نشریه علمی پدافند غیرعامل

سال سیزدهم، شماره ۲، تابستان ۱۴۰۱، (پیاپی ۵۰): صص ۱۰۶-۹۷

علمی- ترویجی

تحلیل ایستای ساختار فایل اجرایی جهت شناسایی و خوشه‌بندی بدافزارهای ناشناخته

حمید تنها[۱]، مصطفی عباسی[۲*]

تاریخ دریافت: ۱۴۰۰/۰۹/۰۶

تاریخ پذیرش: ۱۴۰۰/۱۲/۱۷

چکیده

یکی از روش‌های محبوب شناسایی بدافزار، تطبیق الگوی امضای فایل بدافزار با پایگاه داده امضای بدافزارها است. پایگاه داده امضای بدافزار از قبل استخراج شده و به‌طور مداوم به‌روزرسانی می‌گردد. بررسی شباهت داده‌های ورودی با بهره‌گیری از امضاهای ذخیره شده موجب بروز مشکلات ذخیره‌سازی و هزینه محاسبات می‌گردد. علاوه بر این، شناسایی مبتنی بر تطبیق الگوی امضای بدافزاری در زمان تغییر کد بدافزار در بدافزارهای چند ریخت، با شکست مواجه می‌شود. در این مقاله با ترکیب روش تحلیل ایستای ساختار فایل اجرایی و الگوریتم‌های یادگیری ماشین، روش مؤثری جهت شناسایی بدافزارها ارائه شده است. مجموعه داده برای آموزش و ارزیابی روش پیشنهادی شامل ۳۶۵۶۷ نمونه بدافزاری و ۱۷۲۹۵ فایل بی‌خطر است و در روش پیشنهادی، بدافزارها را در ۷ خانواده، خوشه‌بندی می‌نماید. نتایج نشان می‌دهد که روش پیشنهادی قادر است با دقت بیش از ۹۹ درصد و با نرخ هشدار اشتباه کمتر از ۰/۴ درصد بدافزارها را از فایل‌های سالم تشخیص و خوشه‌بندی نماید. روش پیشنهادی نسبت به روش‌های مشابه، دارای سربارهای پردازشی بسیار کم بوده و مدت زمان پویش فایل‌های اجرایی به‌طور متوسط ۰/۲۴۴ ثانیه طول است.

کلید واژه‌ها: تشخیص بدافزار، ساختار فایل اجرایی، تحلیل ایستا، خوشه‌بندی، یادگیری ماشین

[۱] کارشناس ارشد فناوری اطلاعات، پژوهشگر، دانشگاه جامع امام حسین(ع)، تهران، ایران

[۲] پژوهشگر، دانشگاه جامع امام حسین(ع)، تهران، ایران - (moabbasi@ihu.ac.ir)- نویسنده مسئول

۱- مقدمه

بدافزار به نرم‌افزار بدخواهی اطلاق می‌شود که با اهداف مختلفی نظیر خراب‌کاری، جاسوسی، سرقت و غیره توسعه می‌یابند. روند توسعه و انتشار بدافزارها در سال‌های اخیر شتاب چشمگیری داشته است به گونه‌ای که بر اساس گزارش‌های منابع امنیتی معتبر در سال ۲۰۱۳ روزانه بیش از ۱۰۰۰۰۰ بدافزار در روز منتشر می‌شده است. این میزان انتشار در سال ۲۰۱۹ به میزان روزانه بیش از ۳۵۰۰۰۰ بدافزار جدید افزایش یافته است [۱] و علاوه بر کاربران سامانه‌های نرم‌افزاری خانگی، تجهیزات کنترل صنعتی نیز در معرض این دسته از حملات سایبری قرار دارند [۲]. از جمله دلایل این روند فزاینده می‌توان به تجهیز بدافزارها به موتورهای مبهم سازی، چندریختی و دگردیسی اشاره نمود. این تجهیزات، بدافزار پایه را قادر به تولید نمونه‌های بی‌شماری می‌سازد که در عملکرد یکسان ولی در ظاهر متفاوت هستند [۳]. تفاوت در ظاهر اعضای خانواده یک بدافزار موجب ایجاد تفاوت در امضای آن‌ها می‌گردد، در نتیجه شیوه شناسایی مبتنی بر تطبیق الگو در شناسایی این خانواده از بدافزارها ناکارآمد است [۴ و ۵]. علاوه بر این کاهش سرعت با افزایش حجم بانک اطلاعاتی و تأخیر در اعمال به‌روزرسانی بسیار مشهود خواهد بود [۶]؛ بنابراین تلاش‌ها به سمت شناسایی هوشمندانه بدافزار فارغ از امضاء سوق پیدا کرد. در این مقاله سعی می‌گردد روشی نوین و مؤثر برای شناسایی هوشمندانه بدافزارها و خوشه‌بندی آن‌ها مبتنی بر تحلیل ایستای ساختار فایل‌های اجرایی ارائه گردد.

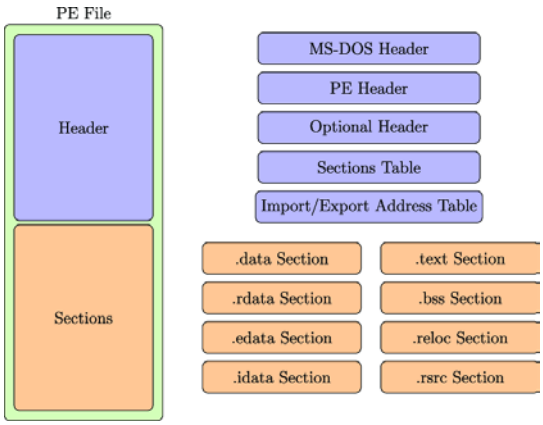
این مقاله در شش بخش سازمان‌دهی شده است. در بخش دوم روش‌های تشخیص بدافزار معرفی و مقایسه شده و در بخش سوم پیشینه موضوع و کارهای انجام شده بیان می‌گردد. در بخش چهارم روش پیشنهادی در تحلیل بدافزار مبتنی بر تحلیل ایستای ساختار فایل اجرایی و نحوه تشخیص و خوشه‌بندی بدافزارها ارائه شده است. نتایج ارزیابی و تحلیل روش پیشنهادی در بخش پنجم بیان شده و در پایان نتیجه‌گیری مقاله ارائه شده است.

۲- بررسی روش‌های تشخیص بدافزار

در این بخش روش‌های شناسایی و تحلیل بدافزار معرفی و مزایا و معایب هر کدام بیان می‌گردد. در پایان این بخش مقایسه کوتاه بین این روش‌ها ارائه شده است و جایگاه روش پیشنهادی و مزایا و محدودیت‌های آن نسبت به سایر روش‌ها عنوان می‌گردد.

۲-۱- ساختار فایل اجرایی قابل حمل

پیش از پرداختن به شیوه‌های موجود در شناسایی بدافزار نیاز است در خصوص ساختار فایل‌های اجرایی قابل حمل مطالبی بیان گردد. این ساختار یک قالب استاندارد است که جهت اجرای فایل‌های اجرایی از نسخه سیستم عامل مایکروسافت ویندوز NT تاکنون توسط مایکروسافت مورد استفاده قرار گرفته [۷ و ۸] و در شکل (۱) نمای کلی ساختار PE ارائه شده است.



شکل (۱): ساختار فایل اجرایی قابل‌حمل در سیستم عامل ویندوز [۹]

همان‌طور که در شکل (۱) مشاهده می‌شود، ساختار فایل اجرایی قابل حمل از قسمت‌هایی[1] مختلفی با اهداف متفاوتی تشکیل شده‌است. اولین قسمت MS DOS Header بوده که از زمان سیستم عامل داس[2] باقی مانده است. قسمت دوم این ساختار، قسمت کد است که بر اساس نوع کامپایلر به‌صورت اختصاری text. یاcode. نامیده می‌شود و در این قسمت کدهای اجرایی قرار می‌گیرد. قسمت‌های دیگر این ساختار شامل قسمت‌های data. که دربردارنده داده‌های مورد نیاز برنامه، idata، شامل داده‌های ورودی و edata. حاوی داده‌های خروجی برنامه است. قسمت idata یکی از مهم‌ترین بخش‌هایی است که در روش پیشنهادی برای تشخیص بدافزار مورد استفاده قرار گرفته و در این قسمت جدول آدرس ورودی مشخص شده است. این جدول دربردارنده نام و آدرس توابع سامانه‌ای است که برنامه مورد نظر حین اجرا به آن‌ها نیاز دارد. در قسمت edata میز جدول آدرس خروجی وجود دارد که نام و آدرس توابعی نگهداری می‌شود که توسط برنامه جهت استفاده سایر برنامه‌ها ارائه می‌گردد. در انتهای این ساختار، اطلاعات مربوط به اشکال‌زدایی برنامه شامل نمادهای اشکال‌زدایی نگهداری می‌شود [۸ و ۱۰].

۲-۲- شناسایی مبتنی بر تحلیل پویا

شیوه‌های شناسایی مبتنی بر تحلیل پویا با هدف جبران نقاط ضعف و محدودیت‌های موجود در روش‌های شناسایی مبتنی بر تحلیل ایستا ارائه شده‌اند. ویژگی بارز این روش‌ها، اجرای فایل مشکوک با هدف دست‌یابی به یک برآورد از میزان خوش‌خیم و یا بدخیمی فایل مورد نظر است [۱۱ و ۱۲].

[1] Section
[2] DOS

۳- پیشینه موضوع و کارهای انجام شده

در این قسمت مروری اجمالی بر برخی از کارهای انجام شده در زمینه شناسایی بدافزار مبتنی بر ویژگی‌های ایستای فایل اجرایی ارائه شده و هدف از این بخش مشخص نمودن جایگاه روش پیشنهادی و مزایا و محدودیت‌های آن است.

روش استفاده شده توسط اسچالتز و همکاران [۱۳] مبتنی بر استفاده از داده‌کاوی برای تشخیص بدافزارها است که در آن خواصی از ساختار فایل اجرایی از جمله نام کتابخانه‌های مورد استفاده، نام و تعداد توابعی استفاده شده از هر یک از کتابخانه‌ها و جستجوی رشته‌های متنی استفاده شده است. در این روش از الگوریتم Ripper برای تولید نقش‌ها، Naïve Byes برای آموزش و n-gram برای جستجوی رشته‌ها استفاده شده است. اگر چه دقت روش پیشنهادی بیش از ۹۷ درصد بیان شده است اما نرخ مثبت کاذب بسیار بالایی دارد.

در پژوهش متیو شولتز و همکاران [۱۴]، با بهره‌گیری از روش‌های داده‌کاوی در ویژگی‌های استخراج شده از فایل اجرایی روشی برای تشخیص بدافزارها ارائه نموده‌اند. در این پژوهش از استخراج سه ویژگی رفتار، رشته‌ها و توالی بایت‌ها و چندین روش دسته‌بندی برای تشخیص برنامه‌های مخرب استفاده شده است. مجموعه داده‌های جهت آموزش و ارزیابی روش پیشنهادی شامل ۳۲۶۵ فایل بدافزاری و ۱۰۰۱ فایل بی‌خطر بوده است. از الگوریتم داده‌کاوی RIPPER، برای ایجاد یک سامانه مبتنی بر نقش و برای اعمال بر روی مجموعه DLLها استفاده و از الگوریتم بیز ساده جهت اعمال بر روی داده‌های رشته‌ای و از الگوریتم n-grams برای آموزش یک دسته‌بندی بیز Multi-Naive با یک استراتژی رأی‌گیری بهره‌برداری شده است. در این روش پیشنهادی با بهره‌گیری از الگوریتم بیز ساده و با استفاده از ویژگی رشته‌ها بهترین دقت حاصل شده است.

روش ارائه شده توسط دی گائو و همکاران [۱۵] مبتنی بر تحلیل ایستا است که بر اساس خواص ساختار فایل اجرایی، بدافزارها را تشخیص می‌دهد. این روش خوشه‌بندی را توسط نسخه‌ای بهینه شده از الگوریتم KNN انجام می‌دهد. مجموعه داده برای آموزش و ارزیابی شامل ۴۱۰ فایل به تفکیک ۳۰۰ بدافزار و ۱۱۰ فایل بی‌خطر بوده است که از این تعداد ۲۸۰ نمونه برای آموزش مدل و ۱۳۰ نمونه برای آزمون استفاده است. بدافزارهای استفاده شده شامل سه خانواده بدافزاری شامل تروجان، درب پشتی و کرم و یک خانواده شامل فایل بی‌خطر است. دقت این روش به‌صورت میانگین ۸۹ درصد بوده و مدل ارائه شده تنها قادر به بررسی فایل‌های اجرایی ۳۲ بیتی هست.

در پژوهش علیرضایی [۱۶]، بر اساس نام کتابخانه‌های بارگذاری شده توسط یک برنامه، بدخواه یا بی‌خطر بودن آن بررسی می‌شود. در روش بیان شده این پژوهش، شناسایی بدافزار از روش کتابخانه‌های پیوندی پویا که در زمان اجرا توسط بدافزار بارگذاری می‌شوند، صورت می‌گیرد. در این روش هیچ تمایزی بین توابع سامانه‌ای موجود در فایل‌های کتابخانه‌ای پیوند پویا وجود نداشته و تنها از بدافزارها جهت مدل‌سازی رفتار مخرب استفاده شده و دقت شناسایی بدافزارها در روش مذکور ۷۵ درصد بوده است.

تشخیص بدافزار با استفاده از توالی API توسط سانگ و همکاران [۱۷] یکی از پژوهش‌های مهم این حوزه است. آن‌ها یک سامانه تشخیص مبتنی بر امضاء را به نام تحلیل ایستای فایل اجرایی مخرب، SAVE، ایجاد کردند که توالی API استخراج شده از برنامه را با دنباله‌ای از یک پایگاه داده امضاءها با بهره‌گیری از ضریب همبستگی پیرسان، کسینوس و ژاکارد تعمیم یافته مقایسه و نتیجه نهایی را بر مبنای میانگین این سه مورد تعیین می‌کند. با استفاده از این بررسی‌های آماری برای تعیین میزان شباهت، SAVE را قادر می‌سازد تا بدافزارهایی ناشناس و غیر قابل شناسایی به روش‌های سنتی، تشخیص داده شوند.

در پژوهش وبر و همکاران [۱۸]، ابزار تجزیه و تحلیل فایل اجرایی PEAT را برای تشخیص مبتنی بر ناهنجاری‌های ساختاری را در یک برنامه دیگر توسعه داده‌اند. PEAT بر این اصل استوار است که کد درج شده در یک برنامه یکپارچگی ساختاری آن را مختل نموده و از این طریق و با استفاده از ابزارهای آماری و مجسم سازی می‌توان میزان مخرب بودن آن را شناسایی نمود. ابزارهای مجسم سازی احتمال یافتن ویژگی‌های خاصی (ترتیبی از بایت‌ها، رشته‌ها، کد دیس اسمبل شده و دسترسی به حافظه از طریق آفست‌های رجیستری) از درج کد را در نواحی خاصی از برنامه برای کاربر مجسم می‌نماید. تحلیل ایستا بر روی تناوب دستورالعمل‌ها، الگوی دستورالعمل‌ها، آفست‌های ثبات، پرش و فراخوانی، آنتروپی مقادیر کدهای عملیاتی و کد و رشته‌ها اعمال شده و نتایج تجربی تنها برای یک برنامه مخرب ارائه گردیده است.

در پژوهش دیگری [۱۹] نویسندگان یک سامانه تشخیص جاسوس‌افزار نظارتی ارائه داده‌اند که از اطلاعات مهم توابع واسط برنامه‌نویسی و کتابخانه‌های پویا و تغییر در رجیستری، سامانه فایل و وضعیت شبکه برای تشخیص جاسوس‌افزار استفاده می‌کند. در این پژوهش از ۱۱۴۷ نمونه جمع‌آوری شده برای آموزش و ارزیابی استفاده شده و برای استخراج ویژگی‌های مؤثر از الگوریتم ماشین بردار پشتیبان بهره‌برداری شده است. نرخ تشخیص صحیح ۹۷/۹ درصد و تشخیص اشتباه ۰/۶۸ درصد بوده است.

در پـژوهش توریسـقام و همکـاران [۲۰]، از مجموعـه‌ای از ویژگـی‌هـای ترکیبـی بـا اسـتفاده از الگـوریتم n-gram، تـوالی دستورالعمل و فراخوانی API ایجاد نمودند و پژوهش خود را با دو مجموعه داده مجزا آموزش و ارزیابی نمودند. مجموعه داده اولیـه شامل ۱۴۳۵ فایل اجرایی که ۵۹۷ تا از آن‌ها خوش‌خـیم و ۸۳۸ فایل دیگر مخرب بوده و مجموعـه داده دوم شـامل ۲۴۵۲ فایـل اجرایی که شامل ۱۳۷۰ فایل خوش‌خـیم و ۱۰۸۲ فایـل بـدخیم بوده است. دقت هر یک از مجموعه ویژگی‌ها با اعمال کراس سـه حالته با استفاده از الگوریتم‌های ماشین بـردار پشـتیبان، درخـت تصمیم، بیز ساده، شبکه بیز و درخت تصمیم تقویت شـده مـورد آزمایش قرار گرفت. برای مجموعه داده اول، بهترین نتیجه بـرای HFS با استفاده از n-grams با اندازه ۶ و دقت دسته‌بنـدی ۹۷/۴ درصد گزارش شده و مجموعه دوم HFS دقت تشـخیص بهتـری نسبت به مجموعه اول داشته است.

۴- روش پیشنهادی

همان‌طور که پیش از این نیز بیان شـد، هـدف از ایـن پـژوهش، کشف و خوشه‌بندی بدافزارها با رویکرد مبتنی بر تحلیل به‌صورت ایستا است، یعنی زمانی که فایل اجرایی بـر روی حافظـه جـانبی ذخیـره شـده اسـت. در ایـن پـژوهش، ویژگـی‌هـایی از سـاختار فایل‌های اجرایی قابل حمل استخراج شده است. این ویژگی‌ها در تعیین ماهیت فایل‌های اجرایی و پیش‌بینی، نـوع رفتـار آن‌هـا در زمان اجرا بسیار مؤثر خواهد بود. در ادامه این ویژگـی‌هـا کـه بـا مکاشـفه در داده‌کـاوی از حجـم انبـوهی شـامل ۳۶۵۶۷ برنامـه مخـرب و ۱۷۲۹۵ برنامـه بـی‌خطـر کـه از مراجـع [۲۳- ۲۱] جمع‌آوری شده، توضیح داده می‌شوند.

۴-۱- ویژگـی‌هـای سـاختاری فایـل‌هـای اجرایـی قابل‌حمل

۴-۱-۱- اختلاف در اندازه خام و مجازی

این ویژگی یکی از مهم‌تـرین و پـروزن‌تـرین مـلاک‌هـا در شـیوه پیشنهادی بوده و در این ویژگی هدف بررسی اختلاف بین انـدازه فایل خام و اندازه مجازی فایل اسـت. انـدازه خـام نشـان دهنـده اندازه فایل بر روی دیسک سخت و اندازه مجـازی نشـان دهنـده اندازه فایل بر روی حافظه اصلی است که از سرآیند فایل اجرایـی قابل استخراج است. این اختلاف در صورتی که بیش از ۱ درصـد حجـم کـل فایـل باشـد بـه‌عنـوان یـک امتیـاز منفـی در شـیوه پیشنهادی مد نظر قرار می‌گیرد و بر اساس این همین روال، هر ۲ درصد اختلاف یک وزن منفی افزوده می‌شود

۴-۱-۲- خواص قسمت‌ها

در این مورد خواص قسمت‌های یک فایل بـه ویـژه قسـمت .text مورد بررسی قرار می‌گیرد. در حالت پیش‌فرض قسمت کـد فقـط خواندنی است. در شیوه پیشنهادی اگر یک فایل اجرایی در زمان تحلیل مشخص شود که قسمت .text آن خاصیت دیگری غیـر از فقط خواندنی دارد به گونـه‌ای کـه قابلیـت نوشـتن در آن وجـود داشته باشد به‌عنوان یک امتیاز منفی با وزن بالا در ارزیـابی قـرار داده می‌شود. دلیل این امر این است که بدافزارهایی کـه نیـاز بـه تغییر کد در زمان اجرا دارند، می‌بایست قادر به نوشتن مجـدد در این قسمت باشند.

۴-۱-۳- خواص سرآیند قسمت‌ها

این بخش ویژگی‌های مهمی از قسمت[1] نظیر نـام، تعـداد، آدرس، اندازه و غیره را دربر می‌گیرد [۲۴]. بر اساس نتایج ایـن پـژوهش مشخص شده بسیاری از بـدافزارها نسـبت بـه فایـل‌هـای سـالم، در ویژگـی‌هـای قسـمت‌هـا، دارای اختلافـات ملموسـی هسـتند. از این جملـه مـی‌تـوان بـه نـام قسـمت‌هـا اشـاره نمـود. تعـداد زیـادی از بـدافزارها حـاوی نـام‌هـای قسـمت نـامفهوم ماننـد .dnn4fh4۶، .Bga1m3ar، .IOu15g4I و غیره هستند در حالی که فایل‌های سالم دارای نام قسـمت‌هـای .text، .data، .rsrc و سـایر بخش‌ها و یا نام‌های معنی‌دار می‌باشند. علاوه بر ایـن، بـه‌صـورت معمول تعداد قسمت‌ها بین ۵ تا ۸ عدد است، تعداد قسـمت‌هـای خارج از این بازه شک برانگیز است. یکی دیگر از ویژگی‌ها، انـدازه قسمت است؛ در واقع منظور از اندازه قسمت، مقایسه قسمت‌های حساس کد و داده با مقدار کـل فایـل اجرایـی اسـت. فایـل‌هـای اجرایی که دارای قسمت کد با انـدازه کوچـک و قسـمت داده بـا اندازه بزرگ می‌باشند، از آن جهت که مشـابه فایـل‌هـای مخـرب بسته‌بندی شـده و نیـز خـود تغییـر هسـتند، مشـکوک بـه نظـر می‌رسند. خاصیت قابل نوشتن بودن قسـمت کـد یکـی دیگـر از ویژگی‌هایی است که در فایل‌های مخرب و فایل‌های سـالم دارای تفاوت ملموس هستند. وجود یا عدم وجود قسمت منـابع معتبـر آخرین ویژگی از این سری ویژگی‌ها است کـه مـی‌تـوان از آن در شناسایی مخرب یا غیر مخرب بودن فایل اجرایـی تصـمیم‌گیـری نمود. در فرآیند تصمیم‌گیری در خصوص وضعیت فایـل اجرایـی، به ازای وجود هر ویژگی مشـکوک امتیـاز ۱- و در صـورت عـدم وجود امتیاز ۱+ به فایل مورد نظر اختصاص داده می‌شود.

۴-۱-۴- آدرس نقطه شروع

نقطه شروع در فایل‌های اجرایی مکانی است کـه فرآینـد اجـرا از آنجـا آغـاز مـی‌گـردد. در حالـت اسـتاندارد آدرس نقطـه شـروع می‌بایست همواره به مکانی درون قسمت کد اشاره نماید [۲۵]. با

[1] Section

این وجود برخی بدافزارها با هدف محافظت از کد و دشوار نمودن فرآیند تحلیل از ابزارهای بسته‌بندی و محافظ استفاده می‌کنند که مشخصه اصلی آن‌ها ایجاد قسمت جدید و قرار دادن نقطه شروع درون آن است. علاوه بر این بدافزارهایی که کد خود را درون فایل‌های اجرایی بر روی دیسک سخت تزریق می‌کنند، جهت در دست گرفتن روال اجرا در زمان بارگذاری فایل اجرایی بر روی حافظه اصلی می‌بایست آدرس نقطه شروع را تغییر دهند که به ابتدای کدهای تزریق شده اشاره می‌کند [۲۶]. در روش پیشنهادی به فایل‌های اجرایی که دارای آدرس نقطه شروع خارج از بازه قسمت کد می‌باشند امتیاز ۱- تعلق می‌گیرد.

۴-۱-۵- نرخ بی‌نظمی

از دیگر ویژگی‌های اصلی جهت شناسایی فایل‌ها مخرب به‌کار رفته شده در روش پیشنهادی می‌توان به نرخ بی‌نظمی اشاره نمود. منظور از بی‌نظمی میزان تنوع بایت‌های تشکیل دهنده فایل اجرایی است. این تنوع برای قسمت‌های کد و داده محاسبه می‌گردد [۲۷]. میزان تنوع بایت‌های تشکیل دهنده فایل اجرایی ساکن در قسمت کد، بسته به معماری پردازنده که می‌تواند ۳۲ و یا ۶۴ بیتی باشد در یک بازه مشخص قرار می‌گیرد. نرخ بی‌نظمی بیشتر از این بازه نشان دهنده بسته‌بندی و یا رمزنگاری شدن فایل اجرایی است [۲۸]. در روش پیشنهادی آستانه نرخ بی‌نظمی 7.3 در نظر گرفته می‌شود و به فایل‌هایی که دارای نرخ بی‌نظمی بیشتر از مقدار آستانه باشند، امتیاز ۱- تعلق می‌گیرد.

۴-۱-۶- کتابخانه‌ها و توابع

از آن‌جهت که هر کتابخانه و توابع مربوط به آن‌ها با هدف ارائه تسهیلات خاصی توسعه یافته‌اند [۲۹ و ۳۰]، این دسته از مؤلفه‌ها نقش مهمی در پیش‌بینی رفتار فایل اجرایی ایفاء می‌کنند که شامل لیست کتابخانه‌های استفاده شده، لیست توابع فراخوانی شده از کتابخانه‌ها و تعداد فراخوانی‌ها از هر کتابخانه توسط فایل اجرایی است. کتابخانه‌ها و توابع مورد استفاده در فایل اجرایی در مکانی به نام جدول توابع ورودی قرار می‌گیرند. پس از استخراج موارد بیان شده از جدول توابع ورودی، بر اساس لیست تهیه شده از توابع خطرناک سیستم عامل (توابعی که در حالت معمول کمتری مورد استفاده قرار می‌گیرند) [۳۱]، با داده‌کاوی بر روی حجم انبوهی از بدافزارها و فایل‌های بی‌خطر، قوانینی جهت تصمیم‌گیری در مورد بدخیم و یا خوش‌خیم بودن فایل اجرایی استخراج می‌شود و این قوانین در دسته‌بندی بدافزارهای کاربردهای متنوع و فراوانی دارد.

۴-۱-۷- آیکون فایل اجرایی

آیکون‌ها می‌توانند در ترغیب کاربران برای اجرای فایل‌های اجرایی، مؤثر باشند؛ از همین روی اغلب تحت عنوان روش‌های مهندسی اجتماعی توسط توسعه دهندگان بدافزار مورد استفاده قرار می‌گیرد [۳۲ و ۳۳]. در طول این پژوهش از مجموع ۳۶۵۶۷ فایل مخرب، ۸۷۳ آیکون و از مجموع ۱۷۲۹۵ فایل سالم، ۱۴۲۹ آیکون استخراج گردید. در شکل (۲) برخی از آیکون‌های جعلی استفاده شده توسط بدافزارها جهت ترغیب و فریب کاربران برای قانونی نشان دادن بدافزار، ارائه شده است؛ استخراج آیکون‌ها به‌صورت برنامه‌نویسی از قسمت rsrc. صورت می‌پذیرد.

شکل (۲): آیکون‌های جعلی استفاده شده در بدافزارها [۳۳]

بنابراین فایل‌های اجرایی که دارای آیکون وابسته به سایر فرمت‌های شناخته شده و پرکاربرد می‌باشند مشکوک تشخیص داده شده و به آن‌ها امتیاز ۲- تعلق می‌گیرد.

۴-۱-۸- اشیاء گرافیکی

وجود یا عدم وجود پنجره‌ها، دکمه‌ها، منوها و سایر عناصر بصری معتبر تشکیل دهنده واسط کاربری در فایل‌های اجرایی می‌تواند به‌عنوان ویژگی دیگری در شناسایی و تشخیص بدافزارها مورد استفاده قرار بگیرد. در این ویژگی، تعیین اعتبار عناصر بصری از اهمیت ویژه‌ای برخوردار است. اعتبار عناصر بصری از طریق بررسی قابل دسترس و مشاهده بودن عناصر تعیین می‌گردد و به فایل‌های اجرایی با عناصر بصری غیر معتبر امتیاز ۳- و در مقابل عناصر بصری معتبر ۳+ تعلق می‌گیرد.

۴-۱-۹- دنباله بایت‌های قابل چاپ

رشته‌های موجود در فایل‌های اجرایی شامل نام کتابخانه‌ها و توابع، امضای نویسنده، نام فایل‌ها، اطلاعات منابع سامانه، برخی تکه کدها و سایر مواردی از این دست است که اغلب بین فایل‌های بدافزار و بی‌خطر مشترک است [۳۴]. علاوه بر این می‌توان به پیام‌ها اشاره نمود که با هدف تعامل با کاربر در فایل‌های اجرایی بی‌خطر مورد استفاده قرار می‌گیرد. در صورت وجود پیام‌های معتبر و قابل دسترس در فایل اجرایی، امتیاز ۱+ به فایل اجرایی تعلق می‌گیرد.

۴-۱-۱۰- دستورالعمل‌های گریز

دستورالعمل‌های گریز، دستورالعمل‌هایی هستند که توسط بدافزارها با هدف شناسایی محیط اجرا مورد استفاده قرار می‌گیرد. از جمله محیط‌های اجرایی که بدافزارها سعی در شناسایی و فرار از آن‌ها دارند می‌توان به محیط‌های جعبه شن [۳۵]، اشکال‌زداها و مقلدهای محصولات ضد بدافزار اشاره نمود [۱۲ و ۳۶]. با بررسی دستورالعمل‌های گریز نیز می‌توانند به پیش‌بینی قابل قبولی از فایل اجرایی مورد بررسی دست یافت. در

روش پیشنهادی برخی از دستورالعمل‌های حساس از جمله دستورالعمل‌های شناسایی ماشین مجازی، دستورالعمل‌های معادل به‌کار رفته جهت اعمال روش‌های دگردیسی، دستورالعمل‌های مقابله با اشکال‌زداها و سایر مواردی از این دست شناسایی‌شده [۳۷] و به فایل‌هایی که از این دسته دستورالعمل‌های استفاده می‌کنند امتیاز ۱- تعلق می‌گیرد.

۴-۱-۱۱- جدول آدرس توابع خروجی

جدول آدرس توابع خروجی در قسمت edata. قرار دارد. این جدول نام و آدرس توابع خروجی توسط برنامه را نشان می‌دهد. این توابع، توسط برنامه دیگری که فایل اجرایی جاری را Import کرده، مورد استفاده قرار می‌گیرد. بسیار غیر معمول است که بدافزار تابعی را برای استفاده دیگر برنامه‌ها ارائه دهد و در مقابل اغلب برنامه‌های بی‌خطر به ویژه کتابخانه‌های پیوند پویا در مجموعه‌های نرم‌افزاری ارائه دهنده این توابع هستند؛ بنابراین در شیوه پیشنهادی وجود جدول EAT معتبر یک امتیاز مثبت برای بی‌خطر بودن فایل در حال تحلیل است.

۴-۲- تشخیص و خوشه‌بندی

در ابن بخش با بهره‌گیری از روش‌های داده‌کاوی بر روی حجم داده‌های اولیه که شامل فایل‌های اجرایی بدافزار و فایل‌های بی‌خطر است، آن‌ها را با توجه به میزان شباهت و نزدیکی خواص ذکر شده در بخش‌های پیشین خوشه‌بندی نموده تا در زمان برخورد با فایل‌های ناشناس ضمن تشخیص خوش‌خیم و یا بدخیم بودن فایل اجرایی، آن‌ها را در یکی از دسته‌های موجود قرار داده شود. در فاز تشخیص می‌بایست میزان خوش‌خیم و یا بدخیم بودن فایل اجرایی محاسبه گردد؛ محاسبه میزان خوش‌خیم و یا بدخیم بودن فایل اجرایی، بر اساس داده‌کاوی از طریق رابطه (۱) صورت می‌گردد:

$$\text{نرخ بدخیمی} = \frac{\sum W \times Pi + \sum W \times Nj}{\sum Wi + Wj} \qquad (۱)$$

در رابطه (۱)، مقدار $\sum W \times Pi$ حاصل جمع امتیازهای مثبت داده شده به فایل اجرایی مورد بررسی با در نظر گرفتن وزن آن‌ها بوده و مقدار $\sum W \times Nj$ حاصل جمع امتیازهای منفی داده شده به همراه وزن آن‌ها است. مقدار $\sum Wi + Wj$ نشان دهنده مجموع امتیازهای مثبت و منفی قابل اعمال به یک فایل در زمان بررسی با در نظر گرفتن وزن آن‌ها هست؛ اگر حاصل رابطه بالا یک عدد مثبت باشد، فایل مورد نظر به‌عنوان یک فایل بی‌خطر و در صورتی که حاصل رابطه منفی باشد به‌عنوان بدافزار تشخیص داده می‌شود. در فرآیند آموزش و ارزیابی روش پیشنهادی از ابزار داده‌کاوی Weka در حالت یادگیری نیمه مدیریت شده و با بهره‌گیری از الگوریتم‌های موجود در آن، استفاده شده است. در فرآیند داده‌کاوی از الگوریتم‌های KStar و Jrip برای تولید قانون و از الگوریتم J48 برای ایجاد درخت تصمیم جهت تعبیه در نرم‌افزارهای تشخیص و شناسایی استفاده شده است. در جدول (۱) یک نمونه قانون برای تشخیص بدافزار جاسوس‌افزار ارائه شده است.

جدول (۱): یک نمونه قانون برای تشخیص بدافزار

```
IF (ExecutableSectionNAme!="text") &&
(NumberOfSection>8) ||
(NumberOFUnknownNamedSection > 0) &&
(DifOfSectionSizeAndFileSize < 0.000040) &&
(SubOfVirtualAndRawSizeOfCodeSection < 000070) &&
(NumberOfPackingInstruction > 1 && PortOpening =
True && KeyboardAPI>3 || ScreenAPI>4)) Then
MalwareType = Malware.Spayware
```

قانون ارائه شده در جدول (۱) با استفاده ویژگی‌های سرآیند قسمت، تعداد قسمت‌های ناشناخته، قسمت اجرایی، اختلاف بین اندازه مجازی، اندازه خام و واسط‌های برنامه‌نویسی کاربردی، میزان مخرب بودن فایل بررسی و در صورت مثبت بودن نتیجه نوع بدافزار را مشخص می‌نماید. روش پیشنهادی قادر است تمامی بدافزارها را در ۷ گروه ویروس، کرم، تروجان، باتنت، جاسوس‌افزار، روت کیت و نرم‌افزار ناخواسته خوشه‌بندی کند. گروه‌های بیان‌شده با الهام از منبع [۳۸] انتخاب شده‌اند. این گروه‌بندی قادر به پوشش انواع کدهای بدخواه و بدافزارها است.

۵- ارزیابی روش پیشنهادی

در این بخش میزان دقت روش پیشنهادی برای کشف و خوشه‌بندی بدافزارها بررسی می‌گردد. جزئیات داده‌های مورد استفاده در این پژوهش که شامل ۳۶۵۶۷ فایل مخرب و ۱۷۲۹۵ فایل سالم در جدول (۲) نشان داده شده است.

با هدف ارزیابی روش پیشنهادی، کار انجام شده با چهار مورد از کارهای مرتبط که اخیراً منتشر شده‌اند مقایسه گردیده است. جهت دست‌یابی به این هدف روش‌های پیاده‌سازی شده در کار شولتز و همکاران [۱۴]، کالتر و همکاران [۳۹]، سیدیکیو و همکاران [۴۰] و گائو و همکاران [۱۵] بر اساس مستندات موجود پیاده‌سازی گردید و خروجی‌های به‌دست آمده به همراه خروجی‌های روش پیشنهاد شده در ادامه ارائه گردیده است. اعتبارسنجی نتایج آزمایش با استفاده از شیوه متقاطع ۱۰ صورت گرفته است. در این شیوه مجموعه داده به‌صورت تصادفی به ۱۰ زیرمجموعه کوچک‌تر تقسیم می‌گردد که در آن ۹ زیرمجموعه

برای آموزش استفاده می‌شود و زیرمجموعه دهم برای آزمون مورد بهره‌برداری قرار خواهد گرفت. روند بیان شده برای هر ترکیب ده‌تایی به میزان ۱۰ بار تکرار گردیده است. علاوه بر این دو سامانه مبتنی بر معماری ۳۲ و ۶۴ بیتی سازوکار بیان شده را به‌صورت مجزا اجرا می‌کنند. سامانه‌های ۳۲ و ۶۴ بیتی دارای پیکربندی سخت‌افزاری مجازی سازی شده یکسان می‌باشند. از جمله این پیکربندی سخت‌افزاری می‌توان به ۶ هسته پردازنده و ۴ گیگابایت حافظه اصلی اشاره نمود. فارغ از معماری، هر دو سامانه سیستم‌عامل ویندوز ۷ نسخه SP3 را اجرا می‌کنند.

جدول (۲): جزئیات مجموعه داده مورد استفاده

عنوان	ویروس	کرم	تروجان	بات‌نت	جاسوس‌افزار	روت کیت	نرم‌افزار ناخواسته	نرم‌افزار سالم
تعداد	۳۰۵۴	۹۴۷۳	۵۶۲۹	۳۹۶۷	۷۷۰۸	۳۹۶۱	۲۷۷۵	۱۷۲۹۵
کتابخانه	۰	۱۰۸۳	۷۶۲	۱۴۲۸	۲۶۹۴	۲۷۳۵	۳۵۷	۴۳۰۸
غیر کتابخانه	۳۰۵۴	۸۳۹۰	۴۸۶۷	۲۵۳۹	۵۰۱۴	۱۲۲۶	۲۴۱۸	۱۲۹۸۷
معماری ۶۴ بیتی	۲۲۳	۴۸۳۱	۲۱۶۶	۱۵۳۳	۲۶۸۷	۹	۱۱۴۴	۳۴۶۶
معماری ۳۲ بیتی	۲۸۳۱	۴۶۴۲	۳۴۶۳	۲۴۳۴	۵۰۲۱	۳۹۵۲	۱۶۳۱	۱۳۸۲۹
UPX	۴۱۶	۱۳۶۸	۵۲۰	۴۸۳	۸۹۴	۱۷۲	۲۹۱	۲۵۶۱
ASPack	۳۰۴	۸۱۰	۳۷۴	۲۱۲	۳۱۷	۴۳	۱۱۶	۷
Themida	۴۷	۹۶	۱۸۶	۹۲	۲۰۳	۲۷	۸۷	۳
Other Packer	۲۶۶	۸۹۳	۴۶۱	۲۴۷	۳۶۸	۷۲۸	۳۰۱	۱۲
Borland C/C++	۲۸۴	۱۰۳	۶۹۴	۳۸۸	۵۷۳	۲۲۴	۲۱۴	۴۳۸۱
Borland Delphi	۱۳۳	۳۸۴	۲۵۸	۲۶۱	۴۲۵	۱۳	۲۴	۸۶۵
Visual Basic	۱۹۷	۳۵۲	۱۳۶	۳۱۶	۱۴۶	۱۷	۱۸۳	۷۳۹
Visual c++	۹۱۹	۳۶۴۸	۲۰۱۵	۵۸۷	۲۹۷۵	۲۱۳۶	۷۶۹	۴۵۰۷
.Net	۸۴	۱۳۵۵	۷۲۲	۷۸۳	۸۶۳	۵۱۰	۵۳۶	۳۶۴۷
Other	۶۳۸	۴۶۴	۲۶۳	۵۹۸	۹۹۴	۹۱	۲۵۴	۵۷۳

در مرحله نخست هدف یافتن میزان دقت روش پیشنهادی و چهار مقاله بیان شده در تشخیص فایل‌های خوش‌خیم و بدخیم است. بررسی میزان تفکیک فایل‌های بدخیم و خوش‌خیم از طریق معیارهای مثبت واقعی (TP)، منفی واقعی (FP)، مثبت کاذب (TN) و منفی کاذب (FN) صورت می‌پذیرد. معیار مثبت واقعی عبارت است تشخیص‌هایی که به درستی فایل ورودی را بدافزار تشخیص داده است. به‌صورت مشابه، منفی واقعی، شامل تشخیص‌هایی است که فایل ورودی را به درستی فایل غیر بدافزار تشخیص داده است. منفی کاذب به مواردی گفته می‌شود که فایل ورودی به اشتباه به‌عنوان بدافزار تشخیص داده نشده است و در مثبت کاذب فایل ورودی به اشتباه بدافزار تشخیص داده می‌شود. در جدول (۳) معیارهای تشخیصی بیان شده برای هر یک از الگوریتم‌های بیان شده در روش‌های چهارگانه و روش پیشنهادی نشان داده شده است. مقادیر برجسته بهترین گزینه از بین مقادیر موجود از ۵ روش شناسایی است. نتایج به‌دست آمده حاصل ۸۵۰ بار تکرار به شیوه متقاطع ۱۰ است. جدول (۳) نشان می‌دهد که روش پیشنهادی دارای بیشترین نرخ تفکیک صحیح و نیز کمترین میزان تفکیک غلط در بین روش‌های بیان شده است. این عملکرد نتیجه انتخاب ویژگی‌های مؤثرتر و انتخاب و بهره‌برداری بهتر از الگوریتم‌های داده‌کاوی است.

جهت بررسی میزان دقت در خوشه‌بندی بدافزار بر اساس معیارهای چهارگانه بالا رابطه (۲) مورد استفاده قرار می‌گیرد.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \qquad (۲)$$

جدول (۳): وضعیت تشخیص بدافزار از فایل‌ها سالم

ردیف	روش	الگوریتم	TP	FN	FP	TN
۱	روش پیشنهادی	*KStar*	۹۹/۲	۰/۲	۸۱/۱	**۰/۱**
		Jrip	۹۹/۶	۰/۰۰۴	۹۸/۹	۰/۳
		J48	۹۵/۱	۰/۰۰۵	۹۹/۸	۰/۲
۲	شولتز و همکاران	*RIPER*	۹۴/۱	۰/۸	۹۳/۵	۰/۴
		Naïve Bayes	۸۵/۴	۱/۱	۹۲/۳	۰/۸
		Multy NB	۸۹/۶	۰/۹	۹۴/۸	۰/۴
۳	کالتر و همکاران	*Naive Bayes*	۹۷/۱	۰/۴	۹۵/۶	۰/۵
		J48	۸۹/۰	۱/۲	۹۴/۲	۰/۷
		TFIDF	۹۳/۸	۲/۰	۹۸/۰	۰/۳
۴	سیدیکیو و همکاران بی‌خطر	*Decision Tree*	۹۷/۴	۰/۴	۹۷/۵	۰/۴
		Bagging	۹۶/۷	۱/۳	۹۸/۳	۱/۲
		Random Forest	۹۴/۷	۱/۱	۹۷/۳	۰/۵
۵	گائو و همکاران	*K means*	۹۸/۳	۰/۲	۹۴/۱	۰/۷
		KNN	۹۵/۵	۰/۶	۸۹/۹	۱/۹

بر اساس [۳۶] میزان شباهت در ساختار فایل‌های اجرایی بدافزار در بازه [۹۳-۱۴] درصد قرار دارد. این میزان تشابه در حالت معمول ۳۵ درصد است. در نتیجه انتخاب ویژگی‌های جامع که دربردارنده تمامی حالات و مؤلفه‌های خوشه‌ها باشد امری بسیار حائز اهمیت است که می‌تواند در نرخ دقت روش‌های پیشنهادی تأثیر مستقیم داشته باشد.

در جدول (۴) میزان دقت خوشه‌بندی روش پیشنهادی و چهار شیوه دیگر ارائه شده است. مقادیر برجسته بهترین میزان خوشه‌بندی را در بین پنج روش مورد ارزیابی نشان می‌دهد. همان‌طور که مشخص است روش پیشنهادی می‌تواند با دقت بالاتری نسبت به سایر روش‌ها، بدافزارها را خوشه‌بندی نماید. نتایج به‌دست آمده حاصل ۷۵۰ بار تکرار به شیوه متقاطع ۱۰ است. همان‌طور که در جدول (۴) نشان داده شده است، شناسایی روت کیت‌ها در مقایسه با سایر خوشه‌ها چالش برانگیزتر است. دلیل این امر ایجاد تغییرات گسترده و عمیق این خوشه بر روی سیستم‌عامل است که موجب می‌گردد آن‌ها از دید بسیاری از نرم‌افزارهای امنیتی و نظارتی مخفی بمانند.

از دیگر معیارهای کارایی می‌توان به منابع مورد نیاز جهت اجرا اشاره نمود. از جمله منابع حائز اهمیت سربار حافظه اصلی و زمان اجرای فایل‌های ورودی است که تأثیر مستقیمی بر روی قابلیت اطمینان و رضایت کاربر نیز دارد. واحد ارزیابی زمان ثانیه / فایل و واحد ارزیابی سربار حافظه اصلی هزار بایت / فایل است.

جدول (۴): میزان دقت خوشه‌بندی درروش پیشنهادی و سایر روش‌ها

ردیف	خوشه	روش پیشنهادی	شولتز	کالتر	سیدیکیو	گائو
۱	ویروس	**۹۹/۱**	۹۶/۹	۹۵/۹	۹۵/۸	۹۶/۲
۲	کرم	**۹۹/۶**	۹۷/۱	۹۷/۵	۹۷/۲	۹۸/۱
۳	تروجان	**۹۹/۳**	۹۵/۳	۹۸/۳	۹۸/۱	۹۷/۳
۴	باتنت	**۹۹/۲**	۹۴/۷	۹۷/۹	۹۷/۹	۹۷/۳
۵	جاسوس‌افزار	**۹۸/۴**	۹۷/۳	۹۵/۶	۹۴/۷	۹۳/۲
۶	روت کیت	**۹۹/۸**	۹۲/۶	۹۳/۳	۹۱/۲	۹۲/۸
۷	نرم‌افزار ناخواسته	**۹۸/۸**	۹۷/۹	۹۷/۲	۹۷/۱	۹۶/۹
۸	بی‌خطر	**۹۹/۷**	۹۸/۴	۹۷/۸	۹۷/۶	۹۸/۳

در جدول (۵)، منابع مورد نیاز روش پیشنهادی به همراه ۴ شیوه دیگر نشان داده شده است. مقادیر برجسته نشان دهنده مقادیر بهینه نسبت به بقیه روش‌ها است. نتایج به‌دست آمده حاصل ۹۵۰ بار تکرار الگوریتم‌های مربوطه بر روی فایل‌های مختلف است. فایل‌های ورودی برای تمامی الگوریتم‌ها یکسان بوده است.

جدول (۵): مقایسه منابع مورد نیاز

ردیف	عنوان روش	زمان پویش	زمان استخراج ویژگی	زمان خوشه‌بندی	حافظه اصلی
۱	روش پیشنهادی	۰/۱۳۰	۰/۷۰	۰/۴۴	۱۴۰
۲	شولتز و همکاران	۰/۲۰۱	۰/۹۴	۰/۷۵	۲۴۷
۳	کالتر و همکاران	۰/۱۹۷	۰/۸۵	۰/۵۲	۲۰۳
۴	سیدیکیو و همکاران	۰/۲۶۴	۰/۷۶	۰/۸۴	۳۱۲
۵	گائو و همکاران	۰/۱۷۸	۰/۹۹	۰/۶۳	۱۹۵

با توجه به بررسی‌های صورت گرفته بر روی معیارهای سنجش و مقایسه روش پیشنهادی با سایر کارهای مرتبط که اخیراً صورت گرفته است، برتری روش پیشنهادی در خصوص تفکیک فایل‌های سالم از مخرب و نیز خوشه‌بندی بدافزارها نسبت به سایر روش‌ها روشن گردید. این برتری در نتیجه انتخاب ویژگی‌های کارآمد و بهره‌گیری از الگوریتم‌های داده‌کاوی مؤثر حاصل شده است.

۶- نتیجه‌گیری

در این مقاله، نشان داده شد که چگونه با بررسی دقیق ساختار فایل اجرایی و بدون نیاز به اجرای فایل مشکوک، می‌توان ویژگی‌هایی از آن فایل استخراج کرد و بر اساس ویژگی‌های استخراج شده در شناسایی و خوشه‌بندی بدافزارهای ناشناخته استفاده نمود. از جمله این ویژگی‌ها می‌توان به اطلاعات مربوط به قسمت‌ها، آدرس نقطه شروع، نرخ بی‌نظمی، لیست کتابخانه‌ها و توابع، رشته‌ها، آیکون‌ها و سایر موارد مرتبط اشاره کرد. در روش پیشنهادی این مقاله، با استفاده از روش‌های یادگیری ماشین مدل رفتاری برای سنجش میزان خوش‌خیم و یا بدخیم بودن فایل اجرایی با انتصاب نمره مثبت و منفی بر اساس فرمول‌های تعریف شده ایجاد نماید. با بهره‌گیری از قوانین تولید شده مبتنی بر داده‌کاوی از حجم انبوهی از بدافزارها و فایل‌های بی‌خطر، میزان مخرب بودن یا سالم بودن فایل‌های مشکوک تعیین و بدافزارهای شناخته شده در ۷ خوشه بدافزاری قرار می‌گیرند. در انتها، میزان کارایی روش پیشنهادی، بر اساس میزان دقت در تشخیص و خوشه‌بندی بدافزارها و فایل‌های بی‌خطر ارزیابی و با روش‌های مرتبط مقایسه شده و نتایج نشان می‌دهد که دقت روش پیشنهادی بیش از ۹۸ درصد بوده است. از جمله عوامل دخیل در بهبود دقت نسبت به سایر روش‌های مورد مقایسه می‌توان به انتخاب ویژگی‌های کارا و مؤثر در بین مجموعه ویژگی‌های ممکن، انتخاب الگوریتم‌های داده‌کاوی سازگار با نوع مسئله و مجموعه داده جامع جهت بررسی و اصلاح نتایج اشاره نمود. توسعه روش پیشنهادی می‌تواند در قالب کاری‌های آتی جهت شناسایی و مقابله با گونه‌های جدیدتر بدافزارها همانند بدافزارهای بدون فایل مورد بررسی و مطالعه قرار گیرد.

۷- مراجع

[1] "Malware Statistics & Trends Report| AV-TEST," https://www.av-test.org/en/statistics/malware/ (Accessed Nov. 25, 2021.

[2] A. Afshar, A. Termechi, A. Golshan, A. Aghayan, H. R. Shahriari, and S. Soleimani, "Review of the Types of Strategies to Improve Security of Industrial Control Systems and Critical Infrastructure," Passiv. Def. Q., vol. 9, no. 2, pp. 1–9, 2018.

[3] K. Kaushal, P. Swadas, and N. Prajapati, "Metamorphic Malware Detection Using Statistical Analysis," Int. J. Soft Comput. Eng., vol. 2, no. 3, pp. 49–53, 2012.

[4] V. P. Nair, H. Jain, Y. K. Golecha, M. S. Gaur, and V. Laxmi, "Medusa: Metamorphic Malware Dynamic Analysis Usingsignature from API," in Proc. of the 3rd Int. Conf. on Security of Information and Networks, pp. 263–269, 2010.

[5] C. S. Veerappan, P. L. K. Keong, Z. Tang, and F. Tan, "Taxonomy on Malware Evasion Countermeasures Techniques," In IEEE World Forum on Internet of Things, WF-IoT-Proceedings, pp. 558–563, May 2018.

[6] J. Saxe and K. Berlin, "Deep Neural Network Based Malware Detection Using Two Dimensional Binary Program Features," In Malicious and Unwanted Software (MALWARE), 10th Int. Conf. on, pp. 11–20, 2015.

[7] Y. Ye, D. Wang, T. Li, D. Ye, and Q. Jiang, "An Intelligent PE-Malware Detection System Based on Association Mining," J. Comput. Virol., vol. 4, no. 4, pp. 323–334, 2008.

[8] "PE Format - Win32 APPS | Microsoft Docs." https://docs.microsoft.com/en-us/windows/win32/debug/pe-format (accessed Nov. 25, 2021).

[9] D. Gibert, C. Mateu, and J. Planes, "The Rise of Machine Learning for Detection and Classification of Malware: Research Developments, Trends and Challenges," J. Netw. Comput. Appl., Mar. 2020.

[10] M. Belaoued and S. Mazouzi, "A Real-Time Pe-Malware Detection System Based on Chi-Square Test and Pe-File Features," In IFIP Int. Conf. on Comput. Sci. and its App., pp. 416–425, 2015.

[11] C.-H. Lin, H.-K. Pao, and J.-W. Liao, "Efficient Dynamic Malware Analysis Using Virtual Time Control Mechanics," Comput. Secur., vol. 73, no.? pp. 359–373, 2018.

[12] A. Afianian, S. Niksefat, B. Sadeghiyan, and D. Baptiste, "Malware Dynamic Analysis Evasion Techniques: A Survey," CoRR, vol. abs/1811.0, 2018.

[13] J. L. C. Candás, V. Peláez, G. López, M. Á. Fernández, E. Alvarez, and G. Díaz, "An Automatic Data Mining Method to Detect Abnormal Human Behaviour Using Physical Activity Measurements," Pervasive Mob. Comput., vol. 15, pp. 228–241, 2014.

[14] M. G. Schultz, E. Eskin, F. Zadok, and S. J. Stolfo, “Data Mining Methods for Detection of New Malicious Executables,” In Proc. 2001 IEEE Symp. on Security and Privacy, S&P 2001, pp. 38–49, 2000.

[15] D. Gao, G. Yin, Y. Dong, and L. Kou, “A Research on the Heuristic Signature Virus Detection Based on the PE Structure,” 2013.

[16] E. Alirezaei, “Behavioral Analysis of Malicious Code,” Kish Paradise Univ. of Tehran, Kish, 2011.

[17] A. H. Sung, J. Xu, P. Chavez, and S. Mukkamala, “Static Analyzer of Vicious Executables (Save),” In 20th Annual Comput. Security App. Conf., pp. 326–334, 2004.

[18] M. Weber, M. Schmid, M. Schatz, and D. Geyer, “A Toolkit for Detecting and Analyzing Malicious Software,” In 18th Annual Computer Security App. Conf., 2002. Proc., pp. 423–431, 2002.

[19] T.-Y. Wang, S.-J. Horng, M.-Y. Su, C.-H. Wu, P.-C. Wang, and W.-Z. Su, “A Surveillance Spyware Detection System Based on Data Mining Methods,” In 2006 IEEE Int. Conf. on Evolutionary Computation, pp. 3236–3241, 2006.

[20] [M. M. Masud, L. Khan, and B. Thuraisingham, “A Scalable Multi-Level Feature Extraction Technique to Detect Malicious Executables,” Inf. Syst. Front., vol. 10, no. 1, pp. 33–45, 2008.

[21] “Inc, V. Malware Sample.” https://virusshare.com/ (Accessed Nov. 25, 2019).

[22] “VirusSign | Malware Research & Data Center, Threat Intelligence, Free Downloads.” https://www.virussign.com/ (Accessed Nov. 25, 2021).

[23] “GitHub - ocatak/malware_api_class: Malware Dataset for Security Researchers, Data Scientists. Public Malware Dataset Generated by Cuckoo Sandbox Based on Windows OS API Calls Analysis for Cyber Security Researchers.” https://github.com/ocatak/malware_api_class (Accessed Nov. 25, 2021).

[24] H. S. Anderson and P. Roth, “Ember: An Open Dataset for Training Static Pe Malware Machine Learning Models,” arXiv Prepr. arXiv1804.04637, 2018.

[25] T. Dube, R. Raines, G. Peterson, K. Bauer, M. Grimaila, and S. Rogers, “Malware Target Recognition via Static Heuristics,” Comput. Secur., vol. 31, no. 1, pp. 137–147, 2012.

[26] J. Demme et al., “On the Feasibility of Online Malware Detection with Performance Counters,” ACM SIGARCH Comput. Archit. News, vol. 41, no. 3, pp. 559–570, 2013.

[27] K. S. Han, J. H. Lim, B. Kang, and E. G. Im, “Malware Analysis Using Visualized Images and Entropy Graphs,” Int. J. Inf. Secur., vol. 14, no. 1, pp. 1–14, 2015.

[28] D. Baysa, R. M. Low, and M. Stamp, “Structural Entropy and Metamorphic Malware,” J. Comput. Virol. hacking Tech., vol. 9, no. 4, pp. 179–192, 2013.

[29] C. Ravi and R. Manoharan, “Malware Detection Using Windows Api Sequence and Machine Learning,” Int. J. Comput. App., vol. 43, no. 17, pp. 12–16, 2012.

[30] G. G. Sundarkumar, V. Ravi, I. Nwogu, and V. Govindaraju, “Malware Detection via API Calls, Topic Models and Machine Learning,” In IEEE Int. Conf. on Automation Sci. and Eng., vol. 2015-Octob, pp. 1212–1217, 2015.

[31] W. Fu, J. Pang, R. Zhao, Y. Zhang, and B. Wei, “Static Detection of Api-Calling Behavior from Malicious Binary Executables,” In 2008 Int. Conf. on Comput. and Elect. Eng., pp. 388–392, 2008.

[32] S. Abraham and I. Chengalur-Smith, “An Overview of Social Engineering Malware: Trends, Tactics, and Implications,” Tech. Soc., vol. 32, no. 3, pp. 183–196, 2010.

[33] J.-S. Kim, W. Jung, S. Kim, S. Lee, and E. T. Kim, “Evaluation of Image Similarity Algorithms for Malware Fake-Icon Detection,” In 2020 Int. Conf. on Information and Communication Tech. Convergence (ICTC), pp. 1638–1640, 2020.

[34] L. Chen, T. Li, M. Abdulhayoglu, and Y. Ye, “Intelligent Malware Detection Based on File Relation Graphs,” In Proc. of the 2015 IEEE 9th Int. Conf. on Semantic Computing (IEEE ICSC 2015), pp. 85–92, 2015.

[35] S. Parsa and F. Jamshidinia, “An Approach to Rootkit Detection Based on Virtual Machine Introspection,” Passiv. Def. Q., vol. 10, no. 2, pp. 33–42, 2019.

[36] B. Lau and V. Svajcer, “Measuring Virtual Machine Detection in Malware Using DSD Tracer,” J. Comput. Virol., vol. 6, no. 3, pp. 181–195, 2010.

[37] Y. Huang, U. Verma, C. Fralick, G. Infantec-Lopez, B. Kumar, and C. Woodward, “Malware Evasion Attack and Defense,” pp. 34–38, 2019,

[38] A. R. A. Grégio, V. M. Afonso, D. S. F. Filho, P. L. de Geus, and M. Jino, “Toward a Taxonomy of Malware Behaviors,” Comput. J., vol. 58, no. 10, pp. 2758–2777, 2015.

[39] J. Z. Kolter and M. A. Maloof, “Learning to detect Malicious Executables in the Wild,” in KDD-2004 - Proc. of the Tenth ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining, pp. 470–478, 2004.

[40] M. Siddiqui, M. C. Wang, and J. Lee, “Detecting Internet Worms Using Data Mining Techniques,” J. Syst. Cybern. Informatics, vol. 6, no. 6, pp. 48–53, 2009.

Static Analysis of the Executable File Structure to Detect and Cluster Unknown Malware

M. Abbasi*, H. Tanha

Abstract

One of the most popular ways to detect malware is to find a match for malware file signature pattern in the malware signature database. The malware signature database is pre-extracted and is constantly updated. Checking the similarity of input data using the stored signatures causes storage problems and increases the calculation costs. In addition, the detection based on adapting the malware signature pattern fails when changing the malware code in polymorphic malware. In this paper, by combining the static analysis of executable file structure and the machine learning algorithms, an effective method for malware detection is presented. The data set for training and evaluation of the proposed method includes 36,567 samples of malware and 17295 benign files, and the malware is clustered in 7 families. The results show that the presented method is able to detect and cluster malware from benign files with an accuracy of more than 99% and a false positive rate less than 0.4%. The proposed method has very low processing overheads compared to similar methods and the average scanning time of executable files is 0.244 second.

Key Words: *Malware Detection, Executable File Structure, Static Analysis, Clustering, Machine Learning*

* Researcher Imam Hossein University, Tehran, Iran (moabbasi@ihu.ac.ir) -Writer-in-Charge