Scientific Journal of Electronical & Cyber Defence
Vol. 10, No. 2, Summer 2022, Serial No. 38

Kashef: A Two-step detector of Windows-based Malicious executable files

E. Shaghaghi*, R. Jalayi, M. A. Javadzadeh

*Master's student, Imam Hossein University (AS), Tehran, Iran

(Received: 07/09/2021, Accepted: 30/11/2021)

ABSTRACT

The growing number of malware is one of the major threats in the field of cyber and malware detection has always been associated with challenges. Windows-based malicious executable files perform malicious activities at the target operating system level or any other application by manipulating features in their header and obscuring their behavior. Detecting suspicious specimens from a large volume of input samples as well as discovering new and unknown malware is one of the researchers' favorite topics. In this study, a combined method has been proposed to determine the level of maliciousness of suspicious executable files. Kashef's proposed method consists of two static modules for extracting executable file header properties, and two behavioral modules for extracting signature-generating properties and a thoughtful behavioral model based on machine learning methods. The purpose of this study is to identify suspicious Windows executable files from a large volume of files and determine their maliciousness level. This method detects malware based on the maliciousness probability assigned to each file. Experiments have been done to determine the malignancy percentage of six malware by four types of detectors. The results show the malignancy percentage for the PE header detector module, to be in the range of 62.7 to 70% and for the Yara-based detector module, to be in the range of 70.8 to 78.2%, whilst for the behavioral signature-based detector module, the malignancy percentage is 98% and for the ML-based detector module using the random forest learning algorithm it is equal to 99%. The experimental results also show that Kashef detected 94% of protected malware with a 2% improvement compared to the achievements of 10 similar rival products, and it detected 98% of unprotected malware, demonstrating a 5% improvement compared to counterpart results of 10 similar products.

Keywords: Malware, Executable file, Malware detection, Behavioral signature, Random forest algorithm.

* Corresponding Author Email: Kpshaghaghi@ihu.ac.ir

نشریه علمی "پدافند الکترونیکی و سایبری"

سال دهم، شماره ۲، تابستان ۱۴۰۱، ص ۱۵۶-۱۴۳

علمی – پژوهشی

کاشف: تشخیص‌گر دو مرحله‌ای فایل‌های اجرایی بداندیش ویندوزی

احسان‌اله شقاقی[1*]، رضا جلایی[2]، محمدعلی جوادزاده[3]

۱- دانشجوی کارشناسی ارشد، ۲و۳- استادیار، دانشگاه جامع امام حسین (ع)، تهران، ایران

(دریافت: ۱۶/ ۰۶/ ۱۴۰۰، پذیرش: ۰۹/ ۰۹/ ۱۴۰۰)

چکیده

رشد روزافزون بدافزارها، از تهدیدات مهم حوزه سایبری است و تشخیص آن‌ها را همواره با چالش‌هایی همراه کرده است. فایل‌های اجرایی بداندیش ویندوزی از طریق دستکاری ویژگی‌های موجود در سرآیند آن‌ها و مبهم‌سازی رفتار خود، فعالیت‌های مخرب را در سطح سیستم عامل هدف و یا هر برنامه کاربردی دیگر انجام می‌دهند. تشخیص نمونه‌های مشکوک بداندیش از میان حجم انبوهی از نمونه‌های ورودی و همچنین کشف بدافزارهای جدید و ناشناخته از موضوعاتی است که همواره مورد تحقیق پژوهشگران است. در این پژوهش، روشی ترکیبی برای تعیین میزان بداندیش بودن فایل‌های اجرایی مشکوک پیشنهاد شده است. روش پیشنهادی کاشف، شامل دو ماژول ایستا، برای استخراج ویژگی‌های سرآیند فایل اجرایی، و دو ماژول رفتاری برای استخراج ویژگی‌هایی برای تولید امضاء و مدل رفتاری بداندیش بر اساس روش‌های یادگیری ماشین است. هدف این پژوهش مشکوک‌یابی فایل‌های قابل اجرای ویندوزی از میان حجم انبوهی از فایل‌ها و تعیین میزان بداندیش بودن آن‌ها است. این روش، بدافزارها را بر اساس میزان احتمال بداندیش بودن اختصاص داده شده به هر فایل تشخیص می‌دهد. آزمایش‌ها، درصد بداندیشی شش نوع بدافزار را برای تشخیص‌گر مبتنی بر سرآیند فایل اجرایی، در بازه ۶۲/۷ تا ۷۰ درصد، برای تشخیص‌گر مبتنی بر یارا، در بازه بین ۷۰/۸ تا ۷۸/۲ درصد، برای تشخیص‌گر مبتنی بر امضای رفتاری، ۹۸ درصد و برای تشخیص‌گر مبتنی بر یادگیری ماشین با استفاده از الگوریتم یادگیری جنگل تصادفی ۹۹ درصد نشان می‌دهد. همچنین نتایج آزمایش‌ها نشان داد که کاشف با تشخیص ۹۴ درصدی بدافزارهای محافظت شده، بهبود دو درصدی در مقایسه با نتایج ۱۰ محصول مشابه دارد و با تشخیص ۹۸ درصدی بدافزارهای محافظت نشده، بهبود پنج درصدی در مقایسه با نتایج ۱۰ محصول مشابه دارد.

کلیدواژه‌ها: بدافزار، فایل اجرایی، تشخیص بدافزار، امضای رفتاری، الگوریتم جنگل تصادفی

۱- مقدمه

بدافزارها نرم‌افزارهایی هستند که در سامانه قربانی فعالیت بداندیش[1] انجام می‌دهند [۱]. این فعالیت ممکن است جمع‌آوری اطلاعات شخصی قربانی، اجرای دستور خاصی بر روی سامانه و یا موارد مشابه باشد. رشد روزافزون بدافزارها، آن‌ها را به یکی از تهدیدات مهم در حوزه سایبری تبدیل کرده است. شرکت امنیتی مکافی[2] گزارشی را از نرخ رشد بدافزارها بین سال‌های ۲۰۱۷ تا یک‌چهارم ابتدایی سال ۲۰۱۹ منتشر کرده است [۲]. روند صعودی این نرخ و افزایش تعداد بدافزارهای جدیدی که در یک‌چهارم ابتدایی سال ۲۰۱۹ کشف شده‌اند از نکاتی هستند که در این گزارش بیان شده است. این نتایج نشان می‌دهد که کشف و شناسایی بدافزارهای جدید به یک چالش عمده در سال‌های اخیر تبدیل شده است.

برای تشخیص بدافزارها روش‌های متنوعی وجود دارد، عمر اصلان و رفیک سامت در پژوهش خود [۳] شرح جزئیات این روش‌ها را ارائه کرده‌اند. هر یک از روش‌های تشخیص، به ویژگی‌هایی نیاز دارند که از تحلیل بدافزارها به‌دست می‌آیند. برای تحلیل و احصای ویژگی‌های مورد نیاز از روش‌های تحلیل ایستا و رفتاری استفاده می‌شود. موهانتا در پژوهش خود [۴] هر یک از این روش‌ها را شرح داده است.

این پژوهش روشی پیشنهاد می‌کند که بدافزارهای جدید و ناشناخته را از میان حجم انبوهی از ورودی‌ها تشخیص می‌دهد. مدل سه لایه‌ای پیشنهادی، متشکل از دو مرحله غربال‌گر و یک مرحله تشخیص‌گر است. در این روش برای کاهش حجم ورودی و دست‌یابی به نمونه‌های مشکوک، ابتدا نرم‌افزارهای معتبر بی‌خطر و سپس بدافزارهای شناخته شده جداسازی می‌شوند.

* رایانامه نویسنده مسئول: Kpshaghaghi@ihu.ac.ir

[1] Malicious
[2] Mcafee

ناشر: دانشگاه جامع امام حسین (ع)

کشف بدافزارهای جدید توسط تشخیص‌گر، و از میان نمونه‌های مشکوک صورت می‌گیرد. بدین منظور روش تشخیص ترکیبی شامل دو ماژول تشخیص‌گر ایستا و دو ماژول تشخیص‌گر رفتاری پیشنهاد می‌شود. ماژول‌های تشخیص‌گر ایستا شامل تشخیص‌گر مبتنی بر سرآیند PE و تشخیص‌گر مبتنی بر قوانین یارا[1] است. و ماژول‌های تشخیص‌گر رفتاری شامل تشخیص‌گر مبتنی بر امضای رفتاری و تشخیص‌گر مبتنی بر یادگیری ماشین است. در این مرحله هر یک از ماژول‌ها به‌صورت مستقل میزان بداندیش بودن نمونه‌های مشکوک را محاسبه می‌کنند. با محاسبه حد آستانه تشخیص، در صورتی که حاصل جمع به‌دست آمده از حد آستانه بزرگ‌تر یا مساوی باشد، آن نمونه مشکوک به‌عنوان بدافزار جدید تشخیص داده می‌شود.

در ادامه نوآوری‌های پژوهش فهرست می‌شوند.

- استفاده از دو ماژول غربال‌گر برای کاهش حجم بالای ورودی

  این پژوهش، دو مرحله غربال‌گری پیشنهاد می‌کند که ورودی‌های ناشناس را تا حد زیادی کاهش می‌دهد. یافتن فایل‌های مشکوک با دقت بالاتر برای کشف بدافزارهای جدید؛ از مزیت‌های این غربالگری است.

- استفاده از زیررشته‌های مشترک[2]

  در اغلب روش‌های ارائه شده مشابه، از زیردنباله‌های مشترک[3] استفاده می‌شود. این پژوهش از زیررشته‌های مشترک برای یافتن امضاهای رفتاری بداندیش استفاده می‌کند.

- دسته‌بندی فراخوانی‌های سامانه‌ای جهت استفاده از تعداد بیشتری از فراخوانی‌ها و ارائه توصیف رفتار سطح بالای نمونه‌های مشکوک در قالب لیست اقدام، یکی دیگر از نوآوری‌هایی است که در این پژوهش به‌منظور تولید امضاهای رفتاری بداندیش استفاده شده است.

- برای بررسی رفتار بدافزارها، معمولاً از فراخوانی‌های سامانه‌ای آن‌ها استفاده می‌شود. به‌طور مرسوم و در اغلب تحقیقات مشابه از تعداد تکرار فراخوانی‌ها برای این کار استفاده می‌شود. از جمله نوآوری‌های این پژوهش، استفاده از تعداد تکرار دسته فراخوانی‌ها به‌منظور ایجاد مدل تشخیص است.

- استفاده از روش وزن‌دهی برای انتخاب تعداد محدودتری از قوانین یارا و ویژگی‌هایی از سرآیند PE که احتمال بداندیشی بیشتری دارند.

ادامه پژوهش به ترتیب زیر نگارش شده است. در بخش دو، مفاهیم پایه بیان شده است، در بخش سه پژوهش‌های مشابه در این حوزه بیان شده است. در بخش چهار روش پیشنهادی ارائه می‌شود و در بخش پنج روش پیشنهادی مورد ارزیابی قرار می‌گیرد. درنهایت و در بخش شش نیز نتیجه‌گیری و پیشنهادهایی برای کارهای آینده ارائه شده است.

۲- مفاهیم پایه

در این بخش مفاهیم پایه مورد استفاده در این پژوهش ارائه می‌شوند.

۲-۱- سرآیند PE

فایل اجرایی قابل حمل[4]، قالب فایل‌های اجرایی در سیستم عامل ویندوز مانند exe.، dll.، ocx.، sys. و drv. است. این نام‌گذاری به دلیل قابلیت اجرا شدن، مستقل از نوع معماری سخت‌افزاری است که سیستم عامل ویندوز در آن اجرا می‌شود [۵]. زمانی که یک فایل اجرایی کامپایل می‌شود بخشی به نام سرآیند PE به آن اضافه می‌شود که توصیف‌گر ساختار فایل است. سرآیند PE برای تحلیل ایستا و تشخیص بدافزارها استفاده می‌شود. به‌عنوان مثال اطلاعاتی نظیر اندازه فایل، آدرس شروع فایل، فهرست کتابخانه‌ها و توابع مورد نیاز فایل، تعداد و اندازه بخش‌های[5] به‌کار رفته در فایل و موارد دیگر در سرآیند PE قرار دارند. وجود برخی ویژگی‌ها و مقادیر آن‌ها در سرآیند PE نشان دهنده بداندیش بودن یک نمونه اجرایی از لحاظ ویژگی‌های ساختاری است.

۲-۲- یارا

یارا[6] پروژه متن بازی است که شامل پایگاه داده‌ای از الگوهای ایستای بداندیش شناخته شده است. این پایگاه داده به‌صورت ترکیبی منطقی از این الگوها است که به هر یک از آن‌ها قانون[7] گفته می‌شود. یارا با استفاده از تطابق الگوهای بداندیش باینری و متنی و به‌صورت ایستا بدافزارها را تشخیص می‌دهد [۶]. پروژه یارا شامل قوانین ضد مهندسی معکوس[8]، ضد دیباگ[9]، ضد

1 Yara
2 Common substring
3 Common subsequence

4 Portable Executable
5 Section
6 Yara
7 Rule
8 Anti Reverse Engineering
9 Anti Debug

ماشین مجازی[1] و ضد جعبه شن[2] است که در بدافزارها کاربرد گسترده‌ای دارد.

۲-۳- کتابخانه مرجع NSRL

کتابخانه مرجع NSRL[3]، پروژه‌ای است که توسط وزارت امنیت داخلی ایالات متحده آمریکا[4] و مؤسسه ملی استانداردها و فناوری[5]، برای جمع‌آوری و تهیه پروفایلی از تمامی نرم‌افزارهای معتبر و بی‌خطر ایجاد شده است [۷]. در این کتابخانه، امضاهای دیجیتال به‌صورت پایگاه داده‌ای از درهم‌سازهای MD5 برنامه‌های بی‌خطر وجود دارد. از این پایگاه داده که به‌طور مستمر به‌روزرسانی می‌شود، به‌عنوان مرجعی برای امضاهای برنامه‌های بی‌خطر استفاده می‌شود.

۳- مروری بر پژوهش‌های مرتبط

در این بخش آن دسته از پژوهش‌هایی که در حوزه تشخیص بدافزار مورد استفاده قرار گرفته‌اند و همچنین پژوهش‌هایی که به‌طور مستقیم با روش پیشنهادی مرتبط هستند، مورد بررسی قرار می‌گیرند.

در پژوهش رامان [۸] تمامی ویژگی‌های سرآیند PE بررسی شده است و ویژگی‌هایی که می‌توانند بیشترین تمایز میان بدافزارها و نرم‌افزارهای معتبر بی‌خطر را ایجاد کنند نیز ارائه شده است. در این تشخیص‌گر ایستا که برای استفاده در ضد بدافزارها نیز توصیه شده است اشاره‌ای به چگونگی برخورد تشخیص‌گر در برابر بدافزارهای مبهم نشده است. شیجو و سلیم در پژوهش خود [۹] از هر دو روش ترکیبی جهت شناسایی بدافزارها استفاده کرده‌اند. در این تحقیق از تعداد تکرار اطلاعات رشته‌ای قابل چاپ[6] به‌عنوان ویژگی‌های ایستا و از ۳-گرام[7] و ۴-گرام فراخوانی‌های سامانه‌ای به‌عنوان ویژگی‌های رفتاری استفاده شده است. با استفاده از بردارهای به‌دست آمده از ویژگی‌های ایستا و رفتاری و استفاده از الگوریتم‌های یادگیری ماشین، مدلی برای تشخیص بدافزار پیشنهاد شده است. تحقیق جاری با استفاده از ویژگی‌های ترکیبی سعی در افزایش دقت تشخیص بدافزارها دارد ولی در بخش ایستای این روش به‌منظور ایجاد مدل و تشخیص فایل‌های ناشناس، صرفاً از رشته‌ها استفاده شده است و از رشته‌های مبهم صرف‌نظر می‌شود. در صورتی که اکثر بدافزارها به گونه‌ای طراحی می‌شوند که تحلیل‌گران بدافزار نتوانند از چنین روش‌های ساده و پیش پا افتاده، اطلاعاتی جهت تحلیل بدافزار به‌دست بیاورند. از جمله این روش‌ها مبهم‌سازی است که در این تحقیق برای مقابله با آن هیچ راه حلی ارائه نشده و به سادگی از آن عبور می‌شود. در نتیجه ادغام این ویژگی‌ها با سایر ویژگی‌های به‌دست آمده از تحلیل پویا می‌تواند دقت تشخیص را به شدت کاهش دهد زیرا مدل تشخیص به خوبی آموزش داده نشده است. در نتیجه این روش قطعاً دارای خطای مثبت کاذب و منفی کاذب بالایی خواهد بود.

سامی و همکاران در پژوهش خود [۱۰] از روش اکتشافی مبتنی بر رفتار، برای تشخیص بدافزارها استفاده می‌کنند. آن‌ها همچنین برای طراحی مدل تشخیص از سه نوع ویژگی بر اساس فراخوانی‌های سامانه‌ای بدافزار استفاده می‌کنند. این سه ویژگی بر اساس نام تابع سامانه‌ای - آرگومان، نام تابع سامانه‌ای - مقدار بازگشتی و نام تابع سامانه‌ای - آرگومان- مقدار بازگشتی است. اطلاعات و ویژگی‌های به‌دست آمده از این مرحله شامل تمامی ترکیبات ذکر شده از ۱۲۶ تابع سامانه‌ای است که متعلق به کتابخانه‌های advapi32.dll، kernel32.dll، ntdll.dll، user32.dll، wininet.dll و ws2_32.dll است. مدل ایجاد شده با استفاده از نام تابع سامانه‌ای - آرگومان - مقدار بازگشتی، در مقایسه با تعداد سه ضد بدافزار، نتایج بهتری به‌دست آورده است. از جمله مزیت‌های روش پیشنهاد شده در این تحقیق، توانایی در تشخیص بدافزارهای جدید و ناشناخته، مانند بدافزارهای مبهم و چندریخت است. از جمله مزیت‌های این روش می‌توان به شناسایی بدافزارهای جدید و ناشناخته و همچنین بدافزارهایی که از روش‌های معروف فرار از تشخیص مانند عملیات مبهم‌سازی استفاده کرده و همچنین برای شناسایی بدافزارهای پیچیده مانند بدافزاهای چندریخت اشاره کرد که عملکرد بسیار بهتری نسبت به روش‌های ایستا دارد.

از نقاط ضعف این پژوهش ضعف الگوهای یادگیری برای شناسایی بدافزارهای جدید است. از آنجایی که نتایج آزمایش‌ها در این پژوهش، نام تابع سامانهای – آرگومان – خروجی را به‌عنوان ویژگی‌های مناسب جهت ایجاد الگوی شناسایی معرفی کرده است، برای شناسایی بدافزارهایی که از روش‌های غیر معمول‌تر برای عملیات خود استفاده می‌کنند با مشکل مواجه می‌گردد. به‌عنوان مثال چنانچه برای بدافزارها معمولاً برای نوشتن برخی فایل‌ها (مثلاً فایل تنظیمات خود و مانند آن) از پوشه temp استفاده کنند و الگوی تشخیص بر اساس این آرگومان ایجاد شده باشد و حال بدافزاری جدید برای نوشتن در یک فایل از پوشه جدید استفاده کند سامانه قادر به شناسایی آن نخواهد بود. همچنین این روش نیز مانند سایر روش‌هایی که صرفاً بر روی نوع خاصی از ویژگی‌ها متمرکز هستند نیز دارای

[1] Anti Virtual Machine
[2] Anti Sandbox
[3] National Software Reference Library
[4] DHS: United States Department of Homeland Security
[5] NIST: National Institute of Standards and Technology
[6] Printable String Information
[7] Gram

نقاط ضعفی مانند عدم شناسایی صحیح بدافزارهایی است که با فراخوانی‌های هرز و گمراه کننده سعی در گمراه کردن سامانه تشخیص دارند.

فوجیه فان و همکاران [۱۱] روشی اکتشافی مبتنی بر ویژگی‌های ایستا در شناسایی و تشخیص بدافزاها ارائه کرده‌اند. در این روش، بیشترین اختلاف میان فایل‌های بدافزار و بی‌خطر مشخص می‌شود. روش پیشنهاد شده با استفاده از دیس اسمبل کردن[۱] نمونه‌های آزمایشی و استفاده از الگوریتم اپریوری[۲]، بدافزارها و نرم‌افزارهای معتبر بی‌خطر را متمایز کرده است. درنهایت الگوی تشخیص با استفاده از الگوریتم ANN[۳] ارائه شده است. این تحقیق روشی صرفاً ایستا بوده و در برابر بدافزارهای مبهم عملکرد مناسبی نخواهد داشت. زیرا برای استخراج صحیح الگوی دنباله کدهای اجرایی باید کد اصلی را در اختیار داشت که لازمه این کار رفع ابهام[۴] کردن ابتدایی فایل است. البته در این تحقیق جهت رفع این موضوع به استفاده از ضد فشرده‌سازهای خودکار اشاره شده است. ولی این ابزارها در ضد فشرده[۵] کردن بسیاری از بدافزارها، به خصوص مواردی که با فشرده‌سازهای[۶] سفارشی و ناشناخته فشرده شده‌اند ناموفق عمل می‌کند.

از دیگر نقاط ضعف این پژوهش ناتوانی در تشخیص بدافزارهای فراریخت است که در آن کد بدافزار عملاً جایگزین مقادیر معادل می‌شود. این موضوع نیز باعث افزایش نرخ خطای منفی کاذب تشخیص‌گر است.

آندره سوسانتوو احمد منور [۱۲] روشی ترکیبی برای تشخیص بدافزارها ارائه کرده‌اند. در این تحقیق بخش تشخیص‌گر ایستا، طی دو مرحله، بدافزار بودن نمونه‌های ورودی را بررسی می‌کند. این دو مرحله شامل بررسی معتبر بودن تولید کننده[۷] نمونه ورودی و بررسی آن نمونه با ضد بدافزار ClamAV است. تشخیص‌گر رفتاری نیز مبتنی بر قوانین عمل می‌کند. برای این منظور سه موتور api call rule engine، rule engine registery و rule engine file طراحی شده است که قوانین بداندیش را تولید می‌کنند. قانون اول تغییراتی است که در رجیستری یا فایل جهت اجرای خودکار بدافزار در زمان بوت شدن سامانه ایجاد می‌شود. قانون دوم تغییراتی است که در پوشه AppData برای مخفی کردن بدافزار اعمال می‌شود. و دسته سوم قوانینی هستند که پوشه‌های سامانه‌ای ویندوز نظیر Windows، System32 و مانند آن را به‌منظور دور زدن UAC[۸] سامانه، تغییر می‌دهند. ضعف این تحقیق در بخش تشخیص‌گر ایستا است. این روش، مدل خاصی برای استفاده از ویژگی‌های ایستا ارائه نکرده است. ضعف دیگر آن در بخش تشخیص‌گر رفتاری است که فقط بر اساس سه دسته قانون مذکور عمل می‌کند. بدین ترتیب، بدافزارهایی با رفتار مغایر با قوانین مد نظر، تشخیص داده نمی‌شوند.

هانسن و همکاران [۱۳] با استفاده از تحلیل رفتاری و بر اساس ترتیب و تکرار فراخوانی‌های APIها، مدلی برای تشخیص و دسته‌بندی خانواده‌های بدافزارها ارائه کرده‌اند. در این تحقیق تعداد مشخصی بدافزار و فایل بی‌خطر در جعبه شن کوکو[۹] اجرا می‌شوند و APIهای فراخوانی شده به همراه آرگومان‌های ورودی‌شان لاگ می‌شوند. در این بخش از ابزار InetSim استفاده شده است تا چنانچه بدافزار قصد برقراری ارتباط اینترنتی داشت، سرویس مورد نظر به‌صورت جعلی[۱۰] فراهم شود. در ادامه در بخش Data Extraction برای مشخص کردن بدافزار بودن یا نبودن یک فایل برای ورود به مرحله یادگیری از ضد بدافزار مرکب ویروس توتال[۱۱] استفاده شده است. درنهایت با اجرای نمونه داده‌های به‌دست آمده، ویژگی‌های مورد نظر که همان ترتیب و تعداد فراخوانی‌های APIها است استخراج می‌شود. برای تشخیص بدافزار ابتدا تولی APIها و سپس آرگومان‌های آن‌ها به‌صورت مجزا و در ادامه آن‌ها را در قالب مد نظر قرار گرفته است، سپس برای افزایش دقت روش، تعداد دفعاتی که هر API فراخوانی شده است نیز شمارش می‌شود. تعداد تکرار APIهایی که از لحاظ رفتاری با APIهای دیگر موجود متفاوت هستند و تعداد تکرار DLLهای پرکاربرد نیز به‌عنوان بخش‌های دیگر قالب مد نظر جهت شناسایی بدافزارها درنظر گرفته می‌شوند.
این پژوهش مبتنی بر اجرای فایل بوده و در آن برای مقابله با روش‌های شناسایی از تحلیل و تشخیص هیچ راهکاری ارائه نشده است. همچنین استفاده از توالی‌های API در صورت عدم انتخاب صحیح توالی‌ها می‌تواند منجر به افزایش نرخ خطای مثبت کاذب در تشخیص‌گر شود.

گالال و همکاران [۱۴] روشی برای تشخیص بدافزار بر اساس اجرای بدافزار و فایل‌های بی‌خطر و استخراج APIهای مورد استفاده در آن‌ها ارائه کرده‌اند. در این پژوهش ابتدا تعدادی فایل بدافزار و بی‌خطر در محیط مجازی اجرا شده و بر اساس روش قلاب‌اندازی[۱۲]، APIهایی که از نظر محقق بیشترین تمایز میان بدافزارها و فایل‌های بی‌خطر را ایجاد می‌کنند، استخراج می‌شود.

[1] Disassemble
[2] Apriori
[3] All Nearest Neigbors
[4] Deobfuscate
[5] Unpack
[6] Packer
[7] Vendor
[8] User Account Control
[9] Cuckoo
[10] Fake
[11] Virustotal
[12] Hooking

در گام دوم APIها به چهار دسته مجزا از هم تقسیم می‌شوند که عبارتند از: استفاده کننده از دستگیره‌ها[1]، تولیدکننده دستگیره‌ها، آزاده کننده دستگیره‌ها و APIهایی که از دستگیره‌ها استفاده نمی‌کنند. درنهایت با استفاده از توالی APIهای به‌دست آمده و ایجاد لیست اقدام[2] مدل تشخیص ایجاد می‌گردد. از جمله نقاط ضعف روش، ایجاد لیست اقدام با تمرکز بر توابعی است که بر اساس دستگیره‌ها با یکدیگر پیوند دارند. این موضوع می‌تواند باعث افزایش نرخ خطای مثبت کاذب و منفی کاذب در تشخیص بدافزار گردد. زیرا ممکن است توالی APIهای به‌دست آمده مبتنی بر دستگیره‌ها در میان نمونه‌های بدافزار و بی‌خطر دارای اشتراک‌های زیادی باشد.

۴- روش پیشنهادی کاشف

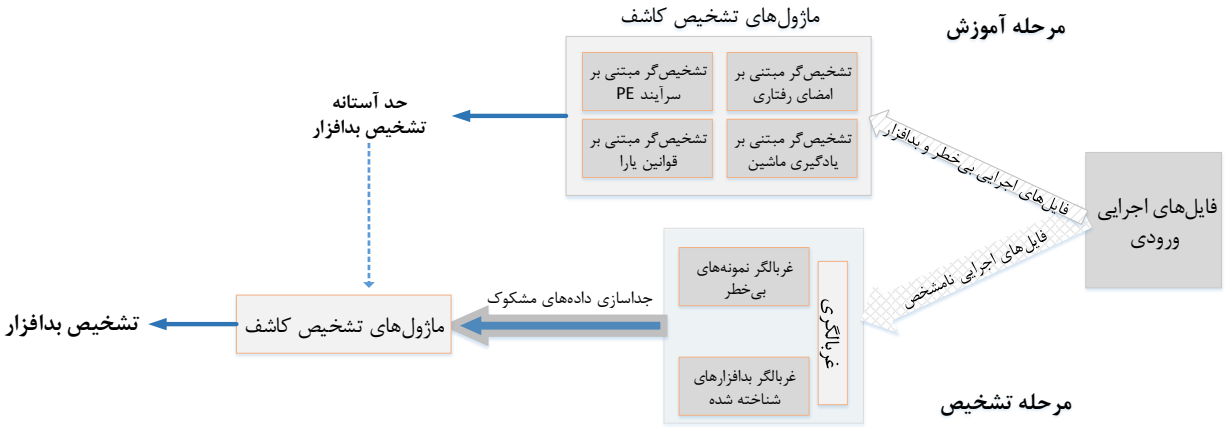
روش پیشنهادی "کاشف" شامل دو مرحله یادگیری و تشخیص است. در شکل (۱) روش تشخیص "کاشف" نشان داده شده است. در مرحله یادگیری، چهار ماژول تشخیص‌گر مبتنی بر سرآیند PE، تشخیص‌گر مبتنی بر قوانین یارا، تشخیص‌گر مبتنی بر یادگیری ماشین و تشخیص‌گر مبتنی بر امضای رفتاری، با استفاده از مجموعه داده‌ای از بدافزارها و نرم‌افزارهای معتبر بی‌خطر آموزش داده می‌شوند. با محاسبه میزان بداندیشی نمونه‌های ورودی (بی‌خطر و بدافزار) و وزن آن‌ها نسبت به دیگر ماژول‌ها، حد آستانه تشخیص بدافزار محاسبه می‌شود.

در مرحله تشخیص، ابتدا بدافزارها و نرم افزارهای معتبر بی‌خطر شناخته شده، توسط دو ماژول مستقل غربالگر بدافزارهای شناخته شده و غربالگر نرم‌افزارهای معتبر بی‌خطر، غربال‌گری می‌شوند. در صورتی که نمونه ناشناس توسط هر کدام از این ماژول‌ها شناسایی شود، توسط همان ماژول به‌عنوان بدافزار یا نمونه بی‌خطر تشخیص داده می‌شود. نمونه‌های مشکوک نیز در پایگاه داده جهت بررسی در مرحله تشخیص، ذخیره می‌شوند. دو ماژول تشخیص ایستا و دو ماژول تشخیص رفتاری، وظیفه تشخیص بدافزارهای جدید را بر عهده دارند. در بخش تشخیص‌گر ایستا، ماژول تشخیص‌گر مبتنی بر سرآیند PE بر اساس ویژگی‌های خاصی از این سرآیند، میزان بداندیش بودن نمونه‌های مشکوک را ارزیابی می‌کند. ماژول تشخیص‌گر مبتنی بر قوانین یارا نیز بر اساس الگوهای ضد مهندسی معکوس[3] میزان بداندیش بودن نمونه‌های مشکوک را مشخص می‌کند.

بخش تشخیص‌گر رفتاری بر اساس فراخوانی‌های سامانه‌ای نمونه‌های مشکوک، نسبت به بدافزار بودن آن‌ها تصمیم‌گیری می‌کند. این بخش دارای دو ماژول تشخیص‌گر مبتنی بر تطابق امضاهای بداندیش و ماژول تشخیص‌گر مبتنی بر یادگیری ماشین است.

۴-۱- مرحله اول: یادگیری

در این مرحله، ماژول‌های "کاشف" جهت تعیین میزان احتمال بداندیش بودن نمونه‌های مشکوک آموزش داده می‌شوند. همچنین، از طریق محاسبه وزن کلی هر یک از این ماژول‌ها، حد آستانه تشخیص بدافزار محاسبه می‌شود. در ادامه نحوه آموزش این چهار ماژول ارائه می‌شوند.



شکل (۱): روش تشخیص کاشف

[1] Handle
[2] Action List

[1] Anti-reverse Engineering

۴-۱-۱- آموزش ماژول تشخیص‌گر مبتنی بر سرآیند PE

در این ماژول، اطلاعات سرآیند PE ۶۰۰۰ نمونه بی‌خطر و بدافزار استخراج می‌شود و با استفاده از رابطه (۱) وزن‌دهی می‌شوند. در این ماژول ابتدا وزن تک تک ویژگی‌ها برای نمونه‌های بدافزار و بی خطر محاسبه می‌شود. در ادامه چون به دنبال انتخاب ویژگی بداندیش هستیم حاصل تفاضل وزن بداندیش یک ویژگی با وزن بی خطری همان ویژگی را به‌دست آورده و در صورت مثبت شدن به‌عنوان ویژگی کاندید درنظر می‌گیریم.

برای انتخاب ویژگی نهایی و کمتر شدن خطا، مجددا وزن ویژگی کاندید را از وزن بداندیشی اولیه آن کم کرده و وزن نهایی بداندیشی را به‌دست می‌آوریم.

$$W_i = \frac{f_i}{N} \qquad (۱)$$

که در آن، f_iتعداد تکرار ویژگی iام، N مجموع کل ویژگی‌ها و W_i وزن ویژگی iام است به‌طوری که $\sum_{i=1}^{n} w_i = 1$.

این ماژول، با توسعه پارسری[1] به زبان پایتون[2] پیاده‌سازی شده است. شاخص انتخاب ویژگی کاندید برای این ماژول مطابق رابطه (۲)، مثبت شدن تفاضل وزن هر ویژگی در بدافزار و نرم‌افزارهای معتبر بی‌خطر است.

$$C_i = W_i\_W'_i \qquad S>0 \qquad (۲)$$

که در آن، C_i وزن ویژگی iام به‌عنوان ویژگی کاندید، W_i وزن ویژگی iام برای نمونه‌های بداندیش و W'_i وزن ویژگی iام برای نرم‌افزارهای معتبر بی‌خطر است.

وزن نهایی بداندیش ویژگی منتخب نیز از رابطه (۳) به‌دست می‌آید.

$$S_i = W_i\_C_i \qquad (۳)$$

که در آن، S_i وزن نهایی ویژگی منتخب، W_i وزن بداندیش ویژگی iام و C_i وزن ویژگی iام به‌عنوان ویژگی کاندید است.

جدول (۱) ویژگی‌های منتخب و وزن نهایی بداندیش آن‌ها را نشان می‌دهد.

۴-۱-۲- آموزش ماژول تشخیص‌گر مبتنی بر قوانین یارا

برای آموزش این ماژول، ابتدا قوانین ضد مهندسی معکوس یارا برای تعداد ۶۰۰۰ بدافزار و نمونه بی‌خطر بررسی می‌شوند. در صورت مطابقت هر قانون، تعداد آن‌ها محاسبه می‌شوند. سپس هر قانون وزن‌دهی می‌شود. شاخص انتخاب قانون برای این ماژول، مثبت شدن حاصل تفریق وزن هر قانون در بدافزارها از نرم‌افزارهای معتبر بی‌خطر است. در جدول (۲) قوانین منتخب حاصل از آزمایش‌ها به همراه وزن نهایی بداندیش برای هر یک از آن‌ها نشان داده شده است.

جدول(۱): وزن بداندیش ویژگی‌های منتخب سرآیند PE

ویژگی‌های منتخب سرآیند PE	وزن‌های نهایی بداندیش
FileEntropy	۲۰/۷۳۵
Characteristics (RELOCS_STRIPPED)	۱۵/۹۵۸
Characteristics (BYTE_RESERVED_LO)	۱۲/۳۱۱
Characteristics (BYTE_RESERVED_HI)	۱۳/۳۰۱
MajorLinkerVersion: MinirLinkerVersion	۴/۵۳۱
SizeOfCode /SampleSize	۰/۱۱۱
SizeOfUninitializedData / SampleSize	۰/۱۲۹
SizeOfImage / Size	۰/۰۵۵
AddressOfEntryPoint / SamplesSize	۰/۳۹۹
BaseOfData/ SamplesSize	۰/۰۸۳
RawSize	۱۵/۵۲۱
VirtualSize /RawSize	۰/۹۷۳
Characteristics(IMAGE_ SCN_CNT_UNINITIALIZED_ DATA)	۶/۹۰۹
Characteristics(IMAGE_ SCN_MEM_SHARED)	۳/۹۴۳
SubXLanguage=0	۵/۰۴۱

جدول(۲): وزن بداندیش قوانین ضدمهندسی معکوس یارا

قانون	وزن نهایی بداندیش
ThreadControl__Context	۹/۶۸۹۲
DebuggerPattern__RDTSC	۱۹/۸۷۸
DebuggerPattern__CPUID	۳۱/۷۴۵
DebuggerPattern__SEH_Inits	۲۸/۵۷۳
SEH__vba	۳/۶۲۸۸
DebuggerCheck__RemoteAPI	۰/۷۲۰
Check_Qemu_Description	۰/۷۲۱
Check_Qemu_DeviceMap	۰/۷۲۱
Check_VBox_Description	۰/۷۲
Check_VBox_DeviceMap	۰/۷۲۱
Check_VBox_VideoDrivrs	۰/۷۲
Check_VMWare_DeviceMap	۰/۷۲۱
Check_Wine	۰/۷۲۱
Check_FilePaths	۰/۷۲۱

[1] Parser
[2] Python

۴-۱-۳- آموزش ماژول تشخیص‌گر مبتنی بر امضای رفتاری

وظیفه این ماژول به‌دست آوردن امضاهای رفتاری بداندیش است. این امضاها با استفاده از دسته‌بندی APIهایی که از لحاظ عملکرد شبیه به هم هستند به‌دست می‌آیند. بدین منظور با توجه به تعامل بدافزار با سیستم عامل، لیست اقدام[1] در قالب چند دسته مجزا تشکیل می‌شود. توابع مرتبط با این لیست اقدام از مرجع MSDN[2] گردآوری شده است. به‌عنوان مثال کلیه توابع API که فایل و دایرکتوری ایجاد می‌کنند در یک دسته با نام A قرار می‌گیرند. در مجموع تعداد ۱۲۲۲ تا API به‌دست آمد. در جدول (۳) طبقات مختلف و کد اختصاصی به هر طبقه نشان داده شده است.

جدول (۳): طبقات مختلف APIها وکد دسته‌ها

کد	دسته	کد	دسته
N	Synchronization	A	File/Directory Create
O	Device Control	B	File Write
P	Socket Comm / Internet com	C	File Read
Q	Network Information	D	File/Directory Info
R	Internet Open/Read	E	Registry Create
S	Win-Service	F	Registry Read
T	Get System Information	G	Registry Write
U	File/Directory Delete	H	Registry Delete
V	COM/OLE/DDE	I	Process/thread Create
W	Memory	J	Process/thread Info
X	System	K	Process/thread Write
Y	Enumeration	L	Hooking
Z	Encryption	M	debugging and ErrorHandling

جدول (۴) نیز چند نمونه از دسته‌ها، کد مرتبط و تعداد APIهای آن را به همراه مثال نشان می‌دهد.

جدول (۴): نمونه‌ای از دسته‌بندی APIها به همراه نام آن دسته

دسته	کد	تعداد کل APIها	مثال
File/Directory Create (open)	A	۱۵	CreateDirectoryA-CreateDirectoryW-CreateFile2-…
File Write	B	۱۷	WriteFile – WriteFileEx – CopyFilE - ….
File Read	C	۵	ReadFile - ReadFileScatter – ReadFileEx - …

برای آموزش این ماژول، بعد از قرار گرفتن فراخوانی‌های APIهای مرتبط با ۱۰۰ نمونه بدافزار و نمونه بی‌خطر در پایگاه داده اسکیوال‌لایت[3]، APIهای موجود در این پایگاه داده، با جدول (۳) مقایسه و با کد دسته معادل آن جایگزین می‌شوند. امضاهای بداندیش طی دو مرحله به‌دست می‌آیند. ابتدا توالی فراخوانی‌ها به‌صورت زیررشته‌های مشترک[4] استخراج می‌شوند. سپس امضاهای بداندیش به‌دست آمده و زیر رشته‌های بدافزارها و نرم‌افزارهای معتبر بی‌خطر مقایسه می‌شوند و موارد مشترک بین آن‌ها حذف می‌گردند.

نحوه استخراج این زیررشته‌های مشترک میان توالی فراخوانی‌ها به‌صورت مقایسه‌ای - پیمایشی است. الگوریتم (۱)، روش استخراج زیررشته‌های مشترک را نشان می‌دهد.

الگوریتم (۱): نحوه استخراج زیررشته‌های مشترک

Input:
S_i: Selected system call sequence
C_i: System call sequences
L: System call sequence length
T: S_i splited vector
D: C_i splited vector

Output:
F: signature vector

1. **for** i: = 3 to 100 **do**
2. split S_i with length of i
3. insert to T vector
4. **end for**
5. **for** j: = 3 to 100 **do**
6. $base = 0$
7. $chunk\_Size = j$
8. **While** $chunk\_Size <> L$
9. split C_i from $C_i[base]$ to C_i[chunk_size]
10. insert to D vector
11. $base = base + 1$
12. $chunk\_Size = chunk\_Size + 1$
13. **end while**
14. **end for**
15. **for** each $T_i \in T$ **do**
16. **if** $T_i = D_i$ **then**
17. insert T_i to F
18. **end if**
19. **end for**
20. **return** F

الگوریتم (۱) توالی فراخوانی‌های یک نمونه بدافزار به‌عنوان مبنا و توالی فراخوانی‌های سایر بدافزارها را به ترتیب با نام‌های S_i و C_i به‌عنوان ورودی می‌گیرد و حاصل تطابق زیررشته‌های مشترک در توالی‌ها را در بردار F قرار می‌دهد. ابتدا توالی‌های ۳ تایی تا ۱۰۰ تایی از توالی مبنا جداسازی شده و در بردار T قرار می‌گیرند

[1] Action list
[2] MicroSoft Developer Network
[3] Sqlite
[4] Common SubStrings

(خط ۱ تا ۴). در ادامه، توالی‌های ۳ تایی تا ۱۰۰ تایی بردار C_i بر اساس مقادیر base و chunk_size که ایندکس‌های ابتدا و انتهایی بردار C_i هستند، جداسازی می‌شوند (خطوط ۵ تا ۱۴). درنهایت توالی‌های بردار T با توالی‌های بردار D مقایسه می‌شوند، و توالی‌های یکسان در بردار امضای F قرار می‌گیرند (خطوط ۱۳ تا ۲۰).

بعد از انجام استخراج و مقایسه توالی‌ها، تعداد ۱۸۴۰ امضای بداندیش به‌دست آمد. چند نمونه از این امضاهای بداندیش به همراه توضیح آن‌ها در جدول (۵) نشان داده شده است.

جدول (۵): نمونه‌های امضاهای رفتاری بداندیش و اقدام متناظر

نوع امضا	اقدام متناظر
YAP	ایجاد سوکت - ایجاد فایل یا دایرکتوری – پیمایش سامانه [1]
KTZM	اشکال‌زدایی و مدیریت خطا – رمزنگاری – گرفتن اطلاعات از سامانه – نوشتن توسط پردازش یا نخ
YMMWQR	خواندن یا ایجاد ارتباط اینترنتی – اطلاعات شبکه – حافظه – اشکال‌زدایی و مدیریت خطا - اشکال‌زدایی و مدیریت خطا - پیمایش سامانه

۴-۱-۴- آموزش ماژول تشخیص‌گر مبتنی بر یادگیری ماشین

وظیفه این ماژول تشخیص بدافزار با استفاده از انتخاب ویژگی‌های مناسب و مبتنی بر روش‌های یادگیری ماشین است. این ویژگی‌ها تعداد تکرار[2] فراخوانی‌های هر دسته است. برای انتخاب این ویژگی‌ها، ابتدا فراخوانی‌های مختلف ۶۰۰۰ نمونه بدافزار و نمونه بی‌خطر دسته‌بندی می‌شوند و سپس تعداد آن‌ها محاسبه می‌شود. مجموعه دادگان نیز با نسبت ۷۰ درصد داده‌های آموزشی و ۳۰ درصد داده‌های آزمون، آماده‌سازی می‌شوند. برای انتخاب ویژگی‌های مناسب، ابتدا عملیات نرمال‌سازی و سپس انتخاب ویژگی انجام می‌گیرد. بدین منظور، الگوریتم‌های cfb، فیشر[3]، Gain Ratio و با الگوریتم‌های یادگیری ماشین Jrip، J48 و جنگل تصادفی[4] مورد آزمایش قرار می‌گیرند.

همان‌طور که در جدول (۶) نشان داده شده است، الگوریتم‌های انتخاب ویژگی cfb و جنگل تصادفی، نرخ مثبت درست[5] بیشتر و نرخ مثبت نادرست[6] کمتری نسبت به سایر الگوریتم‌ها دارند. در نتیجه الگوریتم جنگل تصادفی با استفاده از الگوریتم انتخاب ویژگی cfb برای ایجاد مدل تشخیص در این ماژول مورد استفاده قرار گرفت.

جدول (۶): نتایج تشخیص مبتنی بر الگوریتم‌های یادگیری ماشین

معیار ارزیابی / الگوریتم	نرخ مثبت درست	نرخ مثبت نادرست	دقت
Jrip	۰/۹۷	۰/۰۲	۰/۹۷
J48	۰/۹۷	۰/۰۲	۰/۰۹۷
جنگل تصادفی	**۰/۹۹**	**۰/۰۱**	**۰/۹۹**

۴-۱-۵- محاسبه وزن بداندیشی هر یک از ماژول‌ها

برای محاسبه وزن بداندیش ماژول‌ها، از تعداد ۱۰۰ نمونه بدافزار به‌عنوان ورودی استفاده می‌شود. سهم هر یک از ماژول‌ها در تشخیص بدافزار در جدول (۷) نشان داده شده است. برای مثال، در روش "کاشف"، زیرماژول یارا حداکثر به میزان ۲۲/۰۴ درصد در تشخیص بدافزار مشارکت دارد.

جدول (۷): حداکثر نمره بداندیش نهایی هر زیرماژول

ماژول	وزن بداندیشی
سرآیند PE	۲۰/۳۳
یارا	۲۲/۰۴
الگوهای API بداندیش	۲۸/۵۲
یادگیری ماشین	۲۹/۱۱
مجموع	۱۰۰

۴-۱-۶- محاسبه حد آستانه تشخیص

در "کاشف"، هر یک از چهار ماژول در تشخیص بدافزارها نقش دارند. پارامتر T، حد آستانه تشخیص "کاشف" است. برای این منظور ابتدا تعداد ۱۰۰۰ نمونه بدافزار جدید انتخاب شده و با انجام ۱۰ آزمایش مستقل، حد آستانه تشخیص "کاشف" از رابطه (۱) محاسبه می‌شود.

$$T = \frac{S}{N} \qquad (۱)$$

که در آن، T حد آستانه تشخیص، $S = \sum_{i=1}^{n} d_i$، مجموع نمرات بداندیشی برای ۱۰۰۰ نمونه بدافزار است. به عبارت دیگر d_i، میزان احتمال بداندیشی برای بدافزار iام است. در نهایت، N تعداد کل نمونه آزمایش‌ها است. میانگین به‌دست آمده عددی بین صفر و ۱۰۰ و نرمالایز شده است. زیرا مجموع نمرات بداندیشی چهار ماژول تشخیص‌گر برای یک نمونه عددی بین

1 Socket
2 Frequency
3 Fisher
4 Random Forest
5 True Positive Rate
6 False Positive Rate

صفر تا ۱۰۰ بوده و در صورت تقسیم مجموع نمرات بدافزارها بر تعداد بدافزارها (یعنی ۱۰۰۰)، حاصل عددی بین صفر تا ۱۰۰ خواهد بود.

درنهایت در صورتی که میزان احتمال بداندیشی برای یک نمونه ناشناس، بزرگ‌تر یا مساوی حد آستانه تشخیص باشد، آن نمونه به‌عنوان بدافزار جدید، تشخیص داده می‌شود.

۴-۲- مرحله دوم: تشخیص

وظیفه این مرحله تشخیص بدافزارهای ناشناخته است. در این مرحله، ورودی‌ها با استفاده از ماژول‌های غربالگر بدافزارهای شناخته شده و غربالگر نرم‌افزارهای معتبر بی‌خطر، غربال و جداسازی می‌شوند. چهار ماژول تشخیص‌گر مبتنی بر سرآیند PE، تشخیص‌گر مبتنی بر قوانین یارا، تشخیص‌گر مبتنی بر امضای رفتاری و تشخیص‌گر مبتنی بر یادگیری ماشین، میزان بداندیش بودن نمونه‌های مشکوک را با هدف کشف بدافزارهای جدید انجام می‌دهند.

خروجی هر یک از این چهار ماژول، احتمال بداندیش بودن نمونه‌های ورودی را نشان می‌دهد. این احتمال از رابطه (۲) به‌دست می‌آید.

$$P_t = \sum_{i=1}^{4} p_i \qquad 0 < p_t < 1 \tag{۴}$$

که در آن، p_i احتمال بداندیشی نسبت داده شده توسط هر یک از ماژول‌ها بر اساس وزنی که به آن ماژول نسبت داده شده (با توجه به جدول (۷))، و p_t احتمال بداندیش بودن نمونه مشکوک است. برای مثال، اگر هر کدام از ۴ ماژول تشخیص‌گر، نمره بداندیشی ۷۰ را به یک نمونه مشکوک اختصاص دهند، این اعداد بر حسب وزن آن ماژول نسبت به سایر ماژول‌ها دوباره محاسبه می‌شوند بنابراین احتمال بداندیش بودن ماژول تشخیص‌گر سرآیند PE، ۱۴/۲۳۱ ($\frac{70\times20.33}{100}$) است. اگر برای ماژول‌های دیگر نیز این محاسبه انجام شود، حاصل جمع احتمال بداندیش بودن کل ماژول تشخیص‌گر، عددی بین ۰ و ۱۰۰ است.

در ادامه، این ماژول‌ها مورد بحث و بررسی قرار می‌گیرند.

۴-۲-۱- ماژول غربال‌گر برنامه‌های بی‌خطر

این ماژول با استفاده از تطبیق امضاهای دیجیتال پایگاه داده درهم‌ساز MD5 برنامه‌های بی‌خطر با داده‌های ورودی، نمونه‌های مشکوک را جداسازی می‌کند. این نمونه‌ها در صورت عدم تطابق امضای نمونه ناشناس ورودی با امضاهای موجود، به‌عنوان نمونه مشکوک برچسب‌زنی می‌شوند و به ماژول غربال‌گر بدافزارهای شناخته شده ارسال می‌شوند. در غیر این صورت، نمونه ورودی به‌عنوان نمونه بی‌خطر اعلام می‌شود.

ماژول غرباگر برنامه‌های بی‌خطر از NSRL استفاده می‌کند. بدین منظور با استفاده از زبان برنامه نویسی پایتون، کدی توسعه داده شد که ضمن استفاده آخرین نسخه از پایگاه داده مذکور، به‌صورت خودکار ابتدا درهم‌ساز MD5 برنامه‌های ناشناس ورودی را محاسبه کرده و سپس با امضاهای موجود در پایگاه داده مقایسه می‌کند. نمونه بی‌خطر، از تطابق امضای به‌دست آمده از برنامه ورودی با هر یک از امضاهای موجود در NSRL مشخص می‌شود.

۴-۲-۲- ماژول غربال‌گر بدافزارهای شناخته شده

وظیفه این ماژول غربال‌گری بدافزارهای شناخته شده بر اساس امضاهای آن‌ها است. این ماژول، فایل‌های مشکوک را توسط یک ضد بدافزار چند موتوره غربال می‌کند. این کار با استفاده از روش تشخیص مبتنی بر امضاء و به‌صورت موازی و هم‌زمان، بدافزارهای شناخته شده را غربال می‌کند. انجام مرحله اول ضمن کاهش چشمگیر حجم نمونه‌های ورودی، نمونه‌های مشکوک را نیز کاهش می‌دهد. به‌عبارت دیگر برای کاهش بار پردازشی، از روش تشخیص مبتنی بر امضای ضد بدافزارها به‌عنوان روش تکمیلی و برای بهبود عملکرد روش استفاده شده است.

۴-۲-۳- ماژول تشخیص‌گر مبتنی بر سرآیند PE

این ماژول، ویژگی‌های بداندیش موجود در سرآیند PE نمونه‌های مشکوک را با روش تحلیل ایستا بررسی می‌کند. این ویژگی‌ها در مرجع [۱۵] فهرست شده‌اند. وزن هر کدام از این ویژگی‌ها از رابطه (۵) محاسبه می‌شود.

$$W_i = \frac{f_i}{N} \tag{۵}$$

که در آن، f_iتعداد تکرار ویژگی iام، N تعداد کل ویژگی‌ها و W_i وزن ویژگی iام است به‌طوری که $\sum_{i=1}^{n} w_i = 1$.

۴-۲-۴- ماژول تشخیص‌گر مبتنی بر قوانین YARA

این ماژول با استفاده از تطبیق قوانین ضد مهندسی معکوس موجود در پایگاه داده یارا و وزن‌دهی به قوانین عمل می‌کند. بدافزارها، با استفاده از انتخاب قوانینی تشخیص داده می‌شوند که بیشترین تمایز را میان بدافزارها و نرم‌افزارهای معتبر بی‌خطر ایجاد می‌کنند.

این ماژول نیز با استقاده از رابطه (۲)، قوانین انتخاب شده را وزن‌دهی می‌کند.

۴-۲-۵- ماژول تشخیص مبتنی بر امضاهای رفتاری بداندیش

این ماژول مسئول تطابق مجموعه‌ای از امضاهای رفتاری بداندیش با توالی[1] به‌دست آمده از فراخوانی‌های توابع API نمونه مشکوک است. تعیین بداندیش بودن رفتار نمونه‌های مشکوک به این صورت است که ابتدا این نمونه‌ها در محیطی ایزوله و ایمن اجرا شده و توالی فراخوانی‌های API آن‌ها بر اساس جدول (۳) دسته‌بندی و احصاء می‌شوند. سپس وجود امضاهای رفتاری بداندیش در توالی مذکور بررسی می‌شود. در صورت تطابق هر کدام از این امضاها، آن نمونه به‌عنوان بدافزار تشخیص داده می‌شود.

۴-۲-۶- ماژول تشخیص رفتاری بدافزار مبتنی بر روش‌های یادگیری ماشین

این ماژول نیز وظیفه تشخیص نمونه‌های بدافزار را بر عهده دارد. این کار از طریق مدل تشخیص مبتنی بر یادگیری ماشین انجام می‌شود. همان‌طور که در مرحله آموزش بیان شد؛ روش یادگیری جنگل تصادفی نسبت به روش‌های دیگر نتایج بهتری دارد. بنابراین، نمونه‌های مشکوک توسط مدل تشخیص مبتنی بر الگوریتم جنگل تصادفی به دو دسته بدافزار و بی‌خطر دسته‌بندی می‌شوند.

۵- آزمایش‌ها

در این بخش، نتایج روش پیشنهادی با انجام مجموعه‌ای از آزمایش‌های تشخیص بدافزار نشان داده می‌شوند. این آزمایش‌ها بر اساس سنجه‌های ارزیابی و حد آستانه تشخیص، مورد ارزیابی و تحلیل قرار گرفته و ارائه می‌شوند.

۵-۱- مجموعه دادگان

برای انجام آزمایش‌ها از دو مجموعه داده مجزای بدافزار و بی‌خطر استفاده شده است. مجموعه داده‌های بدافزار از پایگاه داده‌های Virussign، VirusShare، TheZoo، Virustotal و Vxheaven انتخاب شده است. در مجموع تعداد ۶۰۰۰ نمونه بدافزار در خانواده‌های مختلف شامل تروجان، ویروس، کرم، تبلیغ افزار[2]، دانلود کننده[3] و درب پشتی[4] به‌دست آمد. انتخاب انواع مختلف بدافزار به دلیل نیاز به بررسی رفتارهای مختلف بدافزار است.

برنامه‌های بی‌خطر نیز از انواع برنامه‌های کاربردی و DLLهای پیش‌فرض سیستم عامل تمیز[5] ویندوز ۳۲ بیتی نسخه ۷، انتخاب شده است. در مجموع تعداد ۱۰۰۰ برنامه بی‌خطر شامل فایل‌های exe.، dll. و sys. به‌دست آمد. جدول (۸) تعداد و تنوع مجموعه داده به‌دست آمده را نشان می‌دهد.

جدول (۸): نمونه‌های داده جمع‌آوری شده

	خانواده فایل	تعداد
بی‌خطر	exe.، dll.، sys.	۱۰۰۰
بدافزار	Adware Backdoor Downloder Trojan Virus Worm	۱۰۰۰

۵-۲- سکوی آزمایش

برای انجام آزمایش‌ها، محیطی مجزا و امن به‌منظور جلوگیری از آلوده شدن سیستم عامل میزبان ایجاد می‌شود. بدین منظور و برای احصای ویژگی‌های رفتاری، از آخرین نسخه جعبه شن کوکو[6] استفاده می‌شود و تمامی برنامه‌های مقاوم‌سازی در برابر بدافزارها بر روی آن نصب می‌شود. همچنین با توجه به حجم بالای نمونه‌های داده، برنامه‌ای توسعه داده شده است که با استفاده از ویژگی Restful API کوکو، برنامه‌ها را به جعبه شن کوکو ارسال و گزارش‌های آن را در قالب فایل json ذخیره می‌کند.

برای استخراج ویژگی‌های ایستا نیز از ماشین مجازی ویندوز XP ۳۲ بیتی بر روی سیستم عامل میزبان اوبونتو[7] نسخه ۱۸,۰۴,۵ استفاده شده است. آزمایش‌ها نیز از طریق پردازنده اینتل Corei7 با توان پردازش ۳/۵ گیگاهرتز و حافظه موقت ۱۶ گیگابایت انجام شده است.

۵-۳- سنجه‌های ارزیابی

سنجه‌های ارزیابی روش پیشنهادی، نرخ تشخیص صحیح و نرخ هشدار نادرست است. به این صورت که نرخ تشخیص صحیح، تعداد بدافزارهایی است که نسبت به کل بدافزارها، صحیح و به‌عنوان بدافزار تشخیص داده شده است. نرخ هشدار نادرست نیز تعداد بدافزارهایی است که نسبت به کل بدافزارها به‌عنوان نمونه بی‌خطر معرفی شده است.

[1] Sequence
2 Adware
3 Downloader
[4] Backdoor

5 Fresh Operating System
[6] Cuckoo
[7] Ubuntu

۵-۴- روش‌شناسی[1] تشخیص

هدف روش تشخیص، تعیین میزان احتمال بداندیش بودن نمونه‌های مشکوک و تشخیص بدافزارهای جدید است. بدین منظور هر یک از ماژول‌ها، با احتساب وزن تشخیص آن‌ها نسبت به وزن تشخیص ماژول‌های دیگر، احتمال بداندیش بودن نمونه‌های مشکوک را محاسبه می‌کنند. جمع جبری احتمال به‌دست آمده، میزان بداندیش بودن نهایی نمونه‌های مشکوک را نشان می‌دهد.

آزمایش تشخیص بدافزارهای جدید به گونه‌ای تنظیم شد که نرخ خطای مثبت نادرست قابل قبول برای حد آستانه تشخیص، ۰/۳ درصد باشد. بدین ترتیب با تعریف حد آستانه تشخیص بدافزار، نمونه‌هایی که احتمال بداندیشی آن‌ها بیشتر یا مساوی حد آستانه باشد به‌عنوان بدافزار و بقیه موارد به‌عنوان نرم‌افزارهای معتبر بی‌خطر گزارش می‌شوند.

همان‌طور که در روش پیشنهادی بیان شد، این حد آستانه تشخیص با استفاده از مجموعه داده بدافزار و محاسبه وزن تشخیص ماژول‌ها به‌دست می‌آید. بدین ترتیب، میانگین احتمال بداندیشی به‌دست آمده از داده‌های بدافزار با مرتبه تکرار ۱۰ آزمایش، به‌عنوان حد آستانه تشخیص بدافزار درنظر گرفته می‌شود.

۵-۵- عملکرد هر یک از ماژول‌ها

در این بخش به‌منظور محاسبه حد آستانه تشخیص، عملکرد هر چهار ماژول به‌عنوان یک تشخیص‌گر مستقل نشان داده شده است. انجام مستقل هر یک از این ماژول‌ها برای ارزیابی میزان کارایی این ماژول‌ها در مواجه با نمونه‌های بدافزاری انجام می‌شود. به عبارت دیگر نقش هر یک از ماژول‌ها به‌طور جداگانه مورد بررسی و ارزیابی قرار می‌گیرد.

۵-۵-۱- عملکرد تشخیص‌گر مبتنی بر سرآیند PE

برای بررسی و ارزیابی ماژول تشخیص‌گر مبتنی بر سرآیند PE، شش آزمایش مستقل با تعداد ۱۰۰۰ نمونه از هر خانواده بدافزار انجام می‌شود. جدول (۹) نتایج حاصل از انجام آزمایش‌ها را نشان می‌دهد.

جدول (۹): نتایج آزمایش ارزیابی عملکرد ماژول تشخیص‌گر PE

نوع آزمایش	کرم	ویروس	تروجان	دانلود کننده	درب پشتی	تبلیغ افزار
تشخیص صحیح	۶۵/۴	۶۵/۸	۶۸/۴	۷۰	۶۴/۶	۶۲/۷

همان‌طور که در جدول (۹) نشان داده شده است، درصد تشخیص صحیح ماژول تشخیص‌گر مبتنی بر سرآیند PE در بازه ۶۲/۷ الی ۷۰ درصد تغییر کرده است. در نتیجه با توجه به ضعف روش‌های تشخیص ایستا که با تغییرات ظاهری اندک در بدافزار قادر به تشخیص نیستند، نتایج به‌دست آمده مطلوب ارزیابی می‌شود.

۵-۵-۲- عملکرد تشخیص‌گر مبتنی بر قوانین یارا

برای بررسی و ارزیابی ماژول تشخیص‌گر مبتنی بر قوانین یارا، شش آزمایش مستقل با تعداد ۱۰۰۰ نمونه از هر خانواده بدافزار انجام می‌شود. جدول (۱۰) نتایج حاصل از انجام آزمایش‌ها را نشان می‌دهد.

جدول (۱۰): نتایج آزمایش ارزیابی عملکرد ماژول تشخیص‌گر یارا

نوع آزمایش	کرم	ویروس	تروجان	دانلود کننده	درب پشتی	تبلیغ افزار
تشخیص صحیح	۷۴/۳	۷۵/۵	۷۰/۸	۷۸	۷۸/۲	۷۱/۶

همان‌طور که در جدول (۱۰) نشان داده شده است، درصد تشخیص صحیح ماژول تشخیص‌گر مبتنی بر قوانین یارا در بازه ۷۰/۸ الی ۷۸/۲ درصد تغییر کرده است. در نتیجه با توجه به ضعف روش‌های تشخیص ایستا که با تغییرات ظاهری اندک در بدافزار قادر به تشخیص نیستند، نتایج به‌دست آمده مطلوب ارزیابی می‌شود.

۵-۵-۳- عملکرد تشخیص‌گر مبتنی امضاهای رفتاری

برای بررسی و ارزیابی ماژول تشخیص‌گر مبتنی بر سرآیند PE، تعداد ۱۰۰۰ نمونه از هر خانواده بدافزار انتخاب و آزمایش می‌شود. جدول (۱۱) نتایج حاصل از انجام آزمایش‌ها را نشان می‌دهد.

جدول (۱۱): نتایج آزمایش ارزیابی عملکرد ماژول تشخیص‌گر PE

تعداد بدافزار	تعداد تشخیص صحیح	درصد تشخیص صحیح
۱۰۰۰	۹۸۰	۹۸

همان‌طور که در جدول (۱۱) نشان داده شده است، درصد تشخیص صحیح ماژول تشخیص‌گر مبتنی بر امضای رفتاری ۹۸

1 Methodology

درصد است که نشان می‌دهد درصد تشخیص صحیح روش‌های تشخیص رفتاری نسبت به روش‌های تشخیص ایستا بیشتر است.

۴-۵-۵- عملکرد تشخیص‌گر مبتنی بر یادگیری ماشین

برای بررسی و ارزیابی ماژول تشخیص‌گر مبتنی بر سرآیند PE، تعداد ۱۰۰۰ نمونه از هر خانواده بدافزار انتخاب و آزمایش می‌شود. جدول (۱۲) نتایج حاصل از انجام آزمایش‌ها را نشان می‌دهد.

جدول (۱۲): نتایج آزمایش ارزیابی عملکرد ماژول تشخیص‌گر PE

تعداد بدافزار	تعداد تشخیص صحیح	درصد تشخیص صحیح
۱۰۰۰	۹۹۰	۹۹

نتیجه آزمایش نشان می‌دهد که درصد تشخیص صحیح روش‌های تشخیص رفتاری نسبت به روش‌های تشخیص ایستا بیشتر است.

۵-۶- ارزیابی روش پیشنهادی

در این بخش، روش تشخیص "کاشف" در دو مرحله ارزیابی می‌شود. مطابق بخش ۶-۱-۴، میزان حد آستانه تشخیص، عدد ۶۵ به‌دست آمد. به همین منظور در مرحله اول، کارایی تشخیص "کاشف" در شناسایی بدافزارها با تعداد ۱۰ محصول معتبر ضد بدافزار، مطابق سنجه‌های ارزیابی مقایسه می‌شود. و در مرحله دوم نیز، میزان خطای "کاشف" در مواجه با نرم‌افزارهای معتبر بی‌خطر مورد ارزیابی قرار می‌گیرد.

ماژول‌های غربال‌گر "کاشف" صرفاً برای کاهش ابعاد و تعداد نمونه‌های شناخته شده اعم از بی خطر و بدافزار و برای افزایش سرعت و کاهش زمان پردازش است. به عبارت دیگر این دو غربال‌گر وظیفه مشکوک‌یابی و ارسال نمونه‌های ناشناخته به ماژول تشخیص‌گر را دارند.

نوآوری کاشف در ماژول تشخیص‌گر (با ۴ ماژول ایستا و رفتاری) بوده و در ارزیابی‌های این مرحله فقط از همین ماژول‌ها استفاده شده و ماژول‌های غربال‌گر در ارزیابی‌ها مشارکت داده نشده‌اند.

۵-۶-۱- مرحله اول: مقایسه ارزیابی روش تشخیص کاشف با روش‌های مشابه

در این مرحله، دو دسته بدافزار ۱۰۰ تایی به‌عنوان ورودی "کاشف" و ۱۰ محصول معتبر ضد بدافزار مورد آزمایش قرار می‌گیرند. دسته اول شامل بدافزارهایی است که در آن‌ها هیچ روش محافظتی استفاده نشده است. نتایج حاصل از این مرحله در جدول (۱۳) نشان داده شده است.

جدول (۱۳): مقایسه عملکرد کاشف با ۱۰ ضدبدافزار در شناسایی بدافزارهای محافظت نشده

روش تشخیص	نرخ تشخیص صحیح	نرخ تشخیص نادرست
اوست[1]	۸۳	۱۷
اویرا[2]	۷۸	۲۲
بیت‌دیفندر[3]	۹۰	۱۰
کومودو[4]	۹۳	۷
نورتن[5]	۸۸	۱۲
ویندوز دیفندر[6]	۸۶	۱۴
ClamAV	۷۴	۲۶
مکافی[7]	۸۳	۱۷
F-port	۷۷	۲۳
پادویش	۶۸	۳۲
کاشف	**۹۸**	**۲**

نتایج، تشخیص صحیح بدافزارهای شناخته شده را با نرخ ۹۸ درصد نشان می‌دهد. این نتایج در مقایسه با دیگر ضد بدافزارها در جدول (۱۳) نشان داده شده است. دو نمونه‌ای که توسط "کاشف" به‌عنوان بدافزار معرفی نشده‌اند، با میانگین احتمال بداندیشی ۵۵ درصد گزارش شده‌اند.

دسته دوم شامل بدافزارهایی است که در آن‌ها روش‌های محافظتی استفاده شده است. با توجه به ادعای "کاشف" در تشخیص بدافزارهای جدید و ناشناخته، برای ارزیابی آن، از بدافزارهای محافظت شده نیز استفاده شده است. استفاده از روش‌های محافظتی معمولاً برای FUD[8] کردن بدافزارها و با استفاده از ایجاد تغییرات ایستا و رفتاری در آن‌ها انجام می‌گیرد. در این مرحله، تعداد ۱۰۰ بدافزار که با استفاده از ابزار PELock محافظت شده‌اند به‌عنوان ورودی "کاشف" و ۱۰ محصول معتبر ضد بدافزار مورد آزمایش قرار می‌گیرند. نتایج حاصل از این

[1] Avast
[2] Avira
[3] Bitdifender
[4] Comodo
[5] Norton
[6] Windows Defender
[7] McAfee
[8] Fully Undetectable

مرحله در جدول (۱۴) نشان داده شده است.

جدول (۱۴): مقایسه عملکرد کاشف با ۱۰ ضد بدافزار در شناسایی بدافزارهای محافظت شده

روش تشخیص	نرخ تشخیص صحیح	نرخ تشخیص نادرست
اوست	۷۶	۲۴
اویرا	۹۲	۸
بیت‌دیفندر	۸۵	۱۵
کوموِدو	۸۸	۱۲
نورتن	۷۷	۲۳
ویندوز دیفندر	۸۲	۱۸
ClamAV	۶۲	۳۸
مکافی	۷۷	۲۳
F-port	۷۵	۲۵
پادویش	۶۲	۳۸
کاشف	**۹۴**	**۶**

۵-۶-۲- مرحله دوم: مقایسه ارزیابی روش تشخیص کاشف در برابر نرم‌افزارهای معتبر بی‌خطر

در این مرحله تعداد ۱۰۰۰ نمونه بی‌خطر آزمایش می‌شوند. پس از اجرای نرم‌افزارهای معتبر بی‌خطر در سامانه تشخیص "کاشف"، هیچ یک از نمونه‌ها، بدافزار تشخیص داده نشدند. میانگین درصد احتمال بداندیشی برای تعداد ۱۰۰۰ نمونه مذکور ۲۵ درصد به‌دست آمد. این میزان بداندیشی نیز به جهت وجود برنامه‌هایی است که از دید "کاشف" دارای رفتار بداندیش هستند. مانند مرورگر فایرفاکس که به جهت استفاده از توابع متعدد جهت برقراری ارتباط با اینترنت برای سامانه تشخیص "کاشف"، رفتاری بداندیش است.

۶- نتیجه‌گیری و پیشنهاد پژوهش‌های آینده

رشد روزافزون بدافزارها از یک طرف آن‌ها را به یکی از تهدیدات مهم حوزه فناوری اطلاعات تبدیل کرده است و از طرف دیگر، تشخیص بدافزارهای جدید را با چالش جدی مواجه کرده است. بنابراین روش‌های متنوعی نیز برای تشخیص بدافزارها ارائه شده است که هر کدام دارای مزیت‌ها و نقص‌هایی هستند. در این پژوهش، روشی سه لایه‌ای برای تشخیص بدافزارهای جدید ویندوزی ارائه شده است. روش پیشنهادی "کاشف" با استفاده از غربال‌گری چند سطحی، نرم‌افزارهای معتبر بی‌خطر و بدافزارهای شناخته شده را جداسازی می‌کند و کاهش میزان خطای سامانه تشخیص از مزایای این غربالگری است. "کاشف" با استفاده از ترکیب دو ماژول تشخیص‌گر ایستا و دو ماژول تشخیص‌گر رفتاری، بدافزارهای جدید را تشخیص می‌دهد. این روش، با محاسبه میزان بداندیش بودن نمونه‌های مشکوک، فرآیند تحلیل بدافزار را سرعت می‌دهد و باعث ایجاد سطحی از فرآیند خودکارسازی تحلیل بدافزار می‌شود. نتایج "کاشف" نشان می‌دهد که تشخیص بدافزارهای جدید ویندوزی با سرعت و دقت بیشتری نسبت به روش‌های مشابه انجام می‌گیرند. اضافه کردن ماژول‌هایی که بر اساس تشخیص رفتاری و مبتنی بر ناهنجاری، بدافزارها را تشخیص می‌دهند و همچنین توسعه و ایجاد سازوکاری جهت تعیین میزان شباهت بدافزارهای کشف شده به خانواده‌های بدافزار و برای بهبود عملکرد سامانه تشخیص "کاشف"، مواردی هستند که برای پژوهش‌های آینده پیشنهاد می‌شوند.

۷- مراجع

[1] M. Egele, T. Scholte, E. Kirda, and C. Kruegel, "A Survey on Automated Dynamic Malware-analysis Techniques and Tools," *ACM Comput. Surv.*, vol. 44, no. 2, pp. 1-42, 2012.

[2] R. Samani, C. Beek, and L. Scientist, "McAfee Labs Threats Report," August 2019.

[3] O. Aslan and R. Samet, "A Comprehensive Review on Malware Detection Approaches," *IEEE Access*, vol. 8. pp. 6249–6271, 2020.

[4] A. Mohanta, A. Saldanha, A. Mohanta, and A. Saldanha, Antivirus Engines, , A Comprehensive Approach to Detect and Analyze Modern Malware: 1st ed.Edition. Apress , 2020.

[5] P. A. Puranik, "Static Malware Detection using Deep Neural Networks on Portable Executables," August. 2019.

[6] "YARA Project, 2019;" [Online]. Available: https://en.wikipedia.org/wiki/Yara_International.

[7] "NSRL, 2018;" [Online]. Available: https://www.nist.gov/itl/ssd/software-quality-group/national-software-reference-library-nsrl.

[8] K. Raman, "Selecting Features to Classify Malware," *InfoSec Southwest 2012*, pp. 1–5, 2012.

[9] P. V. Shijo and A. Salim, "Integrated Static and Dynamic Analysis for Malware Detection," *Procedia Computer Science*, vol. 46, pp. 804–811, 2015.

[10] Z. Salehi, A. Sami, and M. Ghiasi, "MAAR: Robust Features to Detect Malicious Activity Based on API Calls, their Arguments and Return Values," *Engineering Applications of Artificial Intelligence*, vol. 59. pp. 93–102, 2017.

[11] Y. Fan, Y. Ye, and L. Chen, "Malicious Sequential Pattern Mining for Automatic Malware Detection," *Expert Syst. Appl.*, vol. 52, pp. 16–25, 2016.

[12] A. Susanto and A. Z. A. Munawar, "AHMDS: Advanced Hybrid Malware Detector System," *Proc. 2016 Int. Conf. Data Softw. Eng. ICoDSE*, 2017.

[13] S. S. Hansen, T. M. T. Larsen, M. Stevanovic, and J. M. Pedersen, "An Approach for Detection and

Family Classification of Malware Based on Behavioral Analysis," 2016 Int. Conf. Comput. Netw. Commun. ICNC 2016, 2016.

[14] H. S. Galal, Y. B. Mahdy, and M. A. Atiea, "Behavior-Based Features Model for Malware Detection," J. Comput. Virol. Hacking Tech., vol. 12, no. 2, pp. 59–67, 2016.

[15] J. Yonts, "Attributes of Malicious Files," sans institute, 2012.