest.permission

A. ARANAC ARCHITECTURE

ARANAC has a multilayer architecture as shown in Figure 2, which is composed of three tiers: *i) data collection*, *ii) device analysis*, and *iii) access control*. Firstly, at the data collection tier, the aggregate permission information of all apps installed on each mobile device is monitored and collected through a monitoring tool installed on the final device.

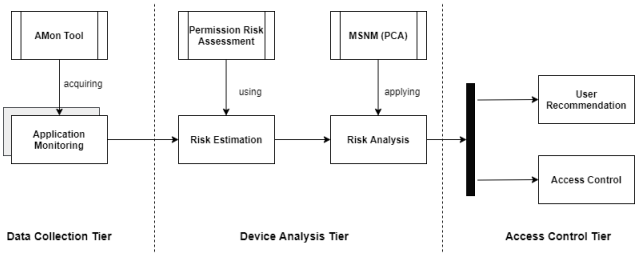

FIGURE 2. ARANAC architecture.

At the device analysis tier, the permission information collected over the set of mobile devices in the corporate environment is analyzed in order to dynamically detect potential anomalies and conclude whether such anomalies should lead to the access rejection of mobile devices in the network. To that end, the analysis is carried out in two stages. First, a *risk estimation* associated with each permission is performed by an assessment module of the permission set available in Android. After that, a *risk analysis* procedure is completed by the application of the methodology MSNM in order to detect threats due to the installed applications on a given device.

At the access control tier, the results of this analysis are subsequently addressed by the *user recommendation* and the *access control* modules. On the one hand, the access control module applies the corresponding granting or restriction of network accessibility according to the risk analysis performed. On the other hand, the user recommendation stage informs final users about potential anomalies and possible solutions to be carried out in order to regain access to the network.

In the next section, a description of the previous tiers and modules is detailed.

B. APPLICATION MONITORING

Threat detection procedures rely on some kind of monitoring process aimed at gathering specific operational information of the target system. We have developed a specific monitoring tool named *AMon*[3] (short for 'Android Monitoring'). AMon is a JAVA app oriented to multidimensional device data gathering in Android environments [61]. It collects disparate sources of information over time, from network usage to protection options enabled on the device. AMon is developed for Android Oreo, and it is compatible with Android Pie. Some of its functionalities are either limited for older versions of Android or are implemented to be compatible with previous

[3]Publicly available at https://github.com/nesg-ugr/AMon

and newer versions. The current minimal supported version is Android Marshmallow, but it could be downgraded at the cost of some functionalities.

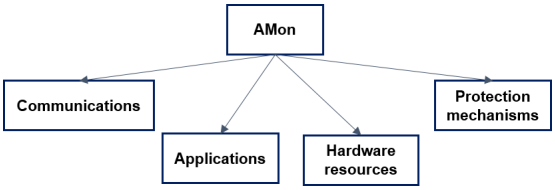

FIGURE 3. Collection modules of AMon.

Due to the different data sources gathered by AMon, its global functionalities are distributed around four different modules as shown in Figure 3:

- The *communications module* implements a local VPN to track outgoing network traffic and obtain data and statistics from it. This module is based on the NetGuard project by Marcel Bokhorst [62]. NetGuard is a firewall application able to log and store traffic, develop traffic statistics, and prevent other applications from connecting to the Internet. In AMon, we only retrieve data from packet headers (to maintain the privacy of communications) in addition to some statistics, the IP address (Android 6.0+) and the MAC address.
- The *applications module* provides a list of installed applications (packages) along with the corresponding permissions, and a timestamp indicating when they are collected. Permissions are an essential feature from a security point of view, since they are a major source of malware infections [63]. It is important to note that the permission related information collected by AMon concerns all the possible permissions a given app could request (extracted from the manifest file of the app) and not necessarily those granted by the user.
- The *hardware resources module* provides access to the state of the device hardware. This information can be split into three groups: *(i)* information related to the device and obtained by the `Build` class of the Android API, such as the brand, model, manufacturer, SDK, number of cores, RAM size, and battery capacity; *(ii)* information on the use of resources by the device, such as CPU, RAM and battery consumption; and *(iii)* communication related information, *i.e.*, if Wi-Fi, Bluetooth, Mobile Data, GPS, etc. are active.
- Finally, the *protection mechanisms module* provides access to some security related checks, such as if the device is rooted, if a mechanism (PIN, pattern, or password) to lock the device is active, and whether developer options or the software installation from unknown sources are enabled or not. While the presence of one of those situations is not really a security fault, they represent potential risks.

As shown, AMon collects both static and dynamic information over time. In this work, we focus on app permissions to perform subsequent risk analysis. Each app/package has an associated list of 158 permissions (in Android API level 29), and each device can have hundreds, or even thousands, of applications/packages installed. In addition, storage will be significantly increased by the fact that our application stores a snapshot of the permissions each time a new app is installed or existing app permissions are changed. Therefore, to reduce the quantity of information handled, permissions are encoded here in binary format: every permission is represented by one bit, equal to a value of 1 if set or 0 if otherwise.

In summary, AMon collects apps and their permissions as a tuple per device: $< Id_i, t_i, App_i, Ps_i >$, where Id_i is the identifier of mobile device, t_i is the timestamp when the permissions are captured, $App_i = \{app_1, app_2, \ldots, app_n\}$ is the set of applications hosted on the mobile device, and $Ps_i = \{ps_1, ps_2, \ldots, ps_n\}$ is the set of permissions for each inspected application. Each set ps_j includes a boolean list of active permissions, that is, $ps_j = \{per_1, per_2, \ldots, per_m\}$ where a binary value 1 means that the permission is active and a value 0 that the permission is not enabled.

Despite the long list of detailed permissions that can be gathered through AMon, in order to respect users' privacy, we will only collect here an aggregate of permissions corresponding to the entire set of installed applications. This provides a high level of privacy which is often a requirement for BYOD environments.

C. PERMISSION RISK ASSESSMENT

The estimation of the risk associated to permissions allows us to highlight those permissions that may involve greater threat. That risk estimation will facilitate better risk analysis of the applications installed on a mobile device.

Generally, there are three approaches to assign a risk score to permissions: *i)* permission frequency ranks found in malware applications; *ii)* taxonomies based on inspections of the permission set; and *iii)* hybrid strategies. The first approach evaluates the most frequent permissions requested by malware applications and/or the usage of permission patterns [59], [64]. Then, an analysis based on this risk ranking determines whether applications hold permissions susceptible to be exploited by malware. In the second approach, the risk estimation is assigned based on the inspection of permissions classified in categories, taking into account a specific feature such as protection-level or permission-group, among others [65]. In this case, the risk rank is set independently of known malware patterns and applications. Finally, the hybrid approach defines new metrics based on a combination of different risk ranks. For instance, in [66] a risk score for permissions is defined with five sub-scores that take into account the permission frequency between benign or malicious applications, the monetary cost, the protection-level, and the relation with private information.

In our proposed method, the risk estimation is based on a re-evaluation of certain permissions belonging to normal or dangerous classes as defined by Android. In that sense, a normal Android permission is classified as dangerous if it

is frequently found in malicious applications. Subsequently, for evaluating the associated risk, we adopt the thread modeling technique based on the well-known STRIDE approach from Microsoft [67], where STRIDE is the abbreviation of *Spoofing, Tampering, Repudiation, Information Disclosure, Denial of Service and Elevation of Privilege*. Although other threat-detection models exist, as explained by Shevchenko [68], STRIDE provides a design-centric approach which allows for investigating the security properties of Android platforms independently of the kind of security attack in question.

Granting a permission can imply a potential risk to be estimated in term of security or user privacy, since enabling a permission provides the application with access to device resources or the ability to invoke specific functionalities through the Android API. Therefore, each permission can pose a potential threat that can be classified into STRIDE categories. To rank the risks for each permission, we use the DREAD model [69] to evaluate the likelihood of an attack to exploit the threats associated by a particular permission. The DREAD model consists of the evaluation of potential Damage, Reproducibility, Exploitability, Affected Users and Discoverability that a risk could have associated with the activation of a permission. This way, a risk value is calculated through Eq. (1) as follows:

$$\text{Risk} = \text{Damage} + \text{Reproducibility} + \text{Exploitability} + \text{Affected Users} + \text{Discoverability} \quad (1)$$

Values from 1 (low) to 3 (high) are assigned to each component in Eq. (1) with the overall sum providing a result in the range 5 to 15. Analyzing the overall ratings, we can assign a high risk for a rating in the range 12-15, medium risk for values in the range 8-11, and low risk for values 5-7. Fig. 4 shows how the DREAD value is estimated for each permission.

Damage potential – How much are assets affected?
- 1 – None
- 2 – Information disclosure
- 3 – Complete data destruction

Reproducibility – How easily can the attack be reproduced?
- 1 – Very hard or impossible
- 2 – Complex steps are required for authorized user
- 3 – Easy steps for authorizer user

Exploitability – How easily can the attack be launched?
- 1 – Advanced programming and network knowledge
- 2 – With existing public exploits, using available attack tools
- 3 – By using just an app

Affected users – What is the number of affected users?
- 1 – None
- 2 – Individual user
- 3 – All users

Discoverability – How easily can the vulnerability be found?
- 1 – Very hard, requires source code or administrative access
- 2 – Can be easily discovered using tools
- 3 – The information is visible for the app

FIGURE 4. Template for evaluating the risk associated to each permission by the DREAD model.

To simplify the calculation of the DREAD value for the corresponding 158 permissions in Android, we select the taxonomy of sensitive permissions proposed by Olukoya *et al.* in [53]. In this taxonomy, the permission set is grouped into four levels of sensitive permissions guided by three risk indicators: protection-level of Android, permission-group of Android, and demoted permission (permission with a protection level that can be downgraded). Once the sensitive level is set, it is easier to apply the DREAD value by considering the template shown in Fig. 4. The risk estimation for some permissions is shown in Fig. 5.

Permission	STRIDE	Sensitive Level	D	R	E	A	D	Sum	Value
WRITE_SETTING	STID	1	3	2	3	2	2	12	HIGH
WRITE_EXTERNAL_STORAGE	TID	2	2	2	3	2	2	11	MEDIUM
WRITE_CONTACTS	STID	2	2	2	3	2	2	11	MEDIUM
WRITE_CALL_LOG	STD	1	3	2	3	2	2	12	HIGH
WRITE_CALENDAR	TID	2	2	2	2	2	2	11	MEDIUM
USE_SIP	SRD	1	3	2	3	2	2	12	HIGH
SYSTEM_ALERT_WINDOW	STD	1	3	2	3	2	2	12	HIGH
SET_PROCESS_LIMIT	D	2	1	0	3	2	2	8	MEDIUM
SET_ALWAYS_FINISH	D	2	1	0	3	2	2	8	MEDIUM
REORDER_TASKS	STID	2	1	0	3	2	2	8	MEDIUM
RECORD_AUDIO	ID	2	2	2	3	2	2	11	MEDIUM
RECEIVE_WAP_PUSH	STID	1	3	2	3	2	2	12	HIGH
RECEIVE_SMS	S	1	3	2	3	2	2	12	HIGH
RECEIVE_MMS	ST	1	3	2	3	2	2	12	HIGH
RECEIVE_BOOT_COMPLETED	D	2	2	0	3	2	2	9	MEDIUM
REBOOT	STD	2	1	0	3	2	2	8	MEDIUM
READ_SMS	I	1	3	2	3	2	2	12	HIGH
READ_PHONE_STATE	I	1	3	2	3	2	2	12	HIGH

FIGURE 5. Risk estimation of some Android permissions based on the DREAD model.

The application of permission risk assessment in ARANAC enhances the risk rank which helps the detection of risky devices in the subsequent device risk analysis.

D. DEVICE RISK ANALYSIS

The information collected by the monitoring module and evaluated through the risk model is passed onto a subsequent analysis module, which employs an anomaly detection approach based on machine learning. This module applies access control so that anomalous, suspicious devices are not granted access, while 'normal' devices are. This approach results in a data-driven and dynamic risk-based access control, where the acceptable level of risk to grant network access depends on the level of risk of the devices already in the network. It is dynamic since it is based on the accumulate of permissions in a device, which can change over time. Therefore, changes due to the installation of new apps can result in the revocation of both network access as well as operation in the corporate environment.

Our choice for the machine learning anomaly detection technique is the Multivariate Statistical Network Monitoring (MSNM) approach [11], based on Principal Component Analysis (PCA), due to its capability to provide diagnostic information about anomalies. Therefore, we can obtain information about why a given device was not granted access, and the device owner could use it to restore an acceptable security level in the device, *e.g.*, by uninstalling specific apps so that access can be restored.

PCA is applied to data sets where M variables/features are measured on N observations/individuals. This data can be arranged in a matrix $\mathbf{X}$ of N rows and M columns. For the specific implementation of this paper, observations correspond to individual devices and features correspond to aggregates of a

specific permission, computed as the total number of apps in the device with that permission granted. Thus, each row of $\mathbf{X}$ contains a vector of aggregates of the M permissions in a specific mobile device.

In PCA, the original features are linearly transformed into the Principal Components (PCs), eigenvectors of $\mathbf{XX} := \mathbf{X}^T \cdot \mathbf{X}$, typically for mean-centered $\mathbf{X}$. In our case, we auto-scale (normalize to zero mean and unit variance) $\mathbf{X}$, to homogenize the scale of common and uncommon permissions. Furthermore, we optionally multiply each column of the auto-scaled $\mathbf{X}$ by the corresponding risk level so that the PCA model is focused on permissions of high risk.

The PCA model is a matrix factorization that follows the expression:

$$\mathbf{X} = \mathbf{T}_A \cdot \mathbf{P}_A^t + \mathbf{E}_A \tag{2}$$

where A is the number of PCs, $\mathbf{T}_A$ is the $N \times A$ score matrix, $\mathbf{P}_A$ is the $M \times A$ loading matrix and $\mathbf{E}_A$ is the $N \times M$ residual matrix.

For the detection of anomalies with MSNM, two statistics are monitored: the Q-statistic (Q-st), which compresses the residuals, and the D-statistic (D-st) or Hotelling's T2 statistic, which is computed from the scores. For an observation, the D-st and Q-st can be obtained from the following equations:

$$D_n = \mathbf{t}_n \cdot (\Sigma_T)^{-1} \cdot \mathbf{t}_n^t \tag{3}$$

$$Q_n = \mathbf{e}_n \cdot \mathbf{e}_n^t \tag{4}$$

where Σ_T represents the covariance matrix of the scores in the calibration data. The D-statistic and Q-statistic can be interpreted as the anomalous level of an observation in the model and residual sub-spaces of PCA, respectively. The closer to 0 these statistics are, the more normal the corresponding observation.

With the statistics computed from the calibration data, upper control limits (UCL), which are basically detection thresholds to establish anomalous objects, can be marked in the charts with a certain confidence level [11]. When the system is calibrated and control limits are calculated, it can be applied to new incoming data (new devices). An anomaly is identified when either the D-st or the Q-st exceed the corresponding UCL.

Upon detection of an anomaly, the affected device can be granted access or rejected (*access control* module). Moreover, a subsequent diagnostic step is performed to identify the features associated to the anomaly through the tool oMEDA [70]. The output of oMEDA is a $1 \times M$ vector where each element contains the contribution of the corresponding feature to the detected anomaly. Those contributions with larger magnitude are considered relevant. In the present proposal, the diagnosis will point to which permissions make a specific device too anomalous to be granted access. With this information, the device owner can be properly notified to identify the apps that are using such permissions and uninstall them if desired (*user recommendation* module).

A specific contribution of this paper is the derivation of time series control charts for the maximum values of the D-st and the Q-st in the devices of the network. The time series plot is useful to provide security analysts with insightful visual analytics on the security state of the network devices over time. To obtain these control charts, we simply compute the maximum D-st/Q-st values from the set of devices with access or requesting to access the network in a given sampling time and normalize it by the corresponding UCL. The result consists of a couple of time charts for the normalized D-st and the Q-st.

IV. EXPERIMENTATION

A. SETUP

We designed and deployed an experimental setup in the private network of the University of Granada (UGR), in the south of Spain. This is an example of a BYOD network, in which students and academic staff use their personal mobile devices to connect to the Internet and to internal and external network services.

Fig. 6 shows the specific deployment of ARANAC at the UGR network. The system architecture is composed of three main nodes: *i)* mobile devices' service, *ii)* central server for device risk analysis, and *iii)* access control module, using a client-server paradigm. Algorithm 1 shows the general access control procedure performed by ARANAC corresponding to the architectural description given in Figures 2 and 6.

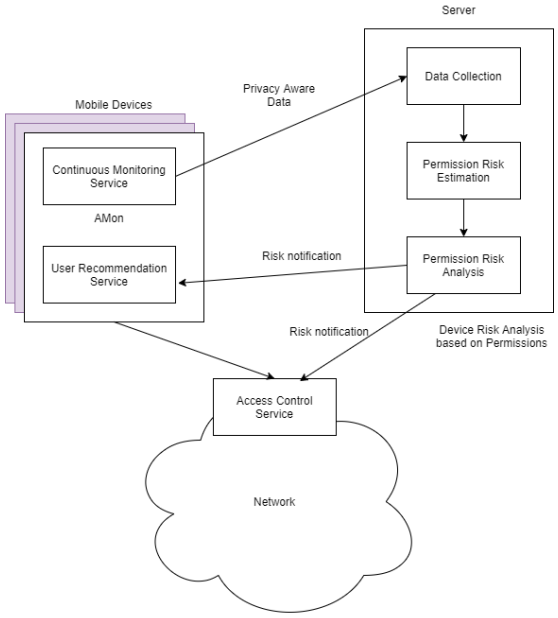

FIGURE 6. Deployment of ARANAC in a BYOD environment in the University of Granada network.

The monitoring system service based on AMon is installed on each mobile device to capture an aggregate of 158 app permissions corresponding to all the apps installed and to send to a central server, which stores the information into a database for subsequent device risk analysis. This information is

Algorithm 1 General ARANAC Algorithm

Data: D-st: D-statistics of normality model; Q-st: Q-statistics of normality model; (R0..Rn): Risk associated with every permission;
Result: Access granted or not
begin
  if *(Device i tries to connect to network)* **then**
    Dev-i(p0,...,pn)=Sum the permissions of all apps;
    Weight every permission with its risk estimation;
    Calculate D-i and Q-i for Dev-i;
    if *(D-i >= D-st || Q-i >= Q-st)* **then**
      `// Device is anomalous`
      Notify "Denied to Dev-i" to the Access Control Service;
      Calculate anomalous permissions with oMeda;
      Notify the user of the permissions to be corrected;
      Notify to the Access Control Service "No access granted";
    else
      `// Device is "normal"`
      Notify "Access granted to Dev-i" to the Access Control Service;
      Notify "Access granted" to the Access Control Service;

periodically sent to the server, with the period of time elapsed between successive samples being configurable in the setup.

The data associated to the user devices are captured over time as shown in Fig. 7, with a total of more than 80 mobile devices monitored over a period of 337 days. The global data are publicly available at https://github.com/nesg-ugr/mdsm-dataset.

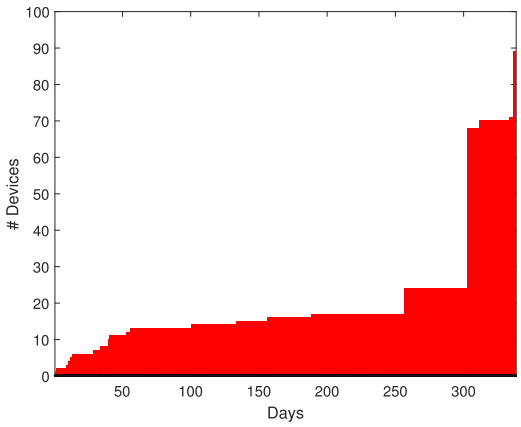

FIGURE 7. Evolution of the number of mobile devices monitored over time.

A relevant aspect regarding data collection is the need to comply with the current regulations on data protection and privacy through the General Data Protection Regulation (GDPR). Although AMon collects anonymized information of the monitored mobile devices, a specific campaign was conducted to find volunteers to participate in the experiment. Therefore, a special document to provide consent from users was approved to cover the different goals of the experiment as well as to guarantee the legality of the performed activities.

After analysing device risks from the data collected by means of the MSNM methodology, the risk information will be sent to the user recommendation service, installed in our case on each mobile device, in order to inform about the detected anomalies and the corresponding recommendations personalized to each user. Likewise, information about access granting or revoking is sent to the access control service installed on the Internet gateway.

B. RESULTS

In the following subsections, we analyze the access-related data obtained in two different ways. First, we analyze the data without including the risks associated to the permissions. For each single day t (for $t = 1$ to $t = 337$), we apply standard pre-processing to the feature matrix $\mathbf{X}_t$, containing N_t devices (see Figure 7) times 158 permissions. In particular, we auto-scale the data as discussed before. On a second approach, each column in the auto-scaled matrix $\mathbf{X}_t$ is multiplied by the risk factor associated to the each permission following Eq. (1). We apply MSNM over these two variants and discuss observed differences.

1) ANALYSIS WITHOUT CONSIDERING PERMISSION RISK MODEL

Figures 8 and 9 show the time series control charts for the maximum values of the D-st and the Q-st, respectively, normalized by their corresponding UCLs at 99% confidence level [11]. These plots are useful to spot days where at least one anomalous device attempted to enter the network during the entire capture.

Before day 303 (12-Jan-2020), both the maximum D-st and the maximum Q-st remain below the control limits, illustrating that no device exceeded the later. Under this situation, all the devices have been granted access to the network. Both charts show an abrupt change at day 303 (12-Jan-2020), where the control limits are exceeded. After spotting this situation in the time series plot, we can proceed by inspecting the MSNM model at day 303.

Fig. 10 shows the anomaly detection results on day 303 where each dot in the scatter plot represents a device, with a unique anonymized index.[4] Observations above the horizontal UCL (for the Q-st) and/or to the right of the vertical UCL (for the D-st) are identified as anomalous and denied access. This is the case of devices 312, 394, and to a lesser

[4]It is important to mention that the device identifier values are independent of the number of devices considered.

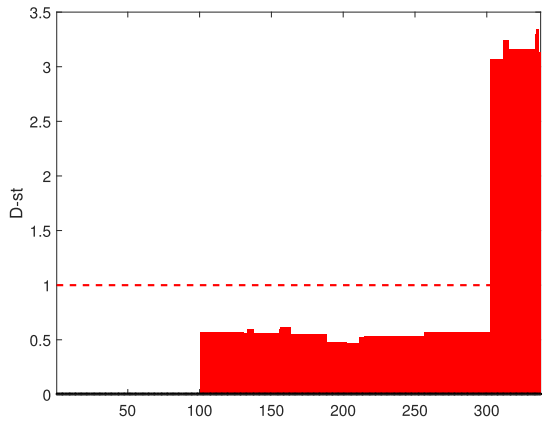

FIGURE 8. Time series control chart for the maximum value of D-st.

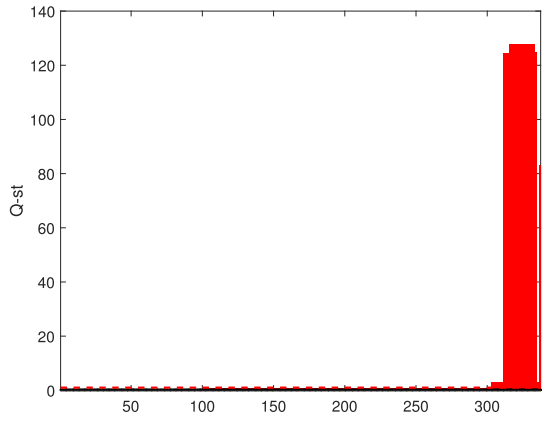

FIGURE 9. Time series control chart for the maximum value of Q-st.

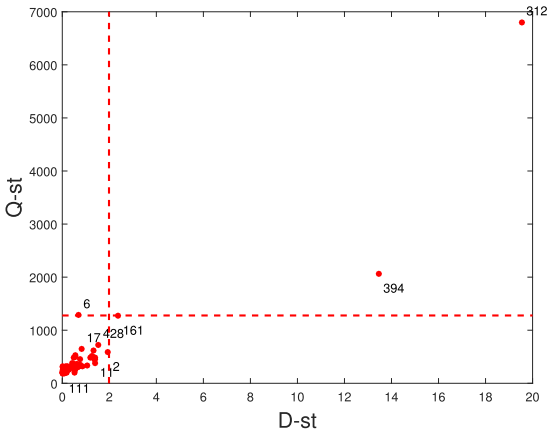

FIGURE 10. MSNM plot (with the MEDA Toolbox): scatter plot with the D-statistic vs the Q-statistic and control limits at 95% of confidence.

extent, 161. In this case, the three devices requested access for the first time on day 303. However, at each time t, MSNM re-evaluates the risk of all devices represented in $\mathbf{X}_t$, and may dynamically revoke access to a previously accepted device or grant access to a previously denied device, responding to a change in its aggregate of permissions.

The ability to diagnose a revocation in MSNM is a useful feature to allow users to improve the security level of devices

TABLE 1. Anomalous features and devices without using the risk assessment model.

# Device	Anomalous Features
312	BIND_CARRIER_SERVICES, WRITE_CALENDAR, BIND_CHOOSER_TARGET_ SERVICE, BIND_NOTIFICATION_LISTENER _SERVICE, ACCESS_FINE_LOCATION, WRITE_CONTACTS, EXPAND_STATUS_BAR, BIND_VOICE_INTERACTION, NFC_TRANSACTION_EVENT, INSTALL_SHORTCUT, CAPTURE_AUDIO_OUTPUT, ACCESS_NETWORK_STATE, REQUEST_COMPANION_RUN_IN_BACKGROUND, ANSWER_PHONE_CALLS, ADD_VOICEMAIL, MODIFY_PHONE_STATE, DIAGNOSTIC, WRITE_SETTINGS, READ_CONTACTS, BLUETOOTH_PRIVILEGED, CHANGE_CONFIGURATION, KILL_BACKGROUND_PROCESSES, WRITE_VOICEMAIL, BIND_TV_INPUT, BIND_VR_LISTENER_SERVICE, BROADCAST_STICKY, GET_ACCOUNTS, REQUEST_PASSWORD_COMPLEXITY, SET_ANIMATION_SCALE, SET_PROCESS_LIMIT, SET_WALLPAPER, SIGNAL_PERSISTENT_PROCESSES, SYSTEM_ALERT_WINDOW, READ_INPUT_STATE, PERSISTENT_ACTIVITY, BIND_INCALL_SERVICE, CHANGE_WIFI_STATE, BROADCAST_WAP_PUSH, CONTROL_LOCATION_UPDATES, READ_SMS, READ_SYNC_STATS, READ_PHONE_NUMBERS, USE_FULL_SCREEN_INTENT, SET_TIME_ZONE, BIND_QUICK_SETTINGS_TILE, RECEIVE_BOOT_COMPLETED,
394	REQUEST_COMPANION_USE_DATA_IN_BACKGROUND, DELETE_CACHE_FILES, WAKE_LOCK, READ_SYNC_STATS, REQUEST_PASSWORD_COMPLEXITY, SET_ANIMATION_SCALE, SET_PROCESS_LIMIT, SET_WALLPAPER, SIGNAL_PERSISTENT_PROCESSES, SYSTEM_ALERT_WINDOW, WRITE_VOICEMAIL, BLUETOOTH_PRIVILEGED, BROADCAST_STICKY, SET_TIME_ZONE, DIAGNOSTIC, BROADCAST_PACKAGE_REMOVED, RECEIVE_BOOT_COMPLETED, CHANGE_CONFIGURATION, MODIFY_PHONE_STATE, ADD_VOICEMAIL, BIND_CALL_REDIRECTIION_SERVICE, WRITE_SETTINGS

with denied access. This is illustrated with devices 312 and 394 in $\mathbf{X}_{303}$ using the oMEDA diagnosis technique [71], which allows for identifying the features related to an anomalous observation detected by MSNM. The results obtained are shown in Table 1. oMEDA detects that device 312 has 46 anomalous permissions while device 394 has 22, taking as reference the rest of normal devices in $\mathbf{X}_{303}$. In ARANAC, a device with denied access receives the list of anomalous permissions. This maintains user privacy to a certain level, since the access control system does not have information about the specific apps causing the situation. With this list, the software in the device can identify apps that reduce the security level of the device and ask the user to uninstall them to (re)gain access to the network.

To assess how accurate MSNM and oMEDA were in the detection and diagnosis on day 303, we checked the accumulated permissions of the devices in $\mathbf{X}_{303}$ for one of the

features highlighted in Table 1: ANSWER_PHONE_CALLS. Authors in [72] show the dangerous nature of this permission. The result is shown in Figure 11, where we can see that the number of apps with the mentioned permission in the anomalous devices is more than ten times the expected amount for the rest of the devices. This shows that the anomaly detection and diagnosis is performing correctly.

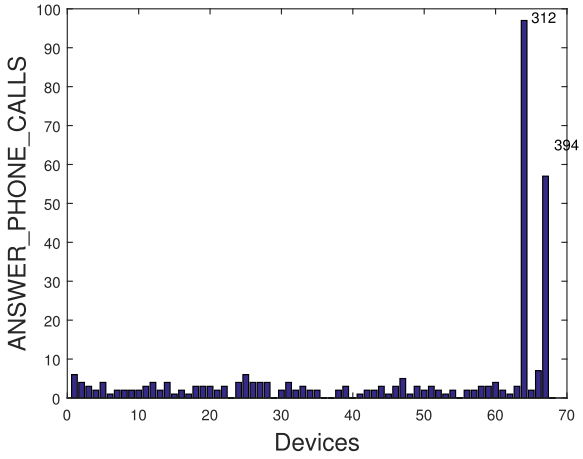

FIGURE 11. Number of apps with permission ANSWER_PHONE_CALLS in the set of devices.

2) ANALYSIS CONSIDERING PERMISSION RISK MODEL

We repeated the previous analysis but now multiplied the column of each matrix $\mathbf{X}_t$ by its corresponding risk factor proposed in Section III-C. We obtained very similar results in terms of detection: no anomalies was found until day 303 and on that day, devices 312 and 394 showed up as clearly anomalous. However, the diagnostic results to differ if considering or not the associated risk. This is illustrated for device 312 in Figures 12 and 13.

When we did not consider the risk model, the diagnosis of device 312 in $\mathbf{X}_{303}$ reported 46 anomalous permissions. Figure 12 shows the security level of these permissions according to the risk model: most permissions are of value 8. If we use the risk factor of each permission in the model, the diagnosis only reports 13 anomalous permissions, and all of them with a risk level above 11. Clearly, the use of risk information in MSNM biases diagnostic results towards high-risk permissions. In Table 2, the diagnostic results for both anomalous devices 312 and 394 using the risk model are reported. We can see that the permissions reported for device 394 have also been reduced in number when considering the risk model.

Based on the above, the ARANAC version that includes the risk model provides a more selective detection of potentially dangerous permissions according to the risk model. That is to say, this version systematically highlights permissions of high risk, while the other provides a more disparate set of risks and a longer list of permissions. Additionally, from the point of view of the final user, the results provided by the second access approach are more suitable to regain network access.

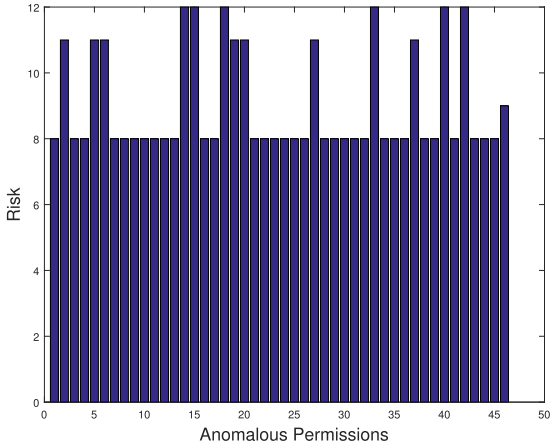

FIGURE 12. Risk levels of anomalous permissions on device 312, day 303, without ARANAC Risk Assessment Model.

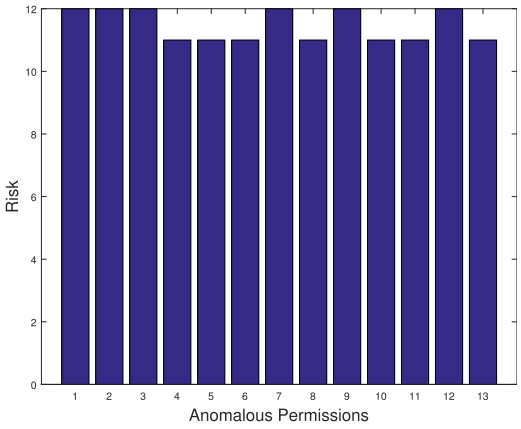

FIGURE 13. Risk levels of anomalous permissions on device 312, day 303, with ARANAC Risk Assessment Model.

TABLE 2. Anomalous features and devices considering risk assessment model.

# Device	Anomalous Features
312	ANSWER_PHONE_CALLS, ADD_VOICEMAIL, WRITE_SETTINGS, WRITE_CALENDAR, ACCESS_FINE_LOCATION, WRITE_CONTACTS, SYSTEM_ALERT_WINDOW, READ_CONTACTS, READ_SMS, BLUETOOTH_PRIVILEGED, GET_ACCOUNTS, READ_PHONE_NUMBERS, CHANGE_WIFI_STATE
394	SYSTEM_ALERT_WINDOW, ADD_VOICEMAIL, WRITE_SETTINGS, ANSWER_PHONE_CALLS, BLUETOOTH_PRIVILEGED, READ_PHONE_NUMBERS, READ_SMS

V. CONCLUSION AND FUTURE WORK

This work introduces ARANAC, a novel access control methodology suitable for use in corporate networks. It is based on the risk estimation for the app permissions installed on a mobile device and it is designed so that the associated risk is compared with the rest of the devices in the environment in order to grant or deny individual access. This is expected to improve the overall security of the corporate environment.

ARANAC is composed of three main modules developed by the authors: a monitoring tool, a risk estimation model, and

a ML-based detection methodology to determine suspicious devices. Although the monitoring tool, AMon, is able to collect a variety of information, we focus here on device app permissions. Moreover, such information is privacy-respectful as it is expressed as a set of 1's and 0's (denoting the set of 158 permissions considered in current Android platforms), where no information about the specific apps installed in each case is released. The risk estimation is based on STRIDE and DREAD models to assign risk values to each of the permissions. Finally, the anomaly-based detection module, named MSNM, determines which devices present suspicious permissions and, from that, accepts or denies access to the corporate network. In addition, MSNM has diagnostic capabilities to extract which specific permissions are contributing to the suspiciousness of the device, which would allow the affected device to solve the problem (*e.g.*, by uninstalling certain apps) and try to regain access.

We have evaluated ARANAC in a real university environment, with the results obtained showing a successful performance by the system as a security related mechanism to control access to corporate environments. In that sense, the use of the risk assessment method significantly reduces the number of permissions detected as dangerous and highlights those with highest risk. This will facilitate the work of users to deactivate the permissions that result in access rejection and users could then try connecting to the network again.

As a future work, we consider it of interest to extend the risk model to estimate risk values associated to the rest of features collected by the monitoring tool AMon (resource usage, communications, etc.). This will provide a more complete view of the overall threat involved in granting certain mobile devices access to corporate networks.

REFERENCES

[1] GSMA. (2020). *The Mobile Economy 2020, GSMA Report*. [Online]. Available: https://www.gsma.com/mobileeconomy/wp-content/uploads/2020/03/GSMA_MobileEconomy2020_Global.pdf

[2] Cisco. (2019). *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017–2022 White Paper*. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-738429.html#_Toc953327 and https://s3.amazonaws.com/media.mediapost.com/uploads/CiscoForecast.pdf

[3] Accenture. (2020). *2020 Cyber ThreatScape Report*. [Online]. Available: https://www.accenture.com/_acnmedia/PDF-136/Accenture-2020-Cyber-Threatscape-Full-Report.pdf

[4] IDC. (2020). *Smartphone Market Share, IDC Report*. [Online]. Available: https://www.idc.com/promo/smartphone-market-share

[5] P. Faruki, A. Bharmal, V. Laxmi, V. Ganmoor, M. S. Gaur, M. Conti, and M. Rajarajan, "Android security: A survey of issues, malware penetration, and defenses," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 2, pp. 998–1022, 2nd Quart., 2015.

[6] Kaspersky. *Android Mobile Security Threats*. Accessed: Feb. 2021. [Online]. Available: https://www.kaspersky.com/resource-center/threats/mobile

[7] S. Garg and N. Baliyan, "Android security assessment: A review, taxonomy and research gap study," *Comput. Secur.*, vol. 100, Jan. 2021, Art. no. 102087.

[8] K. Liu, S. Xu, G. Xu, M. Zhang, D. Sun, and H. Liu, "A review of Android malware detection approaches based on machine learning," *IEEE Access*, vol. 8, pp. 124579–124607, 2020.

[9] Z. Fang, W. Han, and Y. Li, "Permission based Android security: Issues and countermeasures," *Comput. Secur.*, vol. 43, pp. 205–218, Jun. 2014.

[10] H. F. Atlam, A. Alenezi, R. J. Walters, G. B. Wills, and J. Daniel, "Developing an adaptive risk-based access control model for the Internet of Things," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber, Phys. Social Comput. (CPSCom) IEEE Smart Data (SmartData)*, Jun. 2017, pp. 655–661.

[11] J. Camacho, A. Pérez-Villegas, P. García-Teodoro, and G. Maciá-Fernández, "PCA-based multivariate statistical network monitoring for anomaly detection," *Comput. Secur.*, vol. 59, pp. 118–137, Jun. 2016.

[12] M. Taleby, Q. Li, M. Rabbani, and A. Raza, "A survey on smartphones security: Software vulnerabilities, malware, and attacks," *Int. J. Adv. Comput. Sci. Appl.*, vol. 8, no. 10, pp. 25–30, 2017.

[13] Kaspersky. (2020). *IT Threat Evolution Q2 2020. Mobile Statistics*. [Online]. Available: https://securelist.com/it-threat-evolution-q2-2020-mobile-statistics/98337/

[14] Blackberry. (2020). *Mobile Malware and APT Espionage: Prolific, Pervasive, and Cross-Platform*. [Online]. Available: https://www.blackberry.com/us/en/forms/enterprise/mobile-malware-report

[15] M. N. L. Polla, F. Martinelli, and D. Sgandurra, "A survey on security for mobile devices," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 1, pp. 446–471, 1st Quart., 2013.

[16] J. Khan, H. Abbas, and J. Al-Muhtadi, "Survey on mobile user's data privacy threats and defense mechanisms," *Procedia Comput. Sci.*, vol. 56, pp. 376–383, Jan. 2015.

[17] S. Farhan, M. Ali, M. Kamran, Q. Javaid, and S. Zhang, "A survey on security for smartphone device," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 4, pp. 206–219, 2016.

[18] USHS and US Department of Homeland Security. (2017). *Study on Mobile Device Security*. [Online]. Available: https://insidecybersecurity.com/sites/insidecybersecurity.com/files/documents/may2017/cs05052017_DHS_Mobile_Device_Security.pdf

[19] *Mobile Device Security. Corporate-Owned Personally-Enabled (COPE)*, document NIST SP 1800-21, NIST, 2019.

[20] B. Liao, Y. Ali, S. Nazir, L. He, and H. U. Khan, "Security analysis of IoT devices by using mobile computing: A systematic literature review," *IEEE Access*, vol. 8, pp. 120331–120350, 2020.

[21] V. Sharma, I. You, K. Andersson, F. Palmieri, M. H. Rehmani, and J. Lim, "Security, privacy and trust for smart mobile-Internet of Things (M-IoT): A survey," *IEEE Access*, vol. 8, pp. 167123–167163, 2020.

[22] O. C. Karaduman, S. Kaygisiz, A. Buldu, K. Yildiz, and D. Cetinol, "Developing a security software for Android-based systems ('secand')," *Anthropologist*, vol. 17, no. 1, pp. 37–43, Jan. 2014.

[23] R. Stoleriu and M. Togan, "A secure screen and app lock system for Android smart phones using face recognition," in *Proc. 13th Int. Conf. Commun. (COMM)*, Jun. 2020, pp. 133–138.

[24] S. Arshad, A. Khan, M. A. Shah, and M. Ahmed, "Android malware detection & protection: A survey," *Int. J. Adv. Comput. Sci. Appl.*, vol. 7, no. 2, pp. 463–475, 2016.

[25] R. Riasat, M. Sakeena, C. Wang, A. H. Sadiq, and Y. J. Wang, "A survey on Android malware detection techniques," *Trans. Comput. Sci. Eng.*, pp. 1–8, Jan. 2017.

[26] A. Souri and R. Hosseini, "A state-of-the-art survey of malware detection approaches using data mining techniques," *Hum.-Centric Comput. Inf. Sci.*, vol. 8, no. 1, pp. 1–22, Dec. 2018.

[27] Y. S. I. Hamed, S. N. A. AbdulKader, and M. S. M. Mostafa, "Mobile malware detection: A survey," *Int. J. Comput. Sci. Inf. Secur.*, vol. 17, no. 1, pp. 1–10, 2019.

[28] V. Kouliaridis, K. Barmpatsalou, G. Kambourakis, and S. Chen, "A survey on mobile malware detection techniques," *IEICE Trans. Inf. Syst., Special Sect. Secur., Privacy, Anonymity Trust Cyberspace Comput. Commun.*, vol. 103-D, no. 2, pp. 204–211, 2020.

[29] W. Enck, P. Gilbert, B. G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth, "TaintDroid: An information-flow tracking system for real-time privacy monitoring on smartphones," in *Proc. 9th USENIX Conf. Operating Syst. Design Implement.*, 2010, pp. 393–407.

[30] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: Behavior-based malware detection system for Android," in *Proc. 1st ACM Workshop Secur. Privacy Smartphones Mobile Devices*, 2011, pp. 15–26.

[31] D. Arp, M. Spreitzenbarth, M. Hübner, H. Gascon, and K. Rieck, "Drebin: Effective and explainable detection of Android malware in your pocket," in *Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, 2014, pp. 1–15.

[32] A. Silva and J. Simmonds, "BehaviorDroid: Monitoring Android applications," in *Proc. IEEE/ACM Int. Conf. Mobile Softw. Eng. Syst.*, May 2016, pp. 19–20.

[33] P. Feng, J. Ma, C. Sun, X. Xu, and Y. Ma, "A novel dynamic Android malware detection system with ensemble learning," *IEEE Access*, vol. 6, pp. 30996–31011, 2018.

[34] Y. Yao, L. Zhu, and H. Wang, "Real-time detection of passive backdoor behaviors on Android system," in *Proc. IEEE Conf. Commun. Netw. Secur. (CNS), 1st Int. Workshop Syst. Secur. Vulnerability (SSV)*, May 2018, pp. 1–9.

[35] P. Wijesekera, A. Baokar, A. Hosseini, S. Egelman, and D. Wagner, "Android permissions remystified: A field study on contextual integrity," in *Proc. 24th USENIX Conf. Secur. Symp.*, 2015, pp. 499–514.

[36] L. Aron and P. Hanacek, "Dynamic permission mechanism on Android," *J. Softw.*, vol. 11, no. 12, pp. 1224–1230, Dec. 2016.

[37] N. Kashmar and M. Adda, "From access control models to access control metamodels: A survey," in *Advances in Information and Communication*. Cham, Switzerland: Springer, 2020, pp. 892–911.

[38] A. K. Malik, N. Emmanuel, S. Zafar, H. A. Khattak, B. Raza, S. Khan, A. H. Al-Bayatti, M. O. Alassafi, A. S. Alfakeeh, and M. A. Alqarni, "From conventional to state-of-the-art IoT access control models," *Electronics*, vol. 9, no. 10, p. 1693, 2020.

[39] B. Shebaro, O. Oluwatimi, and E. Bertino, "Context-based access control systems for mobile devices," *IEEE Trans. Depend. Sec. Comput.*, vol. 12, no. 2, pp. 150–163, Mar. 2015.

[40] N. K. Thanigaivelan, E. Nigussie, A. Hakkala, S. Virtanen, and J. Isoaho, "CoDRA: Context-based dynamically reconfigurable access control system for Android," *J. Netw. Comput. Appl.*, vol. 101, pp. 1–17, Jan. 2018.

[41] W. Ren, L. Zeng, R. Liu, and C. Cheng, "F2AC: A lightweight, fine-grained, and flexible access control scheme for file storage in mobile cloud computing," *Mobile Inf. Syst.*, vol. 2016, pp. 1–10, Jan. 2016.

[42] A. Oglaza, R. Laborde, P. Zaraté, A. Benzekri, and F. Barrère, "A new approach for managing Android permissions: Learning users' preferences," *EURASIP J. Inf. Secur.*, vol. 2017, no. 1, pp. 1–16, Dec. 2017.

[43] J. Abdella, M. Özuysal, and E. Tomur, "CA-ARBAC: Privacy preserving using context-aware role-based access control on Android permission system," *Secur. Commun. Netw.*, vol. 9, no. 18, pp. 5977–5995, Dec. 2016.

[44] Y. Baseri, A. Hafid, and S. Cherkaoui, "Privacy preserving fine-grained location-based access control for mobile cloud," *Comput. Secur.*, vol. 73, pp. 249–265, Mar. 2018.

[45] F. Wu, F. Luo, K. Chen, W. Lin, and Z. Lu, "A light-weight kernel-level mandatory access control framework for Android," in *Proc. IEEE Int. Conf. Power, Intell. Comput. Syst. (ICPICS)*, Jul. 2019, pp. 353–358.

[46] K. Muthumanickam and P. C. S. Mahesh, "A collaborative policy-based security scheme to enforce resource access controlling mechanism," *Wireless Netw.*, vol. 26, no. 4, pp. 2537–2547, May 2020.

[47] S. Dutta, S. S. L. Chukkapalli, M. Sulgekar, S. Krithivasan, P. K. Das, and A. Joshi, "Context sensitive access control in smart home environments," in *Proc. IEEE IEEE 6th Intl Conf. Big Data Secur. Cloud (BigDataSecurity) Intl Conf. High Perform. Smart Comput., (HPSC) IEEE Intl Conf. Intell. Data Secur. (IDS)*, May 2020, pp. 35–41.

[48] H. F. Atlam, M. A. Azad, M. O. Alassafi, A. A. Alshdadi, and A. Alenezi, "Risk-based access control model: A systematic literature review," *Future Internet*, vol. 12, no. 6, pp. 1–23, 2020.

[49] D. Choi, D. Kim, and S. Park, "A framework for context sensitive risk-based access control in medical information systems," *Comput. Math. Methods Med.*, vol. 2015, pp. 1–15, May 2015.

[50] D. Hintze, M. Muaaz, R. D. Findling, S. Scholz, E. Koch, and R. Mayrhofer, "Confidence and risk estimation plugins for multi-modal authentication on mobile devices using CORMORANT," in *Proc. 13th Int. Conf. Adv. Mobile Comput. Multimedia*, Dec. 2015, pp. 384–388.

[51] H. F. Atlam, R. J. Walters, G. B. Wills, and J. Daniel, "Fuzzy logic with expert judgment to implement an adaptive risk-based access control model for IoT," *Mobile Netw. Appl.*, pp. 1–13, Jan. 2019.

[52] A. Rahmati, E. Fernandes, K. Eykholt, and A. Prakash, "Tyche: A risk-based permission model for smart homes," in *Proc. IEEE Cybersecur. Develop. (SecDev)*, Sep./Oct. 2018, pp. 29–36.

[53] O. Olukoya, L. Mackenzie, and I. Omoronyia, "Permission-based risk signals for app behaviour characterization in Android apps," in *Proc. 5th Int. Conf. Inf. Syst. Secur. Privacy (ICISSP)*, vol. 1, 2019, pp. 183–192.

[54] S. Kumar and S. K. Shukla, "The state of Android security," in *Cyber Security in India* (IITK Directions), vol. 4, S. Shukla and M. Agrawal, Eds. Singapore: Springer, 2020, doi: 10.1007/978-981-15-1675-7_2.

[55] R. Kumar, X. Zhang, R. U. Khan, and A. Sharif, "Research on data mining of permission-induced risk for Android IoT devices," *Appl. Sci.*, vol. 9, no. 2, pp. 1–23, 2019.

[56] E. Alepis and C. Patsakis, "Unravelling security issues of runtime permissions in Android," *J. Hardw. Syst. Secur.*, vol. 3, no. 1, pp. 45–63, Mar. 2019, doi: 10.1007/s41635-018-0053-2.

[57] F. Alswaina and K. Elleithy, "Android malware family classification and analysis: Current status and future directions," *Electronics*, vol. 9, no. 6, p. 942, Jun. 2020, doi: 10.3390/electronics9060942.

[58] W. Wang, M. Zhao, Z. Gao, G. Xu, H. Xian, Y. Li, and X. Zhang, "Constructing features for detecting Android malicious applications: Issues, taxonomy and directions," *IEEE Access*, vol. 7, pp. 67602–67631, 2019.

[59] V. Moonsamy, J. Rong, and S. Liu, "Mining permission patterns for contrasting clean and malicious Android applications," *Future Gener. Comput. Syst.*, vol. 36, pp. 122–132, Jul. 2014.

[60] A. Sadeghi, H. Bagheri, J. Garcia, and S. Malek, "A taxonomy and qualitative comparison of program analysis techniques for security assessment of Android software," *IEEE Trans. Softw. Eng.*, vol. 43, no. 6, pp. 492–530, Jun. 2017.

[61] J. A. Gómez-Hernández, P. García-Teodoro, J. A. Holgado-Terriza, G. Maciá-Fernández, J. Camacho-Páez, and M. Robles-Carrillo, "AMon: A monitoring multidimensional feature application to secure Android environments," in *Proc. IEEE Secur. Privacy Workshops (SPW)*, 2021, pp. 31–36, doi: 10.1109/SPW53761.2021.00013.

[62] M. Bokhorst. *NetGuard: A Simple Way to Block Access to the Internet Per Application*. Accessed: Jul. 14, 2021. [Online]. Available: https://github.com/M66B/NetGuard/

[63] J. K. Santosh, S. Chakravarty, and K. V. P. Ravi, "Feature selection and evaluation of permission-based Android malware detection," in *Proc. 4th Int. Conf. Trends Electron. Informat. (ICOEI)*, Jun. 2020, pp. 795–799.

[64] W. Wang, X. Wang, D. Feng, J. Liu, Z. Han, and X. Zhang, "Exploring permission-induced risk in Android applications for malicious application detection," *IEEE Trans. Inf. Forensics Security*, vol. 9, no. 11, pp. 1869–1882, Nov. 2014.

[65] Y. Zhauniarovich and O. Gadyatskaya, "Small changes, big changes: An updated view on the Android permission system," in *Proc. Int. Symp. Res. Attacks, Intrusions, Defenses (RAID)*, in Lecture Notes in Computer Science, vol. 9854, 2016, pp. 346–367.

[66] S. Yoo, H. R. Ryu, H. Yeon, T. Kwon, and Y. Jang, "Visual analytics and visualization for Android security risk," *J. Comput. Lang.*, vol. 53, pp. 9–21, Aug. 2019.

[67] Microsoft. (2009). *The STRIDE Threat Model*. [Online]. Available: https://docs.microsoft.com/en-us/previous-versions/commerce-server/ee823878(v=cs.20)?redirectedfrom=MSDN

[68] N. Shevchenko. (2018). Threat modeling: 12 available methods. Software Engineering Institute. [Online]. Available: https://insights.sei.cmu.edu/blog/threat-modeling-12-available-methods/

[69] A. Shostack, "Experiences threat modeling at Microsoft," in *Proc. MODSEC@MoDELS*, 2008, pp. 1–11.

[70] J. Camacho, "Observation-based missing data methods for exploratory data analysis to unveil the connection between observations and variables in latent subspace models," *J. Chemometrics*, vol. 25, no. 11, pp. 592–600, 2011.

[71] J. Camacho, A. Pérez-Villegas, R. A. Rodríguez-Gómez, and E. Jiménez-Mañas, "Multivariate exploratory data analysis (MEDA) toolbox for MATLAB," *Chemometric Intell. Lab. Syst.*, vol. 143, pp. 49–57, Apr. 2015.

[72] P. Calciati, K. Kuznetsov, A. Gorla, and A. Zeller, "Automatically granted permissions in Android apps: An empirical study on their prevalence and on the potential threats for privacy," in *Proc. 17th Int. Conf. Mining Softw. Repositories (MSR)*. New York, NY, USA: Association for Computing Machinery, Jun. 2020, pp. 114–124, doi: 10.1145/3379597.3387469.

J. A. GÓMEZ-HERNÁNDEZ is currently an Associate Professor with the Department of Languages and Computer Systems, University of Granada, Spain, and a Researcher with the Network Engineering & Security Group (NESG). His research interests include computer and network security, especially related with operating systems and digital forensics.

J. CAMACHO is currently an Associate Professor with the Department of Signal Theory, Networking and Communications, and a Researcher with the Information and Communication Technologies Research Centre, CITIC for its initials in Spanish, and with the Network Engineering & Security Group (NESG), University of Granada, Spain. His research interests include exploratory data analysis, anomaly detection and optimization with multivariate techniques applied to data of very different nature, including industrial processes, chemometrics, and communication networks. He is especially interested in the use of exploratory data analysis to big data for cybersecurity and network metrics.

J. A. HOLGADO-TERRIZA (Member, IEEE) is currently an Associate Professor with the Department of Languages and Computer Systems, University of Granada, and a Researcher with the Concurrent System Group. His research interests include the Internet of Things and agent-based systems applied to ambient intelligence, smart homes, instrumentation, and industry. He is also interested in the development of real time, embedded and mobile systems for cyber-physical systems, wearables, and consumer electronic devices.

P. GARCÍA-TEODORO is currently a Full Professor with the Department of Signal Theory, Telematics and Communications, University of Granada, Spain. He is also the Head of the Research Group Network Security & Engineering Group (NESG), University of Granada. His current research interests include computer and network security, especially focused on anomaly-based intrusion detection and denial of service attacks.

G. MACIÁ-FERNÁNDEZ received the M.S. degree in telecommunications engineering from the University of Seville, Spain, and the Ph.D. degree in telecommunications engineering from the University of Granada, Spain. He is currently an Associate Professor with the Department of Signal Theory, Telematics and Communications, University of Granada. He is also a Researcher with the Information and Communication Technologies Research Centre, CITIC. His research interests include system and network security, with special focus on intrusion detection, ethical hacking, network information leakage, and denial of service.