مجله علمی پژوهشی «علوم و فناوری‌های پدافند نوین»
سال پنجم، شماره ۲، تابستان ۱۳۹۳؛ ص ۱۵۷-۱۴۷

محافظت از سیستم عامل در مقابل جاسوس‌افزارها و منحرف‌سازی آن‌ها

دانیال جواهری[1*]، سعید پارسا[2]

۱- مربی دانشکده تحصیلات تکمیلی، گروه رایانه، دانشگاه آزاد اسلامی واحد بروجرد

۲ – دانشیار دانشکده مهندسی رایانه، دانشگاه علم و صنعت ایران

(دریافت: ۹۳/۰۲/۰۷، پذیرش: ۹۳/۰۶/۱۲)

چکیده

در این مقاله روش جدیدی برای شناسایی، رهگیری و مقابله با جاسوس‌افزارها به ویژه ضبط کننده‌های صفحه کلید، مسدود کننده‌ها و تصویربردارها از صفحه نمایش ارائه شده است. در این روش تشخیص جاسوس‌افزارها بر مبنای تحلیل رفتار آنها به صورت پویا انجام می‌شود و پس از تشخیص وجود جاسوس‌افزار، اقدام به رهگیری برای شناسایی پردازه، فایل‌های اجرایی روی دیسک سخت و تعاملات جاسوس‌افزار با شبکه در یافتن مقصد مورد نظر جاسوس‌افزار و اطلاعات سرقتی می‌نماید. روش پیشنهادی پس از رهگیری، اقدام به مقابله با بدافزارهای مذکور می‌کند این مقابله شامل ختم اجباری پردازه رهگیری شده، حذف فایل اجرایی آن از دیسک سخت یا منحرف‌سازی جاسوس‌افزارها با تحویل اطلاعات غلط و تغییر مقصد آنها می‌باشد در این طرح، رهگیری و مقابله توسط دایورهایی در سطح هسته سیستم عامل، پیشنهاد و پیاده‌سازی شده است تا به صورت مستقل از توابع سامانه‌ای سطح کاربر و محدودیت‌های اعمالی سیستم عامل عمل نماید. این مقاله همچنین، امنیت سامانه پیشنهادی را مورد بررسی قرار داده و راهکاری برای مقابله با مسدود کننده‌ها و نحوه ساخت یک صفحه کلید مجازی امن و غیر قابل رهگیری نیز ارائه می‌دهد تا بتوان به هدف اصلی این مقاله که ایمن کردن محیط سیستم عامل از وجود جاسوس‌افزارها می‌باشد، دست یافت. در این مقاله کارایی روش پیشنهادی از نظر میزان دقت در کشف و مقابله مؤثر با جاسوس‌افزارها ارزیابی شده و نشان داده می‌شود که روش پیشنهادی می‌تواند با دقت نزدیک به ۹۶ درصد وجود جاسوس‌افزارها را تشخیص و با موفقیت ۱۰۰ درصد سیستم عامل را از وجود جاسوس‌افزارهای کشف شده تمیز دهد و با مقایسه با برترین برنامه‌های ضد جاسوس‌افزارها در دنیا نشان داده می‌شود که روش پیشنهادی از ابعاد مختلف کاملاً قابل رقابت و در مواردی برتر از نمونه‌های خارجی است.

کلید واژه‌ها: ضد جاسوس افزار، ضد ضبط کننده صفحه کلید، ضد مسدود کننده، منحرف‌سازی جاسوس‌افزار، تحلیل پویا بدافزار.

Protection of Operation System against Spywares and their Diversion

D. Javaheri[*], S. Parsa

College of Post-Graduate Education, Islamic Azad University, Borujerd Branch

(Received: 27/04/2014; Accepted: 03/09/2014)

Abstract

In this article, a new method for detection and interception of Spywares specifically key loggers, blockers and screen recorders is proposed. After detecting a malicious behavior, at run time by dynamic behavioral analysis, its corresponding process and executable file are located. All the interaction of the underlying network are logged and analyzed to extract the destination and source of the stolen information which was support to be transferred by the spyware. After the malicious code is analyzed, the process in the main memory is terminated and its executable and image files are removed from the hard disk, in addition it can deliver junk information to spyware or caused diversion of its destination. The proposed method tracks and intercepts malicious code through the kernel drivers belonging to the operation system. In this way, all the system functions in user mode and all the limitations and constraint imposed by the operating system can be bypassed and ignored. In this article, the security of the proposed method is also considered and a new method for interception of blockers and construction of secure virtual keyboards is presented. In this way, the main target of the proposed method to secure the operation system environment of any spywares can be achieved. Finally, the accuracy of detection and success reaction against spywares are evaluated. The accuracy was 96% and reaction rate was 100%. Comparing these results with top famous anti-spyware application proved that the proposed methods is competitive and is better in some features.

Keywords: Anti-Spyware, Anti-Key Logger, Anti Blocker, Diversion Spyware, Dynamic Malware Analysis.

[*] Corresponding Author E-mail: d.javaheri@iaub.ac.ir

۱. مقدمه

از دیر باز محبوبیت سیستم عامل ویندوز که در این مقاله به اختصار سیستم عامل ذکر می‌شود در بین کاربران سامانه‌های رایانه‌ای چه کاربران شخصی و چه اداری بسیار چشمگیر بوده است. این امر سبب شده تا این سیستم عامل به عنوان هدف اصلی برای برنامه‌نویسان قرار گیرد. در این راستا برنامه‌نویسان بدخواه نیز عمده حملات خود را معطوف به این سیستم عامل نموده‌اند. از جمله خطرناک‌ترین این حملات، حمله با قصد سرقت اطلاعات و جاسوسی از فعالیت‌های کاربران می‌باشد. در این مقاله ضمن تشریح عملکرد این بدافزارها، راهکاری برای شناسایی، رهگیری و مقابله با آنها ارائه می‌شود و کارایی آن مورد ارزیابی و مقایسه قرار می‌گیرد.

در ادامه این مقاله، در قسمت دوم به معرفی و تشریح نحوه عملکرد جاسوس‌افزارها پرداخته می‌شود. در قسمت سوم روش پیشنهادی برای کشف جاسوس‌افزارها ارائه شده و روش پیاده‌سازی آن نیز شرح داده می‌شود. در قسمت چهارم راهکارهایی برای رهگیری این بدافزارها و نحوه مقابله با آنها نیز ذکر می‌شود و در قسمت پنجم به ارزیابی و مقایسه روش پیشنهادی پرداخته می‌شود.

۲. نحوه عملکرد جاسوس‌افزارها

تمام ابزارهای جاسوسی در زمره بدافزارها قرار می‌گیرند که خود با توجه به سازوکار عملکرد یا نوع اطلاعات سرقتی تقسیم‌بندی می‌شوند. سه دسته بسیار خطرناک و پر کاربرد این بدافزارها، ضبط کننده‌های صفحه کلید[1]، مسدود کننده‌ها[2] و تصویر بردارها[3] از صفحه نمایش می‌باشند [۱]. در این قسمت مقاله نحوه عملکرد این بدافزارها را برای شناخت و مقابله شرح داده می‌شود. رفتار این بدافزارها بسیار شبیه یک دیگر است. از این حیث که همگی برای رسیدن به اهداف خود، یعنی سرقت اطلاعات نیازمند قلاب اندازی[4] به امکانات سیستم عامل به ویژه توابع سامانه‌ای هستند. هدف از قلاب اندازی این است تا با واسط قرار گرفتن بین منابع سیستم عامل و درخواست‌های وارده شده از سوی کاربر یا برنامه‌های کاربردی نصب شده روی سامانه قربانی، اقدام به ثبت و سرقت اطلاعات لازم نمایند. بسیاری از این بدافزارها می‌توانند اطلاعات سرقت شده را در زمان‌هایی خاص با استفاده از اتصال شبکه برای مقصدی خاص ارسال کنند.

۲-۱. قلاب اندازی و انواع آن

قلاب اندازی، یک مفهوم استفاده شده برای به دست آوردن کنترل جریان اجرایی برنامه بدون تغییر و Compile مجدد کد منبع آن است. این کار، توسط متوقف سازی فراخوانی تابع و هدایت مجدد آن به کد سفارشی شده، به دست می‌آید. با ارائه کد سفارشی، هر عملیاتی را می‌توان اجرا نمود. پس از آن قابلیت‌های اصلی تابع می‌تواند اجرا شده و نتیجه می‌تواند یا به سادگی برگشت داده شود و یا تغییر داده شده و برای انتقال کنترل به کدی که تابع قلاب شده را فراخوانی کرده، برگشت داده شود. بنابراین قلاب سازی یک ابزار مناسب و کاملی را برای تحلیل مخرب‌ها به صورت پویا فراهم می‌کند [۲ و ۳]. قلاب‌های درون خط به طور مستقیم، بایت‌های کد تابع را در حافظه بازنویسی می‌کنند. به طور خاص، تنها چند دستورالعمل با یک دستورالعمل پنج بایتی پرش jmp به تابع قلاب جایگزین می‌شوند. دستورالعمل‌های جایگزین شده در تابع trampoline که به عنوان نقطه ورودی جدید به فراخوانی API اصلی استفاده می‌شود ذخیره می‌شوند. در تابع قلاب، ثبت اطلاعات و اصلاح آرگومان‌ها قبل از اجرای تابع API اصلی می‌تواند انجام شود [۴]. این نوع از قلاب اندازی مستلزم تغییر در فایل‌های سیستم عامل است. بنابراین پیاده‌سازی آن بسیار سخت بوده و گاهی سبب ناسازگاری در سیستم عامل می‌شود.

روش دیگر در قلاب اندازی، تغییر آدرس وقفه‌های سامانه‌ای است. اما پیاده‌سازی این روش بسیار سخت و در بسیاری از مواقع غیر ممکن است. زیرا به ندرت اطلاعاتی در خصوص وقفه‌های سامانه‌ای منتشر شده است و نمی‌توان دریافت که یک وقفه با یک کد خاص برای چه رویدادی تعریف شده است. از آنجا که جدول توصیف‌گر این وقفه‌ها[5] پایین‌ترین سطح در هسته سیستم عامل قرار دارد، تعداد آنها نیز زیاد است. شناسایی آنها با راهکارهایی نظیر اشکال‌زدایی هسته سامانه توسط ابزارهایی مثل Windbg هم در عمل غیر ممکن است.

روش دیگری که بسیاری از بدافزارها برای قلاب اندازی استفاده می‌کنند. تغییر آدرس توابع سامانه‌ای[6] است. این آدرس‌ها در جدولی به نام SSDT و SSDT Shadow ذخیره می‌شوند. بنابراین یکی از مناسب‌ترین نقاط برای قلاب اندازی توسط بدافزارها جدول توصیف‌گر توابع سامانه‌ای[7] و جدول توصیف‌گر توابع سامانه‌ای سایه[8] می‌باشد. این دو جدول مشخص کننده آدرس توابع سامانه‌ای سطح هسته در حافظه سامانه است بنابراین تمام درخواست‌های گذار از سطح کاربر و رسیدن به سطح هسته نیاز به یافتن آدرس توابع مورد نظرشان در این جدول‌ها هستند.

تفاوت دو جدول SSDT و SSDT Shadow در این است که جدول SSDT نگاه دارنده آدرس توابع سامانه‌ای برای کار با هسته سیستم عامل و جدول SSDT Shadow حاوی آدرس توابع سامانه‌ای مرتبط با علیات‌های گرافیکی و پنجره‌های سیستم عامل است.

گذار از سطح کاربر به سطح هسته توسط دستور SysEnter برای سیستم عامل‌های ویندوز، از نسخه XP به بعد و قبل از آن توسط وقفه شماره Int 2e H صورت می‌گیرد. تقریباً تمامی بدافزارهای جاسوسی با قلاب اندازی به این جدول‌ها و جایگزین کردن آدرس

[1] Key Loggers
[2] Blockers
[3] Screen Recorders
[4] Hooking

[5] Interrupt Descriptor Table
[6] System Function Address Patching
[7] System Service Descriptor Table (SSDT)
[8] Shadow SSDT

توابع موجود در آنها با آدرس توابع جعلی که خود از پیش مهیا کرده‌اند، کنترل را به دست گرفته و ضمن سرقت اطلاعات مورد نظرشان از بافرهای ورودی و پارامترهای ارسال شده به توابع مورد نظر، دوباره تابع سامانه‌ای اصلی را با پارامترهای درخواست شده فراخوانی می‌کنند تا در روند اجرایی برنامه‌ها خللی ایجاد نشود [۵ و ۶]. در ادامه به طور خاص بر عملکرد ضبط کننده‌های صفحه کلید، مسدود کننده‌ها و تصویربردارها از صفحه نمایش تمرکز می‌کنیم.

۲-۲. ضبط کننده‌های صفحه کلید

ضبط کننده‌های صفحه کلید برنامه‌هایی هستند که اقدام به ثبت کلیدهای فشرده شده توسط کاربر در زمان فعال بودن سامانه یا زمان‌هایی حساس می‌کنند. منظور از این زمان‌های حساس، مواقعی است که کاربر در حال انجام عملیات مهمی با رایانه خود است مثلاً می‌خواهد وارد رایانامه خود شود یا خرید اینترنتی انجام دهد و یا در حال نوشتن سند متنی باشد. اما یک بدافزار جاسوسی چگونه این زمان‌ها را تشخیص می‌دهد؟ بدافزارهای جاسوسی با استفاده از منابع سامانه‌ای می‌توانند این زمان‌ها را تشخیص دهند. مثلاً زمانی که یک پنجره مرورگر اینترنت باز است یا زمانی که پنجره‌ای از یک برنامه ویرایشگر متن باز است، عمل می‌کنند. جاسوس‌افزارها زمان‌های مذکور را با استفاده از توابع سامانه‌ای FindWindow و GetWindow و جستجوی وجود پنجره فعالی با نام مرورگرهای اینترنت نظیر IE و یا ویرایشگر متن نظیر Word تشخیص می‌دهند. در صورت بازگشت مقدار صحیح از توابع مذکور بلافاصله عملیات جاسوسی خود شامل ثبت کلیدهای فشرده شده یا تصویربرداری از صفحه را آغاز می‌کنند. روشی دیگر استفاده از توابع سامانه‌ای Process32First و Process32Next برای جستجو در بین پردازه‌های فعال سامانه جهت یافتن پردازه‌ای خاص مانند پردازه مرورگر اینترنت است. بدافزارهای مذکور جستجوی فوق را در حلقه‌هایی در زمان فعالیت کاربر انجام می‌دهند. این روش با فیلتر کردن زمان‌های کم اهمیت و محدود شدن تجسس به زمان‌های حساس سبب می‌شود حجم اطلاعات سرقتی به شدت کاهش یافته و عملیات تحلیل اطلاعات سرقتی برای سارقین تسهیل شود. مزیت دیگر این روش زمانی نمایان می‌شود که این اطلاعات سرقتی قرار است از طریق اتصال شبکه به سامانه دیگری ارسال شود که با کاهش حجم اطلاعات ارسالی می‌توان علاوه بر کوتاه شدن زمان ارسال از مشکوک شدن کاربر به ویژه در مراکزی که حجم ترافیک توسط مدیر شبکه کنترل می‌شود جلوگیری کرد.

ضبط کننده‌های صفحه کلید برای رسیدن به مقصود خود که اطلاع از کلیدهای فشرده شده است اقدام به قلاب اندازی به توابع سامانه‌ای NtUserGetAsyncKeyState و NtUserGetKeyState از جدول SSDT Shadow می‌کنند. توابع فوق مقدار دریافتی از توابع بالاتر در سطح کاربر را به درایورهایی که مدیریت صفحه کلید را برعهده دارند تحویل می‌دهد. هدف ضبط کننده‌های صفحه کلید این است که با واسط قرار گرفتن بین این توابع سامانه‌ای و درخواست‌های دریافت شده از کاربر ضمن آگاه شدن از کلیدهای فشرده شده با خواندن مقادیر پارامترهای توابع مذکور، آنها را ثبت کنند. ضبط کننده‌های صفحه کلید عمدتاً این اطلاعات را در فایل‌های متنی مخفی ذخیره کرده تا در زمانی مناسب آنها را برای مقصدی خاص از طریق اتصال شبکه به صورت رمزنگاری شده ارسال کنند [۶ و ۷] علت رمزنگاری اطلاعات ارسالی این است که در صورت شنود بسته‌ها در شبکه توسط ابزارهای نظارتی مانند دیواره‌های آتش[1] به سرقت اطلاعات، مشکوک نشوند.

۲-۳. مسدود کننده‌ها

سیستم عامل ویندوز به صورت پیش فرض یک درایور برای مدیریت صفحه کلید و موشواره نصب می‌کند. تمام صفحه کلیدها و موشواره‌ها با درایور استاندارد سیستم عامل سازگار هستند. برخی از صفحه کلیدها و موشواره‌ها که دارای دکمه‌های اضافی برای عملکردهای خاص منظوره هستند که درایور استاندارد سیستم عامل پاسخگوی امکانات اضافی آنها نیست، ناچار هستند برای استفاده از امکانات اضافی درایور خاص خودشان را نصب کنند. اما این درایورها چگونه عمل می‌کنند؟ بسیاری از آنها فیلتر درایورهایی هستند که با قرار گرفتن در بالای درایور اصلی صفحه کلید یا موشواره در فضای حافظه مدیر ورودی/خروجی[2] سیستم عامل با استفاده از امکانات سکوی فیلترینگ سیستم عامل یا WFP[3] اقدام به ضبط کردن بسته‌های ارسال ارسال شده به سمت دستگاه سخت‌افزار و تفسیر آنها می‌کنند [۹].

برنامه‌نویسان بدخواه با علم به این موضوع و وجود همچنین امکانی در سیستم عامل شروع به نوشتن فیلتر درایورهایی برای مدیریت صفحه کلید و موشواره کردند. این فیلتر درایورها با قرار گرفتن بالاتر از درایور استاندارد سامانه در فضای حافظه مدیر ورودی/خروجی سیستم عامل برای صفحه کلید یا موشواره تمام درخواست‌های دریافت شده از سمت کاربر را ضبط می‌کنند. برخی از این بدافزارها پس از ضبط بسته‌های دریافتی از کاربر آنها را حذف می‌کنند. بدین ترتیب ارتباط کاربر با صفحه کلید و موشواره قطع می‌شود. این دسته از بدافزارها تحت عنوان مسدود کننده‌ها[4] شناخته می‌شوند.

سازوکار دیگر مسدود کننده‌ها قلاب اندازی به تابع سامانه‌ای SendInput از کتابخانه User32.dll در رینگ ۳ و یا تابع معادل آن در رینگ ۰ سیستم عامل یعنی NtUserSendInput است و از این طریق درخواست‌های وارده به تابع مذکور را پاسخ نمی‌دهند بدین ترتیب کلیدهای فشرده شده توسط کاربر هیچ‌گاه به سیستم عامل اعمال نمی‌شود. اغلب از قلاب به تابع دوم در سطح هسته برای این منظور استفاده می‌شود.

اغلب این بدافزارها پس از قطع کردن ارتباط کاربر با صفحه کلید و موشواره یا به اصطلاح بلاک کردن سامانه با نمایش یک پیغام با عناوین جعلی نظیر پلیس سایبری به کاربر، وی را متهم به غیر

[1] Wireshark
[2] I/O Manager
[3] Windows Filtering Platform
[4] Blocker

قانونی بودن نسخه سیستم عامل یا یک نرم‌افزار نصب شده روی رایانه قربانی می‌کنند و قربانی را برای باز کردن سامانه و عدم ثبت سوء پیشینه ملزم به پرداخت مبلغی به عنوان جریمه می‌کنند. کلاهبرداران اینترنتی در این روش بیشتر توجه خود را به کشورهایی که قانون حق تکثیر[1] را رعایت نمی‌کنند (به ویژه در شرق آسیا) معطوف می‌نمایند.

پس از گسترش این دسته از بدافزارها شرکت‌های تولید کننده برنامه‌های ضد ویروس نظیر Kaspersky اقدام به ارائه ابزارهایی برای مقابله با این دسته از بدافزارها نمودند. در این مقاله نیز روشی برای مقابله با این دسته از بدافزارها ارائه شده است این روش بستر لازم برای تولید یک صفحه کلید امن را نیز مهیا می‌سازد.

۲-۴. تصویربردارها

سازوکار عملکرد بدافزارهای جاسوسی که اقدام به تصویربرداری از صفحه کاربر می‌کنند بدین صورت است که برای گرفتن تصویر از صفحه یک پنجره درخواست خود برای عکس‌برداری از آن پنجره یا صفحه میز کار سامانه، در قالب یک دستگیره[2] به تابع سامانه‌ای GetWindowDC از کتابخانه User32.dll می‌دهند. تابع مذکور به عنوان مقدار بازگشتی تمام محتویات آن پنجره اعم از منوها، برچسب‌ها و مواردی دیگر را در قالب ساختار DC (یک ساختار در سیستم عامل با نام Device Context) باز می‌گرداند. مقدار باز گشتی از این تابع به تابع سامانه‌ای دیگری به نام BitBlt متعلق به کتابخانه GDI32.dll به عنوان آرگمان ورودی داده می‌شود. خروجی این تابع یک تصویر از DC مذکور می‌باشد. سایر پارامترهای این تابع خصوصیات تصویر نظیر طول و عرض آن را مشخص می‌کنند. ساختار توابع سامانه‌ای GetWindowDC و BitBlt به صورت زیر است:

```
HDC GetWindowDC(
_In_  HWND hWnd);
BOOL BitBlt(
  _In_  HDC hdcDest,
  _In_  int nXDest,
  _In_  int nYDest,
  _In_  int nWidth,
  _In_  int nHeight,
  _In_  HDC hdcSrc,
  _In_  int nXSrc,
  _In_  int nYSrc,
  _In_  DWORD dwRop);
```

این ابزارها نیاز به استفاده از توابع سامانه‌ای FindWindow و GetWindow دارند. هدف از فراخوانی توابع مذکور ضبط تصاویر از پنجره‌های برنامه‌ای خاص خواهد بود. بدفزارهای تصویربردار با جستجوی نام یک پنجره با توابع سامانه ای مذکور دستگیره‌ای برای تصویربرداری ایجاد می‌کنند [۶ و ۱۰] هدف از این کار محدود کردن تصویربرداری به یک پنجره خاص نظیر مرورگر اینترنت است. برنامه‌های امنیتی و برنامه‌های حساس برای جلوگیری از جستجو شدن پنجره‌هایشان توسط بدافزارها، اقدام به نام‌گذاری تصادفی برای آنها می‌کنند. این مورد در برنامه‌های اشکال‌زدا نظیر OllyDbg مشاهده شده است که با نام‌گذاری تصادفی پنجره‌هایش از جستجوی آنها برای کشف محیط اشکال‌زدایی توسط بدافزار جلوگیری می‌کند [۱۱].

در ادامه این مقاله به بیان روش پیشنهادی می‌پردازیم. طرح کلی روش پیشنهادی در دو قسمت ارائه می‌شود. در قسمت اول به نحوه کشف جاسوس‌افزارها پرداخته می‌شود و در قسمت دوم نحوه مقابله با جاسوس‌افزارهای کشف شده توضیح داده می‌شود.

۳. روش پیشنهادی برای کشف جاسوس‌افزارها

روش پیشنهادی برای کشف جاسوس‌افزارها رفتار آنها را ملاک قرار داده و با تحلیل رفتاری به صورت پویا (در زمان اجرا) اقدام به شناسایی و رهگیری جاسوس‌افزارها می‌کند. علت استفاده از روش تحلیل رفتاری پویا در مجهز شدن بدافزارهای جدید به انواع امکانات ضد تحلیل رفتاری ایستا نظیر مبهم‌سازی کد[3]، چند ریختی[4] و دگردیسی می‌باشد که امکان کشف با استفاده از روش های مبتنی برامضاء و تحلیل رفتاری ایستا را بسیار سخت و گاهی غیر ممکن می‌کند [۸].

روش پیشنهادی برای استخراج رفتار یک بدافزار تعاملات آن با سیستم عامل را درنظر می‌گیرد تا بر اساس این تعاملات بتواند ماهیت جاسوس افزارها را تشخیص دهد. این مهم در روش پیشنهای توسط درایورهایی در سطح هسته سیستم عامل پیاده‌سازی شده است. علت استفاده از درایورهای سطح هسته علاوه بر استخراج رفتار، مقابله با بدافزار نیز می‌باشد. در روش پیشنهادی پس از کشف، مقابله موثری با جاسوس‌افزار می‌شود که پیاده‌سازی آن ملزم استفاده از درایورهای سطح هسته برای رهایی از محدودیت های سیستم عامل در سطح کاربر مانند UAC[5] و مقابله با تمهیدات تدافعی بدافزار است. برای نظارت بر منابع سامانه‌ای و استخراج رفتار یک برنامه بر اساس تعاملاتش با سیستم عامل، یک درایور اقدام به بارگذاری جدول‌های SSDT و SSDT Shadow به صورت مستقیم از فایل‌های ntoskrnl.exe و win32k.exe می‌کند [۱۲ و ۱۳] بارگذاری مستقیم از فایل‌های مالک جدول‌ها به جای خواندن آنها از حافظه برای شناسایی این بدافزارها بسیار مهم است سپس در ادامه آدرس‌های اصلی برای تمام مدخل‌های این جداول محاسبه می‌شود و ساختار آنها برای استفاده در درایور برنامه تعریف می‌شود. این ساختارها به صورت زیر است [۸]:

```
typedef struct _KSYSTEM_SERVICE_TABLE
{
  PULONG  ServiceTableBase;
  PULONG  ServiceCounterTableBase;
  ULONG   NumberOfService;
  ULONG   ParamTableBase;
}
```

[1] Copyright
[2] Handle
[3] Code Obfuscation
[4] Polymorphism
[5] User Access Control

لازم به ذکر است که هر دو جدول SSDT و SSDT Shadow زیرمجموعه جدول توصیف‌گر سامانه یا SDT[1] هستند. بنابراین نیازمند تعریف و بارگذاری جدول SDT نیز هستیم که ساختار آن به صورت زیر است [۱۴]:

```
typedef struct _KSERVICE_TABLE_DESCRIPTOR
{
  KSYSTEM_SERVICE_TABLE  ntoskrnl;
  KSYSTEM_SERVICE_TABLE  win32k;
  KSYSTEM_SERVICE_TABLE  notUsed1;
  KSYSTEM_SERVICE_TABLE  notUsed2;
}
```

۳-۱. شناسایی ضبط کننده‌های صفحه کلید

اولین قدم بعد از بارگذاری جداول، یافتن آدرس‌های اصلی برای رویه‌های موجود در آنها است. فرمول زیر طریقه یافتن این آدرس‌ها را به ما نشان می‌دهد [۱۲ و ۱۳].

محاسبه آدرس اصلی برای توابع سامانه‌ای در جدول توصیف‌گر سرویس‌های سامانه:

$$\text{Original Address} = \text{ServiceTableBase} + \text{SysEnterAddress} \times 4 \quad (۱)$$

در فرمول فوق، آدرس اصلی برای هر ردیف از جدول که نمایان گر آدرس یک تابع سامانه‌ای است از جمع آدرس آن سرویس نسبت به پایه جدول و چهار برابر آدرس دستور SysEnter است، به دست می‌آید. دستور SysEnter عمل تبدیل دستورات سطح کاربر به دستورات سطح هسته را در سیستم عامل بر عهده دارد [۶].

تا کنون توانستیم آدرس‌های اصلی توابع موجود در جدول SSDT را استخراج نماییم پس از این باید آدرس‌های به دست آمده را با آدرس‌های موجود در جدول SSDT جاری (جدولی که هم اکنون در حافظه سامانه است) مقایسه نماییم. علت این مقایسه یافتن اختلاف در آدرس‌ها برای روتین‌های مورد نظر است. هرگونه مغایرت، در آدرس اصلی و جاری برای توابع NtUserGetAsyncKeyState و NtUserGetKeyState از جدول SSDT Shadow موید وجود یک ضبط کننده صفحه کلید در سامانه است. به دلیل حساسیت زیاد، ساختارهای مذکور هیچ گاه توسط مایکروسافت مستندگذاری نشدند. در شکل (۴) مراحل الگوریتم کشف ضبط کننده صفحه کلید آورده شده است.

۳-۲. شناسایی مسدود کننده‌ها

الگوریتم شناسایی این دسته از بدافزارها نیز کاملاً مشابه با الگوریتم شناسایی ضبط کننده‌های صفحه کلید است با این تفاوت که وجود قلاب به تابع سامانه‌ای NtUserSendInput کنترل می‌شود که در صورت قلاب بودن تابع مذکور وجود یک مسدود کننده در سامانه قطعی است. تابع مذکور به دلیل حساسیت زیاد هیچ‌گاه از سوی مایکروسافت مستندگذاری نشده است. در ادامه الگوریتم کشف ضبط کننده‌های صفحه کلید آورده شده است.

۱) بار گذاری مستقیم جداول SSDT و SSDT Shadow از فایل‌های سامانه‌ای ntoskrnl.exe و win32k.exe

۲) تعریف ساختار جدول SDT

۳) تعریف ساختار جدول SSDT

۴) تعریف ساختار جدول SSDT Shadow

۵) محاسبه آدرس اورجینال توابع مذکور در جداول فوق

۶) خواندن جدول SSDT و SSDT Shadow جاری (در حافظه سامانه)

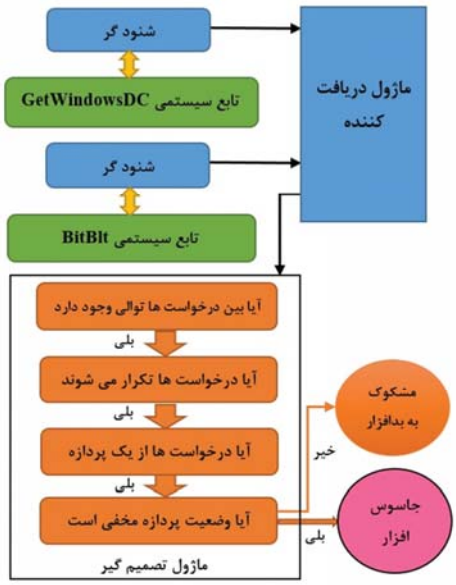
۷) مقایسه آدرس‌های اصلی و آدرس‌های جاری برای توابع مذکور

۸) وجود مغایرت بین آدرس‌ها نشان‌گر ایجاد قلاب به توابع مذکور و وجود یک شنودگر است.

۳-۳. شناسایی تصویربردارها

در خصوص شناسایی برنامه‌های تصویربردارها روش پیشنهادی بدین صورت عمل می‌کند که یک درایور در سطح هسته سیستم عامل با قلاب اندازی به تابع‌های سامانه‌ای GetWindowDC و BitBlt تمام درخواست‌های واصله به این توابع را شنود می‌کند. سپس با در نظر گرفتن میزان تکرار درخواست‌ها به این توابع برای یک پردازه خاص و همچنین وجود توالی بین فراخوانی‌ها برای این دو تابع به جاسوسی بودن آن پردازه مشکوک خواهد شد. در ادامه با بررسی حالت آن پردازه در صورتی که در حالت مخفی یا اجرا در پس‌زمینه باشد جاسوسی بودن آن تأیید می‌شود. قوانین فوق به صورت‌های دیگر نیز در پایگاه دانش روش پیشنهادی تعریف شده است. عناصر اصلی فرمول کشف رفتار تصویربردارها شامل تعداد تکرارها، کران‌های تعریف شده برای این تکرارها، وجود توالی بین دو تابع مذکور در کنار حالت پردازه و یکتا بودن آن است. قوانین تولید شده بر اساس استخراج دانش از پارامترهای فوق، تصمیم‌گیری در خصوص جاسوسی بودن یا نبودن درخواست‌ها را مشخص می‌کند. در صورت تأیید جاسوسی بودن بلافاصله ضمن اعلام به کاربر، پردازه مربوطه به زیرسامانه‌های رهگیری و مقابله کننده برای رهگیری و مقابله معرفی می‌شود. شرح وظیفه و نحوه عملکرد این زیر آنها در قسمت بعد آورده شده است. در شکل زیر الگوی کشف تصویربردارها نشان داده شده است:

[1] System Descriptor Table



شکل ۱. الگوریتم کشف تصویربردارها

بزرگ‌ترین چالش موجود برای پیاده‌سازی این روش، بارگذاری و قلاب اندازی به جدول SSDT Shadow است. برای بارگذاری جدول SSDT آدرس آن با استفاده از یک تابع خروجی در ntoskrnl.exe به نام KeServiceDescriptorTable به دست می‌آید [۶] ولی آدرس جدول SSDT Shadow را باید خود محاسبه کنیم. سختی این کار در آن است که این آدرس در نسخه‌های مختلف ویندوز متفاوت می‌باشد به عنوان مثال در ویندوز XP آدرس جدول SSDT Shadow به اندازه ۶۴ بایت قبل از آدرس جدول SSDT است. بنابراین با یافتن آدرس جدول SSDT می‌توانیم آدرس جدول SSDT Shadow را نیز پیدا کنیم، ولی در نسخه‌های مختلف ویندوز ۷ و ۸ این جدول بعد از SSDT در حافظه بارگذاری می‌شود و بسته به نسخه ویندوز آدرس بارگذاری جدول SSDT Shadow متفاوت است به عنوان مثال در سیستم عامل ویندوز ۷ نسخه Professional با نسخه Ultimate متفاوت است. در روش پیشنهادی با استفاده از تابع سامانه‌ای PsGetVersion ابتدا نسخه سیستم عامل را به دست آورده سپس متناسب با نسخه ویندوز اقدام به یافتن آدرس جدول SSDT Shadow می‌شود. میزان اختلاف آدرس جدول SSDT Shadow با جدول SSDT برای تمام نسخه‌های مختلف سیستم عامل ویندوز محاسبه شده و در پایگاه دانش روش پیشنهادی قرار داده شده است تا بر اساس نسخه ویندوز با اضافه و کم کردن مقادیر محاسبه شده نسبت به آدرس جدول SSDT که همیشه با رویه‌ای ثابت به دست می‌آید آدرس جدول SSDT Shadow هم به دست آید.

۴. روش پیشنهادی برای مقابله با بدافزارهای جاسوسی

در روش پیشنهادی، مقابله با جاسوس‌افزارهای کشف شده در دو مرحله انجام می‌شود. در مرحله اول پس از تشخیص وجود جاسوس‌افزار فعال بر روی سامانه عملیات رهگیری برای تعیین جزئیات شامل پردازه مربوط به جاسوس‌افزار و فایل اجرایی آن روی دیسک سخت به صورت بلادرنگ آغاز می‌شود و در گام بعدی با بدافزار مقابله می‌شود یعنی پردازه آن به صورت اجباری ختم و فایل اجرایی آن نیز حذف می‌شود.

طرح معماری برای پیاده‌سازی روش پیشنهادی از زیرسامانه‌های رهگیری و مقابله تشکیل شده است. زیرسامانه رهگیری[1] وظیفه رهگیری بدافزار کشف شده برای تعیین شماره پردازه مربوط به بدافزار و مسیر فایل اجرایی آن بر روی دیسک سخت را برعهده دارد و زیرسامانه مقابله[2] وظیفه مقابله با بدافزار رهگیری شده برای ختم پردازه و حذف فایل اجرایی آن را عهده دار است. زیرسامانه رهگیری از سه ماژول رهگیر پردازه[3]، رهگیر فایل[4] و رهگیر شبکه[5] برای رهگیری پردازه، فایل اجرایی و تعاملات شبکه تشکیل شده است. زیر سامانه مقابله، از ماژول‌های خاتمه دهنده اجباری[6] و حذف کننده مستقل[7] برای ختم اجباری پردازه و حذف مستقل فایل از سامانه، پس از رهگیری استفاده می‌کند. همه مؤلفه‌های مذکور توسط ماژول تصمیم‌گیر که یک برنامه کاربردی می‌باشد در سطح کاربر می‌باشد کنترل و مدیریت می‌شوند.

۴-۱. زیرسامانه رهگیری

پس از کشف بدافزار، وجود آن به ماژول‌های رهگیر[8] از زیرسامانه رهگیری اعلام می‌شود. در این زیرسامانه ابتدا درخواست به ماژول رهگیر پردازه داده می‌شود. وظیفه این ماژول یافتن شماره شناسایی پردازه مذکور است که این مهم را با استفاده از تابع سامانه‌ای سطح هسته PsGetProcessId انجام می‌دهد. بلافاصله شماره پردازه به دست آمده تحویل ماژول رهگیر فایل که دومین ماژول رهگیر در زیرسامانه رهگیری است، داده می‌شود. این ماژول نیز در هسته سیستم عامل با استفاده از تابع سامانه‌ای PsGetProcessImageFileName وارد حافظه پردازه هدف شده و با باز کردن ماژول اصلی[9] آن، اقدام به تعیین مسیر فایل اجرایی آن پردازه می‌نماید. همزمان که ماژول دوم از زیرسامانه رهگیری در حال رهگیری محل فایل اجرایی پردازه بدافزار است. ماژول سوم نیز با دریافت شماره پردازه مذکور، بررسی می‌کند که آیا بدافزار با شبکه در تعامل بوده یا خیر و در صورتی که اطلاعاتی را از طریق شبکه ارسال کرده مقصد آن را جستجو می‌کند. این ماژول با قلاب انداختن به توابع سامانه‌ای مرتبط با شبکه در سطح کاربر از طریق تزریق کتابخانه پیوند پویا[10] به حافظه اختصاص یافته به جاسوس‌افزار [۱۹] می‌تواند این تعاملات را رهگیری نماید. شرح این رهگیری به صورت زیر است:

---

[1] Interceptor Subsystem
[2] Reaction Subsystem
[3] Process Interceptor
[4] File Interceptor
[5] Network Interceptor
[6] Force Terminator
[7] Independent Eliminator
[8] Interceptor Modules
[9] Main Module
[10] Dynamic Link Library

بدافزارهای مذکور برای ارسال داده نیاز دارند تا یک سوکت به مقصدشان ایجاد کنند. این سوکت از ترکیب آدرس IP و Port و نوع پروتکل مورد نظر به دست می‌آید. توابع سامانه‌ای Socket, WSASocket امکان تعریف یک سوکت را فراهم می‌کند که بهترین محل برای رهگیری جهت کشف مقصد ارسالی می‌باشد. پس از آن با استفاده از توابع سامانه‌ای Connect, WSAConnect می‌توان به سوکت ایجاد شده متصل شد. پس از برقراری اتصال نوبت به ارسال داده می‌رسد. تابع سامانه‌ای Send ، SendTo و WSASend امکان ارسال اطلاعات را از طریق سوکت ایجاد شده مهیا می‌سازند. ساختار توابع Send به صورت زیر است [۱۵]:

```
int send(
  _In_  SOCKET s,
  _In_  const char *buf,
  _In_  int len,
  _In_  int flags);
```

۴-۲. منحرف سازی جاسوس‌افزار

در روش پیشنهادی می‌توان با ضبط پارامتر دوم از تابع Send یعنی بافر آن، به اطلاعات ارسالی دست پیدا کرد یا در آنها تغییری ایجاد نمود. هدف از تغییر در اطلاعات بافر، تحویل اطلاعات غلط به جاسوس‌افزار برای سرقت است. لازم به ذکر است که تمام توابع مذکور در کتابخانه WSock32.dll قرار دارند. در زیرسامانه رهگیری تمام درخواست‌های کار با شبکه برای پردازه‌های مختلف ثبت می‌شود. این اطلاعات شامل شماره و نام پردازه، نوع بسته ارسالی، آدرس IP و پورت مقصد و زمان ارسال است. بدین ترتیب در زمان کشف ابزار جاسوسی می‌توان با تفسیر اطلاعات ثبت شده برای آن به مقصد و زمان ارسال داده‌ها پی برد. به عنوان گزینه‌های جانبی بانک اطلاعاتی حاوی آدرس IP تمام کشورهای جهان در برنامه قرار داده شده تا بتوان کشور و شهر و مقصد بسته‌ها را نیز به کاربر معرفی نمود. علت ثبت تمام اطلاعات کار با شبکه در این است که بعد از کشف یک برنامه جاسوسی بتوان تعاملاتی که در گذشته با شبکه وجود داشته و اطلاعات سرقت شده را شناسایی نمود. همچنین در روش پیشنهادی می‌توان با تعریف یک سوکت با آدرس IP و Port دلخواه و جایگزینی آن به عنوان پارامتر اول تابع مذکور مقصد جاسوس‌افزارها را به مقصدی دلخواه برای به دست آوردن اطلاعات سرقت شده منحرف ساخت. پس اتمام کار ماژول دوم در زیرسامانه رهگیری، مسیر فایل اجرایی مشخص شده به همراه شماره پردازه به زیرسامانه مقابله کننده تحویل می‌شود.

۴-۳. زیرسامانه مقابله کننده

روش پیشنهادی عملیات مقابله را به صورت پیش فرض انجام می‌دهد، مگر آنکه کاربر با فعال کردن حالت دستی، مقابله را منوط به اجازه خویش از طریق محاوره یا عدم مقابله در صورت تعریف به عنوان استثناء در فایل تنظیمات روش پیشنهادی نماید. با تعبیه شدن این ویژگی در طرح معماری پیشنهادی، کاربر در صورتی که بخواهد توسط ابزاری از صفحه نمایش خود تصویربرداری کند برنامه برای آن هشدار یا اقدامی انجام نمی‌دهد.

خاتمه دهنده اجباری: در زیرسامانه مقابله، ابتدا ماژول ختم کننده اجباری با دریافت شماره پردازه مورد هدف، اقدام به خاتمه دادن به آن به صورت اجباری می‌کند. این ماژول یک درایور در سطح هسته می‌باشد که با استفاده از امتیاز عدم اعمال محدودیت توسط سیستم عامل برای درایورها در سطح هسته، با فراخوانی روتین ZwTerminateProcess اقدام به ختم اجباری برنامه هدف می‌کند. علت اینکه ختم پردازه مذکور از ابتدا صورت نگرفت در این است که برای رهگیری و تعیین محل فایل اجرایی، پردازه بدافزار باید حتماً در حالت اجرا باشد.

حذف کننده مستقل: پس از ختم اجباری پردازش، نتیجه عملیات به ماژول دوم در زیرسامانه مقابله (ماژول حذف کننده مستقل) داده می‌شود و این ماژول همانند ماژول قبلی درایوری در هسته سیستم عامل می‌باشد که به صورت مستقل از توابع سامانه‌ای سطح کاربر اقدام به حذف فایل اجرایی بدافزار هدف می‌کند. این استقلال از توابع سامانه‌ای سطح کاربر سبب رهایی از محدودیت‌های سامانه‌ای برای حذف فایل‌های حفظت شده و سامانه‌ای است چرا که این بدافزارها یقیناً با تمام توان از حذف شدنشان جلوگیری می‌کنند. برای رسیدن به این هدف از روش‌هایی نظیر تغییر خصیصه‌های فایل، برای سامانه‌ای یا فقط خواندنی معرفی کردن آن بهره می‌برند تا برنامه‌های دیگر و حتی خود کاربر سامانه حتی با دسترسی مدیر[1] نتواند آنها را حذف نماید. این محدودیت‌ها برای درایور در سطح کرنل مطرح نیست و یک درایور در سطح هسته سیستم عامل می‌تواند آنها را به صورت مستقل از سطح کاربر حذف نماید. تنها محدودیت ممکن وجود دستگیره‌ای باز به فایل مورد نظر است که در مرحله قبل پردازه مالک آن دستگیره شناسایی و به صورت اجباری خاتمه داده شد تا علاوه بر متوقف کردن برنامه جاسوسی برای حذف آن محدودیتی نداشته باشد.

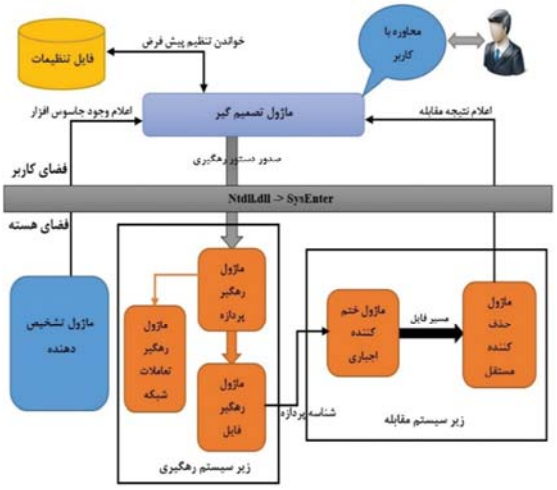
راه دیگر این است که با استفاده از یک فایل سامانه فیلتر درایور و بارگذاری آن در ارتفاعی بالاتر از درایور فایل سامانه در فضای مدیر ورودی/خروجی اقدام به ارسال IRP[2] به دیسک سخت جهت حذف فایل به صورت مستقیم و مستقل از سیستم عامل نمود. در این مقاله از این روش به دلیل پیچیدگی در پیاده‌سازی و همچنین ایجاد لختی در سیستم عامل صرف نظر شده و از همان روش قبلی استفاده شده است.

همان‌طور که در قسمت دوم این مقاله گفته شد، بدافزارهایی تحت عنوان مسدود کننده‌ها که به دلیل تشابه بسیار زیاد سازوکار عملکرد آنها در دسته بدافزارهای جاسوسی قرار گرفته‌اند، ممکن است در روند اجرایی برنامه‌ها به ویژه برنامه‌های ضد ویروس و تحلیل‌گر اخلال ایجاد کنند. در روش پیشنهادی، برنامه در ابتدای شروع به کار یک درایور جداگانه برای مدیریت صفحه کلید و موشواره در سیستم

[1] Administrator
[2] Input/Output Request Packet

عامل نصب می‌کند. ساخت این درایور از ابتدا کار سختی می‌باشد، بنابراین برای سهولت کار از کلاس‌های آماده مایکروسافت استفاده شده است. کلاس‌های ذکر شده امکان ساخت یک درایور استاندار صفحه کلید ۱۲۳ کلیده را فراهم می‌سازد [۱۶]. در روش پیشنهادی هنگام نصب برنامه درایورهای مذکور به عنوان سرویس در سیستم عامل نصب شده و پس از آن برنامه ما تمام تعاملات خود با صفحه کلید و موشواره را از طریق درایور خودش انجام می‌دهد و هیچ گونه نیازی به استفاده از درایور سیستم عامل ندارد. این عامل سبب می‌شود تا در زمان حمله یا آلودگی به بدافزار مسدود کننده که شرح عملکرد آنها در قسمت دوم این مقاله توضیح داده شد در روند اجرایی برنامه پیشنهادی هیچ‌گونه اخلالی ایجاد نشود. در قسمت پایانی این مقاله توضیح داده می‌شود که چگونه از این درایور برای تولید یک صفحه کلید امن نیز استفاده خواهد شد.

در تمام مراحل کشف، رهگیری و مقابله پیام‌هایی برای آگاهی و یا کسب اجازه به کاربر نشان داده می‌شود. در ابتدا پس از تشخیص، پیامی به عنوان هشدار که نشان دهنده وجود بدافزار جاسوسی و نوع آن است، نشان داده می‌شود. در مرحله بعد به صورت بلادرنگ برنامه رهگیری، پردازه یا پردازه‌های مربوط به بدافزار جاسوسی را شناسایی می‌نماید. پس از پایان هر گام پیام موفق یا عدم موفقیت نیز به کاربر نشان داده می‌شود. در شکل زیر معماری روش پیشنهادی، از جمله زیرسامانه‌ها و ماژول‌های رهگیری و مقابله به خوبی نمایان است.



شکل ۲. معماری روش پیشنهادی

۴-۴. ارتباط بین درایورها

یکی از نکات بسیار مهم در روش پیشنهادی ارتباط بین درایورها و امنیت آن به ویژه در سطح هسته و همچنین ارتباط درایورهای سطح هسته با برنامه‌های کاربردی است. عمده‌ترین روش‌های ارتباط بین یک درایور و برنامه سطح کاربر عبارتند از: تعریف حافظه اشتراکی، نگاشت فایل و خط لوله که با بررسی‌های انجام شده در روش‌های ارتباطی موجود، خط لوله مناسب‌ترین روش برای سامانه پیشنهادی بوده است، دلیل آن هم در امنیت و سرعت ارتباطی است. خط لوله‌های ارتباطی به دو دسته خط لوله نامدار[1] و خط لوله بدون نام[2] تقسیم می‌شوند. از خط لوله‌های بدون نام برای ارتباط بین برنامه بر روی یک رایانه به صورت محلی استفاده می‌شود و از خط لوله نامدار برای ارتباط بین برنامه‌ها بر روی چند ماشین به صورت شبکه استفاده می‌شود. در روش پیشنهادی از خط لوله نامدار استفاده شده است [۱۷].

اتصال ارتباط، از طریق خط لوله[3] با فراخوانی لینک‌های نمادین[4] درایور مقصد امکان پذیر می‌شود [۱۸]. این لینک‌های نمادین توسط توابع سامانه‌ای سطح هسته IOCreateSymboliclink و IOCreateUnprotectedSymboliclink ساخته می‌شوند. تفاوت این دو تابع در این است که لینک‌های نمادین ساخته شده با تابع اول فقط در سطح هسته توسط درایورها قابل استفاده است و از دید برنامه‌های سطح کاربر مخفی می‌ماند ولی لینک‌های نمادین ساخته شده با تابع دوم توسط برنامه‌های سطح کاربر هم قابل استفاده است.[۲۰]

به دلیل آنکه معماری روش پیشنهادی برای کشف، رهگیری و مقابله با بدافزارهای مورد نظر نیازمند درایورهایی در سطح هسته با بالاترین سطح دسترسی سیستم عامل است. باید امکان هرگونه سوء استفاده از این درایورها از بین برود. برای این مهم در معماری پیشنهادی دو لایه امنیتی طراحی شده است. در لایه اول درایور، درخواست دریافت شده از سمت برنامه سطح کاربر را برای مشخص شدن پردازه و آدرس فایل اجرایی، رهگیری می‌کند تا اصالت آن تصدیق شود. این اصالت به معنای متعلق بودن درخواست به ماژول تصمیم‌گیری در معماری روش پیشنهادی است. پس از تصدیق لایه اول، در لایه دوم یک کلید از سمت برنامه درخواست کننده به صورت رمزنگاری شده به طرف مقابل ارسال می‌شود. در طرف مقابل، پس از رمزگشایی، کلید مورد اعتبارسنجی قرار می‌گیرد، که در صورت معتبر بودن، درخواست وارده عملیاتی می‌شود. معتبر بودن کلید منوط به قابل تولید بودن آن توسط الگوریتمی خاص است که از قبل در کد برنامه‌ها در هر دو سمت تعبیه شده است. کلید مذکور در هر بار فراخوانی تغییر می‌کند. علت این امر هم جلوگیری از کشف آن توسط هرگونه برنامه اشکال‌زدا و یا نظاره‌گر است. کنترل‌های امنیتی ذکر شده برای جلوگیری از ارسال فرمان به درایورها توسط برنامه‌ای به غیر از ماژول تصمیم‌گیر در معماری روش پیشنهادی است.

۴-۵. صفحه کلید امن

تا کنون در صورت استفاده از برنامه‌های ضد ویروس نظیر Kaspersky متوجه ابزار جالب صفحه کلید امن شده‌اید و یا در دروازه‌های پرداخت‌های الکترونیکی بانک‌ها نیز صفحه کلید مجازی را

[1] Named Pipe
[2] Unnamed Pipe
[3] Pipeline
[4] Symbolic Link

مشاهده کرده‌اید. بسیار جالب است بدانید این ابزارها چه گونه ساخته می‌شوند. در نمونه ابزارهایی که توسط بانک‌ها و سایر وب سایت‌ها در سمت سرور ساخته می‌شوند، کار ساده‌تر است. بدین ترتیب که از عکس‌هایی جهت نشان دادن اعداد و حروف استفاده می‌شود تا کاربر بدون نیاز به تایپ کردن، متناسب با تصویری که کلیک می‌کند، سرور با استفاده از یک دستور ساده آن را در فیلد مربوطه تایپ می‌نماید. در زبان‌های .Net کلاس SendKeys.Send() یک عبارت را در فیلد فعال از فرم جاری چاپ می‌کند بنابراین کاربر از فشردن هرگونه کلید از صفحه کلیدش معاف می‌شود اما اوضاع در سمت کاربر برای طراحی یک صفحه کلید مجازی امن کاملاً متفاوت است. سه راه برای طراحی این ابزارها پیشنهاد می‌شود که در روش پیشنهادی از گزینه سوم استفاده شده است. راه اول آنکه بدنه تابع سامانه‌ای برنامه مذکور در متن برنامه صفحه کلید کپی می‌شود [۱۰] بدین ترتیب برنامه با نشان دادن عکس‌هایی به کاربر با استفاده از تابع درونی خود و مستقل از توابع سامانه‌ای آن را برای کاربر در محل مشخص شده توسط مکان‌نما، ثبت می‌کند بدین ترتیب نیاز به هیچ فراخوانی سامانه‌ای وجود ندارد تا ضبط کننده‌ها با گوش دادن به آنها بتوانند به اطلاعات کلیدهای فشرده شده دست پیدا کنند.

از تابع سامانه‌ای GetKeyState بدین منظور می‌توان استفاده نمود ساختار این تابع به صورت زیر است [۲۱]:

```
SHORT WINAPI GetKeyState( _In_  int nVirtKey);
```

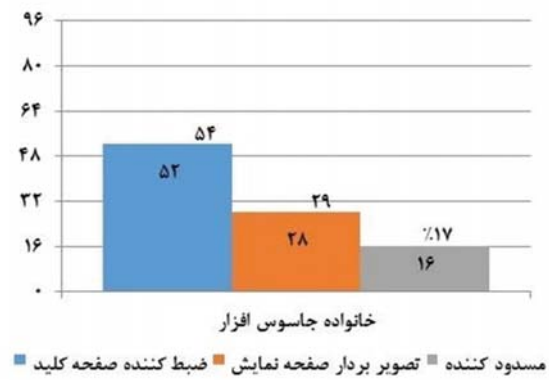
در روش دوم برنامه درخواست خود را به یک درایور سطح هسته سیستم عامل می‌دهد تا در آنجا برنامه سطح هسته پس از اطمینان از عدم وجود قلاب به توابع سامانه‌ای مذکور که در قسمت دوم این مقاله توضیح داده شد اقدام به فراخوانی آنها برای ثبت مقدار مورد نظر کند و در صورتی که برنامه وجود قلاب به توابع مذکور را شناسایی کند، می‌تواند با بارگذاری مجدد جدول SSDT Shadow و یافتن آدرس‌های اصلی، توابع مذکور را با آدرس‌های اصلیشان فراخوانی کند تا در خواست‌های کاربر از رهگیری توسط بدافزارهای جاسوسی ضبط کننده صفحه کلید در امان بماند.

در روش سوم از درایورهای جداگانه برای ساخت صفحه کلید امن استفاده می‌شود این درایورها وظیفه مدیریت دستگاه‌های صفحه کلید و موشواره را بر عهده دارند تا مستقل از درایور سیستم عامل در شرایط آلوده شدن به یک مسدود کننده یا شنود توسط یک ضبط کننده، درخواست‌های خود را به صفحه کلید و موشواره بدهند. بدین ترتیب امکان شنود این درخواست‌ها از بین می‌رود. طریقه ساخت این درایور در قسمت دوم این مقاله بیان شد. وجود این درایور برای مقابله با مسدود کننده‌ها کاملاً ضروری است زیرا در غیاب آن درخواست‌های صفحه کلید و موشواره باید از طریق توابع سامانه‌ای پاسخ داده شوند که این توابع از جمله SendInput توسط مسدود کننده از بین رفته‌اند.

۵. نتایج و بحث

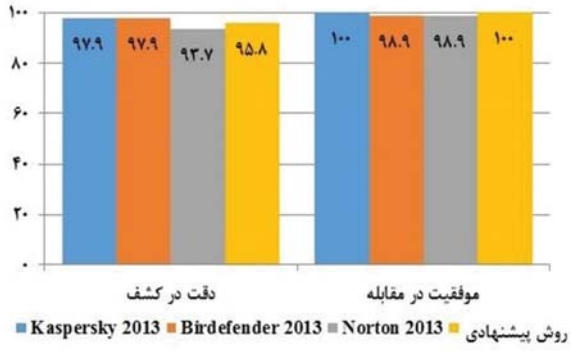
در این قسمت به ارزیابی میزان دقت روش مذکور در شناسایی جاسوس‌افزارها پرداخته و روش پیشنهادی را با چند روش مطرح دنیا مقایسه می‌کنیم.

در ارزیابی روش پیشنهادی از ۹۶ نمونه بدافزار از مراجع [۱] و [۲۲] استفاده شده است این نمونه بدافزارها شامل ضبط کننده‌های صفحه کلید، تصویربردارها و مسدود کننده‌ها است. ذکر این نکته ضروری است که برخی از نمونه بدافزارها هر دو امکان ضبط صفحه کلید و تصویربرداری را به صورت هم‌زمان داشتند مانند جاسوس‌افزار GoldenEye و MaxKeylogger برای تشخیص نوع جاسوس‌افزار، در روش پیشنهادی رفتاری که زودتر کشف شود ملاک قرار خواهد گرفت. همچنین روش پشنهادی امکان رهگیری، شناسایی و مقابله هم‌زمان با چند بدافزار را دارد.



شکل ۳. نمودار تعداد و درصد نمونه بدافزارهای استفاده شده برای آزمون روش پیشنهادی

نمونه بدافزارهای فوق بر روی سیستم عامل ویندوز ۷، نسخه ۳۲ بیتی پس از نصب برنامه‌های امنیتی زیر اجرا شده است. علت اجرای نمونه بدافزارها، کشف آنها فارغ از استفاده از امضاء و بر پایه تحلیل رفتاری پویا و مقابله مؤثر با آنها می‌باشد. نتایج این مقایسه در شکل (۴) و جدول (۱) آورده شده است.



شکل ۴. مقایسه روش پیشنهادی با روش چند برنامه ضد جاسوس‌افزار مطرح دنیا از نظر درصد دقت در تشخیص و درصد موفقیت در مقابله

جدول ۱. مقایسه روش پیشنهادی با چند برنامه ضد جاسوس‌افزار مطرح دنیا از نظر قابلیت‌ها و امکانات

ردیف	برنامه ضد جاسوس افزار	صفحه کلیدامن	رهگیری مقصد جاسوس افزار	منحرف‌سازی جاسوس افزار
۱	Kaspersky 2013	بلی	بلی	خیر
۲	Bitdefender 2013	بلی	خیر	خیر
۳	Norton 2013	خیر	خیر	خیر
۴	روش پیشنهادی	بلی	بلی	بلی

در شکل (۱) نرخ دقت، شاخص نسبت تعداد جاسوس‌افزارهای شناسایی شده توسط هر برنامه نسبت به کل نمونه جاسوس‌افزارها بوده و نرخ واکنش، شاخص تعداد واکنش‌های موفق برنامه در ختم پردازه و حذف یا قرنطینه فایل جاسوس‌افزار نسبت به تعداد جاسوس‌افزار کشف شده توسط هر برنامه می‌باشد. امکان صفحه کلید امن هم پس آلوده کردن محیط توسط چند ضبط کننده صفحه کلید که به صورت تصادفی از نمونه‌های جمع‌آوردی شده، انتخاب گردیدند مورد آزمون قرار گرفت و ملاک آن هم عدم وجود سابقه دکمه‌های فشرده شده توسط صفحه کلید امن در فایل سابقه ضبط شده توسط جاسوس‌افزار بوده است.

امکان رهگیری مقصد جاسوس‌افزار مشخص کننده مقصد مورد نظر جاسوس‌افزار جهت سرقت اطلاعات می‌باشد که روش پیشنهادی این مورد را با دقت کشور، شهر و فراهم کننده سرویس اینترنت[1] مقصد مشخص کند.

اماکن منحرف‌سازی جاسوس‌افزارها در تغییر مقصد اطلاعات سرقت شده و دادن اطلاعات غلط به جاسوس‌افزار تعریف شده [۲۳] و فقط منحصر به روش پیشنهادی این مقاله می‌باشد و دیگر روش‌ها و برنامه‌های موجود این امکان را ندارند.

لازم به ذکر است با عنایت به اینکه برای تحلیل رفتاری پویا نیاز به اجرای جداگانه هر نمونه بدافزار می‌باشد، بنابراین آزمون در یک مقیاس بهینه صورت گرفته و از حیث مقیاس‌پذیری قابل تعمیم به مقیاس‌های بزرگ‌تر می‌باشد. با مقایسه انجام شده، نشان داده شد که روش پیشنهادی از نظر دقت در تشخیص، توانایی در مقابله مؤثر و امکانات جانبی نسبت به برترین برنامه‌های دنیا نه تنها چیزی کم ندارد بلکه در برخی موارد برتری هم دارد.

در روش فوق با توجه به این که کشف جاسوس‌افزارها بر پایه تحلیل رفتار آنها به صورت پویا انجام می‌شود، بنابراین نیاز به یک محیط امن برای اجراء بی‌خطر بدافزارها ضروری است در مرجع [۲۳] روشی جدید برای طراحی و پیاده‌سازی یک محیط امن و هوشمند پیشنهاد شده است.

[1] Internet Service Provider

۶. نتیجه‌گیری

در این مقاله روشی کارا برای شناسایی، رهگیری و مقابله با جاسوس‌افزارها شامل ضبط کننده‌های صفحه کلید، مسدود کننده‌ها و تصویربردارها پیشنهاد و توضیح داده شد. روش پیشنهادی با استفاده از الگوها و الگوریتم‌های ذکر شده، وجود جاسوس‌افزارها را تشخیص می‌دهد. روش پیشنهادی پس از تشخیص وجود جاسوس‌افزار اقدام به رهگیری، مقابله و یا منحرف‌سازی آنها می‌کند در روش پیشنهادی رهگیری جاسوس‌افزار شامل تشخیص پردازه، فایل اجرایی و تعاملات با شبکه و مقابله شامل ختم اجباری پردازه و حذف فایل اجرایی و منحرف‌سازی شامل تحویل اطلاعات غلط و تغییر مقصد جاسوس‌افزار می‌باشد. نحوه پیاده‌سازی روش پیشنهادی برای تولید ابزاری مؤثر در کشف و مقابله با جاسوس‌افزارها نیز تا حد امکان تشریح شد. در پایان روش پیشنهادی از جهات مختلف ارزیابی شد که حاصل آن دقت نزدیک به ۹۶ درصدی در تشخیص و موفقیت ۱۰۰ درصدی در مقابله با جاسوس‌افزارها بود و برای اثبات کارایی با برترین برنامه‌های دنیا مقایسه شد.

۷. تشکر و قدردانی

از آقایان امیر گوران اوریمی و امیر محمدزاده لاجوردی که در پیاده‌سازی روش پیشنهادی و در طرح ضد ویروس بومی، ما را یاری نمودند، تشکر و قدردانی می‌شود.

۸. مراجع

[1] "Virus Sign Malware Data Base"; http://www.virussign.com/, 2014.

[2] Schönbein, C. "PyBox - A Python Sandbox"; Diploma Thesis, May 2011.

[3] Engelberth, M.; Gobel, J.; Schonbein, C.; Freiling, C. "PyBox A Python Sandbox."; In Proc. of Make Available to a Broad Public Recent Findings in Informatics of Computer Science and Information Systems, 2011, 137-138.

[4] Plohmann, D.; Leder, F. "GI Graduate Workshop on Reactive Security for PyBox", University of Bonn, Germany, 2010.

[5] Mohammadzadeh Lajevardi, A. "Design and Implementation of a Behavior-Based Method for Malware Detection"; M.Sc. Thesis, Iran University of Science and Technology, Tehran, 2013 (In Persian).

[6] Schreiber, B. "Undocumented Windows 2000 Secrets: A Programmer's Cookbook"; Addison Wesley Longman Publishing Co, Boston, MA, USA, 2001.

[7] Parsa, S.; Mohammadzadeh Lajevardi, A.; Amiri, M. J. "Propose a Method for Attack to Malware Detector Tools with Hiding System Calls"; In Proc. of 18th Iran Computer Conf., Sharif University of Technology, Iran, 2013. (In Persian)

[8] Javaheri, D. "Design and Implementation a Secure and Intelligent Environment for Safe Malware Analysis"; M.Sc. Thesis, Islamic Azad University, Borujerd Branch, 2014 (In Persian).

[9] Russinovich, M.; Solomon, D.; Ionescu, A. "Windows Internals Part1"; 6th, 2012.

[10] Madou, M.; Anckaert, B.; Moseley, P.; Debray, S.; Sutter, B.; Bosschere, K. "Software Protection through Dynamic Code

Mutation"; In Proc. of Sixth Int. Conf. on Information Security Applications, Heidelberg 2006, 194-206.

[11] Bayer, U.; Moser, A.; Krügel, C.; Kirda, E. "Dynamic Analysis of Malicious Code"; In Proc. of J. in Computer Virology 2006, 2, 67-77.

[12] Hu, Y.; Chen, L.; Xu, M.; Zheng N.; Guo, Y. "Unknown Malicious Executable Detection Based on Run-time Behavior"; In Proc. of Fifth Int. Conf. on Fuzzy Syst. and Knowledge Discovery, China, 2008, 4, 391-395.

[13] Weiqin, M.; Duan, P.; Liu, S.; Guofei, G.; Liu, J. "Shadow Attacks: Automatically Evading System-Call Behavior Based Malware Detection"; In Proc. of J. in Computer Virology 2012, 8, 1-13.

[14] Blunden, A. "The Rootkit Arsenal"; 2th, 2012.

[15] Petzold, C. "Programming Windows"; 6th, 2013.

[16] "Keyboard Filter Driver"; http://code.msdn.microsoft.com/windowshardware/Kbfiltr-WDF-Version-685ff5c4

[17] Silberschatz, P.; Galvin, B.; Gagne, G. "Operating System Concepts"; 9th, 2012.

[18] Tanenbaum, A. S.; Woodhull, A. S., "Operating Systems Design and Implementation"; 3th, the Minix Book.

[19] Berdajs J.; Bosnić, Z. "Extending Applications Using an Advanced Approach to Dll Injection and Api Hooking"; Software: Practice and Experience 2010, 40, 567-584.

[20] Reeves, R. D. "Windows 7 Device Driver"; 1st, 2010.

[21] Hoglund, G.; Butler, J. "Rootkits: Subverting the Windows Kernel"; 1th, 2005.

[22] "CW Sand Box Data"; http://pi1.informatik.uni-mannheim.de/malheur/, 2014.

[23] Javaheri, D. "Design and Implementation a Secure and Intelligent Environment for Safe Malware Analysis"; M.Sc. Thesis, Islamic Azad University, Borujerd Branch, 2014 (In Persian).