Scientific Journal of Electronical & Cyber Defence
Vol. 11, No. 2, Summer 2023, Serial No. 42

Presenting A Method Based on Nearest Neighbors and Hamming Distance in Order to Identify Malicious Applications

M. Deypir

*Associate Professor, Faculty of Computer and Information Technology, Shahid Sattari Aviation University, Tehran, Iran

(Received: 09/09/2022, Accepted: 04/03/2023)

ABSTRACT

Nowadays, Android-based devices such as smart phones, tablets, and recently virtual reality headsets have found increasing usage in our daily lives. Along with the development of software for these devices, new malicious applications are released by intruders, which are more difficult to identify and deal with because they use more sophisticated methods. Although methods have been provided to calculate the security risk and identify malicious apps, but with the expansion of the level and depth of their threats, the need for new methods in this field is still required. In this study, we have presented a new algorithm to calculate the security risk of Android apps, which can be used to identify malicious apps from benign ones. In this algorithm, to estimate the security risk of an input app, the nearest neighbors of the type of malicious apps and the nearest neighbors of the type of normal apps are determined separately using Hamming distance. Then, based on the criteria presented in this article, the security risk of an unknown input app can be computed. After implementing this algorithm and adjusting the parameter of the number of neighbors with the help of real data, extensive various experiments were conducted in order to evaluate the proposed method. In these experiments, the proposed method was compared with three previously known methods in the context of detecting malicious apps, using four different datasets. The results show the higher detection rate of the proposed method in most cases.

Keywords: Malware, Hamming distance, Nearest neighbor, Security risk.

* Corresponding Author Email: Mdeypir@gmail.com

نشریه علمی "پدافند الکترونیکی و سایبری"
سال یازدهم، شماره ۲، تابستان ۱۴۰۲، ص۸۱-۹۰

علمی – پژوهشی

ارائه روشی مبتنی بر راهکار نزدیک‌ترین همسایه‌ها و فاصله همینگ به‌منظور شناسایی برنامه‌های مخرب

محمود دی‌پیر

دانشیار، دانشکده رایانه و فناوری اطلاعات، دانشگاه هوایی شهید ستاری، تهران، ایران

(دریافت: ۱۴۰۱/۰۶/۱۸، پذیرش: ۱۴۰۱/۱۲/۱۳)

چکیده

امروزه دستگاه‌های مبتنی بر اندروید مثل تلفن‌های همراه هوشمند، تبلت‌ها و اخیراً هدست‌های واقعیت مجازی، کاربرد روزافزونی در زندگی روزمره ما پیدا کرده‌اند. همراه با توسعه نرم‌افزارها برای این دستگاه‌ها، برنامه‌های مخرب جدیدی توسط نفوذگران منتشر می‌شود که شناسایی و مقابله با آن‌ها مشکل‌تر است چون از روش‌های پیچیده‌تری استفاده می‌کنند. اگرچه تاکنون روش‌هایی برای محاسبه خطر امنیتی و شناسایی برنامه‌های مخرب ارائه شده‌اند، اما با گسترش سطح و عمق تهدیدات آن‌ها، نیاز به روش‌های جدید در این زمینه همچنان احساس می‌شود. در این مقاله الگوریتم جدیدی به‌منظور محاسبه خطر امنیتی برنامه‌های اندروید ارائه داده‌ایم که می‌تواند در شناسایی برنامه‌های مخرب از برنامه‌های مفید به کار رود. در این الگوریتم برای محاسبه خطر امنیتی یک برنامه ورودی، به کمک فاصله همینگ نزدیک‌ترین همسایه‌ها از نوع برنامه‌های مخرب و نزدیک‌ترین همسایه‌ها از نوع برنامه‌های بی‌خطر به طور جداگانه مشخص می‌شوند. سپس بر اساس معیاری که در این مقاله ارائه شده است، خطر امنیتی برنامه ورودی محاسبه می‌گردد. پس از پیاده‌سازی این الگوریتم و تنظیم پارامتر تعداد همسایه به کمک مجموعه داده‌های واقعی، آزمایش‌های گسترده و متنوعی به‌منظور ارزیابی روش پیشنهادی صورت گرفت. در این آزمایش‌ها، روش پیشنهادی با سه روش شناخته شده قبلی در زمینه تشخیص برنامه‌های مخرب، به کمک چهار مجموعه داده مختلف، مقایسه شد. نتایج حاصل نشان‌دهنده نرخ تشخیص بالاتر روش پیشنهادی در اغلب موارد است.

کلیدواژه‌ها: بدافزار، فاصله همینگ، نزدیک‌ترین همسایه، خطر امنیتی

۱- مقدمه

امروزه استفاده از سیستم‌عامل اندروید محدود به دستگاه‌های موبایل نمی‌شود و طیف وسیعی از وسایل جمله تبلت‌ها، هدست‌های واقعیت مجازی، دستگاه‌های مسیریاب نصب شده روی خودروها، تلویزیون‌های هوشمند و غیره به این سیستم‌عامل مجهز شده‌اند. ارائه بدافزارهایی مثل بدافزار برادر بزرگ[1] [۱] که به‌صورت راه دور از اعمال کاربر در محیط واقعیت مجازی فیلم‌برداری می‌کند، نشان داد که این‌گونه دستگاه‌ها هم از حملات امنیتی توسط بدافزارها مصون نیستند. از طرف دیگر، فنون مورد استفاده توسط نفوذگران هم گسترش یافته است به نحوی که بسیار از بدافزارها روش‌های شناسایی مختلف را فریب داده و راه خود را برای واردشدن به سیستم میزبان پیدا می‌کنند. به‌عنوان نمونه، در سال ۲۰۲۲ خانواده‌ای از بدافزارها در گوگل‌پلی پس از سه میلیون دانلود و نصب توسط کاربران مختلف، به کمک محققین امنیتی شناسایی شده‌اند[۲]. این بدافزارها عموماً اعمال خرابکارانه‌ای چون سرقت داده‌های شخصی، انجام سوء استفاده‌های مالی، ایجاد درخواست‌های ساختگی و غیره را روی سیستم‌های قربانی انجام می‌دهند. بنابراین به‌منظور حفظ حریم شخصی افراد و محافظت از دارایی‌های آنها و همچنین جلوگیری از سوء استفاده از سیستم‌های آن‌ها، نیازمند راهکارهای جدید و قدرتمندتری هستیم که نسبت به روش‌های گذشته توان تشخیصی بالاتری داشته باشند.

در این مقاله معیار جدیدی بر مبنای ایده نزدیک‌ترین همسایه‌های مخرب و مفید به‌منظور اندازه‌گیری خطر امنیتی یک برنامه ارائه شده که نسبت به معیارهای ارائه شده قبلی کارائی بهتری دارد یعنی قادر است درصد بیشتری از بدافزارها را شناسایی کند. در این تحقیق، به‌منظور محاسبه دقیق خطر امنیتی از فاصله همینگ برای شناسایی نزدیک‌ترین همسایه‌ها استفاده شده است. بر اساس معیار پیشنهادی الگوریتمی توسعه داده شده است که قادر است با تخمین دقیق خطر امنیتی هر برنامه ورودی، کاربر را در تصمیم‌گیری برای نصب یا عدم نصب برنامه یاری رساند. برای ارزیابی روش پیشنهادی، از چهار مجموعه داده متفاوت شامل برنامه‌های مفید و مخرب، استفاده کرده‌ایم. آزمایش‌های ما نشان داد که روش پیشنهادی نسبت به روش‌های قبلی در شناسایی برنامه‌های مخرب به طور ملموسی بهتر عمل می‌کند.

* رایانامه نویسنده مسئول: Mdeypir@gmail.com

[1] Big Brother

ناشر: دانشگاه جامع امام حسین (ع)

در بخش بعد برخی از کارهای تحقیقاتی انجام شده مرتبط با امنیت اندروید معرفی شده‌اند. در بخش سوم، صورت‌مسئله بیان می‌شود. در بخش چهارم معیار پیشنهادی معرفی شده و نحوه محاسبه آن با استفاده از الگوریتم پیشنهادی تشریح شده است. در بخش پنجم آزمایش‌های مربوط به ارزیابی و مقایسه معیار پیشنهادی با معیارهای قبلی، ارائه شده است. در این بخش با استفاده از مجموعه داده‌های واقعی متشکل از صدها بدافزار و هزاران نرم‌افزار مفید شناخته شده اندروید، معیار پیشنهادی با معیارهای ارائه شده قبلی از نظر نرخ تشخیص بدافزارها، مقایسه خواهد شد. در نهایت این مقاله در بخش ششم جمع‌بندی و نتیجه‌گیری می‌شود.

۲- مروری بر تحقیقات گذشته

برای شناسایی بدافزارها در سیستم‌عامل‌های مختلف از جمله در سیستم‌عامل اندروید روش‌های گوناگونی تا کنون ارائه شده‌اند. می‌توان این روش‌ها را به دو دسته کلی تقسیم‌بندی کرد. دسته اول روش‌هایی هستند که خروجی به‌صورت صفر و یک دارند یعنی هر برنامه ورودی را به‌عنوان بدافزار یا نرم‌افزار مفید تشخیص می‌دهند. این روش‌ها عمدتاً از یادگیری ماشین به‌منظور ساخت مدل استفاده می‌کنند. اما روش‌های دسته دوم برای هر برنامه ورودی، میزان یا درصد ریسک امنیتی را تخمین می‌زنند. در دسته اول تلاش بر این است که با استفاده از داده‌های موجود، یک مدل دسته‌بندی دو کلاسه آموزش داده شود به‌نحوی‌که هر برنامه ورودی را بتوان به یکی از دو کلاس بدافزار یا برنامه مفید دسته‌بندی کرد. از بین مدل‌های مختلف موجود، روش‌های مبتنی بر ماشین بردار پشتیبان [۳،۴،۵] بیز ساده [۶،۷،۸] شبکه‌های عصبی[۹] و شبکه‌های عصبی عمیق[۱۰،۱۱] نسبت به سایر مدل‌ها، پرکاربردتر بوده و کارایی بهتری از خود نشان داده‌اند. ایجاد مدل دسته‌بندی برای شناسایی بدافزارها دارای نقاط ضعفی است. اول اینکه مدل‌ها معمولاً دارای خطا هستند، دوم اینکه با گذشته زمان و پیداشدن روش‌های نفوذ جدید کارایی خود را از دست می‌دهند و سوم اینکه با شناسایی این مدل‌ها به کمک روش‌های جعبه سفید و جعبه سیاه، نفوذگران قادر به فریب و عبور از آن‌ها خواهند شد به‌نحوی‌که می‌توانند نمونه‌های بدافزار را به‌صورت نرم‌افزار مفید نشان دهند[۱۲،۱۳]. اما دسته دوم روش‌های هستند که به‌جای تعیین دقیق دسته‌بندی یک برنامه ورودی، میزان خطر امنیتی را برای آن محاسبه می‌کنند. این روش‌ها مشکلات دسته اول را ندارند یا کمتر با آن مواجه هستند چون اولاً هدف، تعیین دسته‌بندی برنامه ورودی به بدافزار یا نرم‌افزار نیست که مدل دسته‌بندی دارای خطا باشد. دوم اینکه اگر بر اساس پارامترهای درستی ایجاد شوند کارایی آن‌ها چندان کاهش نمی‌یابد. سوم اینکه نفوذگران قادر به شناسایی نوع مدل یا روش آن‌ها نیستند چون از مدل‌های معمول یادگیری ماشین استفاده نمی‌کنند. بنابراین دلایل، دسته دوم نسبت به دسته اول روش‌ها برتری دارند.

گیت و دیگران تأثیر ارائه مؤثر خطر امنیتی به‌جای لیست مجوزهای مورد نیاز برنامه‌های اندروید، به کاربران را بررسی کرده اند. از آنجایی که کاربران افراد فنی نیستند، آن‌ها نشان دادند که محاسبه و ارائه میزان خطر امنیتی، در تصمیم‌گیری و انتخاب‌های آن‌ها می‌تواند مؤثر باشد زیرا آن‌ها از نصب برنامه‌های پر خطر و مخرب اجتناب خواهند کرد[۱۴]. پنگ و همکاران مدل‌هایی احتمالی به‌منظور محاسبه خطر امنیتی برنامه‌های اندروید ارائه کردند که می‌تواند به‌منظور رتبه‌بندی برنامه‌های اندروید بر حسب میزان خطر امنیتی آن‌ها به کار رود[۱۵]. این مدل‌ها در مقاله[۱۶] توسعه پیدا کردند و چندین معیار جدید به‌منظور محاسبه خطر امنیتی برنامه‌ها ارائه شدند. از بین آن‌ها مدل RSS نسبت به سایر مدل‌ها مناسب‌تر بود زیرا در عین سادگی، نرخ تشخیص بدافزار بالاتری را ارائه می‌داد. به این معنی که اگر لیستی از برنامه‌های مفید و مخرب به‌صورت مخلوط داشته باشیم و خطر امنیتی را برای آن‌ها محاسبه کنیم، به برنامه‌های مخرب نسبت به برنامه‌های مفید، میزان خطر امنیتی بالاتری بر اساس این معیار، تخصیص داده خواهد شد. در [۱۷] معیار جدیدی بر اساس بهره اطلاعاتی به‌منظور تخمین مخاطرات امنیتی نرم‌افزارهای اندروید ارائه شد. این معیار، بهره اطلاعاتی مجوزها را به‌منظور محاسبه خطر امنیتی به کار می‌برد. بر همین اساس در مقاله[۱۸] معیاری بنام ERS به‌منظور تخمین خطر امنیتی ارائه گردید که نسبت به RSS و دیگر معیارها کارایی خوبی از خود نشان داده است.

در مقاله [۱۹] بر اساس تحلیل ایستای برنامه‌ها، روشی به‌منظور تخمین ریسک امنیتی برنامه‌های اندروید ارائه شده است که تمرکز بر استفاده برنامه از داده‌ها به‌منظور جلوگیری از سرقت داده‌های شخصی دارد. در راهکار ترکیبی و توسعه یافته این روش[۲۰]، به کمک تحلیل ایستا و پویای برنامه، چگونگی استفاده از داده‌ها مورد پایش قرار می‌گیرد. این روش به برنامه‌هایی که استفاده از داده‌ها در آنها با هدف و کاربردشان مغایر باشد، مقدار خطر امنیتی بالایی را نسبت می‌دهد. در این روش اگر برنامه‌ای قصد جمع‌آوری داده‌های محلی شخصی و ارسال آن‌ها را داشته باشد، میزان خطر امنیتی محاسبه شده برای آن را افزایش می‌دهد. مشکل این روش خاص‌منظوره بودن آن است چون تمرکز بر داده‌های شخصی دارد و میزان خطر را بر حسب نحوه استفاده از آن‌ها می‌سنجد در حالی که بدافزارها می‌توانند طیف گسترده‌ای از خطرات امنیتی و سوء استفاده‌های مستقیم و غیر مستقیم را برای کاربران می‌توانند ایجاد کنند، مانند در اختیار گرفتن سیستم قربانی و استفاده از آن به‌عنوان زامبی برای

حمله به سایر اهداف. محاسبه خطر امنیتی برنامه‌های اندروید بر اساس نمونه در [۲۱] مورد بررسی قرار گرفت. در این روش، به‌جای استفاده از ویژگی‌های مختلف به‌صورت جداگانه، کل نمونه‌های قبلی برنامه‌ها و بدافزارها برای محاسبه خطر امنیتی برنامه ورودی مورد استفاده قرار می‌گیرد. در این روش به کمک بردار ویژگی برنامه‌های مفید و بدافزارها، مرکز هر دسته به دست می‌آید. سپس به کمک معیاری ساده، خطر امنیتی برنامه ورودی از طریق فاصله اقلیدسی بردار ویژگی آن تا مرکز برنامه‌ها و بدافزارها، محاسبه می‌شود.

صرف نظر از دسته‌بندی فوق برای تشخیص بدافزار، نیاز به استخراج ویژگی‌های لازم داریم زیرا هر دو دسته فوق، بر مبنای برنامه‌ها و بدافزارهای شناخته شده قبلی، عمل می‌کنند. این ویژگی‌ها عموماً با تحلیل ایستا و دیکامپایل کردن برنامه‌ها یا با تحلیل‌های رفتاری زمان اجرا، قابل استخراج هستند. اغلب تحقیقات انجام شده در زمینه شناسایی بدافزارهای اندروید، استخراج ویژگی را به‌عنوان مرحله اول روش خود معرفی کرده اند. مثلاً در [۲۲] ویژگی‌های متنوعی چون مجوزها، توابع، آدرس لینک‌های مورد استفاده، قصدها، ویژگی‌های سخت‌افزاری و مؤلفه‌ها، از برنامه‌ها و بدافزارهای موجود استخراج شده‌اند و از آن‌ها به‌منظور ساخت مدل بردار پشتیبان استفاده شده است. برخی از تحقیقات بر استفاده از مجوزهای درخواستی نرم‌افزارها تمرکز دارند و به تشخیص نرم‌افزارهای مخرب یا مشکوک پرداخته‌اند] ۲۳،۲۴]. برخی نیز از تحلیل ایستای کد نرم‌افزار، به‌صورت استخراج توابع برنامه‌نویسی مورد استفاده و مطابقت آن با برخی الگوهای موجود بدافزارها، برای تشخیص بدافزارهای جدید استفاده کرده‌اند [۲۸-۲۵]. تعدادی از محققین نیز روش هایی را ارائه داده‌اند که با تحلیل رفتاری نرم‌افزار در حال اجرا و استخراج ویژگی‌های پویا، سعی در تشخیص نرم‌افزارهای مخرب اندرویدی داشته‌اند [۳۳-۲۹].

باررا و همکاران [۳۴] روشی را برای ارزیابی عملی مدل‌های امنیتی بر اساس مجوز، به کمک نقشه‌های خودسازمانده[1] ارائه داده‌اند. آنها روش خود را به‌منظور تحلیل توزیع مجوزها بر روی هزار برنامه اعمال کرده و نشان دادند که چگونه استفاده از مجوز با دسته‌بندی برنامه‌ها ارتباط پیدا می‌کند. در [۳۵] تلاش شده است که با دیکامپایل کردن و تحلیل کد به‌دست‌آمده نرم‌افزارها، نشت داده را تشخیص دهند. انک و همکاران [۳۶] سامانه‌ای را توسعه دادند که ترکیب مجوزهای خطرناک را به‌منظور چگونگی برآورده‌کردن سیاست‌های امنیتی به کار می‌برد. در این سیاست‌ها به‌صورت دستی، ترکیب مجوزهای خطرناکی چون مجوزهای دستیابی به مکان و اینترنت در نظر گرفته شده است. این ترکیبات خود با تحلیل بدافزارهای شناخته شده به دست می‌آیند. ابزاری بنام MAST در [۳۷] توسعه داده شده که نرم‌افزارهایی که به احتمال زیاد بدافزار هستند را بر اساس تحلیل کد و تحلیل مجوزها تشخیص می‌دهد. ابزار PScout [۳۸] با استفاده از تحلیل ایستای کد اندروید، چگونگی نگاشت مجوز به تابع را در اندروید بررسی می‌کند. این ابزار نشان داد که سیستم مجوزهای اندروید حداقل افزونگی را داشته و این مسئله با توسعه اندروید و ارائه نسخه‌های جدید نیز پایدار باقی مانده است. برخی از تحقیقات نیز به‌صورت خاص به استخراج و مهندسی ویژگی‌ها پرداخته‌اند که از آن جمله می‌توان به [۳۹-۴۱] اشاره کرد. در [۳۹] تلاش شده که با بررسی منابع در سطوح مختلف یک برنامه اندرویدی ویژگی‌های مناسبی برای تشخیص بدافزار استخراج شود. در [۴۰] به‌جای مهندسی ویژگی‌ها به‌صورت دستی از روشی بر مبنای یادگیری عمیق و شبکه‌های عصبی کانولوشنال به‌منظور تحلیل ویژگی‌های لازم برای شناسایی بدافزارها استفاده شده است. هدف ما در مقاله پیش رو ارائه معیاری دقیق‌تر و قوی‌تر به‌منظور سنجش ریسک امنیتی نرم‌افزارها در اندروید است. از این نظر، تحقیق ما با تحقیقات ارائه شده در مقالات[۲۱-۱۴] نزدیکی بیشتری دارد یعنی دسته دوم از روش‌های شناسایی بدافزارهای اندرویدی که تمرکز آن‌ها بر محاسبه خطر امنیتی است.

۳- بیان مسئله

همان‌طور که در بخش قبل گفته شد محاسبه خطر امنیتی یکی از روش‌های شناسایی بدافزارهاست که می‌تواند مورد استفاده فروشگاه‌های برنامه نیز قرار گیرد. خطر امنیتی، مقداری است که با احتمال بدافزار بودن یک برنامه اندرویدی رابطه مستقیمی دارد. یعنی هرچه این مقدار بیشتر باشد احتمال بدافزار بودن یک نرم‌افزار بیشتر است، اما این مقدار خود از جنس احتمال نیست و از قوانین احتمالی تبعیت نمی‌کند. یعنی هیچ محدودیتی از نظر بازه مقادیر آن وجود ندارد. تنها کافی است برای بدافزارها مقدار بیشتری نسبت به نرم‌افزارها بدهد. نرم-افزارهای مخرب یا مشکوک معمولاً دارای خطر امنیتی بالایی هستند چون با استفاده از مجوزهایی که دریافت می‌کنند یا توابعی که استفاده می‌کنند و یا منابع سخت‌افزاری و نرم‌افزاری که در اختیار می‌گیرند، قادرند مخاطراتی را برای کاربر خود ایجاد کنند. استفاده از این منابع توسط برنامه اندروید A را می‌توان توسط بردارهای ویژگی دودویی به فرم $F = \{f_1,f_2,f_3,\ldots,f_n\}$ کدگذاری کرد که در آن مقدار f_i وضعیت استفاده یا عدم استفاده از منبع i توسط این برنامه را نشان می‌دهد. از این بردار ویژگی به‌منظور نمایش هر برنامه در فضای چندبعدی استفاده می‌گردد.

[1] Self-organizing maps

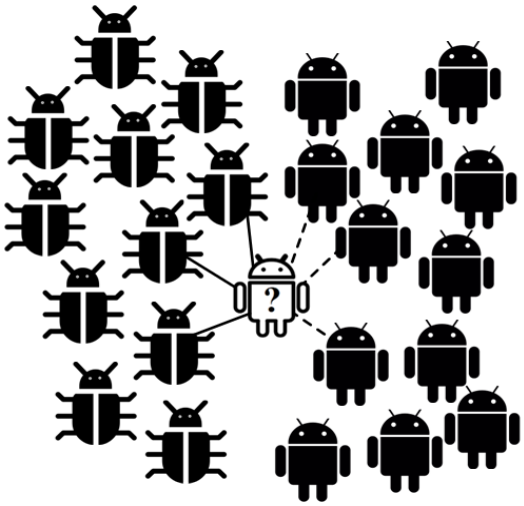
در نهایت به‌منظور محاسبه خطر امنیتی از مقادیر بردارهای ویژگی برنامه‌ها استفاده می‌شود. این خطر امنیتی با احتمال مخرب بودن برنامه ارتباط مستقیمی دارد.

اما بالابودن خطر امنیتی لزوماً به معنای مخرب بودن برنامه نیست زیرا برخی از نرم‌افزارهای مفید به دلیل قابلیت‌هایی که دارند از مجوزهای زیاد و حساسی استفاده و منابع مختلفی را در اختیار می‌گیرند تا خدمات لازم را برای کاربران فراهم کنند. بنابراین، برای تشخیص بهتر بدافزارها از نرم‌افزارها و جلوگیری از تشخیص اشتباه نیاز به معیاری داریم که الگو و رفتار بدافزارها و نرم‌افزارهای فعلی را به‌خوبی به کار برده و بتواند به نرم‌افزارهای مخرب، خطر امنیتی بالا و به نرم‌افزارهای مفید، تا حد امکان، خطر امنیتی پایینی نسبت دهد. بنابراین هدف ما ارائه معیار و روشی است که تخمین مناسب و معقولی برای میزان خطر امنیتی برنامه‌های اندروید به دست آورد. با توجه به گستردگی سطح و عمق تهدیدات بدافزارهای اندروید در سال‌های اخیر، مطلوب ما این است که معیار و روشی که ارائه می‌دهیم از نمونه-های مشابه خود قوی‌تر باشد یعنی نرخ تشخیص بدافزار بالایی داشته باشد. به بیان دیگر، اگر مخلوطی از برنامه‌های مفید و مخرب را بر حسب میزان خطر امنیتی تخمین شده توسط این معیار به‌صورت نزولی مرتب کنیم، تعداد بدافزار بیشتری در ابتدای لیست قرار گیرند و انتهای لیست اغلب شامل نرم‌افزارهای مفید با ریسک پایین باشد.

۴- روش پیشنهادی

برای محاسبه میزان خطر امنیتی برنامه‌ها، استفاده از اطلاعات گذشته مربوط به برنامه‌های مفید و مخرب ضروری است. اما چگونگی استفاده از آنها تأثیر زیادی در موفقیت هر روش دارد. از هر برنامه اندروید، صرف نظر از مفید یا مخرب بودن، ویژگی‌های مختلفی، قابل استخراج است. منظور از ویژگی ها، مشخصاتی مانند مجوزهای لازم، توابع مورد استفاده، منابع نرم‌افزاری و سخت‌افزاری، آدرس‌های شبکه و غیره هستند. این ویژگی‌ها با مهندسی معکوس برنامه‌های مفید و مخرب توسط ابزارهای مربوطه، قابل استخراج هستند. برخی از روش‌های گذشته، ویژگی‌های تأثیرگذار در مفید یا مخرب بودن برنامه‌های گذشته را شناسایی و از آن‌ها در محاسبه خطر امنیتی استفاده می‌کنند. مثلاً اینکه کدام مجوزها یا توابع در بدافزارهای قبلی نرخ استفاده بیشتری دارند یا اینکه کدام مجوزها یا توابع کمتر در برنامه‌های مفید استفاده شده‌اند. بنابراین، چنین روش‌هایی به دنبال بخش‌های تأثیرگذار در بردار ویژگی برنامه‌ها هستند.

اما در روش پیشنهادی تلاش می‌کنیم که از کل فضای ویژگی‌ها به‌منظور محاسبه ریسک امنیتی استفاده کنیم. ایده کلی به این صورت است که برای هر برنامه ناشناخته ورودی، ابتدا k نزدیک‌ترین همسایه برنامه مفید و k نزدیک‌ترین همسایه برنامه مخرب به دست می‌آیند. یعنی ابتدا نزدیک‌ترین همسایه‌ها به طور جداگانه برای برنامه‌های مفید و مخرب شناسایی می‌شوند و سپس بر اساس آنها میزان خطر امنیتی برنامه ورودی محاسبه می‌گردد. شکل(۱) شمای کلی ایده روش پیشنهادی را نشان می دهد. در این شکل برنامه‌های مخرب و مفید به ترتیب در سمت چپ و راست برنامه ورودی نشان داده شده‌اند. برنامه ورودی ناشناخته که می‌خواهیم برای آن میزان خطر امنیتی را محاسبه کنیم با رنگ سفید و علامت ? مشخص شده است. در اینجا k=3 در نظر گرفته شده است. همان‌طور که این شکل نشان می‌دهد، با محاسبه فاصله همه برنامه‌های مفید و مخرب با برنامه ناشناخته ورودی به‌صورت جداگانه با برنامه ورودی، ۳ نزدیک‌ترین همسایه از نوع برنامه مخرب و سه نزدیک‌ترین همسایه از نوع برنامه مفید شناسایی شده است. به کمک فرمول‌هایی که توضیح خواهیم داد میزان خطر امنیتی برای برنامه ناشناخته ورودی، محاسبه می شود.



شکل (۱). محاسبه فاصله تا نزدیک‌ترین همسایه‌های برنامه مفید و همسایه‌های برنامه مخرب.

با توجه به شکل (۱) محاسبه خطر امنیتی باید به نحوی باشد که هرچه فاصله برنامه ورودی با برنامه‌های مفید بیشتر باشد، میزان خطر امنیتی بیشتر شود و هرچه فاصله با برنامه‌های مخرب بیشتر باشد میزان خطر امنیتی کمتر شود. به عبارت دیگر خطر امنیتی با فاصله با برنامه‌های مفید نسبت مستقیم و با فاصله با برنامه‌های مخرب رابطه معکوس داشته باشد. برای به دست آوردن نزدیک‌ترین همسایه‌ها از هر نوع، از فاصله همینگ استفاده می‌شود. علت استفاده از فاصله همینگ ماهیت داده‌هاست چون هر برنامه با یک بردار بیتی مشخص می‌شود. پس از به دست آوردن نزدیک‌ترین همسایه‌ها از هر نوع، میانگین فاصله تا هر دسته از همسایه‌ها محاسبه می‌گردد. بنابراین دو مقدار میانگین به دست می‌آید. میانگین فاصله تا k نزدیک‌ترین

همسایه از نوع برنامه مفید با برنامه ورودی که به‌صورت زیر محاسبه می‌شود:

$$HM(X) = \frac{\sum_{j=1}^{K}\sum_{i=1}^{|F|}|X_i - NM_{ji}|}{K} \quad (۱)$$

در این فرمول HM منظور میانگین فاصله همینگ برنامه X تا k نزدیک‌ترین همسایه از نوع بدافزار است. X_i و NM_{ji} به ترتیب iامین ویژگی از برنامه ورودی و iامین ویژگی از jامین نزدیک‌ترین همسایه از نوع بدافزار هستند. متغییر F مجموعه ویژگی‌ها و |F| تعداد این ویژگی‌ها هستند. هر ویژگی یک متغییر دودویی است که استفاده یا عدم استفاده از مجوز، تابع یا هر منبع دیگری را برای هر برنامه را نشان می‌دهد. به طور مشابه، پس از بدست آوردن نزدیک‌ترین همسایه‌ها از نوع برنامه مفید، میانگین فاصله همینگ برنامه X (HB(x))تا k نزدیک‌ترین همسایه از نوع برنامه مفید به‌صورت زیر محاسبه می‌شود:

$$HB(X) = \frac{\sum_{j=1}^{K}\sum_{i=1}^{|F|}|X_i - NB_{ji}|}{K} \quad (۲)$$

در اینجا NB_{ji} به معنی i امین ویژگی از j امین نزدیک‌ترین همسایه از نوع برنامه مفید است. سایر موارد، مشابه فرمول قبل هستند. پس از محاسبه این دو مقدار می‌توان ریسک امنیتی برای برنامه ناشناخته X را به‌صورت زیر محاسبه کرد:

$$Risk(X) = \frac{HB(X)}{HM(X)} = \frac{\frac{\sum_{j=1}^{K}\sum_{i=1}^{|F|}|X_i - NB_{ji}|}{K}}{\frac{\sum_{j=1}^{K}\sum_{i=1}^{|F|}|X_i - NM_{ji}|}{K}} = \frac{\sum_{j=1}^{K}\sum_{i=1}^{|F|}|X_i - NB_{ji}|}{\sum_{j=1}^{K}\sum_{i=1}^{|F|}|X_i - NM_{ji}|} \quad (۳)$$

در این فرمول صورت و مخرج به ترتیب مجموع فاصله همینگ تا k نزدیک‌ترین همسایه از نوع برنامه مفید و مجموع فاصله همینگ تا k نزدیک‌ترین همسایه از نوع برنامه مخرب هستند. در واقع ایده اصلی این معیار بر این اساس است که هرچه فاصله همینگ برنامه ناشناخته تا نزدیک‌ترین برنامه‌های مخرب بیشتر باشد، خطر امنیتی کاسته می‌شود. در نقطه مقابل، هر چه فاصله تا نزدیک‌ترین برنامه‌های بی‌خطر بیشتر باشد، خطر امنیتی افزایش می‌یابد.

بر اساس مراحل ذکر شده و معیار ارائه شده فوق، الگوریتمی را طراحی کرده‌ایم که میزان خطر امنیتی برای یک برنامه ناشناخته ورودی را محاسبه می‌کند. این الگوریتم در شکل(۲) نشان داده شده است. این الگوریتم را NRS(Nearest Neighbour Risk Score) نامیده‌ایم چون با ایده k نزدیک‌ترین همسایه، خطر امنیتی را برای یک برنامه ورودی محاسبه می‌کند. نمادهای مورد استفاده در این الگوریتم در جدول(۱) نشان داده شده‌اند و ما بر اساس تعریف این نمادها، این الگوریتم را توضیح می‌دهیم. الگوریتم شکل(۲) مجموعه برنامه‌های شناخته شده قبلی، برنامه ورودی ناشناخته و K را به‌عنوان پارامترهای ورودی دریافت می کند. در خطوط اول و دوم، به ترتیب مجموعه برنامه‌های مفید و مجموعه برنامه‌های مخرب، از مجموعه اصلی به دست می‌آیند. در خط ۳ دو متغیری که برای ذخیره حاصل‌جمع فواصل نزدیک ترین همسایه‌های مفید و مخرب لازم هستند، مقداردهی اولیه می‌شوند.

```
Procedure NRS(SA, x, K)
begin
1.  SBA = {∀ benign App ∈ SA};
2.  SMA = {∀ malicious App ∈ SA};
3.  s_b = 0; s_m=0;
4.  for I =1 to K do
5.    nb = first member of SBA;
6.    nm = first member of SMA;
7.    h_b = HD(x, nb);
8.    h_m= HD(x, nm);
9.    for each b_i ∈ SBA do
10.     h_i = HD(x, b_i);
11.     if h_i ≤ h_b then
12.       nb = b_i;
13.       h_b = h_i
14.   Endfor;
15.   SBA = SBA − {nb}
16.   s_b+= h_b
17.   for each m_i ∈ SMA do
18.     h_i = HD(x, m_i);
19.     if h_i ≤ h_m then
20.       nm = m_i;
21.       h_m = m_i;
22.   Endfor;
23.   SMA = SMA − {nm}
24.   s_m+= h_m
25. Endfor;
26. return s_b / s_m ;
end
```

شکل (۲). الگوریتم محاسبه ریسک امنیتی بر اساس معیار پیشنهادی

در خطوط ۴ تا ۲۵ به تعداد K بار محاسبات لازم برای به دست آوردن نزدیک‌ترین همسایه‌های مفید و مخرب انجام می شوند که در ادامه توضیح خواهیم داد. در خطوط ۵ تا ۸ اولین برنامه‌ها از مجموعه مفیدها و مخرب‌ها به‌عنوان مقداردهی اولیه نزدیک‌ترین همسایه مفید و مخرب انجام می‌گیرد و فواصل همینگ آنها تا برنامه ورودی محاسبه می‌شوند. در خطوط ۹ تا ۱۴ با بررسی همه اعضاء متعلق به مجموعه برنامه‌های مفید و محاسبه فاصله همینگ آنها با برنامه ورودی، نزدیک‌ترین برنامه مفید را شناسایی می‌کنیم. در خط ۱۵ این برنامه را از مجموعه برنامه‌های مفید حذف کرده تا در دور بعدی بتوانیم سایر نزدیک ترین همسایه‌ها از این نوع را شناسایی کنیم. سپس مقدار فاصله همینگ آن را با مجموع جزئی فواصل همینگ تا نزدیک‌ترین همسایه‌های شناخته شده، جمع می‌کنیم تا در پایان بقیه دورها بتوانیم مجموع فواصل همینگ نزدیک‌ترین همسایه‌ها را در متغیر مربوطه داشته باشیم.

همین کارها را در خطوط ۱۷ تا ۲۴ برای برنامه‌های مخرب

انجام می‌دهیم تا در نهایت نزدیک‌ترین همسایه از نوع برنامه مخرب با برنامه ورودی را به دست آوریم و همچنین متغیرهای مربوطه را به‌روزرسانی کنیم. با اجرای K دور به همین روش، در نهایت مجموع فواصل K نزدیک‌ترین برنامه مفید و مجموع فواصل K نزدیک‌ترین برنامه مخرب به ترتیب در متغیرهای s_b و s_m قرار خواهند داشت. در نهایت در خط ۲۶ ریسک امنیتی برنامه ورودی با تقسیم این مقادیر بدست خواهد آمد.

جدول (۱). جدول نمادها

نام نماد	مفهوم نماد
SA	مجموعه برنامه مخرب و مفید
x	برنامه ناشناخته ورودی
K	تعداد نزدیک‌ترین همسایه
SBA	مجموعه برنامه‌های مفید
SMA	مجموعه برنامه‌های مخرب
nb	نزدیک‌ترین همسایه از نوع برنامه مفید
nm	نزدیک‌ترین همسایه از نوع برنامه مخرب
HD	تابع محاسبه فاصله همینگ دو برنامه
h_b	فاصله تا نزدیک‌ترین همسایه از نوع برنامه مفید
h_m	فاصله تا نزدیک‌ترین همسایه از نوع برنامه مخرب
b_i	عضوی از مجموعه برنامه‌های مفید
m_i	عضوی از مجموعه برنامه‌های مخرب
sb	مجموع فواصل تا نزدیک‌ترین همسایه‌های برنامه مفید
sm	مجموع فواصل تا نزدیک‌ترین همسایه‌های برنامه مخرب

در واقع این الگوریتم با محاسبه K نزدیک‌ترین همسایه از نوع برنامه مفید و K نزدیک‌ترین همسایه از نوع برنامه مخرب، اطلاعات لازم به‌منظور تخمین خطر امنیتی برنامه ورودی را فراهم می‌کند. در بخش بعد نشان می‌دهیم که چگونه با استفاده از داده‌های واقعی می‌توان مقدار مناسبی برای K به دست آورد تا به نرخ تشخیص خوبی برسیم. با توجه به اینکه الگوریتم پیشنهادی برای به دست آوردن نزدیک‌ترین همسایه‌ها و در نهایت محاسبه خطر امنیتی همه برنامه‌های مفید و مخرب قبلی موجود در حافظه را مورد توجه قرار می‌دهد، می‌توان نتیجه گرفت که مقدار محاسبه شده نهایی برای خطر امنیتی مقدار واقع‌بینانه‌ای خواهد بود زیرا بدافزارنویسان و نفوذگران به یکباره الگوهای شناخته شده قبلی را تغییر نمی‌دهند و اگر قرار است برنامه ورودی، یک برنامه مخرب باشد، با برنامه‌های مخرب قبلی فاصله همینگ زیادی نخواهد داشت و از طرفی از برنامه‌های مفید به‌اندازه کافی دور خواهد بود. بنابراین با توجه به فرمول(۳) میزان خطر امنیتی قابل توجهی برای آن محاسبه خواهد شد. بر عکس این حالت زمانی رخ می‌دهد که برنامه ورودی یک برنامه مفید باشد و خطر امنیتی حاصل میزان کمی خواهد بود. فرمول (۳) برای هر برنامه ورودی مقدار پیوسته‌ای برای خطر امنیتی محاسبه می‌کند. در اینجا اگر به‌مرورزمان تغییری در الگوهای مورد استفاده نفوذگران اتفاق بیفتد یا نفوذگر تلاش کند که سیستم را فریب دهد، منجر به تغییر در میزان خطر امنیتی خواهد شد که مقداری پیوسته است و مانند روش‌های یادگیری ماشین صفر و یک نیست که پاسخ اشتباه سبب شود که کاربر را در تصمیم‌گیری گمراه کند چون برای محدوده‌های پرخطر، کم‌خطر و بی‌خطر بازه‌هایی از مقادیر را می‌توان در نظر گرفت. تغییر اندک در مقدار ریسک همچنان نتیجه را در محدوده درست نگه خواهد داشت. به‌طورکلی نوآوری‌های روش پیشنهادی نسبت به سایر روش‌های تخمین خطر امنیتی برنامه‌ها را می‌توان در سه بخش خلاصه کرد. اول، استفاده از فاصله همینگ که برای اولین‌بار به‌منظور تخمین خطر امنیتی استفاده شده است. دوم، استفاده از نزدیک‌ترین همسایه‌های مفید و مخرب که به طور جداگانه به‌منظور محاسبه خطر امنیتی برنامه‌های ناشناخته به کار می‌روند. سوم ارائه معیار جدید برای محاسبه خطر امنیتی برنامه‌های اندروید که بر اساس آن الگوریتمی به‌منظور محاسبه این فرمول طراحی و پیاده‌سازی شده است. در بخش بعد الگوریتم پیشنهادی را به کمک داده‌های واقعی با روش‌های قبلی ارائه شده در این زمینه مقایسه خواهیم کرد.

۵- ارزیابی

به‌منظور ارزیابی، علاوه بر روش پیشنهادیNRS، مهم‌ترین معیارهای ارائه شده در تحقیقات قبلی را برای مقایسه با روش پیشنهادی خود انتخاب و پیاده‌سازی کرده‌ایم. این معیارها RSS [۱۶]، IRS [۲۱] و ERS [۱۸] هستند. با توجه به نتایج آزمایش-های ارائه شده در مقالات مربوطه، از بین روش‌های موجود، این سه روش کارایی خوبی از خود نشان داده و هر کدام نسبت به روش‌های قبل از خود بهتر عمل کرده‌اند. از طرف دیگر رویکرد محاسبه ریسک در [۱۰] حالت خاصی را مدنظر قرار داده و یک معیار کلی محسوب نمی‌گردد، به همین خاطر در آزمایش‌های ما مورد استفاده قرار نگرفته است. برای ارزیابی و مقایسه این معیارها، از چهار مجموعه داده توصیف شده در جدول(۲) استفاده کرده‌ایم که این مجموعه داده‌ها قبلاً در تحقیقات مختلف استفاده شده‌اند. در جدول (۲)، نام هر مجموعه داده، نوع برنامه‌های موجود در هر مجموعه داده، تعداد و نوع ویژگی‌های موجود، مشخص شده‌اند. با توجه به این جدول، تلاش کرده‌ایم از مجموعه داده‌های متنوعی مربوط به سال‌های مختلف برای ارزیابی و مقایسه روش‌ها استفاده کنیم تا نتایج ما قابل اعتمادتر باشند.

همان‌طور که گفته شد یک معیار خوب سنجش و تخمین خطر می‌بایست برای بدافزارها عدد بالایی تولید کند و برای نرم‌افزارهای مفید و غیر مخرب عدد کمی به دست آورد تا سبب تمایز این دو شود و بتوانیم نرم‌افزارهای پر خطر که عموماً همان بدافزارها هستند را شناسایی کنیم.

جدول(۲). مجموعه داده‌های مورد استفاده در ارزیابی معیار پیشنهادی

ردیف	نام مجموعه داده	تعداد و نوع برنامه	تعداد ویژگی	نوع ویژگی
۱	Market2011[14]	۱۳۶۵۳۴نرم‌افزار مفید	۱۲۲	مجوز
۲	Malwares2011 [31]	۸۰۸ بدافزار	۱۲۲	مجوز
۳	Drebin2014 [22]	۱۲۳۴۵۳ نرم‌افزار مفید و ۵۵۶۰ بدافزار	۳۸۵	مجوز، تابع سیستمی، قصدها، آدرس‌ها، منابع
۴	Deypir2019 [18]	۲۱۸۰ نرم‌افزار مفید و ۱۰۱۴ بدافزار	۱۳۵	مجوز

بنابراین، در آزمایش‌ها، تمرکز بر روی توانایی تشخیص[1] این معیارهاست. یعنی معیاری از نظر ما موفق است که بتواند به طور نسبی برای بدافزارها مقدار خطر امنیتی بیشتری محاسبه کند. یعنی اگر برای همه نرم‌افزارها و بدافزارها مقدار خطر را بر اساس یک معیار محاسبه کرده و سپس لیست کلی برنامه‌ها را به ترتیب نزولی مقدار خطر مرتب کنیم، بدافزارهای بیشتری نسبت به نرم‌افزارهای مفید در بالای لیست قرار گیرند. در آزمایش اول هدف پیداکردن مقدار پارامتر K برای روش پیشنهادی است و در آزمایش دوم، هدف مقایسه نرخ تشخیص روش پیشنهادی با سه روش ذکر شده قبلی است. با توجه به جدول(۲) مجموعه داده اول فقط شامل نرم‌افزار مفید است و مجموعه داده دوم فقط شامل بدافزار است، اما مجموعه داده سوم و چهارم هر کدام به‌تنهایی شامل نرم‌افزار و بدافزار هستند. به همین دلیل ما در آزمایش‌های خود، مجموعه داده‌های اول (Market2011) و دوم(Malwares2011) را در یک لیست واحد قرار می‌دهیم اما برای مجموعه داده‌های سوم و چهارم نیازی به این کار نیست چون شامل هر دو نوع هستند و نرم‌افزارهای مفید و بدافزارها در یک لیست واحد قرار دارند. در هر مورد با استفاده از ۹۰٪ لیست حاصل مدل خود را می‌سازیم. سپس با استفاده از ۱۰٪ باقیمانده به آزمایش هر معیار می‌پردازیم. به این صورت که با استفاده از مدل به‌دست‌آمده، خطر امنیتی آنها را محاسبه کرده و سپس به‌صورت نزولی مرتب کرده‌ایم. حال در هر بار، درصدهای مختلفی را از برنامه‌های بالای لیست مربوطه را انتخاب کرده و بررسی می‌کنیم که چه درصدی از بدافزارها در این بخش از لیست قرار گرفته‌اند. به درصدهای انتخاب شده لیست، سطح هشدار[2] و به درصدهای شناسایی شده بدافزارها، نرخ تشخیص گوئیم. اگرچه بازه مقادیر محاسبه‌شده در معیارهای مختلف متفاوت است، با استفاده از این روش ارزیابی، تفاوت در مقادیر تأثیری در مقایسه‌ها نخواهد داشت زیرا سطح هشدار در اینجا عدد خاصی نیست، بلکه نسبی است. بدیهی است که هرچه معیار قوی‌تر باشد، درصد بیشتری از بدافزارها در بالای لیست قرار می‌گیرد و نرخ تشخیص بیشتر خواهد بود. در هر مورد میزان تشخیص را بر حسب سطح هشدار به دست آورده‌ایم. در آزمایش اول تنها پارامتر روش پیشنهادی یعنی تعداد همسایه‌ها(K) است را تنظیم کرده‌ایم. عدد این پارامتر برای بدافزارها و برنامه‌های مفید یکسان است. بنابراین در اولین آزمایش، تلاش می‌کنیم که مقدار مناسبی برای این پارامتر به دست آوریم؛ بنابراین کارایی روش پیشنهادی برای مقادیر مختلف K را به دست می‌آوریم تا به مقدار مناسب برسیم. نرخ تشخیص روش پیشنهادی برای مقادیر مختلف k=1,2,3,5,10 را به دست آورده و به این نتیجه رسیدیم که بعد از K=5 تغییر ملموسی دیده نمی‌شود. در واقع بین K=5 تا K=10 اختلاف بسیار کمی وجود داشت. بنابراین پس از K=10 دیگر نیازی به افزایش بیشتر این متغیر نیست چون افزایش آن نه‌تنها تأثیری در کارایی ندارد بلکه زمان اجرا را افزایش می‌دهد. بنابراین مقدار نهایی این پارامتر را K=10 در نظر گرفته‌ایم.

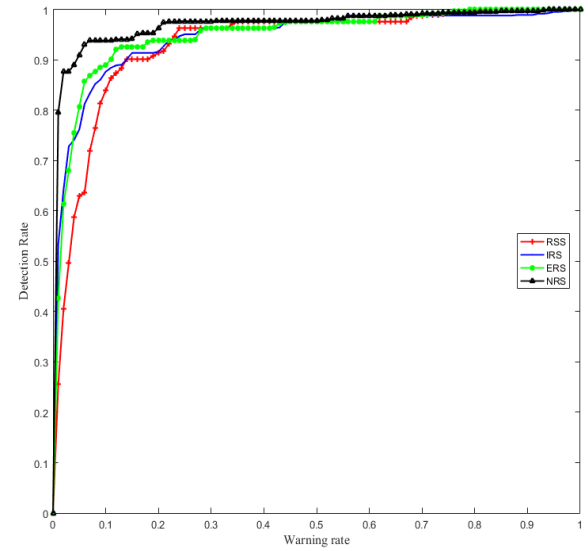
در ادامه، کارایی روش پیشنهادی با مقدار K=10 را با سایر روش‌ها مقایسه کرده‌ایم. شکل (۳) نمودار *ROC* برای معیار پیشنهادی و سایر معیارها روی مجموعه داده‌های مختلف را نشان می‌دهد. در این شکل محور افقی سطح هشدار و محور عمودی نرخ تشخیص بدافزار است. به عبارت دیگر، با انتخاب یک درصد مشخص از کل لیست در محور افقی، محور عمودی درصد بدافزارهای تشخیص داده شده را در این لیست نشان می‌دهد. همان‌طور که در بخش‌های مختلف این شکل پیداست، روش پیشنهادی نسب به همه روش‌های قبلی، به‌ویژه در مجموعه داده-های ادغام شده اول و دوم (بخش الف) و مجموعه داده چهارم (بخش ج) کارایی قابل توجهی از خود نشان می‌دهد. روش‌های ERS، IRS و RSS در رتبه‌های بعدی کارایی قرار دارند. در بخش الف) برتری روش پیشنهادی از همان ابتدا نسبت به سایر معیارها قابل توجه است که نشان‌دهنده قدرت بیشتر معیار پیشنهادی از نظر نرخ تشخیص است. در بخش الف میزان اختلاف نرخ تشخیص روش پیشنهادی نسبت به سایر روش‌ها در نرخ‌های هشدار پایین و متوسط قابل توجه است که نشان‌دهنده توان تشخیصی بالای معیار پیشنهادی است. اما با افزایش سطح هشدار کارایی همه معیارها به هم نزدیک می‌شود. در بخش ج) نرخ تشخیص روش پیشنهادی در سطوح هشدار پائین بسیار نزدیک به معیارهای قبلی است اما با افزایش سطح هشدار، اختلاف نرخ

[1] Detection Rate
[2] Warning Rate

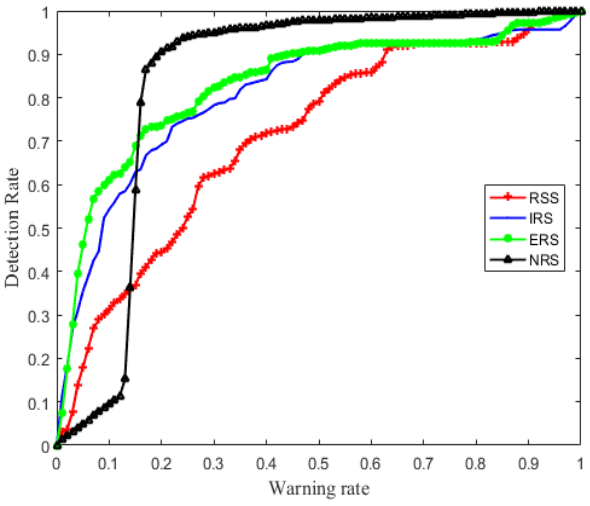
تشخیص روش پیشنهادی نسبت به سایر معیارها ملموس‌تر می‌شود.

در مورد مجموعه داده سوم (بخش ب) این شکل دیده می‌شود که برای نرخ‌های هشدار پایین، روش پیشنهادی کارایی خوبی از خود نشان نداده است اما با افزایش سطح هشدار، نرخ تشخیص آن افزایش یافته است و نسبت به سه روش دیگر، نرخ تشخیص بالاتری از خود نشان داده است. در این آزمایش مجموعه داده‌ای مورد استفاده قرار گرفته است که بعد بالایی دارد و باتوجه‌به جدول (۲) تعداد ویژگی‌های آن حدوداً سه برابر مجموعه‌های دیگر است. چون روش پیشنهادی یک روش بر اساس نمونه است، از تمام مقادیر این ویژگی‌ها استفاده می‌کند و نمی‌تواند در نرخ‌های هشدار پایین کارایی خوبی را از خود نشان دهد. اما با افزایش نرخ هشدار، کارایی بسیار خوبی از خود نشان داده و همان‌طور که شکل مربوطه نشان می‌دهد در نهایت برتری چشمگیری را نسبت به سایر روش‌ها ایجاد می‌کند. همان‌طور که از بخش‌های مختلف شکل (۳) پیدا است، به طور کلی روش پیشنهادی میزان نرخ تشخیص بالاتری را به خود اختصاص داده است که نشان می‌دهد معیار قوی‌تری برای محاسبه خطر امنیتی و در نهایت تشخیص بدافزارهاست.

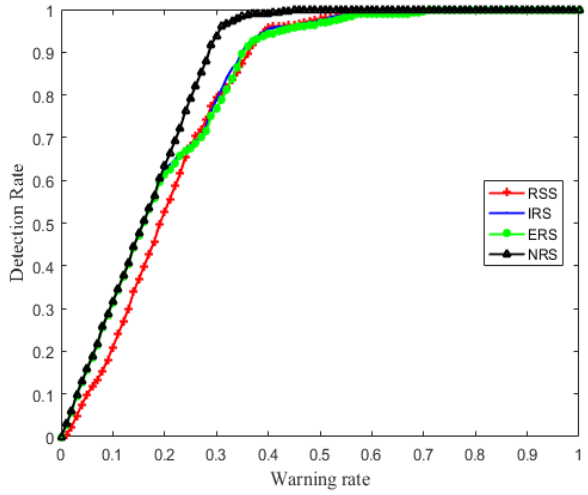
علت برتری روش پیشنهادی نسبت به روش‌های قبلی این است که در روش پیشنهادی از ایده نزدیک‌ترین همسایه‌های بدافزار و نزدیک‌ترین همسایه‌های نرم‌افزار، جداگانه برای محاسبه ریسک استفاده شده است. از طرفی با توجه به ماهیت دودویی داده‌ها، فاصله همینگ استفاده شده است. علاوه بر این، روش پیشنهادی فاصله همینگ را در فضای چندبعدی نمونه‌ها محاسبه کرده و از اطلاعات سایر نمونه‌های بد و خوب به‌درستی استفاده می‌کند، حال آنکه در روش‌های ERS و RSS از ویژگی‌ها به طور جداگانه برای محاسبه ریسک استفاده می‌شود. در واقع در معیارهای ERS و RSS، ریسک هر ویژگی جدا حساب می‌شود، درصورتی که در روش پیشنهادی کل نمونه‌های بد و خوب برای محاسبه ریسک با توجه به معیار فرمول (۳) برای محاسبه ریسک استفاده می‌گردند. در معیار IRS نیز اگرچه از کل فضای چندبعدی نمونه‌ها با هم بهره‌برداری می‌شود اما فاصله‌ها تا مرکز نمونه‌های بد و خوب محاسبه می‌گردد و از اطلاعات نزدیک‌ترین همسایه‌ها استفاده نمی‌شود. از طرفی IRS از فاصله اقلیدسی استفاده کرده است که با طبیعت داده‌ها سازگار نیست. بنابراین روش پیشنهادی نسبت به سایر معیارهای محاسبه ریسک بهتر عمل می‌کند.



الف) مقایسه نمودارهای ROC در ادغام مجموعه داده‌های اول و دوم



ب) مقایسه نمودارهای ROC در مجموعه داده سوم



ج) مقایسه نمودارهای ROC در مجموعه داده چهارم

شکل(۳). مقایسه نرخ تشخیص معیارهای مختلف به کمک نمودارهای ROC آنها

۶- جمع‌بندی و نتیجه‌گیری

با توجه به گسترش استفاده از سیستم‌عامل اندروید در دستگاه‌ها و سامانه‌های مختلف و همچنین پیچیده‌تر شدن حملات و آسیب‌پذیری‌های امنیتی آن از طریق بدافزارها، نیاز به ارائه روش‌های قوی‌تر برای شناسایی و مقابله با این تهدیدات احساس می‌شود. در این مقاله معیار جدیدی برای محاسبه خطر امنیتی برنامه‌های اندروید معرفی شد و بر مبنای آن الگوریتم جدیدی توسعه یافت. آزمایش‌های صورت‌گرفته روی داده‌های واقعی مربوط به برنامه‌های بی‌خطر و بدافزارهای اندروید نشان داد که روش پیشنهادی از نظر نرخ تشخیص بسیار بهتر از روش-های اخیر عمل می‌کند. برتری روش پیشنهادی نسبت به روش-های قبلی دو دلیل اصلی دارد. اول اینکه معیار پیشنهادی از نزدیک‌ترین همسایه‌های برنامه مفید و مخرب استفاده می‌کند که سبب استفاده از کل بردار ویژگی برنامه‌های قبلی به‌منظور تخمین بهتر ریسک امنیتی می‌شود. دوم اینکه استفاده از فاصله همینگ با توجه به ماهیت دودویی بردارهای ویژگی نسبت به سایر انواع فاصله مثل فاصله اقلیدسی، مناسب‌تر است. برای ادامه این تحقیق می‌توان از یادگیری عمیق به‌منظور تشخیص دقیق‌تر برنامه‌های مخرب استفاده کرد. استفاده از روش‌های فرا ابتکاری و بهینه‌سازی به‌منظور تنظیم پارامترهای روش یادگیری در شناسایی برنامه‌های مخرب، از دیگر زمینه‌های تحقیقاتی آینده خواهد بود.

۷- مراجع

[1] Inside, “Hackers remotely connect to VR devices via Big Brother malware,” https://inside.com/xr/posts/hackers-remotely-connect-to-vr-devices-via-big-brother-malware-299588,” 2022.

[2] B. Toulas, “New Android malware on Google Play installed 3 million times,” https://www.bleepingcomputer.com/news/security/new-android-malware-on-google-play-installed-3-million-times/, 2022.

[3] L. Wen and H. Yu, “An Android malware detection system based on machine learning,” AIP conference proceedings. vol. 1864, No. 1. AIP publishing, 2017.

[4] S. Gunalakshmii and P. Ezhumalai, “Mobile keylogger detection using machine learning technique,”In Proceedings of IEEE International Conference on Computer Communication and Systems, pp. 051–056, 2014.

[5] J. Sahs and L. Khan, “A Machine Learning Approach to Android Malware Detection,” 2012 Eur. Intell. Secur. Informatics Conf., pp. 141–147, 2012.

[6] S. Y. Yerima, S. Sezer, and I. Muttik, “Android Malware Detection Using Parallel Machine Learning Classifiers,” In Eighth international conference on next generation mobile apps, services and technologies, pp. 37–42, 2014.

[7] M. G. Schultz, E. Eskin, E. Zadok, and S. J. Stolfo, “Data Mining Methods for Detection of New Malicious Executables,” Proc. 2001 IEEE Symp. Secur. Priv., p. 38--, 2001.

[8] W. G. Hatcher, D. Maloney, and W. Yu, “Machine learning-based mobile threat monitoring and detection,” 2016 IEEE/ACIS 14th Int. Conf. Softw. Eng. Res. Manag. Appl. SERA 2016, pp. 67–73, 2016.

[9] C. Gavrilu, Drago, Mihai, D. Anton, and L. Ciortuz, “Malware detection using machine learning,” Comput. Sci. Inf. Technol. 2009. IMCSIT’09. Int. Multiconference, pp. 735–741, 2009.]

[10] Y. Chen, Y. Li, A. Tseng, and T. Lin, “Deep Learning for Malicious Flow Detection,” IEEE Access, p. 7, 2018

[11] Rahali, A., Lashkari, A. H., Kaur, G., Taheri, L., Gagnon, F., & Massicotte, F. (2020, November). Didroid: Android malware classification and characterization using deep image learning. In *2020 The 10th international conference on communication and network security* (pp. 70-82).

[12] H. Li, S. Zhou, W. Yuan, X. Luo, C. Gao, S. Chen, Robust android malware detection against adversarial example attacks. In *Proceedings of the Web Conference 2021*, pp. 3603-3612.

[13] H. Li, S. Zhou, W. Yuan, J. Li, and H. Leung,. Adversarial-example attacks toward android malware detection system. *IEEE Systems Journal*, *14*(1), 2019, pp. 653-656.

[14] C. S. Gates, J. Chen, N. Li, and R. W. Proctor, “Effective risk communication for android apps,” *IEEE Transactions on dependable and secure computing*, vol. 11, no. 3, pp. 252-265, 2013.

[15] H. Peng, C. Gates, B. Sarma, N. Li, Y. Qi, R. Potharaju, R., and I. Molloy, “Using probabilistic generative models for ranking risks of android apps,” In *Proceedings of the 2012 ACM conference on Computer and communications security*, ACM, October 2012, pp. 241-252.

[16] C. S. Gates, N. Li, H. Peng, B. Sarma, Y. Qi, R. Potharaju, and I. Molloy, “Generating summary risk scores for mobile applications,” *Dependable and Secure Computing, IEEE Transactions on*, vol. 11, no. 3, pp. 238-251, 2014.

[17] M. Deypir, “Estimating Security Risks of Android Apps Using Information Gain,” *Electronic and Cyber Defense*, vol. 5, no. 1, pp. 73-83, 2017. (in Persian).

[18] M. Deypir, “Entropy-based security risk measurement for Android mobile applications,” *Soft Computing*, vol. *23*, no. 16, pp. 7303-7319, 2019.

[19] H. X. Son, B. Carminati, and E. Ferrari, “A Risk Assessment Mechanism for Android Apps,” In *2021 IEEE International Conference on Smart Internet of Things (SmartIoT)*, August 2021, pp. 237-244.

[20] H. X. Son, B. Carminati, E. Ferrari, “A Risk Estimation Mechanism for Android Apps based on

system for android," In *Proceedings of the 1st ACM workshop on Security and privacy in smartphones and mobile devices*, October 2011, pp. 15-26.

[32] Y. Zhou, and X. Jiang, "Dissecting android malware: Characterization and evolution", In *Security and Privacy (SP), 2012 IEEE Symposium on* May 2012, pp. 95-109.

[33] D. Barrera, H. G. Kayacik, P. C. van Oorschot, and A. Somayaji, "A methodology for empirical analysis of permission-based security models and its application to android," In *Proceedings of the 17th ACM conference on Computer and communications security*, October 2010, pp. 73-84.

[34] D. Barrera, H. G. Kayacik, P. C. van Oorschot, and A. Somayaji, "A methodology for empirical analysis of permission-based security models and its application to android," In *Proceedings of the 17th ACM conference on Computer and communications security*, October 2010, pp. 73-84.

[35] W. Enck, D. Octeau, P. McDaniel, and S. Chaudhuri, "A Study of Android Application Security," In *USENIX security symposium*, August 2011 Vol. 2, p. 2.

[36] W. Enck, M. Ongtang, and P. McDaniel, "On lightweight mobile phone application certification," In *Proceedings of the 16th ACM conference on Computer and communications security*, November 2009, pp. 235-245.

[37] S. Chakradeo, B. Reaves, P. Traynor, W. Enck, "Mast: triage for market-scale mobile malware analysis," In *Proceedings of the sixth ACM conference on Security and privacy in wireless and mobile networks*, April 2013, pp. 13-24.

[38] K. W. Y. Au, Y. F. Zhou, Z. Huang, D. Lie, "Pscout: analyzing the android permission specification," In *Proceedings of the 2012 ACM conference on Computer and communications security*, October 2012, pp. 217-228.

[39] Yang, M., & Wen, Q. (2016, August). Detecting android malware with intensive feature engineering. In *2016 7th IEEE International Conference on Software Engineering and Service Science (ICSESS)* (pp. 157-161). IEEE.

[40] N. Zhang, Y. A. Tan, C. Yang, and Y. Li, "Deep learning feature exploration for android malware detection," *Applied Soft Computing*, vol. 102, 2021.

Hybrid Analysis," *Data Science and Engineering*, 2022, pp. 1-11.

[21] M. Deypir, A. Horri, "Instance based security risk value estimation for Android applications," *Journal of information security and applications*, vol. 40, pp. 20-30, 2018.

[22] D. Arp, M. Spreitzenbarth, M. Hubner, H. Gascon, K. Rieck, and C.E.R.T Siemens, "Drebin: Effective and explainable detection of android malware in your pocket," In *Ndss*, Vol. 14, February 2014,pp. 23-26.

[23] D. Geneiatakis, I. N. Fovino, I. Kounelis, and P. Stirparo, "A Permission verification approach for android mobile applications," *Computers & Security*, vol. 49, pp.192-205, 2015.

[24] B. P. Sarma, N. Li, C. Gates, R. Potharaju, C. Nita-Rotaru, and I. Molloy, "Android permissions: a perspective combining risks and benefits," In *Proceedings of the 17th ACM symposium on Access Control Models and Technologies*, June 2012, pp. 13-22.

[25] A. D. Schmidt, R. Bye, H. G. Schmidt, J. Clausen, O. Kiraz, K. Yüksel, and S. Albayrak, "Static analysis of executables for collaborative malware detection on android," In *Communications, 2009. ICC'09. IEEE International Conference on*, June 2009, pp. 1-5.

[26] Y. Zhou, Z. Wang, W. Zhou, and X. Jiang, "Hey, You, Get Off of My Market: Detecting Malicious Apps in Official and Alternative Android Markets," In *NDSS*, Vol. 25, No. 4, February 2012, pp. 50-52.

[27] Y. Aafer, W. Du, and H. Yin, "DroidAPIMiner: Mining API-level features for robust malware detection in android," In *Security and Privacy in Communication Networks*, 2013, pp. 86-103.

[28] M. Christodorescu, S. Jha, C. Kruegel, "Mining specifications of malicious behavior," In *Proceedings of the 1st India software engineering conference*, ACM, February 2008, pp. 5-14.

[29] K. Rieck, T. Holz, C. Willems, P. Düssel, and P. Laskov, "Learning and classification of malware behavior," In *Detection of Intrusions and Malware, and Vulnerability Assessment*, 2008, pp. 108-125.

[30] A. Shabtai, and Y. Elovici, "Applying behavioral detection on android-based devices," In *Mobile Wireless Middleware, Operating Systems, and Applications*, 2010, pp. 235-249.

[31] I. Burguera, U. Zurutuza, and S. Nadjm-Tehrani, "Crowdroid: behavior-based malware detection