
بسم الله الرحمن الرحيم

دانشکده مهندسی کامپیوتر و فناوری اطلاعات

ارایه روشی بر مبنای پردازش سیگنال جهت تحلیل آلودگی فایل‌های اجرایی

ابوالفضل سرکرده یی

استاد راهنما:

دکتر علی‌اکبر پویان

اساتید مشاور:

دکتر حمید حسن‌پور

مهندس جواد کیا

پایان نامه جهت اخذ درجه کارشناسی ارشد

بهمن ماه ۱۳۹۱

دانشگاه صنعتی شاهرود

دانشکده: مهندسی کامپیوتر و فناوری اطلاعات

پایان نامه کارشناسی ارشد آقای ابوالفضل سرکرده یی

تحت عنوان: ارایه روشی بر مبنای پردازش سیگنال جهت تحلیل آلودگی فایل‌های اجرایی

در تاریخ ۹۱/۱۱/۲۴ توسط کمیته تخصصی زیر جهت اخذ مدرک کارشناسی ارشد مورد ارزیابی و با درجه ..... مورد پذیرش قرار گرفت.

اساتید راهنما	امضاء	اساتید مشاور	امضاء
دکتر علی‌اکبر پویان		دکتر حمید حسن‌پور	
		مهندس جواد کیا	

اساتید داور	امضاء	نماینده تحصیلات تکمیلی	امضاء
دکتر مرتضی زاهدی		مهندس محسن فرهادی	
دکتر حسین مروی			

تقدیم به

پدر و مادر عزیزتر از جانم

و

برادر فداکارم

و

خواهران مهربانم

تشکر و قدردانی

پس از سپاسگذاری از درگاه خداوند منان ، بر خود لازم می دانم از استاد گرانقدر جناب آقای دکتر پویان که انجام این پایان نامه بدون کمک های ایشان امکان پذیر نبود و همچنین بپاس زحماتی که ایشان در طول بیش از ٦ سال برای اینجانب کشیدند، کمال تشکر و قدردانی را داشته باشم ؛

همچنین از اساتید محترم جناب آقای دکتر حسن پور که با راهنمایی هایشان در طول انجام پایان نامه کمک شایانی به من کردند و جناب آقای دکتر زاهدی سپاسگذارم ؛

در پایان از جناب آقای مهندس کیا که مرا در انجام این پایان نامه بسیار یاری رساندند تشکر می کنم.

تعهد نامه

اینجانب **ابوالفضل سرکرده یی** دانشجوی دوره کارشناسی ارشد رشته مهندسی کامپیوتر- گرایش هوش مصنوعی دانشکده مهندسی کامپیوتر دانشگاه صنعتی شاهرود نویسنده پایان‌نامه **ارایه روشی بر مبنای پردازش سیگنال جهت تحلیل آلودگی فایل‌های اجرایی** تحت راهنمائی آقای دکتر **علی-اکبر پویان** متعهد می‌شوم:

- تحقیقات در این پایان‌نامه توسط اینجانب انجام شده است و از صحت و اصالت برخوردار است .
- در استفاده از نتایج پژوهش‌های محققان دیگر به مرجع مورد استفاده استناد شده است .
- مطالب مندرج در پایان‌نامه تاکنون توسط خود یا فرد دیگری برای دریافت هیچ نوع مدرک یا امتیازی در هیچ جا ارائه نشده است .
- کلیه حقوق معنوی این اثر متعلق به دانشگاه صنعتی شاهرود می‌باشد و مقالات مستخرج با نام «دانشگاه صنعتی شاهرود» و یا «*Shahrood University of Technology*» به چاپ خواهد رسید.
- حقوق معنوی تمام افرادی که در به دست آمدن نتایج اصلی پایان‌نامه تأثیرگذار بوده‌اند در مقالات مستخرج از پایان‌نامه رعایت می‌گردد.
- در کلیه مراحل انجام این پایان‌نامه، در مواردی که از موجود زنده (یا بافتهای آنها) استفاده شده‌است، ضوابط و اصول اخلاقی رعایت شده‌است.
- در کلیه مراحل انجام این پایان‌نامه، در مواردی که به حوزه اطلاعات شخصی افراد دسترسی یافته یا استفاده شده‌است اصل رازداری، ضوابط و اصول اخلاق انسانی رعایت شده‌است.

تاریخ

امضای دانشجو

مالکیت نتایج و حق نشر

- کلیه حقوق معنوی این اثر و محصولات آن (مقالات مستخرج ، کتاب ، برنامه های رایانه ای ، نرم افزار ها و تجهیزات ساخته شده است) متعلق به دانشگاه صنعتی شاهرود می باشد . این مطلب باید به نحو مقتضی در تولیدات علمی مربوطه ذکر شود .
- استفاده از اطلاعات و نتایج موجود در پایان نامه بدون ذکر مرجع مجاز نمی‌باشد.

چکیده

با افزایش روزافزون بدافزارهای جدید و ناشناخته و همچنین استفاده از تکنیک های پنهان سازی در بدافزارها ارایه روش هایی جدید جهت شناسایی آن ها بسیار احساس می شود. در آنتی ویروس های تجاری از ترکیب روش های مبتنی بر علامت های هویتی و همچنین روش های مبتنی بر یادگیری جهت شناسایی بدافزارها استفاده می شود. با این وجود به دلیل مناسب نبودن روش های مبتنی بر یادگیری موجود در آنتی ویروس ها جهت تشخیص بدافزارهای جدید، این نرم افزارها برای تشخیص بدافزار بسیار به بروز بودن پایگاه داده وابسته هستند. در نتیجه امروزه بهبود روش های مبتنی بر یادگیری و ارایه روش های جدید در این زمینه بخش اصلی تحقیقات در تشخیص بدافزار را تشکیل می دهد.

شناسایی فایل های اجرایی آلوده شده با کدهای مخرب یکی از اصلی ترین کارهای یک آنتی ویروس می باشد. که برای تشخیص کدهای مخرب ناشناخته و جدید از روش های مبتنی بر یادگیری استفاده می شود. هدف از انجام این پایان نامه تلاش جهت بدست آوردن یک الگوی سیگنالی از فایل های اجرایی می باشد تا بتوان با استفاده از آن الگو و تکنیک های پردازش سیگنال یک روش تشخیص فایل های اجرایی آلوده ارایه کرد.برای این کار ابتدا به بررسی روش های مختلف تحلیل فایل اجرایی پرداخته و سپس روش مناسبی که از تحلیل قسمت منبع فایل اجرایی بدست می آید معرفی می شود. جهت اثبات کارا بودن روش معرفی شده، براساس آن یک بردار ویژگی ۲۵۶ تایی از تکرار ترکیب بایت ها در قسمت آخر فایل اجرایی بدست می آید و با استفاده از تکنیک های دسته بندی متن روشی جهت تشخیص فایل اجرایی آلوده جدید، با دقت ۹۹.۱۰ درصد ارایه می شود.

کلمات کلیدی: فایل آلوده، کد مخرب ناشناخته، آنتی ویروس، پردازش سیگنال، تشخیص فایل آلوده.

لیست مقالات مستخرج از پایان نامه

۱. ابوالفضل سرکرده یی، علی‌اکبر پویان، حمید حسن‌پور، " ارایه روشی هوشمند بر مبنای دسته بندی متن جهت تشخیص کدهای مخرب ناشناخته". یازدهمین کنفرانس سیستم های هوشمند ایران، دانشگاه خوارزمی اسفند۱۳۹۱.

فهرست مطالب

فهرست اشکال

فهرست جداول

فصل اول

۱ مقدمه

۱-۱ مقدمه

با افزایش روز افزون بدافزارها و همچنین استفاده از تکنیک های مختلف جهت پیچیده تر کردن آن ها نیاز به روش های هوشمند برای شناسایی بدافزارها بسیار احساس می شود به طور کلی می توان نرم افزار های کامپیوتری را به دو دسته بدافزار[۱] و خوب افزار[۲] تقسیم کرد. منظور از خوب افزار همه نرم افزارهایی می باشد که توسط کاربر جهت کاربردی خاص بروی کامپیوتر مورد استفاده قرار گرفته و بدون اجازه کاربر توانایی اجرا نداشته باشد و در صورت اجرا هیچ گونه فعالیت خرابکارانه ای را انجام ندهند. از سوی دیگر بدافزار به نرم افزاری گفته می شود که بدون اجازه کاربر اجرا شده و فعالیت های خرابکارانه ای را انجام می دهد که این فعالیت ها می تواند شامل سرقت اطلاعات شخصی افراد، از بین بردن اطلاعات، تحت کنترل گرفتن سیستم کاربر، اخلال در عملکرد سیستم کاربر و... باشد.

به صورت کلی می توان گفت یک بدافزار مجموعه ای از کدها است که جهت انجام فعالیت مخرب در سیستم کاربر اجرا می شود.یکی از روش‌هایی که منجر به اجرای کدهای مخرب[۳] در سیستم کاربر می-شود اجرای این کدها در قالب یک فایل اجرایی آلوده[۴] می‌باشد، به عبارت دیگر آن فایل اجرایی با آن کدهای مخرب آلوده شده و تبدیل به یک فایل اجرایی آلوده شده است.

در نرم افزارهای آنتی ویروس[۵] شناسایی فایل های اجرایی آلوده یا همان بدافزارهایی که از نوع یک فایل اجرایی می باشند به عنوان بخش مهمی از فعالیت های یک آنتی ویروس در نظر گرفته می شود . آنتی ویروس های موجود در شناسایی بدافزارهای جدیدی که قبلا گزارش فعالیت آن ها ارایه نشده است

---

۱ Malicious Software(Malware)

۲ Good Software

۳ Malicious Code

۴ Infected File

۵ Anti Viruss

دچار مشکل هستند. همچنین شناسایی نمونه[۱]های جدید بدافزار های قدیمی با دقت خوبی انجام نمی شود.در ادامه فصل به دسته بندی بدافزارها می پردازیم.

۱-۲ دسته بندی انواع بدافزارها

بدافزارها را می توان براساس نحوه عملکرد آن ها به دسته[۲] های مختلفی تقسیم کرد که هر کدام از دسته ها شامل نمونه[۳] های مختلفی می شود که ممکن است شامل زیر نمونه[۴] هایی نیز باشد. در ادامه مهمترین دسته ها و نمونه ها و زیر نمونه ها معرفی می شود.

۱-۲-۱ویروس[۵]

ویروس بدافزاری است که با تغییر دادن کد یک فایل اجرایی و اضافه کردن کدهای مخرب ، آن فایل را آلوده می کند و زمانی که آن برنامه اجرا شود، مجموعه کدهای مخرب نیز اجرا خواهد شد. این دسته از بدافزارها به نمونه های مختلفی تقسیم می شوند که در ادامه به توضیح آن ها می پردازیم.

۱-۲-۱-۱ ویروس فایل[۶]

در این نمونه، از فایل‌های سیستمیِ سیستم عامل جهت منتشر و توزیع کردن کدهای مخرب استفاده می شود. این نمونه شامل این مواردمی شود: ویروس هایی که از فایل های اجرایی به عنوان میزبان[۱]

---

[۱] Variant

[۲] Category

[۳] Type

[۴] Subtype

[۵] Virus

[۶] File Virus

استفاده کرده و آن ها را آلوده می کنند و یا یک کپی از یک فایل می سازند که این کپی نسخه ای از آن ویروس می باشد و یا ویروس هایی که خود را در مسیرهای مختلف کپی کرده و به ویروس هایی که فایل های سیستمی را آلوده کرده اند لینک می شوند.ویروس اسکریپت[۲] یک زیر مجموعه از این نوع ویروس ها می باشد که با یکی از زبان های اسکریپتی نوشته می شود.

۱-۲-۱-۲ ویروس بوت سکتور[۳]

این نمونه، سکتور بوت یا یک رکورد بوت را آلوده می کند و یا سکتور بوت فعال را تغییر می دهد و زمانی که کامپیوتر بوت می شود ویروس نیز فعال می شود. تعداد زیادی از این نمونه ویروس زمانی که یکبار اجرا شوند مانع بوت شدن کامپیوتر می شوند.

۱-۲-۱-۳ ویروس ماکرو[۴]

این نمونه با زبان ماکرو اسکریپتی نرم افزار های کاربردی مانند پردازش متن، حسابداری و... نوشته می شود که با تغییر دادن ویژگی های این نوع زبان ها از یک فایل آلوده به یک فایل سالم انتقال پیدا می کنند. اغلب این نوع ویروس ها برای نرم افزار های مجموعه آفیس[۵] وجود دارندو به دلیل اینکه این نوع ویروس ها با زبان یک نرم افزار کاربردی نوشته شده اند بروی هر بستر[۶] که آن نرم افزار اجرا شوند آن ویروس می تواند فعالیت کند به همین دلیل این نمونه از ویروس ها به بستر وابسته نیستند.

---

۱ Host

۲ Script Virus

۳ Boot Sector Virus

۴ Macro Virus

۵ Microsoft Office

۶ Platform

۱-۲-۱-۴ ویروس ایمیل[۱]

این نمونه برخلاف موراد قبلی که براساس نحوه آلوده سازی تقسیم بندی شده اند براساس نحوه توزیع ویروس نام گذاری شده است. ایمیل می تواند برای انتقال همه نمونه های قبلی استفاده شود و زمانی که یکی از افرادی که این ایمیل را دریافت کرده اند فایل ضمیمه را اجرا کنند این ویروس از طریق ایمیل آن فرد به آدرس هایی که آن فرد ذخیره کرده است ارسال می شود.

۱-۲-۱-۵ ویروس چند نوعی[۲]

در این نمونه هسته ویروس ها یکسان بوده ولی تغییراتی کمی در پیاده سازی اعمال شده است تا براحتی توسط آنتی ویروس ها شناسایی نشوند.

۱-۲-۲ کرم شبکه

یک بدافزار خود پخش[۳] است که بروی شبکه توزیع می شود و اغلب از طریق اینترنت گسترش پیدا می کند. برخلاف ویروس برای اجرا شدن، تکرار شدن و توزیع شدن به برنامه های دیگر و انجام کارهایی مانند باز کردن ایمیل آلوده وابسته نیست. کرم ها جهت یافتن حفره های امنیتی و نقاط آسیب پذیر در سیستم شبکه را اسکن کرده و با استفاده از آن ها پخش می شوند و خود را در سیستم های آسیب پذیر کپی می کنند. کرم ها اغلب براساس اینکه از چه محیطی برای پخش استفاده می کنند دسته بندی می شوند در برخی موارد نیز براساس سرعت پخش کرم ها نیز این دسته بندی انجام می شود که در ادامه نمونه های مختلف کرم ها را بررسی می کنیم:

---

[۱] Email Virus

[۲] Multi-Variant Virus

[۳] Self-Propagate

کرم ایمیل که با آلوده کردن ضمیمه ایمیل پخش می شود و همچنین کرم هایی که با استفاده از کانال IM[۱] و یا IRC[۲] خود را پخش می کنند و به همین نام شناخته می شوند. کرم اینترنتی نیز که با استفاده پروتکل های وب مانند FTP[۳] و یا استفاده از صفحات وب و آلوده کردن سیستم هایی افرادی که به آن صفحه مراجعه می کنند، خود را پخش می کند همچنین کرم هایی نیز براساس سرعت پخش شدن دسته بندی می شوند به عنوان مثال کرم وارول[٤] که در کمتر از ۱۵ دقیقه سرورهای آسیب پذیر موجود در اینترنت را آلوده می کند . همچنین ممکن است که یک کرم با بکارگیری تعداد زیاد کرم دیگر فعالیت خود را گسترش دهد مانند کرم سوارم [٥] که به صورت هوشمند با تعداد زیادی کرم همکاری می کند.

یکی از تفاوت‌هایی که بین ویروس و کرم وجود دارد این است که ویروس های ابتدایی خود را به صورت فایل به فایل و با تکنیک تکرار فایل منتشر می کردند اما کرم ها از طریق اینترنت وبه صورت میزبان به میزبان منتشر می کردند انتشار یک کرم می تواند از طریق یک لینک مهندسی اجتماعی شده که در یک ایمیل قرار دارد باشد که می تواند نقاط آسیب پذیر میزبان را شناسایی کند.

۱-۲-۳ اسب تروا(تروجان)[٦]

یک بدافزار است که در قالب یک نرم افزار خوب و بدون خطر قرار گرفته است و پس از آن که تروجان برای اولین بار اجرا شد کنترل نرم افزار به مهاجم داده می شود و مهاجم می توانند عملیات خود را انجام

---

[۱] Instant Message

[۲] Internet Relay Chat

[۳] File Transfer Protocol

[٤] Warhol Worm

[٥] Swarm Worm

[٦] Trojan Horse

دهد.. برای جلوگیری از شناسایی تروجان ها از تکنیک Process injection استفاده می شود که در آن پروسس تروجان با فعال شدن یک پروسس به ظاهر سالم فعال می شود و بدین ترتیب پروسس تروجان براحتی شناسایی نمی شود. در ادامه به بررسی نمونه های مختلف تروجان می پردازیم.

۱-۲-۳-۱ تروجان در پشتی[۱]

نام دیگر این نوع تروجان دسترسی از راه دور نیز می باشد این نمونه یک دسترسی از راه دور بروی سیستم آلوده شده[۲] برای مهاجم فراهم می کند .برای مثال از این نمونه می توان تروجان DoS مطرح کرد که اگر گسترش و پراکندگی این تروجان مناسب باشد می توان از آن به عنوان بستری برای حمله های DDoS[۳] استفاده کرد.

۱-۲-۳-۲ تروجان جمع آوری اطلاعات[۴]

در این نمونه اطلاعات به صورت پنهانی از سیستم آلوده شده جمع آوری و برای مهاجم ارسال می شود به این نوع تروجان Stealth ware گفته می شود.Spyware هم از این نمونه است که به آن Spybot نیز گفته می شود در این نمونه، تروجان به صورت مخفی جهت جمع آوری اطلاعات بر روی یک سیستم نصب می شود. Key logger نیز از این نمونه است که خود را در یک نرم افزار مرورگر و یا به عنوان درایور یک دستگاه خود را نصب می کند و داده هایی که با استفاده از کیبورد وارد می‌شود را کنترل می کند و اطلاعات آن را برای مهاجم ارسال می کند. Notifier نیز از این نمونه است که وظیفه‌ی

---

[۱] Backdoor Trojan

[۲] Infected System

[۳] Distributed Denial of Service

[۴] Data-collection Trojan

تایید کردن آلود شدن موفق سیستم را دارد واطلاعات مانند IP سیستم و پورت های باز آن را ارسال می کند.

۱-۲-۳-۳ Rootkit

مجموعه ای از برنامه هاست که مهاجم جهت دسترسی به یک سیستم و پنهان ماندن و شناسایی نشدن استفاده می کند. این نمونه جهت پنهان کردن پروسس ها، فایل ها و اطلاعات رجیستری طراحی شده اند. به عبارت دیگر مهاجم جهت پنهان کردن خود و جلوگیری از ردیابی و همچنین پنهان کردن فعالیت هایی که انجام می دهد از این نوع تروجان استفاده می کند. دو نوع Rootkit وجود دارد که براساس نحوه نصب rootkit دسته بندی شده اند: ا- روتکیت مد هسته ۲- روتکیت مد کاربر. روتکیت مد هسته به این دلیل که براحتی پنهان می شود و قوی تر است بیشتر مورد استفاده قرار می گیرد.

۱-۲-۳-۴ Bot و Botnet

لغت Bot از کلمه Robot مشتق شده است به معنی کارگر و کسی که کار می‌کند.در کامپیوتر برای یک پروسس اتوماتیک نیز از این عبارت استفاده می‌شود Botها با اهداف زیادی در اینترنت فعالیت می کنند گستره کاربرد Botها از موتورهای جستجو گرفته تا Game Botsو IRC Channel Botها می.باشد

می توان گفتBot ها یک فایل اجرایی هستند که در معرض خطر قرار می دهند، کنترل می‌کنند و میزبان را عضو Botnetها می‌کنند. اینگونه نرم افزارها بدون اطلاع کاربر بروی سیستم آن‌ها نصب می شودBot ها می‌توانند مانند ویروس‌ها مخفی ومانند کرم‌ها در شبکه پخش شوند.

Bot ها سیستم ها را آلوده می‌کنند و کنترل آن سیستم را به مهاجم[۱] می دهند تا از راه دور بتواند سیستم را کنترل کند و با ارسال دستورات برای Bot موجود در سیستم آلوده طبق دستورات به اهداف خویش برسد به این سیستم های آلوده Zombie نیز گفته می شود.

۳-۱ تعریف مسئله

در این پایان نامه هدف ارایه الگوی سیگنالی از فایل اجرایی جهت تشخیص فایل اجرایی آلوده از فایل اجرایی سالم است که منظور ما -در این پایان نامه- از فایل اجرایی، فایل اجرایی قابل حمل[۲] ۳۲ بیتی ویندوز می باشد. به عبارت دیگر می خواهیم برای هر فایل اجرایی یک الگوی سیگنالی ارایه کنیم تا براساس آن آلودگی فایل اجرایی به کد مخرب را تشخیص دهیم. این کد مخرب می تواند فعالیت های مخرب مختلفی مانند حذف یا خراب کردن فایل های سیستمی، دزدیدن اطلاعات کاربر، در اختیار مهاجم قرار دادن سیستم کاربر و ... باشد. لازم به ذکر است که فرض ما بر این است که فایل های اجرایی به صورت غیر فشرده[۳] می باشند و همچنین فایل اجرایی باید به صورت ایستا تحلیل شود و آلوده بودن آن به صورت ایستا تشخیص داده شود. تاکنون روشی بر مبنای پردازش سیگنال جهت تشخیص فایل اجرایی آلوده ارایه نشده است و این پایان نامه می تواند آغازی در ارایه روش های تشخیص آلودگی فایل اجرایی بر مبنای پردازش سیگنال باشد.

---

۱ Bot Master

۲ Portable Executable (PE32)

۳ Unpack

۱-۴ مقایسه فایل اجرایی سالم و آلوده

همانطور که گفته شد منظور از آلودگی یک فایل اجرایی این است که آن فایل شامل کدهای مخرب بوده که بدون ایجاد خلل در فعالیت های فایل میزبان آن کدها اجرا شده و به سیستم آسیب می زنند. جهت شناسایی فایل سالم از فایل آلوده می بایست تفاوت های موجود بین این دو نوع فایل اجرایی بررسی شده و باتوجه به این تفاوت ها راهکارهایی جهت شناسایی فایل های آلوده از فایل سالم ارایه کرد.

۱-۴-۱اطلاعات سرآیند[۱] فایل اجرایی

در سرآیند فایل اجرایی اطلاعاتی مانند اندازه فایل، تعداد قسمت ها ، اطلاعات کنترلی و... وجود دارد که این اطلاعات محتوای کلی آن فایل را بیان می کند. پس از آنکه کد مخرب یک فایل سالم را آلوده کرد اطلاعاتی مانند اندازه آن فایل نیز تغییر می کند ولی اطلاعاتی که در سرآیند فایل آلوده وجود دارد همان اطلاعات فایل سالم است. به عبارت دیگر در فایل آلوده اطلاعات سرآیند با محتوی فایل مطابقت ندارد.

به عنوان مثال در فایل سالم اندازه فایل با مقداری که در سرآیند به عنوان اندازه فایل آمده است یکسان است اما در فایل های آلوده در بدافزارهایی که پس از ایجاد تغییرات در محتوای فایل سالم و آلوده کردن آن به کدهای مخرب اطلاعات سرآیند را تغییر نمی دهند بین مقادیر موجود در سرآیند فایل و مقادیر واقعی فایل تفاوت وجود دارد که براساس این تفاوت می توان فایل آلوده را شناسایی کرد یعنی اگر در فایلی اطلاعات سرآیند با محتوی فایل متفاوت باشد آن را به عنوان فایل اجرایی آلوده در نظر می گیریم.

---

[۱] Header

لازم به ذکر است این تفاوت بین فایل سالم و آلوده بسیار رایج بوده و یکی از روش های شناسایی فایل های آلوده وجود این مقادیر متفاوت بوده است اما می توان گفت در بدافزارهای امروزی این عدم تطابق وجود ندارد. به عبارت دیگر پس از آلوده کردن فایل، اطلاعات سرآیند را باتوجه به تغییراتی که فایل رخ داده است تغییر می دهند.

۱-۴-۲ تغییرات اشاره گر دستورالعمل[۱]

تفاوت مهم دیگر بین فایل سالم و آلوده تغییرات اشاره گر دستورالعمل می باشد. در فایل سالم تغییرات این اشاره گر نرمال بوده و تغییرات شدیدی ندارد و از ابتدا تا انتها یک روند دارد. اما در فایل های آلوده به این دلیل که کدهای مخرب به فایل سالم اضافه می شود تغییرات این اشاره گر نرمال نیست. به عنوان مثال معمولا این اشاره گر در فایل آلوده ابتدا به انتهای فایل یعنی نقطه شروع کدهای مخرب پرش[۲] کرده و پس از اجرای آن دستورالعمل ها دوباره به شروع برنامه اصلی برگشته و آن ها را اجرا می کند به عبارت دیگر در فایل های آلوده در ابتدا اشاره گر دستورالعمل به انتهای فایل پرش کرده و سپس به شروع برنامه اصلی بر می گردد که این مورد اصلی ترین تفاوت در تغییرات اشاره گر بین فایل اجرایی آلوده و سالم می باشد که می توان از آن جهت تشخیص فایل سالم از آلوده استفاده کرد.

---

۱ Instruction Pointer(IP)

۲ Jump

۱-۴-۳ تفاوت در قسمت[۱] منبع[۲]

فایل اجرایی از قسمت های مختلفی تشکیل شده است که پس از آلوده شدن یک فایل سالم تغییرات در قسمت منبع محسوس تر است.این قسمت که در انتهای فایل اجرایی قرار دارد در فایل سالم شامل اطلاعات متنی از قبیل توضیحات فایل اجرایی، آیکون، ورژن ، توضیحات کامپایلر و... می باشد و بخش اصلی آن را کاراکترهای متنی تشکیل می دهد.

برای آنکه یک فایل آلوده شود باید کدهای مخرب به آن فایل اضافه شوند بدین منظور کدهای مخرب باید به گونه ای به فایل سالم اضافه شوند که فایل میزبان خراب نشود و پس از آلوده شدن نیز به درستی اجرا شود. بدین منظور کدهای مخرب معمولا در انتهای فایل اجرایی یعنی قسمت آخر آن قرار می گیرند و با تغییر در شروع اجرای فایل ابتدا کد مخرب اجرا شده و پس از کد اصلی برنامه اجرا می شود. به طور کلی می توان گفت در فایل آلوده همواره بخشی از کدهای مخرب در این قسمت از فایل آلوده قرار دارند و براساس آن می توان با تحلیل این قسمت به روشی جهت شناسایی فایل آلوده رسید.

---

[۱] Section

[۲] Resource (.rsrc)

فصل دوم

۲ بررسی روش های تشخیص آلودگی فایل های اجرایی

۲-۱ مقدمه

جهت شناسایی یک فایل اجرایی آلوده در بین فایل های اجرایی سالم روش های مختلفی ارایه شده است که می توان این روش ها را به دو دسته کلی تقسیم کرد:

۱-روش های مبتنی بر علامت های هویتی[۱] ۲-روش های مبتنی بر یادگیری و تخمین[۲]

در دسته اول همانطور که از نام آن مشخص است روش ها براساس علامت[۳] ها و نشان هایی که در فایل ها آلوده وجود دارد کار می کنند و اگر در فایلی آن علامت وجود داشته باشد آن فایل به عنوان بدافزار در نظر گرفته می شود. به عبارت دیگر در این دسته برای هر نوع بدافزار یک علامت یا نشان در نظر گرفته شده و هر فایلی که آن علامت را داشته باشد یک بدافزار از آن نوع شناخته می شود و اگر فایلی هیچ کدام از علامت های انواع مختلف بدافزارها را نداشته باشد به عنوان یک فایل سالم در نظر گفته می شود. نقطه ضعف این روش این است که قدرت شناسایی بدافزارهای جدید را ندارد.

اما در دسته دوم، روش ها براساس ویژگی های کلی که در اکثر بدافزارها وجود دارد فایل های آلوده را شناسایی می کنند به عبارت دیگر در این دسته، روش ها به دنبال الگوهایی می گردند که در اکثر بدافزارها وجود داشته باشد و سپس هر فایلی که آن الگو را داشته باشد را به عنوان فایل آلوده در نظر می گیرند همچنین در این روش ها اگر فایلی الگویی شبیه الگوی فایل آلوده داشته باشد به عنوان یک فایل مشکوک به آلوده بودن شناخته می شود.

در آنتی ویروس های امروزی برای شناسایی فایل های آلوده از هر دو دسته روش مذکور به صورت ترکیبی استفاده می شود و روش های دسته دوم تلاش می کنند ضعف موجود در دسته اول یعنی همان

---

[۱] Signature Base

[۲] Heuristic

[۳] Tag

عدم شناسایی بدافزارهای جدید را پوشش دهند. در ادامه به بررسی این دو دسته روش می پردازیم و سپس بطور کلی این دو دسته روش را باهم مقایسه می کنیم.

۲-۲ روش‌های مبتنی بر علامت‌های هویتی

همانطور که در ابتدا گفته شد نکته اصلی در این روش ها بدست آوردن علامت های هویتی[۱] بازای هر نوع بدافزار می باشد که این علامت ها در پایگاه داده ای ذخیره شده و برای بررسی هر فایل به این پایگاه داده مراجعه می شود و در فایل ورودی علامت های موجود در پایگاه داده جست و جو می گردد. اگر هر یک از علامت ها در فایل پیدا شود آن فایل به عنوان فایل آلوده در نظر گرفته می شود. می توان گفت اصلی ترین چیزی که قدرت تشخیص این دسته از روش ها را تحت تاثیر قرار می دهد بروز بودن پایگاه داده است. نکته دیگر درمورد این روش ها نحوه بدست آوردن علامت های هویتی است که به آن نحوه تحلیل بدافزارها نیز گفته می شود. در واقع با این روش به دنبال نشان ها و علامت هایی می گردیم که در صورت مشاهده آن علامت ها در فایل های دیگر آن فایل ها را به عنوان بدافزار بشناسیم که بعد از بدست آوردن این علامت ها، می بایست آن ها را به پایگاه داده اضافه کرد تا سیستم بتواند بدافزارهایی از آن نوع را شناسایی کند.

۲-۲-۱ نحوه تحلیل بدافزارها

برای آنکه روش های این دسته بتوانند بدافزارهای بیشتری را شناسایی کنند می بایست پس از شروع فعالیت بدافزارهای جدید، آن بدافزارها را تحلیل کرد. همانطور که گفته شد منظور از تحلیل بدافزارها

---

[۱] Signature

نحوه بدست آوردن علامت های هویتی برای بدافزارها می باشد به دو روش می توان بدافزارها را تحلیل کرد که در ادامه به بررسی این روش ها می پردازیم.

۲-۲-۱-۱ تحلیل ایستا[۱]

در این روش جهت بدست آوردن علامت های بدافزارها مجموعه دستورالعمل های تشکیل دهنده یک بدافزار مورد تحلیل قرار می گیرد. در واقع برای تحلیل یک بدافزار به بررسی دستورالعمل های آن پرداخته و وجود مجموعه دستورالعمل ها در یک فایل را به عنوان یک علامت برای آلوده بودن آن فایل در نظر گرفت. که این کار برای انواع بدافزارها انجام شده و برای هر نوع از بدافزارها مجموعه ای از دستورالعمل ها به عنوان علامت در نظر گرفته شده و وجود آن دستورالعمل ها در جای مشخص در یک فایل، نشان دهنده آلوده بودن فایل می باشد.

۲-۲-۱-۲ تحلیل پویا[۲]

در این روش هدف بررسی رفتاری بدافزارها می باشد. به عبارت دیگر برای بدست آوردن علامت مشخص که نشان دهنده آلوده بودن یک فایل باشد فعالیت هایی که یک فایل در زمان اجرا انجام می دهد مورد بررسی قرار می گیرد. برای بدست آوردن علامت های هویتی ابتدا بدافزار در یک محیط امن اجرا می شود سپس فعالیت های بدافزار در زمان اجرا زیرنظر گرفته می شود حال در بین فعالیت های حین اجرای یک بدافزار به دنبال رفتاری می گردیم که می توان آن را به عنوان یک علامت درنظر گرفت و گفت که اگر فایل دیگری چنین رفتار مشابه ای را داشته باشد آن فایل نیز یک بدافزار است این علامت ها نیز به پایگاه داده اضافه خواهد شد.

---

۱ Static Analysis

۲ Dynamic Analysis

۲-۲-۲ نقاط ضعف

در روش های مبتنی بر علامت های هویتی روش کار به این شکل است که با مراجعه به پایگاه داده یک علامت انتخاب شده و در فایل مورد نظر در جای مشخص شده آن علامت جست و جو می‌گردد. در صورت وجود هر یک از علامت های موجود در پایگاه داده در فایل مورد بررسی، آن فایل به عنوان یک بدافزار شناخته می شود. می توان نتیجه گرفت اگر فایل مورد بررسی یک بدافزار جدید باشد که قبلا فعالیت نکرده باشد به این دلیل که علامت های این فایل بدست نیامده است این فایل سالم در نظر گرفته می شود. همچنین اگر علامت های بدافزار جدید بدست آمده باشد اما در پایگاه داده وجود نداشته باشد باز هم این فایل سالم در نظر گرفته نمی شود.

می توان گفت این دسته از روش ها برای تشخیص بدافزار هایی که علامت آن ها بدست آمده است کاملا به بروز بودن پایگاه داده وابسته است و حتی در مورد بدافزارهایی که نوع[۱] جدیدی از بدافزارهای قدیمی- که علامت آن ها در پایگاه وجود دارد- هستند این دسته از روش ها با مشکل مواجه می شوند. همچنین توانایی تشخیص بدافزارهای جدید در این دسته از روش ها وجود ندارد پس نتیجه می گیریم که با استفاده از روش های این دسته هرگز نمی توان به قدرت تشخیص ۱۰۰ درصد رسید به همین دلیل در آنتی ویروس ها همراه با روش های این دسته از روش های مبتنی بر یادگیری و تخمین استفاده می شود تا بتواند قدرت تشخیص را در آنتی ویروس ها افزایش دهد. اما امروزه با وجود استفاده از روش های مبتنی بر یادگیری و تخمین هنوز دقت آنتی ویروس ها به نقطه مطلوب نرسیده است به عنوان نمونه براساس آزمایشی که سایت[۲] مقایسه نرم افزارهای آنتی ویروس در سال ۲۰۰۸ انجام داده است نشان داد که قدرت تشخیص ۱۶ آنتی ویروس مطرح امروزی که پایگاه داده آن ها بروز رسانی نشده بود حداکثر به

---

[۱] Variant

[۲] www.av-comparatives.org

۷۰ درصد رسیده است. پس می توان نتیجه گرفت جهت افزایش دقت آنتی ویروس ها می بایست بر روی بهبود روش های مبتنی بر یادگیری و تخمین کار کرد. در ادامه به بررسی این دسته از روش ها می پردازیم.

۲-۳ روش‌های مبتنی بر یادگیری و تخمین

در این دسته از روش ها تلاش می شود با استفاده از روش های اتوماتیک، بدافزارهایی که توسط روش های مبتنی بر علامت های هویتی قابل شناسایی نیستند را تشخیص داد به عبارت دیگر این دسته از روش ها به عنوان مکمل دسته قبلی جهت رسیدن قدرت تشخیص به ۱۰۰ درصد مورد استفاده قرار می گیرند. برای این کار می بایست در بین بدافزارها به دنبال الگو[۱] هایی بگردیم که در آن ها مشترک باشد.می توان روش های موجود در این دسته را براساس نحوه تحلیل وبررسی فایل اجرایی به چند زیر دسته کلی تقسیم کرد که در ادامه به بررسی آن ها می پردازیم.

۲-۳-۱ تحلیل براساس n بایت استخراج شده از فایل اجرایی[۲]

در این دسته، فایل اجرایی به صورت یک فایل متنی که از ۰ و ۱ تشکیل شده است مورد بررسی قرار می گیرد به عبارت دیگر ما مجموعه ای از فایل های اجرایی را به صورت فایل های متنی متشکل از ۰ و ۱ تبدیل کرده و سپس صورت مسئله مانند دسته بندی فایل های متنی می شود. فایل های متنی ما به دو دسته سالم و آلوده تقسیم می شود. در دسته بندی متون، کلمه به عنوان ویژگی اصلی مورد استفاده قرار می گیرد که در اینجا دنباله n بایت ها مانند کلمه در دسته بندی متون در نظر گرفته می شود.

---

[۱] Pattern

[۲] Byte N-gram

سپس می توان الگوریتم های دسته بندی مانند K-NN و روش های وزن دهی که در دسته بندی متون استفاده می شود را بروی این مسئله اعمال کرد. نکته ای که در اینجا حایز اهمیت است این است که در این روش ها فایل اجرایی به صورت معنی دار بررسی نمی شود. به عبارت دیگر محتوای فایل اجرایی و اینکه مثلا از چه دستورالعمل هایی در آن فایل اجرایی استفاده شده است بررسی نمی شود و فقط از دنباله ای از ۰ و ۱ ها استفاده می شود. می توان انتظار داشت که این روش به دلیل اینکه به صورت معنی داری با فایل اجرایی برخورد نمی کند دارای سرعت خوبی باشد. به عنوان نمونه در [۱] از روش n بایت به عنوان روش استخراج ویژگی استفاده شده و برای دسته بندی نیز روش نزدیک ترین همسایه[۱] مورد استفاده قرار گرفته است؛ پایگاه داده ای که این روش بر روی آزمایش شد مشتمل بر ۶۵ بدافزار و فایل سالم بود که با دقت[۲] ۹۸درصد فایل های آلوده مشخص شدند. در [۲] نیز از همین روش برای استخراج ویژگی استفاده شده است.

۲-۳-۲ براساس اطلاعات سرآیند فایل اجرایی[۳]

در این دسته همانطور که از نام آن پیدا است شناسایی فایل های آلوده براساس اطلاعات موجود در سرآیند فایل انجام می شود.ازاطلاعات موجود در سرآیند از قبیل اطلاعات فیزیکی مثل زمان ایجاد فایل، سایز فایل و... و همچنین اطلاعات ساختار منطقی مثلا اطلاعات مربوط به قسمت ها و... و اطلاعات مربوط به DLL ها توابع استفاده شده در برنامه و موارد دیگر جهت شناسایی فایل های اجرایی آلوده می توان استفاده کرد. به عنوان مثال زمانی که فایل اجرایی سالم آلوده می شود کد های مخرب به انتهای فایل سالم اضافه شده و اندازه فایل زیاد می شود. نکته ای که باعث تشخیص فایل آلوده می شود این

---

[۱] K-NN

[۲] Accuracy

[۳] Portable Executable Header Information

است که اندازه فایل زیاد شده ولی اطلاعات درون سرآیند که شامل اندازه فایل می شود تغییری نکرده است که این تفاوت منجر به شناسایی فایل اجرایی آلوده می شود.

۲-۳-۳ براساس کد عملگر فایل اجرایی[۱]

در زبان ماشین دستورالعمل ها شامل کد عملگر[۲] و یک یا بیشتر عملوند[۳] می شوند که براساس کد عملگر های موجود در فایل اجرایی می توان به آلودگی آن پی برد. در این دسته از روش ها مانند دسته قبل فایل اجرایی به صورت معنی دار مورد بررسی قرار می گیرد. در این دسته کد عملگراستفاده شده در فایل اجرایی بررسی می شود. به عبارت دیگر از فایل اجرایی کد عملگرهای استفاده شده در آن را استخراج می کنیم. در ادامه می توانیم بر اساس روش های مختلف آن فایل را تحلیل کنیم. مثلا مجموعه کدهای عملگر فایل اجرایی را به عنوان یک فایل متنی در نظر گرفته و در ادامه مانند دسته بندی متون عمل کنیم. یکی از کارهایی که می توان انجام داد این است که کد عملگر را به عنوان کلمه در نظر گرفته و دنباله ای از n کلمه(کد عملگر) را به عنوان ویژگی در نظر بگیریم و سپس الگوریتم های دسته بندی را روی آن ها اعمال کنیم.

به عنوان نمونه در [۳] با استفاده از دنباله دستورالعمل های مفید برای تشخیص که از روش های اتوماتیک بدست آمده بودند با دقت ۹۸.۴ درصد فایل های آلوده شناسایی شدند. همچنین در [۴] نحوه توزیع تکرار کد عملگر ها در فایل های آلوده و سالم مورد تحلیل قرار گرفت و این نتیجه حاصل شد که توزیع کد عملگر ها (بخصوص کد عملگرهای کم کاربرد) در فایل آلوده نسبت فایل سالم بسیار متفاوت

---

۱ OpCode Information

۲ Opcode(Operational Code)

۳ Operand

است و در [۵] نیز با استفاده از n کد عملگر[۱] به عنوان ویژگی و بررسی ۵ روش دسته بندی و آزمایش بروی مجموعه ای با بیش از ۳۰۰۰۰ فایل سالم و آلوده دقت ۹۹ درصد بدست آمده است. نکته ای در این دسته حایز اهمیت است این است که در برخی موارد بخصوص در فایل های آلوده نمی توان به طور کامل به دستورالعمل های تشکیل دهنده آن فایل اجرایی را بدست آورد و به عبارت دیگر نمی توان گفت همواره دستورالعمل های بدست آمده از یک فایل اجرایی به طور ۱۰۰ درصد درست است که این موضوع نقطه ضعف بزرگی برای این دسته به حساب می آید.

[۱] Opcode n-gram

فصل سوم

۳ روش های تحلیل فایل های اجرایی آلوده

۳-۱ مقدمه

پس از بررسی تفاوت های موجود بین فایل اجرایی آلوده و فایل اجرایی سالم و همچنین بررسی کارهای انجام شده در تشخیص فایل های آلوده، در این فصل می خواهیم تحلیل فایل های اجرایی آلوده را از جنبه های مختلف بررسی کنیم . در واقع در این فصل تلاش می کنیم بازای هر فایل اجرایی یک الگوی سیگنالی داشته باشیم که بتوان براساس آن فایل های آلوده را شناسایی کرد. برای این کار باید با توجه به تفاوت های فایل های سالم و آلوده به تحلیل فایل های اجرایی پرداخت. بطور کلی دو تحلیل برای ارایه یک الگوی سیگنالی بازای هر فایل اجرایی در این فصل مورد بررسی قرار گرفته است: ۱- براساس کد اسمبلی ۲- براساس بایت های فایل اجرایی

در دسته اول ابتدا فایل اجرایی به کد اسمبلی تبدیل می شود و سپس بروی آن تحلیل های مختلفی انجام می شود پس در این دسته ابتدا باید بروی فایل اجرایی یک عملیات پیش پردازش[۱] انجام داد که از نظر زمانی برای سیستم ما یک نقطه ضعف محسوب می شود. علاوه بر این کد اسمبلی بدست آمده به طور کامل دقیق نیست زیرا نرم افزار های معروف تبدیل فایل اجرایی به اسمبلی[۲] مانند آیدا[۳] نیز در برخی موارد اشتباه می کنند.

در دسته دوم فایل اجرایی براساس بایت های تشکیل دهنده فایل که در مبنای ۱۶ می باشند تحلیل می شوند و تقریبا می توان گفت نسبت به دسته اول پیش پردازش ندارد و در نتیجه دسته دوم بسیار سریعتر از دسته اول می باشد و با توجه به اهمیت زمان در سیستم تشخیص فایل آلوده این امر نقطه قوت بزرگی برای این دسته به حساب می آید.

---

[۱] Preprocessing

[۲] Disassembler

[۳] IDA

۳-۲ براساس تحلیل کد اسمبلی

در این دسته محتوای با معنی فایل اجرایی مورد تحلیل قرار می گیرد و می خواهیم براساس اینکه چه کدهای اسمبلی در فایل اجرایی جود دارد آلودگی فایل را تشخیص دهیم و براساس کد اسمبلی بازای هر فایل اجرایی یک الگوی سیگنالی داشته باشیم. برای این کار اولین مرحله تبدیل فایل اجرایی به کد اسمبلی می باشد تا در ادامه تصمیم بگیریم چگونه کد اسمبلی را تحلیل کنیم. در تبدیل فایل اجرایی به کد اسمبلی با دو مشکل روبرو می شویم:

۱- زمانبر بودن ۲- دارای خطا بودن

در تبدیل فایل اجرایی به کد اسمبلی زمان تبدیل، به اندازه فایل اجرایی بستگی دارد و هرچه فایل بزرگتر باشد در پیش پردازش زمان بیشتری صرف می شود علاوه بر این ممکن است کد اسمبلی بدست آمده از یک فایل اجرایی دارای اشتباه باشد که این امر در فایل های آلوده بیشتر دیده می شود. علت مشکل دوم این است که در بدافزارها بروی قسمت کد مخرب تکنیک های پنهان سازی[۱] اعمال می شود که این امر موجب می شود کد اسمبلی بدست آمده از یک فایل اجرایی دقیق نباشد و برخی دستورات به اشتباه بدست آمده باشند.

نکته دیگری که در این دسته حایز اهمیت است وجود کدهای اضافی و بدون استفاده در فایل های آلوده می باشد. در فایل های آلوده جهت جلوگیری از تحلیل درست فایل، کدهای بی فایده و اضافی زیادی به کد مخرب اضافه می شود مثلا حلقه با تعداد تکرار بالا که در پایان حلقه هیچ مقداری تغییر نکرده است و یا جمع و تفریق یک مقدار با مقادیر مختلف و جایجایی یک مقدار که در نهایت آن مقدار نیز تغییر نمی کند.

---

[۱] Obfuscation

پس از بدست آوردن کد اسمبلی رویکردهای متفاوتی را می توان جهت بدست آوردن الگوی سیگنالی از آن اتخاذ کرد که در اینجا دو رویکرد مورد بررسی قرار می گیرد: ۱- براساس کد عملگر ۲- براساس تغییرات اشاره‌گر دستورالعمل که در ادامه به بررسی آن ها می پردازیم.

۳-۲-۱ براساس کد عملگر

همانطور که گفته شد پس از تبدیل فایل اجرایی به کد اسمبلی با جدا کردن کد عملگر ها از عملوند ها ادامه تحلیل بروی کد عملگرها انجام می شود.چند روش می توان برای تحلیل کد عملگرها ارایه کرد. در اولین روش تعداد تکرار کد عملگر به عنوان الگوی سیگنالی در نظر گرفته می شود که در آن تعداد تکرار هرکدام از کد عملگرها مانند Move, Add, Sub,… بدست آمده و براساس آن نماینده ای برای آن فایل معرفی می شود. در این روش می توان فقط تعدادی از کد عملگرها را در نظر گرفت. به عبارت دیگر پس از آنکه تعداد تکرار همه کد عملگر ها بدست آمد از بین آن ها کد عملگر هایی که در تشخیص فایل سالم از آلوده نقشی ندارد را حذف کرد و سپس براساس تعداد تکرار کد عملگرهای باقی مانده الگوی سیگنالی فایل را بدست آورد.

رویکردی دیگری که می توان در این دسته ارایه کرد این است که تعداد تکرار کلاس های مختلف عملگرها را در نظر گرفت. به عبارت دیگر ابتدا باید کد عملگرها را کلاس بندی کرد، که برای این کار می توان از دسته بندی ارایه شده توسط شرکت سازنده پردازنده استفاده کرد و پس از آن بازای دیده شدن هر کد عملگر یک واحد به کلاسی که آن کد عملگر عضو آن می باشد اضافه کرد و سپس تعداد تکرار هر کلاس مد نظر قرار بگیرد.

با توجه به هزینه‌بر بودن تبدیل فایل اجرایی به کد اسمبلی و همچنین عدم دقت ۱۰۰درصد در این تبدیل، این دسته از روش ها در پایان نامه مورد استفاده قرار نگرفت.

۳-۲-۲ براساس تغییرات اشاره گر دستورالعمل

همانطور که گفته شد در فایل آلوده تغییرات اشاره گر دستور العمل نسبت به تغییرات این اشاره گر در فایل سالم متفاوت می باشد که دلیل آن اضافه شدن کد مخرب به فایل سالم است. بر همین اساس می توان تغییرات مقادیر اشاره گر دستورالعمل را با توجه به زمان به عنوان الگوی سیگنالی هر فایل اجرایی در نظر گرفت. به عبارت دیگر از زمان آغاز اجرا تا پایان اجرای دستورالعمل های یک فایل اجرایی مقادیر این اشاره گر ذخیره می شود و نمودار این تغییرات بر حسب زمان رسم می شود.

جهت بدست آوردن مقادیر اشاره گر دستورالعمل دو رویکرد می توان ارایه کرد.در رویکرد اول فایل اجرایی در یک محیط شبیه سازی شده اجرا شده و در هر لحظه مقادیر این اشاره گر ذخیره شود و سپس نمودار مقادیر مختلف بر حسب زمان رسم می شود. چند نکته در باب این رویکرد قابل ذکر است اول اینکه ایجاد محیط شبیه سازی شده و امن که بتوان در آن فایل را اجرا کرد هزینه بر است. دوم این که هر فایل یکبار به طور کامل اجرا شده و سپس مقادیر اشاره گر آن فایل تحلیل شود این کار بسیار زمان بر است و با توجه به اینکه زمان در تشخیص آلودگی فایل بسیار مهم است این روش مفید نیست. مازاد بر موارد ذکر شده فرض ما بر این است که باید فایل به صورت ایستا تحلیل شود.

در رویکرد دوم ابتدا باید فایل اجرایی به کد اسمبلی تبدیل شود و سپس برای بدست آوردن تغییرات مقادیر اشاره گر دستورالعمل از ابتدای برنامه شروع به بررسی کد عملگر کرده و با توجه به آن مقدار این اشاره گر را بدست آوریم و این کار را تا پایان برنامه ادامه دهیم که در این روش نیز باید ازا ابتدا تا انتهای فایل اجرایی بررسی شود و بسیار زمان بر است و علاوه بر اینکه ماهیت این روش زمان بر است در فایل های آلوده به دلیل وجود بسیار زیاد کدهای زاید و حلقه های با تکرار بالا و تو در تو این روش را از نظر زمانی با مشکل مواجه می کند.همچنین با توجه به اینکه صحیح بدست آوردن مقادیر اشاره گر بسیار وابسته به تبدیل درست و کامل فایل اجرایی به کد اسمبلی می باشد اما همانطور که گفته شد به دلیل پنهان سازی در فایل های آلوده این امر امکان پذیر نیست.

۳-۳ براساس بایت های فایل اجرایی

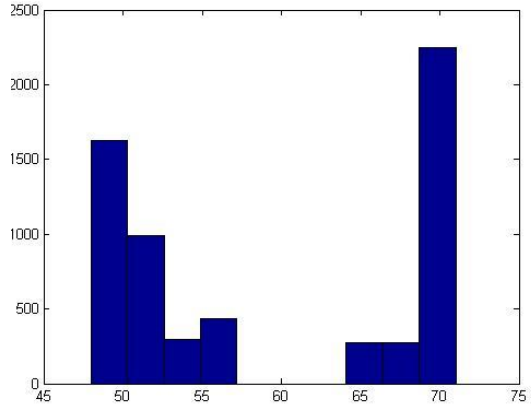
در این دسته محتوای بدون معنی فایل اجرایی تحلیل می شود به عبارت دیگر مجموعه بایت های تشکیل دهنده فایل اجرایی که در ظاهر نمی توان از آن ها به آلودگی فایل رسید معیاری برای تشخیص آلودگی فایل می شود. کارهایی که تاکنون در این دسته انجام شده است بروی کل فایل اجرایی اعمال شده است که منجر به در بر گرفتن بخش زیادی از بایت ها می شود که در تشخیص آلودگی فایل نقشی ندارند.

بر همین اساس ما به دنبال ارایه روشی بودیم تا براساس آن بخشی از فایل اجرایی که بیشترین تغییرات را در هنگام آلودگی دارد؛ مورد تحلیل قرار گرفته و براساس آن الگوی سیگنالی فایل رابدست آوریم. با توجه به اینکه همواره بخشی از کد مخرب در قسمت آخر فایل اجرایی یعنی قسمت منبع قرار دارد به جای آن که کل فایل اجرایی تحلیل شود فقط قسمت منبع مورد تحلیل قرار می گیرد.در این روش، مرحله پیش پردازش شامل جداسازی قسمت آخر فایل اجرایی و ذخیره آن در یک فایل متنی است که شامل بایت های این قسمت در مبنای ۱۶ است. همچنین جهت تحلیل راحت تر فایل های متنی بین هر دو بایت از حرف G به عنوان جدا کننده استفاده کردیم. در ادامه کلیه تحلیل ها بروی قسمت آخر فایل های اجرایی سالم و آلوده که در قالب فایل متنی ذخیره شده اند انجام می شود.

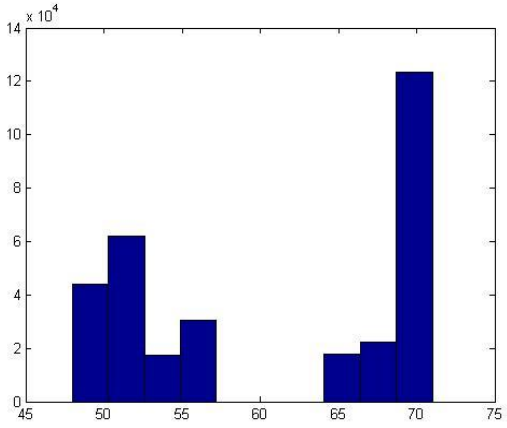
پس از جداسازی قسمت آخر فایل اجرایی، با دو رویکرد به تحلیل این قسمت پرداختیم: ۱- براساس بایت های تشکیل دهنده ۲- براساس ترکیب بایت ها. در ادامه هر کدام از روش ها به طور کامل ارایه می شود.

۳-۳-۱ براساس بایت های تشکیل دهنده

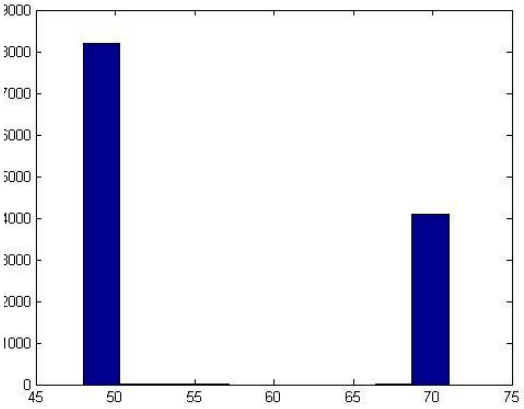
در این روش جهت بدست آوردن الگوی سیگنالی، تعداد تکرار هر کدام از بایت ها مد نظر قرار می گیرد همانطور که گفته شد مجموعه بایت ها در مبنای ۱۶ شامل ارقام 9-0 و حروف A-G که G نقش جدا کننده برای ۲ بایت را دارد می شوند. برای راحتر شدن کار از کد اسکی آن ها که بین ۴۸تا۵۷ برای ارقام و ۶۵تا۷۱ برای حروف می باشد؛استفاده می کنیم.جهت بررسی روش ارایه شده چند نمونه فایل سالم و آلوده در نظر گرفته شد و سپس با جدا کردن قسمت آخر آن فایل ها به بررسی نمودار فرکانس بایت های تشکیل دهنده قسمت آخر پرداختیم .



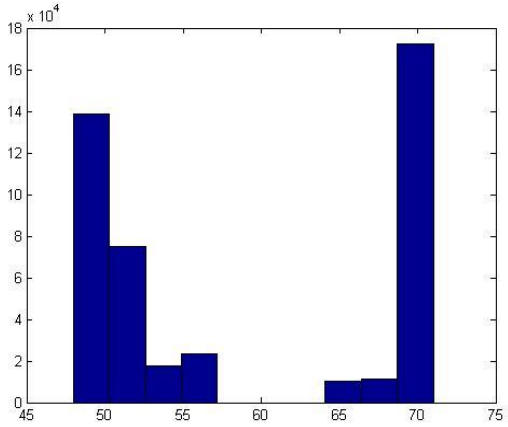
الف-۱



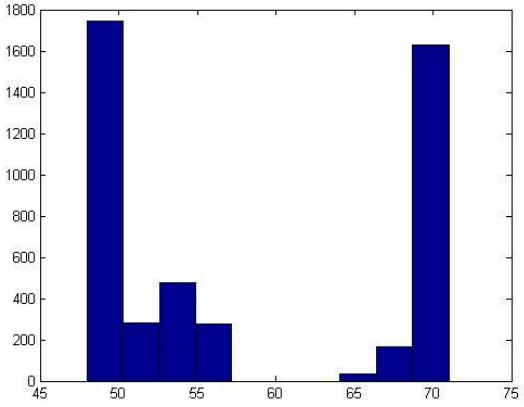
الف-۲



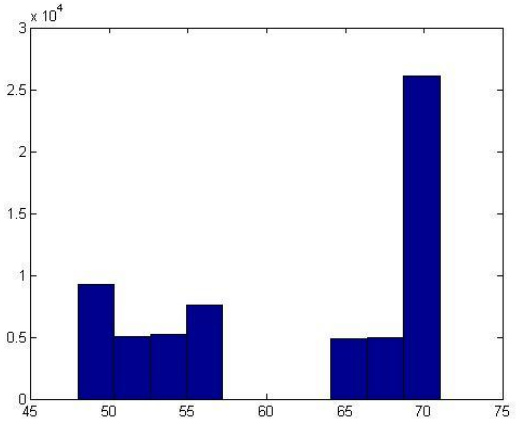
ب-۱



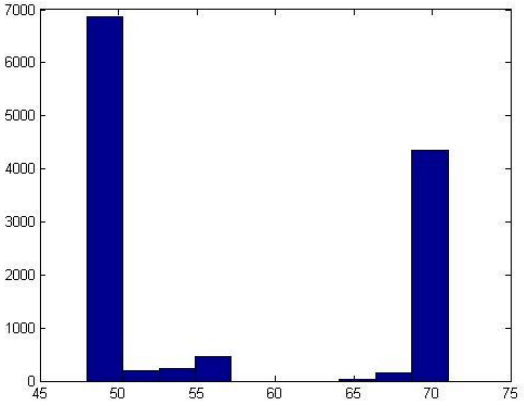
ب-۲



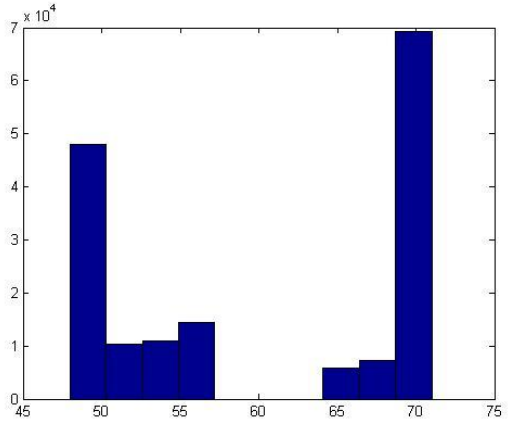
ج-۱



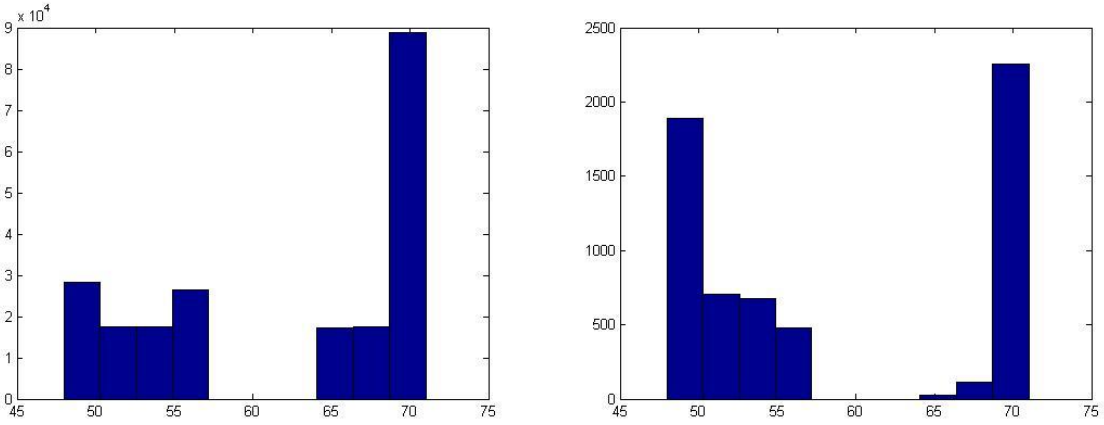
ج-۲



پ-۱



پ-۲



ی-۲ ی-۱

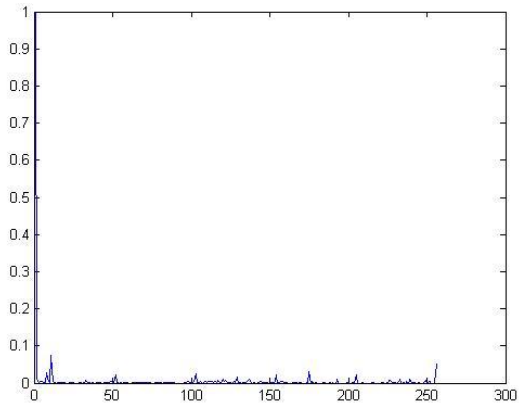
شکل (۳-۱) نمودار فرکانس بایت های تشکیل دهنده قسمت آخر فایل اجرایی. (الف-۱)، (ب-۱)، (ج-۱)، (پ-۱) (ی-۱). برای فایل های اجرایی سالم. (الف-۲)، (ب-۲)، (ج-۲)، (پ-۲) (ی-۲)، برای فایل های اجرایی آلوده

در شکل (۳-۱) نمودار تعداد تکرار بایت های تشکیل دهنده قسمت آخر فایل اجرایی آمده است در این نمودارها محور افقی شامل کد اسکی بایت های تشکیل دهنده این قسمت می باشد و محورعمودی نیز تعداد تکرار آن بایت ها را نشان می دهد. همانطور که گفته شد به این دلیل تحلیل بروی قسمت آخر فایل اجرایی انجام شدکه همواره بخشی از کد مخرب در این قسمت از فایل قرار دارد پس باید ارتباطی بین تکرار بایت ها و وجود کد مخرب در فایل آلوده پیدا کرد و یا اینکه ارتباطی بین وجود کاراکتر متنی در فایل سالم و تعداد تکرار بایت ها در فایل سالم دیده شود. با توجه به دو ارتباط مطرح شده به این نتیجه رسیدیم تا به جای بررسی تعداد تکرار ۱ بایت، به بررسی تکرار ترکیب بایت ها بپردازیم.

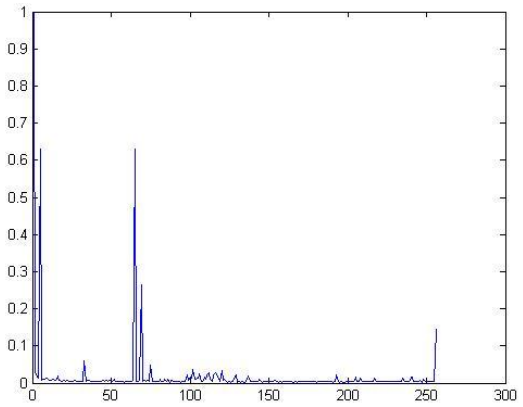
۳-۳-۲ براساس ترکیب بایت های تشکیل دهنده

پس از بررسی نمودار تکرار بایت ها در قسمت آخر فایل اجرایی، مشخص شد براساس آن نمی توان فایل آلوده را تشخیص داد. در این قسمت هدف بررسی ترکیب های مختلف بایت ها جهت رسیدن به بهترین ترکیب می باشد. به عبارت دیگر با توجه به اینکه مبنای تحلیل قسمت آخر فایل اجرایی بر این

اساس است که در قسمت منبع فایل سالم بیشتر کاراکترهای متنی وجود دارد ولی این قسمت در فایل آلوده شامل کدهای مخرب می باشد؛ تعداد تکرار ترکیبی از بایت ها را بررسی کنیم که این اختلاف بین فایل سالم و آلوده را بهتر نشان دهد. با توجه به اینکه کاراکترهای متنی به صورت ۲ بایت در کد اسکی [۱] نشان داده می شوند و همچنین ۲ بایت نماینده بهتری برای کدهای مخرب می باشد؛ اگر تعداد تکرار ۲بایت ها الگوی سیگنالی یک فایل اجرایی را تشکیل دهد جهت تشخیص آلودگی یک فایل مناسب تر است. در مبنای ۱۶ دو بایت ۲۵۶ حالت مختلف می توانند کنار هم قرار بگیرند ودر نتیجه الگوی سیگنالی یک فایل اجرایی نشان دهنده فرکانس این ۲۵۶ حالت مختلف در قسمت آخر فایل اجرایی می باشد.

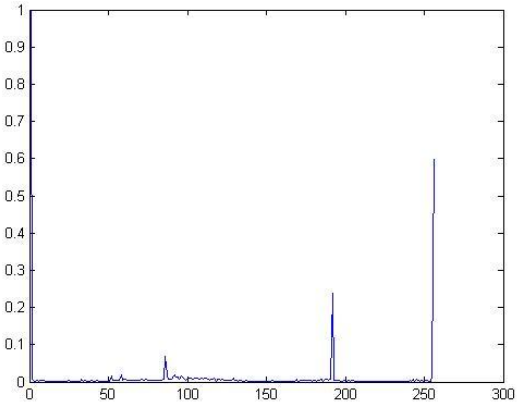


الف-۱

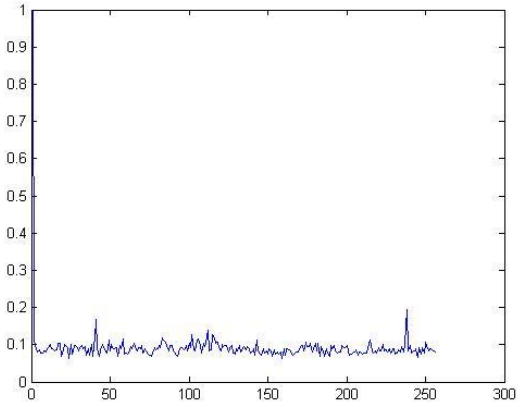


الف-۲

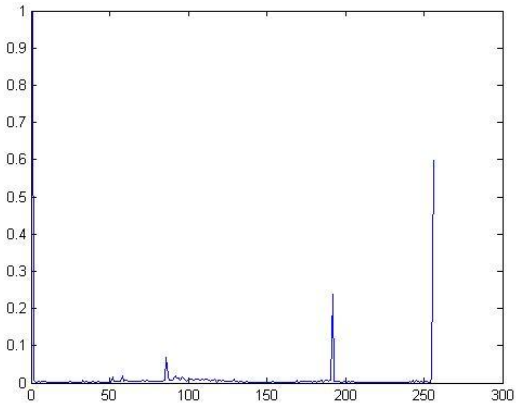
[۱] ASCII



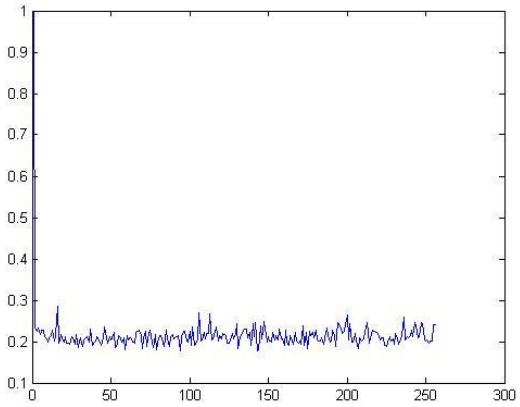
ب-۱



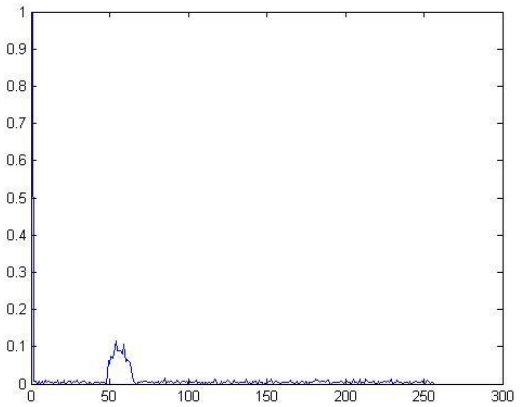
ب-۲



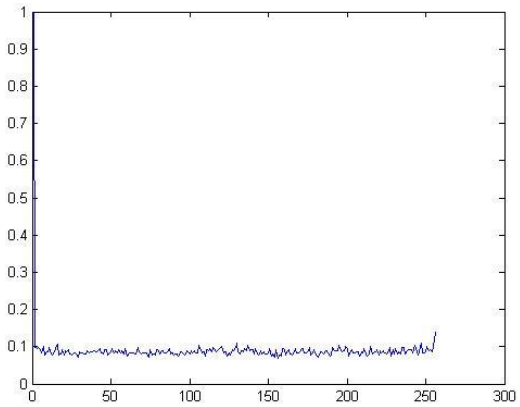
ج-۱



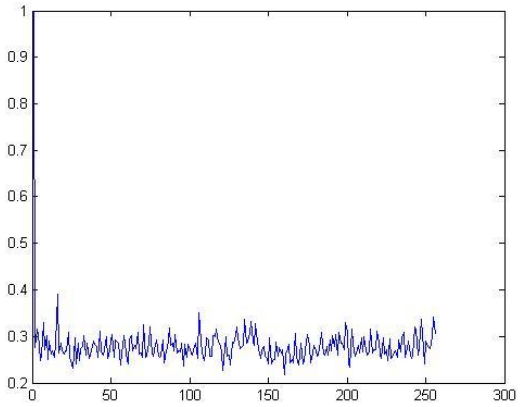
ج-۲



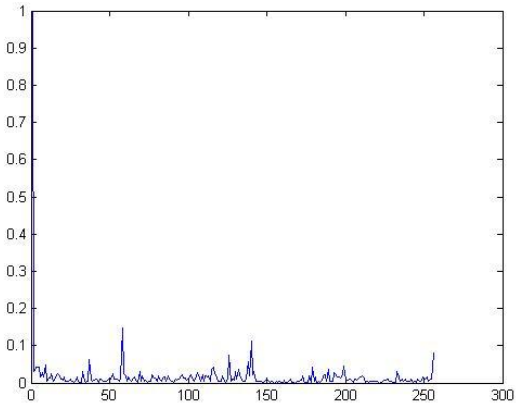
پ-۱



پ-۲



ی-۲



ی-۱

شکل (۳-۲) نمودار تعداد تکرار ۲بایت ها در قسمت آخر فایل اجرایی. (الف-۱)، (ب-۱)، (ج-۱)، (پ-۱) (ی-۱). برای فایل های اجرایی سالم. (الف-۲)، (ب-۲)، (ج-۲)، (پ-۲) (ی-۲)، برای فایل های اجرایی آلوده

در شکل (۳-۲) نمودار تعداد تکرار ۲ بایت ها در قسمت آخر فایل اجرایی آمده است. در این نمودارها محور افقی نشان دهنده ۲۵۶ حالت مختلف ۲بایت ها و محور عمودی تعداد تکرار آن ها می باشد که برای مقایسه بهتر نمودارها مقادیر تعداد تکرار بایت ها نرمال شده اند و تعدادتکرار هر حالت مقداری بین ۰و۱ دارد.در نمودار فایل های سالم مقادیر تکرار حالت های مختلف معمولا کمتر از مقادیر تعداد تکرار در فایل آلوده می باشد. می توان گفت میانگین تعداد تکرار در فایل سالم کمتر از میانگین تعداد تکرار در فایل آلوده می باشد و با تحلیل نمودارهای بدست آمده می توان به این نتیجه رسید که در روش دوم تفاوت بیشتری بین نمودار فایل های سالم وآلوده قابل مشاهده است و می توان براساس آن و اعمال روش های دسته بندی [۱] سیستم تشخیص آلودگی فایل اجرایی را تولید کرد.

[۱] Classification

فصل چهارم

٤ پیاده سازی و ارزیابی نتایج

۴-۱ مقدمه

در ادامه کار می بایست براساس نتایجی که در فصل سوم بدست آمد یک روش مناسب جهت تشخیص فایل آلوده از سالم ارایه کنیم. برای این کار ابتدا باید ویژگی مناسبی را انتخاب و با بررسی روش های مختلف وزن دهی یک روش وزن دهی کارا ارایه کنیم.

در این فصل براساس بهترین روش ارایه شده در فصل سوم یعنی بدست آوردن فرکانس ۲ بایت ها در قسمت آخر فایل اجرایی استخراج ویژگی را انجام می دهیم و از این ویژگی ها به عنوان نماینده هر فایل اجرایی استفاده می کنیم. سپس با اعمال روش های مختلف وزن دهی و همچنین استفاده از روش نزدکترین همسایه به عنوان روش دسته بندی، کارایی هرکدام از روش های وزن دهی را مورد بررسی قرار می دهیم و همچنین تغییرات تعداد همسایگی و رابطه آن با تشخیص فایل های اجرایی آلوده را بررسی می کنیم.

۴-۲ استخراج ویژگی

با توجه به اینکه از فایل اجرایی فقط قسمت آخر آن در قالب یک فایل متنی، معیاری جهت تشخیص آلودگی در نظر گرفته شد می توان برای یافتن ویژگی مناسب، صورت مسئله را به این شکل تغییر داد که می خواهیم فایل های متنی را به دو دسته آلوده و سالم تقسیم و در ادامه کار براساس روش های دسته بندی متن رفتار کنیم.

ویژگی ای که در دسته بندی متن استفاده می شود کلمه است. پس باید در صورت مسئله جدید منظورمان از کلمه را در یک فایل متنی تشکیل شده از بایت هایی در مبنای ۱۶ بیان کنیم. با توجه به اینکه در کد اسکی کاراکتر ها با ۲ بایت نشان داده می شوند و همچنین در کد اسمبلی نیز ۲بایت ارتباط بهتری با دستورالعمل ها دارد هر ۲بایت را به عنوان یک کلمه در نظر می گیریم و باتوجه به اینکه که

بایت ها در مبنای ۱۶ می باشند هر متن از ۲۵۶ کلمه یا ویژگی تشکیل شده است. به عبارت دیگر بازای هر فایل اجرایی یک بردار ویژگی ۲۵۶ تایی داریم.

۴-۳ وزن دهی ویژگی ها

در این قسمت به بررسی روش های مختلف وزن دهی ویژگی ها که برای بدست آوردن بهترین روش وزن دهی مورد بررسی قرار گرفته اند می پردازیم. منظور از وزن دهی به ویژگی این است که مشخص کنیم هر ویژگی به چه میزان می تواند نشانگر محتوای آن متن باشد. اگر فرض کنیم $D=\{d_1,\ldots,d_n\}$مجموعه کل متن ها و $C=\{c_1,\ldots c_m\}$ مجموعه دسته ها باشد، هر متن $d_i \epsilon D$ با بردار ویژگی $d_i=(w_{1i},\ldots,w_{ki})$ نشان داده می شود که در آن k تعداد ویژگی های متمایز در مجموعه D است که تعداد آن در روش ما ۲۵۶ است و w_{ki} وزن ویژگی t_k می باشد.

۴-۳-۱ روش TF[۱]

در این روش وزن ویژگی t_k در متن d_i برابر با تعداد تکرار ویژگی در آن متن می باشد. ایده اصلی این روش بر این اساس است که هرچه تعداد تکرار یک ویژگی بیشتر باشد ارزش آن ویژگی بیشتر است. رابطه (۴-۱) بیانگر روش بدست آوردن این نوع وزن دهی می باشد

$$w_{ki}=tf(t_k,d_i)=\begin{cases}\#(t_k,d_i) & t_k \in vector\ of\ d_i \\ 0 & t_k \notin vector\ of\ d_i\end{cases} \qquad (۱-۴)$$

که در آن $\#(t_k,d_i)$ برابر با تعداد تکرار ویژگی t_k در متن d_i می باشد.

---

[۱] Term Frequency

۴-۳-۲ روش normTF

معمولا طول متن های موجود در مجموعه D برابر نمی باشد برای آنکه طول متن بروی وزن ویژگی تاثیری نداشته باشد از این روش که نرمال شده روش TF می باشد استفاده می کنیم که این امر باعث می شود مقدار وزن ویژگی ها بین (0,1) باشد. برای این کار تعداد تکرار هر ویژگی در هر متن را بر طول آن متن تقسیم می کنیم رابطه (۴-۲) نشان دهنده این نوع وزن دهی می باشد:

$$w_{ki}=\text{normTF}(t_k,d_i)=\frac{tf(t_k,d_i)}{\sqrt{\sum_k (tf(t_k,d_i))^2}} \qquad (۴-۲)$$

که در آن $\text{tf}(t_k, d_i)$ از رابطه(۴-۱) بدست می آید.

۴-۳-۳ روش logTF

ممکن است در یک متن تعداد تکرار یک ویژگی خیلی بزرگتر از ویژگی های دیگر باشد و این امر بروی دسته بندی تاثیر منفی بگذارد برای رفع این اثر نامطلوب و یکسان کردن محدوده اختصاص داده شده از این روش استفاده می کنیم:

$$w_{ki} =\text{logTF}(t_k,d_i)=\log(tf(t_k, d_i)) \qquad (۴-۳)$$

۴-۳-۴ روش ITF[۱]

این روش با استفاده از قوانین آماری ارایه شده است[۶] که در آن معمولا مقدار r یک در نظر گرفته می شود.

$$w_{ki} =\text{ITF}(t_k,d_i)=1- \frac{r}{r+tf(t_k,d_i)} \qquad (۴-۴)$$

---

۱ Inverse Term Frequency

۴-۳-۵ روش Sparck

این روش با استفاده از تیوری های آماری ارایه شده است[۷] و که در آن k تعداد کل ویژگی های متمایز در مجموعه D و $p_k=\sum_D tf(t_k,d_i)$ می باشد.

$$w_{ki}=\text{Sparck}(t_k,d_i)=tf(t_k,d_i)*(k-\log(p_k)) \quad (۴-۵)$$

۴-۳-۶ روش IDF

در این روش وزن ویژگی برای کل مجموعه تعیین می شود[۸] که در آن |D| تعداد کل متن های مجموعه D و $|D(t_k)|$ تعداد متن هایی است که ویژگی t_k در آن ها وجود دارد و در نتیجه هر چه یک ویژگی در تعداد متن بیشتری وجود داشته باشد وزن کمتری می گیرد.

$$w_{ki}=\text{idf}(t_k)=\log\frac{|D|}{|D(t_k)|} \quad (۴-۶)$$

۴-۳-۷ روش TFIDF

همانطور که گفته شد روش IDF به هر ویژگی در کل مجموعه یک وزن می دهد به همین دلیل از ترکیب این روش با روش TF می توان به هر ویژگی به هر متن یک وزن اختصاص داد که در آن $\text{idf}(t_k)$ از رابطه(۴-۶) بدست می آید.

$$w_{ki}=\text{tfidf}(t_k,d_i)=tf(t_k,d_i)*idf(t_k) \quad (۴-۷)$$

۴-۳-۸ روش normTFIDF

برای از بین بردن اثر طول بر روی وزن ویژگی ها در روش TFIDF، از این روش استفاده می شود[۹]:

$$w_{ki}=\text{normTFIDF}(t_k,d_i)=\frac{tfidf(t_k,d_i)}{\sqrt{\sum_k(tfidf(t_k,d_i))^2}} \quad (۴-۸)$$

۴-۳-۹ روش TFRF

در این روش توزیع ویژگی t_k در دسته از پیش تعریف شده c_j استفاده می شود. [۱۰] برای این منظور فاکتور ارتباط[۱] بازای هر ویژگی t_k در دسته c_j به صورت رابطه(۴-۹)تعریف می شود:

$$rf(t_k,c_j)=\log\left(2+\frac{|D(t_k,c_j)|}{\sum_{m=1,m\neq j}^{|C|}|D(t_k,c_m)|}\right) \quad (۴-۹)$$

که در آن $c_j \epsilon C$ و $|D(t_k,c_j)|$ تعداد متن هایی عضو مجموعه D و از دسته c_j که ویژگی t_k را داشته باشند است. $\sum_{m=1,m\neq j}^{|C|}|D(t_k,c_m)|$ مجموع تعداد مستنداتی است که عضو مجموعه D باشند و از دسته ای به جز دسته c_j قرار داشته باشند که دارای ویژگی t_k هستند. همانطور که مشاهده می شود بازای هر ویژگی این فاکتور رابطه ی مستقیم دارد با تعداد متن هایی از همان دسته متن اصلی هستند و دارای آن ویژگی باشند و همچنین رابطه معکوس دارد با تعداد متن هایی از دسته های دیگر که دارای این ویژگی باشند. در نهایت وزن ویژگی t_k از مستند d_i بعد از نرمال سازی از رابطه(۴-۱۰) بدست می آید که در آن c_{d_i} طبقه متن d_i می باشد.

$$w_{ki}=TFRF(t_k,d_i)=\frac{tf(t_k,d_i)*rf\left(t_k,c_{d_i}\right)}{\sqrt{\sum_k\left(tf(t_k,d_i)\right)^2*\left(rf\left(t_k,c_{d_i}\right)\right)^2}} \quad (۴-۱۰)$$

۴-۴ انتخاب روش دسته بندی

پس از اعمال روش های مختلف وزن دهی در مرحله بعدی باید یک روش دسته بندی مناسب انتخاب کرد. با توجه به اینکه روش نزدیک ترین همسایه یکی از موثرترین و سریعترین روش ها در دسته بندی متن می باشد ما نیز این روش را به عنوان روش دسته بندی انتخاب کردیم.در این روش مجموعه ای از

---

[۱] Relevancy Factor(rf)

نمونه ها به عنوان مجموعه آموزش[۱] در نظر گرفته می شود. در زمان تست بازای هر نمونه از مجموعه تست فاصله آن با تمام نمونه های مجموعه آموزش براساس معیار فاصله در نظر گرفته شده بدست می آید سپس براساس تعداد همسایگی در نظر گرفته شده K فاصله ی کمتر بررسی می شود. دسته ای که بیشترین تعداد نمونه در بین K همسایه ی انتخاب شده را داشته باشد به عنوان دسته نمونه تست مورد نظر انتخاب می شود. بر این اساس مقدار K معمولا عددی فرد در نظر گرفته می شود. در این پایان نامه برای تعیین فاصله بین دو بردار از فاصله اقلیدسی[۲] که در رابطه (۴-۱۱) آمده است استفاده می کنیم:

$$\sqrt{\sum_{j=1}^{k}\left(d_{1_j}-d_{2_j}\right)^2} \qquad (۴-۱۱)$$

۴-۵ پایگاه داده

در روش های ارایه شده در زمینه تشخیص فایل های اجرایی آلوده، پایگاه داده استاندارد وجود ندارد و چون کارهایی که در این زمینه انجام می شوند اغلب با هدف تجاری سازی و استفاده در آنتی ویروس ها صورت می گیرد پایگاه داده خود را در اختیار دیگران نمی گذارند و معمولا در هر روشی یک پایگاه داده معرفی می شود. لازم به ذکر است در برخی روش ها از پایگاه داده یکسان استفاده شده بود ولی آن پایگاه داده نیز در دسترس نبود. بر همین اساس در این پایان نامه سعی کردیم پایگاه داده مناسبی ارایه کنیم. برای آزمایش روش پیشنهادی، ۱۶۷ فایل اجرایی آلوده به همراه ۱۶۷ فایل اجرایی سالم مجموعه کل داده های ما را که ۳۳۴ فایل اجرایی می باشد را تشکیل می دهند. که فایل های آلوده شامل دو دسته از کدهای مخرب رایج به نام های Sality[۳] و Virut[۴] می شوند و فایل های سالم نیز از مجموعه

---

۱ Train

۲ Euclidian Distance

۳ W32.Sality

۴ W32.Virut

فایل های اجرایی ویندوز هفت[۱] بدست آمده است. که برای آزمایش به روش نزدیکترین همسایه از ۱/۳ داده ها برای تست و ۲/۳ آن ها برای آموزش استفاده می کنیم . بنا بر این داده تست ما از ۱۱۱ فایل شامل ۵۶ فایل آلوده و ۵۵ فایل سالم تشکیل شده است.

۴-۶ معیارهای ارزیابی

جهت ارزیابی روش های دسته بندی معیار های مختلفی مطرح شده است ولی در دسته بندی فایلهای آلوده و سالم سه معیار اصلی وجود دارد.

۴-۶-۱ نرخ تشخیص درست مثبت[۲]

اولین معیار نرخ تشخیص درست مثبت می باشد که معادل است با تقسیم تعداد فایل هایی که به درستی آلوده تشخیص داده شده اند بر کل فایل های آلوده که در رابطه(۴-۱۲) آمده است:

$$TPR=\frac{TP}{TP+FN} \qquad (۱۲-۴)$$

که در آن TP[۳] برابر است با تعداد فایل های که به درستی آلوده تشخیص داده شده اند و FN[۴] تعداد فایل های آلوده ای که اشتباها سالم در نظر گرفته شده اند. هر چه این عدد به ۱ نزدیکتر باشد یعنی از بین فایل های اجرایی آلوده فایل های بیشتری شناسایی شده اند و در نتیجه کارایی آن روش بیشتر است.

---

۱ Windows 7

۲ TPR(True Positive Ratio)

۳ True Positive

۴ False Negative

۴-۶-۲ نرخ تشخیص غلط مثبت[۱]

معیار دوم نرخ تشخیص غلط مثبت می باشد که معادل است با نسبت تعداد فایل های سالم که اشتباها آلوده تشخیص داده شده اند به کل فایل های سالم که در رابطه (۴-۱۳) آمده است:

$$FPR=\frac{FP}{FP+TN} \quad (۱۳-۴)$$

که در آن FP[۲] نشان دهنده تعداد فایل های سالمی است که اشتباها آلوده تشخیص داده شده اند و TN[۳] برابر است با تعداد فایل هایی که به درستی سالم تشخیص داده شده اند. در این معیار نرخ فایل های سالمی که اشتباها آلوده تشخیص داده می شوند بیان می شود. در یک سیستم تشخیص فایل آلوده هر قدر این معیار به صفر نزدیک شود کارایی سیستم بیشتر است. به این دلیل که اگر این سیستم فایل های سالم زیادی را آلوده تشخیص دهد اشتباها اجازه فعالیت به آن فایل های سالم نمی دهد کارایی کاربر پایین می آید.

۴-۶-۳ دقت

معیار سوم دقت است که معادل با تقسیم تعداد کل فایل هایی که به درستی دسته بندی شده اند تقسیم بر تعداد فایل های مجموعه تست می باشد که در رابطه(۴-۱۴) آمده است:

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (۱۴-۴)$$

در واقع در این معیار دقت سیستم در تشخیص درست فایل های سالم و آلوده ارزیابی می شود و در آن هم تشخیص درست فایل های آلوده نقش دارد و هم تشخیص درست فایل های سالم. هر اندازه مقدار

---

۱ FPR(False Positive Ratio)

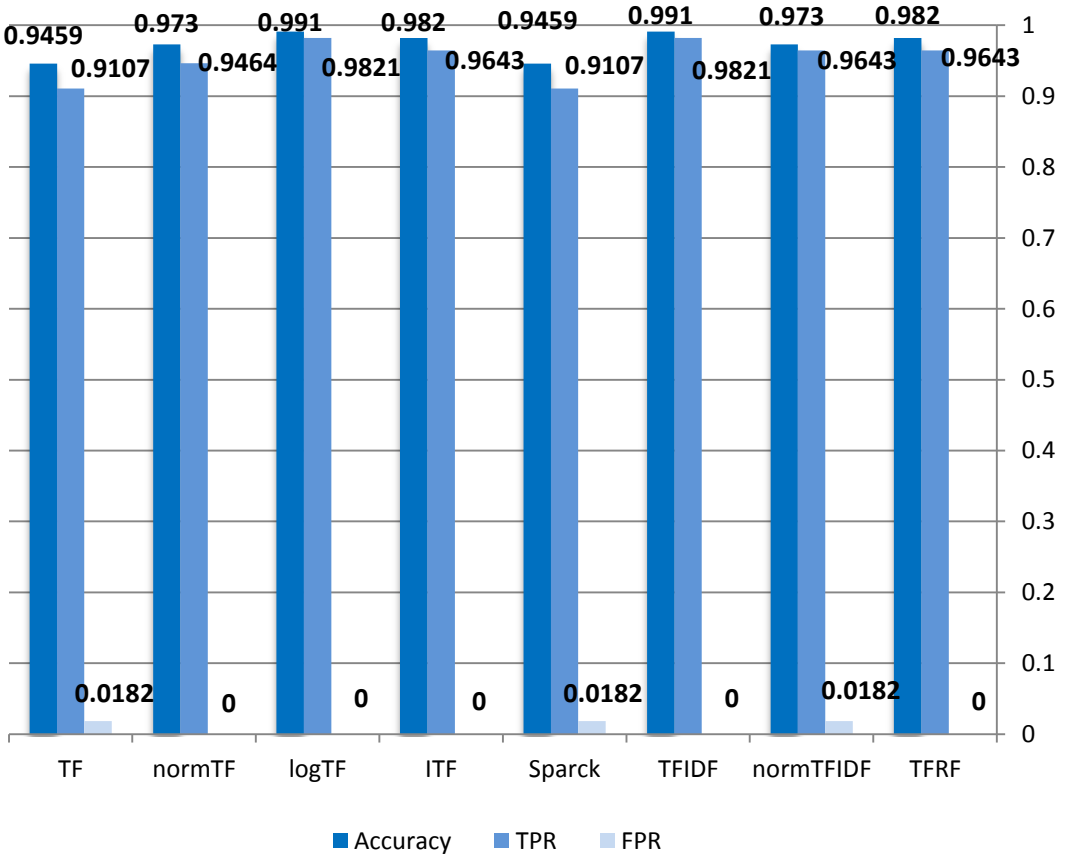
۲ False Positive

۳ True Negative

این معیار به ۱ نزدیکتر باشد یعنی فایل های بیشتری درست تشخیص داده شده اند و در نتیجه کارایی سیستم بیشتر است.

۴-۷ نتایج

پس از پیاده سازی روش پیشنهادی و استفاده از مجموعه پایگاه داده معرفی شده نتایج زیر حاصل شد.



شکل ۴-۱ نمودار کارایی روش های مختلف وزن دهی براساس معیارهای معرفی شده

در شکل (۴-۱) نمودار کارایی روش های مختلف وزن دهی زمانی که در روش نزدکترین همسایه مقدار k برابر با یک باشد آمده است. همانطور که در شکل مشخص است روش های logTF و TFIDF با دقت ۰.۹۹۱۰ و TPR ۰.۹۸۲۱ وFPR صفر بهترین کارایی را دارند و با توجه به اینکه در روش TFIDF

تعداد متن هایی که یک ویژگی در آن ها وجود دارد در ارزش گذاری برای یک ویژگی دخیل می کند پیش بینی می شود در مجموعه هایی با تعداد بیشتر فایل اجرایی کارایی بهتری نسبت به روش logTF داشته باشد.

همچنین برای بررسی تاثیر تغییرات مقدار K در روش دسته بندی برای روش های مختلف وزن دهی مقادیر ۱و۳و۵و۷و۹ برای K در نظر گرفته شد و معیارهای معرفی شده بررسی گردید.

جدول ۴-۱بررسی کارایی روش های مختلف وزن دهی بازای تغییر مقادیر K.(الف) روش TF. (ب) روش normTF. (ج) روش logTF. (د) روش ITF. (ه) روش Sparck. (و) روش TFIDF. (ز) روش normTFIDF. (ح) روش TFRF

TF	k=1	k=3	k=5	k=7	k=9
Accuracy	0.9459	0.964	0.964	0.955	0.955
TPR	0.9107	0.9464	0.9464	0.9464	0.9464
FPR	0.0182	0.0182	0.0182	0.0364	0.0364

الف

normTF	k=1	k=3	k=5	k=7	k=9
Accuracy	0.973	0.964	0.964	0.964	0.964
TPR	0.9464	0.9286	0.9286	0.9286	0.9286
FPR	0	0	0	0	0

ب

logTF	k=1	k=3	k=5	k=7	k=9
Accuracy	0.991	0.982	0.982	0.991	0.991
TPR	0.9821	0.9643	0.9643	1	1
FPR	0	0	0	0.0182	0.0182

ج

ITF	k=1	k=3	k=5	k=7	k=9
Accuracy	0.982	0.982	0.991	0.991	0.982
TPR	0.9643	0.9821	1	1	1
FPR	0	0.0182	0.0182	0.0182	0.0364

د

Sparck	k=1	k=3	k=5	k=7	k=9
Accuracy	0.9459	0.964	0.964	0.955	0.955
TPR	0.9107	0.9464	0.9464	0.9464	0.9464
FPR	0.0182	0.0182	0.0182	0.0364	0.0364

ه

TFIDF	k=1	k=3	k=5	k=7	k=9
Accuracy	0.991	0.982	0.982	0.991	0.991
TPR	0.9821	0.9643	0.9643	0.9821	0.9821
FPR	0	0	0	0	0

و

normTFIDF	k=1	k=3	k=5	k=7	k=9
Accuracy	0.973	0.982	0.973	0.973	0.973
TPR	0.9643	0.9821	0.9643	0.9643	0.9643
FPR	0.0182	0.0182	0.0182	0.0182	0.0182

ز

TFRF	k=1	k=3	k=5	k=7	k=9
Accuracy	0.982	0.982	0.982	0.982	0.982
TPR	0.9643	0.9643	0.9643	0.9643	0.9643
FPR	0	0	0	0	0

ح

در جدول (۴-۱) با تغییر مقادیر K در روش نزدیک ترین همسایه کارایی روش های مختلف وزن دهی آمده است. در روش TF با افزایش مقدار K کارایی افزایش و در روش های TFIDF و logTF با افزایش همسایگی کارایی کاهش پیدا می کند در نتیجه نمی توان گفت که هر چقدر تعداد همسایگی های در

نظر گرفته شده بیشتر باشد کارایی افزایش پیدا می کند و باید با بررسی جوانب مختلف همسایگی مناسب را پیدا کرد.

فصل پنجم

۵ نتیجه گیری و کارهای آینده

۵-۱ نتیجه گیری

در این پایان نامه تلاش شد روشی ارایه شود تا بازای هر فایل اجرایی یک الگوی سیگنالی داشته باشیم و براساس آن بتوانیم فایل های اجرایی آلوده را شناسایی کنیم. به عبارت دیگر رویکردی جدید جهت تشخیص فایل اجرایی آلوده معرفی شد که براساس آن بتوان با استفاده از تکنیک های پردازش سیگنال فایل آلوده را تشخیص داد. تاکنون روشی که برمبنای پردازش سیگنال فایل آلوده را شناسایی کند مطرح نشده است. بدین منظور روش های مختلف تحلیل ایستای فایل اجرایی مورد بررسی قرار گرفت این روش ها در دو دسته کلی الف) براساس کد اسمبلی و ب) براساس بایت های تشکیل دهنده ارایه شد

در تشخیص فایل اجرایی آلوده از سالم دو شاخص جدید در این پایان نامه معرفی شد: الف)تغییرات اشاره گر دستور العمل و ب) تفاوت در قسمت منبع که می توان از آن ها در سیستم های تشخیص فایل آلوده استفاده کرد. با در نظر گرفتن تفاوت در قسمت منبع و بدست آوردن تعداد تکرار ۲ بایت ها در آن، به عنوان نماینده یک فایل اجرایی و استفاده از روش های دسته بندی متون اثبات کردیم که این روش می تواند در جهت تشخیص فایل اجرایی آلوده موثر باشد.

برای اثبات اینکه تحلیل قسمت آخر فایل اجرایی و بدست آوردن الگوی سیگنالی از آن قسمت می تواند شاخص خوبی برای تشخیص آلودگی فایل اجرایی باشد. پس از بدست آوردن تعداد تکرار ۲ بایت ها در قسمت آخر فایل اجرایی ، هر ۲ بایت را به عنوان یک کلمه در نظر گرفتیم و روش های مختلف وزن دهی در دسته بندی متن را بروی آن ها اعمال کردیم. برای دسته بندی از روش نزدیک ترین همسایه و از معیارهای دقت و TPR و FPR برای ارزیابی کارایی روش های وزن دهی استفاده کردیم که در نهایت

روش TFIDF به همراه روش نزدیکترین همسایه با مقدار K=1 با دقت ۰.۹۹۱۰ و TPR ۰.۹۸۲۱ وFPR صفر بهترین کارایی را داشت.

۵-۲ کارهای آینده

برای ادامه کار می توان در دو جنبه روش های این پایان نامه را دنبال کرد. جنبه اول در بهبود روشی که بر مبنای دسته بندی متن ارایه شده بود.که در این زمینه می توان روش ارایه شده را بروی مجموعه داده هایی با تعداد بالا فایل سالم و آلوده آزمایش کرد و سپس براساس نتایج بدست آمده به بهبود روش مذکور پرداخت. همچنین بررسی روش های دیگر دسته بندی مانند درخت تصمیم[۱] ، روش بیزین[۲] و... می تواند بروی این روش های وزن دهی اعمال شود. کار دیگری که می توان در این زمینه ارایه کرد یافتن روش وزن دهی جدیدی است بر مبنای صورت مسئله ای که تعریف شده است. به عبارت دیگر با توجه به اینکه روش وزن دهی موجود برای دسته بندی متن ارایه شده است تلاش شود روش وزن دهی جدیدی با توجه بردار ویژگی ارایه شده مطرح کرد.

در جنبه دوم برای ادامه کار می توان بروی ارایه روش دسته بندی با استفاده از پردازش سیگنال کار کرد. به عبارت دیگر با توجه به اینکه تحلیل قسمت منبع می تواند ما را به الگوی سیگنالی برساند که براساس آن بتوان فایل آلوده را شناسایی کرد؛ به دنبال بدست آوردن الگوی سیگنالی مناسب از قسمت آخر فایل اجرایی باشیم به عنوان مثال می توانیم اختلاف ۲ بایت های مجاور در این قسمت را مورد بررسی قرار دهیم.

---

[۱] Decision Tree

[۲] Navie Bayes

منابع

[1] Abou-Assaleh T, Cercone N, Keselj V, Sweidan R. N-gram based detection of new malicious code. In: Proceedings of the 28th annual international computer software and applications conference; 2004

[2] Moskovitch R, Stopel D, Feher C, Nissim N, Elovici Y. Unknown malcode detection via text categorization and the imbalance problem. In: IEEE Intelligence and Security Informatics, Taiwan; 2008.

[3] Siddiqui M, Wang MC, Lee J. Data mining methods for malware detection using instruction sequences. Artificial Intelligence and Applications 2008.

[4] Bilar D. Opcodes as predictor for malware. International Journal of Electronic Security and Digital Forensic;1:2, 2007

[5] Moskovitch R, Feher, Tzachar N, Berger E, Gitelman M, Dolev S, et al. Unknown malcode detection using OPCODE representation. In: European conference on intelligence and security informatics 2008 (EuroISI08), Esbjerg, Denmark; 2008

[6]E. Leopold and J. Kindermann. Text categorization with support vector machines: How .to repre- sent texts in input space Journal of Machine learning 46 423-444- 2002

[7]K .sparck Jones, Indexing term weighting. Information Storage and Retrieval vol .9pp619-663 ,1995

[8]G. Salton , c.s.Yang, On the specification of termvalue in automatic indexing,journal of documentation, vol29 no4 pp351-357 ,1973

[9]G . Saltpn, J Allan, A. Singhal, Automatic text decomposition and structuring, Inforamtionprocessing and mangment, vol 32 no 2 pp127-138 ,1996

[10]M .LAn, S.Y. Sung, H.B LOW, C.L.Tan, A comparative Study on term Weighting Schemes for Text Categorization, IEEE International on Neural Networks pp 546-551, 2005

[11] Ding Yuxin, Yuan Xuebing, Zhou Di, Dong Li, An Zhancha Feature representation and selection in malicious code detection methods based on static system calls, computers & security 30514-524,2011

[12] Zhang B, Yin J, Hao J, Zhang D, Wang S. Malicious codes detection based on ensemble learning. Autonomic and trusted computing;. p. 468–27. 2007

[13]Menahem E, Shabtai, Rokach L, Elovici Y. Improving malware detection by applying .multi-inducer ensemble. Computational Statistics and Data Analysis 2008

[14] I. Santos et al., Opcode sequences as representation of executables for data-mining- .based unknown malware detection, Inform. Sci. 2011

[15]Karim E, Walenstein A, Lakhotia A, Parida L. Malware phylogeny generation using .permutations of code. Journal in Computer Virology;1(1–2):13–23. 2005

[16] Dolev S, Tzachar N. Malware signature builder and detection for executable code. .Patent application; 2008

[17] Schultz M, Eskin E, Zadok E, Stolfo S. Data mining methods for detection of new malicious executables. In: Proc. ofthe IEEE symposium on security and privacy. p. .178–84. ; 2001

[18] Henchiri O, Japkowicz N. A feature selection and evaluation scheme for computer virus detection. In: Proceedings of ICDM-2006, Hong Kong;. p. 891–95. 2006

[19] M. Christodorescu and S. Jha, "Static Analysis of Executables to Detect Malicious .Patterns", in Twelfth Security Symposium, pp. 169-186, 2003

[20] Dolev S, Tzachar N. Malware signature builder and detection for executable code. Patent application; 2008.

[21] Kolter J, Maloof M. Learning to detect and classify malicious executables in the wild. Journal of Machine Learning Research;7:2721–44. 2006

[22] D. K. S. Reddy, A. K. Pujari. N-gram analysis for computer virus detection. Journal of Compute Virology. France;2: 231239, 2006.

[23] D. Reddy, S. Dash, A. Pujari. New malicious code detection using variable length n-grams. International Conference on Information Systems Security. Germany;276-288; 2006.

[24] Bergeron J, Debbabi M, Desharnais J, Erhioui MM, Lavoie Y, Tawbi N. Static detection of malicious code in executable programs. In: Symposium on requirements engineering for information security (SREIS'01), Indianapolis, Indiana, USA; March. p. 157e166; 2001.

[25] M. Christodorescu, S. Jha, Static analysis of executables to detect malicious patterns, in: Proceedings of the 12th USENIX Security Symposium, , pp. 169–186; 2003.

Abstract

Due to proliferate of new malware and using obfuscation techniques in malware, research in unknown malware detection is essential . In the commercial antivirus, complimentary combination of signature based method and heuristic is used; nevertheless heuristic feeble performance results in reliable detection rate depending on database updating. Therefore today the most research in malware detection conducted to enhance and improve heuristic method.

The detection of files infected by unknown malicious code is important task in antivirus using heuristic methods for unknown malicious code detection. In this thesis has studied various some kinds of methods analyzing portable executable (PE) file and then introducing a signal for each PE file. The most reliable method is concluded based on Resource section analysis of PE files . To show reliable performance of proposed method, an unknown malicious code detection method based on text classification technique with using feature vector extracted from proposed method achive 99.10 accuracy.

Keywords: Infected file, Unknown malicious code, Signal processing, unknown malware detection

Shahrood University of Technology

Faculty of Computer Engineering

Infected Files Analysis Based on Signal Processing

Thesis

Submitted in Partial Fulfillment of the Requirements for the Degree of

Master of Science (M.Sc)

Aboulfazl Sarkardei

Supervisor

Dr. Ali Akbar Pouyan

Associate supervisors

Dr. Hamid Hassanpour

Javad Kia

Date: February 2013