بهبود شناسایی ویروس بر اساس امضاء به کمک الگوریتم ژنتیک خود انطباقی

الهام نیکبخت[۱]، سید محمدحسین معطر[۲]، مجید وفایی جهان[۳]

[۱]دانشگاه آزاد اسلامی واحد مشهد، el.nikbakht@gmail.com

[۲]دانشگاه آزاد اسلامی واحد مشهد، moattar@mshdiau.ac.ir

[۳]دانشگاه آزاد اسلامی واحد مشهد، vafaeijahan@mshdiau.ac.ir

چکیده

ویروس‌های کامپیوتری بسیار باهوش می‌باشند و بطور پیوسته تغییر می‌کنند. آن‌ها فعالیت و عمل ویروس‌های بیولوژیکی را شبیه سازی می‌کنند. در این مقاله هدف پیاده سازی الگوریتمی است که بر اساس سیستم‌های ایمنی بیولوژیکی با استفاده از سیستم ایمنی مصنوعی به مقابله با ویروس‌های کامپیوتری بپردازد. تکامل ویروس‌ها برگرفته از فرایندی مشابه با الگوریتم ژنتیک می‌باشد این مسئله الهام بخش ایده‌ای شد که با توجه به این موضوع مسیر تکامل ویروس پیش‌بینی می‌شود و تغییرات بعدی ویروس تشخیص داده می‌شود و در راستای حذف ویروس مورد استفاده قرار می‌گیرد. در این تحقیق تلاش می‌شود تا با استفاده از الگوریتم ژنتیک خود انطباقی به شناسایی امضاء فایل‌های آلوده بپردازد. امضاء فایل‌ها توسط عملگر برش الگوریتم ژنتیک به بخش‌های مختلفی تقسیم می‌شوند سپس هر یک از بخش‌ها با توجه به میزان شباهت به امضاء ویروس، با استفاده از معادلاتی که در الگوریتم ژنتیک خودانطباقی معرفی شده است، جهش می‌یابند و توسط عملگر برش به تعدادی معین قطعه جدید تقسیم می‌شوند.

جهت ارزیابی روش پیشنهادی (SAGA) و روش های مورد مقایسه (GA) و (EMGA)، امضاء ۱۰۰ ویروس تحت عنوان استخر امضاء در نظر گرفته شده است. علاوه بر این دو مجموعه فایل که هر یک حاوی ۲۰۰ فایل هستند به ترتیب ۲۵٪ و ۷۵٪ ویروسی شده است و ارزیابی‌ها بر روی معیار صحت انجام گرفته است. این نتایج کارایی رویکرد پیشنهادی را نسبت به روش مورد مقایسه نشان می‌دهد. به نحوی که در مجموعه فایل‌های ۲۵٪ و ۷۵٪ صحت روش پیشنهادی نسبت به روش مورد مقایسه به ترتیب ۵٪ و ۳۱٪ بهبود را نشان می‌دهد.

واژه‌های کلیدی

بدافزار تکاملی، چندریختی، شناسایی ویروس بر اساس امضاء، الگوریتم کلونال، الگوریتم ژنتیک خود انطباقی

۱ – مقدمه

از زمان پیدایش اولین کد مخرب کامپیوتری در سال ۱۹۸۶، هر ساله تعداد زیاد و قابل توجه‌ای از کدهای مخرب جدید، ظهور پیدا می‌کنند. این رشد سریع کدهای مخرب، همیشه یک گام جلوتر از تلاش‌های متخصصین امنیتی برای ارائه راه حل‌هایی به منظور تشخیص و حذف آن‌ها از سیستم کاربران بوده است. اگرچه بدون در نظر گرفتن این رشد سریع نیز، روش سنتی مقابله با کدهای مخرب، شامل انتظار برای آلوده شدن تعدادی کامپیوتر، تشخیص کد مخرب، طراحی راه حل مقابله و سپس ارائه آن به کاربران و مشتریان، فرآیندی است طولانی و ناکارآمد؛ چرا که در طول این فرآیند امکان وارد آمدن خسارت توسط کدهای مخرب وجود دارد.

با توسعه اینترنت، سرعت تولید و گسترش کدهای مخرب نسبت به قبل، بسیار بیشتر شده است. طبق یک بررسی انجام شده توسط شرکت اف- سکیور تعداد کدهای مخرب کشف شده، فقط در سال ۲۰۰۷ برابر با مجموع تعداد کدهای مخرب کشف شده در ۲۰ سال قبل از آن بوده است و نسبت به سال ۲۰۰۶ رشد صد درصدی را تجربه کرده است[۱]. در بررسی دیگری که توسط شرکت امنیتی پاندا صورت پذیرفته، نشان داده است که روزانه در حدود ۶۵ هزار کد مخرب توسط موتور امنیتی این شرکت در اینترنت رصد می‌شود.

شرکت معتبر سمانتک معتقد است، مسأله تشخیص مخرب یا سالم بودن یک فایل یک مسئله حل نشدنی است[۲]. این شرکت معتقد است، تولید نرم افزاری که به صورت قطعی فایل‌های مخرب را شناسایی کند و خطای مثبت کاذب صفر داشته باشد و همچنین نیاز به بروز رسانی منظم نیز نداشته باشد عملی غیر ممکن است[۲]. از طرفی سیستم‌های کامپیوتری و ارتباطات زیربنایی دیجیتال امروزی به طور شدیدی در معرض انواع مختلف

حملات مخرب قرار دارند. حملاتی که جنبه‌های مختلفی از جمله باج خواهی، انتقام، تروریستی، دزدی اطلاعات، سوء استفاده‌های مختلف و ... را شامل می‌شود. یکی از شایع‌ترین روش‌های این کار استفاده از کد مخرب است؛ که انواع و کاربردهای مختلفی دارد. از جمله می‌توان به ویروس، کرم، تروجان، جاسوس ابزار و ... اشاره کرد.

تکثیر و انتشار کد مخرب ممکن است اثرات ناخوشایندی برای انواع مختلف کاربران عادی، شرکت‌های تجاری و دولت‌ها که از سیستم‌های کامپیوتری استفاده می‌کنند داشته باشد. به عنوان مثال، اگر یک کپی از یک کد مخرب به کامپیوتری که به شبکه‌ای متصل باشد نفوذ کند، می‌تواند منجر به از دست رفتن، استفاده غیر مجاز و یا تغییر مقدار زیادی از داده‌ها شود. در نتیجه این امر سبب بروز عدم اطمینان کاربران، نسبت به صحت اطلاعات بر روی شبکه خواهد شد.

۲ – کارهای مرتبط

در روش شناسایی بر مبناء امضاء فایلهای مشکوک با تطبیق با لیست امضاء موجود تحلیل می گردد. اگر در این مقایسه تفاوتی وجود داشته باشد فایل تحت آزمایش بعنوان یک برنامه مخرب طبقه بندی خواهد شد.

توسعه دهندگان بدافزار استراتژی پوشش[۱] یا ابهام را برای جلوگیری از شناسایی شدن به روش مبتنی بر امضاء بکار می برند. بعضی از بدافزارها موقع انتقال تغییر یافته و بعضی خودشان را برای فعالیت مخربشان رمز گذاری می‌کنند. بنابراین شناسایی نسخه جدید از بدافزار بخاطر مشکل بدست آوردن امضایشان وقت و نیروی انسانی زیادی می برد و باعث شد که محققین روی روشهای جدید برای یافتن روشهای موثر و کارآمدتر متمرکز شوند. در اینجا روشی که بدافزار خودشان را مخفی میکنند را مورد بررسی قرار میدهیم و برروی معرفی موثرترین روشهای آزمایشگاهی برای شناسایی بدافزار متمرکز می شویم.[۳]

یک چالش بینشی اصلی در سال ۱۹۸۸ در زمانی که یکی از این فعالیت ها فکری یعنی کرم Morris اشتباه عمل کرد رخ داد که قسمت مهمی از اینترنت را آلوده کرد و موجب حذف خدمات اینترنت شد که پاکسازی این آلودگی با چندین میلیون دلار خرج برداشت[۵]. این مسئله موجب توجه خیل جمعیتی به بدافزارها شد که موجب شکل گیری صنعت دفاع از طریق آنتی ویروس شد. سیستم های اولیه تشخیص بد افزارها، به پیمایش ساده بایت وابسته بودند که به برنامه نویسان بدافزارها انگیزه داد تا مخلوقات شان را به تکنولوژی چند شکلی مجهز کنند. در دهه ی اخیر، دو جهش حمله ی روز صفر و بات نت در فناوری بدافزارها رشد حائز اهمیتی داشته است.

در سال ۲۰۰۳، کرم Slammer نزدیک به ٪۹۰ از هاست های تأثیر پذیرش را طی چندین دقیقه بعد از انتشارش آلوده کرد. برنامه نویسان بدافزارها هم اکنون روش روز صفر را بعنوان وسیله ای قدرتمند برای حملات سریع و گسترده به فروش می گذارند[۹]. چنین آسیب پذیرهایی درسه ماهه اول سال ۲۰۱۳ در بالای فهرست تهدیدات سایبری قرار داشت.[۱۶]

بدین دلیل، امکان شکست حملات سایبری ظرفیت های مهاجمی برای ازکار انداختن سرورهای C&C رقیب لازم است تا شناسایی صورت بگیرد و به شبکه های دشمن نفوذ پیدا کرد و در آن خرابکاری ایجاد نمود. بر طبق این اخیراً USAF (نیروی هوایی ایالات متحده) وجود حداقل ۶ سلاح سایبری را در محیط جنگ افزاری اش اعلام کرده است[۷] و DARPA (سازمان پژوهش های تحقیقاتی پیشرفته ی دفاعی) هم اکنون درخواست جدی از تحقیق در جنگ افزار سایبری را دارد که توانایی های (قابلیت های) دفاعی و مهاجمی ملی خود را افزایش دهد[۸].

در سال ۲۰۱۰ ویروس Stuxnet سانتروفیوژهای هسته ای را در تجهیزات بسیار ایمن مرکز غنی سازی اورانیوم در نطنز را آلوده و تخریب کرد[۹]. برای مجهز کردن این کار شاخص، ویروس Stuxnet چهار آسیب پذیری روز صفر بی سابقه را وارد کرد. که از مدارک مخفی برای معتبر نشان دادن خودش استفاده کرد و به سرورهای C&C دسترسی پیدا کرد تا خود را آپدیت کند و از مرزهای ساختمانی عبور کرد تا سانتریفیوژهای کنترل کننده ی PLC ها را آلوده کند. سپس ویروس های DuQu، Flame و Gaus بعنوان سلاح سایبری با همان پیچیدگی شناسایی شد که توسط همان شخص یا اشخاص (ایالات متحده) به کار گرفته شده بود[۱۰]. بدافزار تطابق پذیر واکنشی[۱۱] یک پیشرفت دفاعی می باشد که در آزمایشگاه نرم افزارهای امنیتی در دانشگاه تگزاس در دالاس مورد بررسی و مطالعه قرار گرفته است. با مشاهده چنین بدافزاری تقریباً تمام روش های مبهم بدافزاری یک ضعف مشترک دارند: پیشرفت و جهش های آنها غیر مستقیم است. برای مثال ویروس های چند شکلی مخفی شده، کلیدهای مخفی جدیدی را انتخاب می کنند به این امید که نوشته رمز شامل الگو یا ویژگی متمایز کننده ی رایجی می باشد. مدافعان این ضعف را با آشکار کردن الگوهای ثابت وارد می کنند تا نشانه هایی که تمام متغیرهای بدافزار را یکسان می کند را از بین ببرد. در مقابل، بدافزار تطابق پذیر واکنشی تحت جهشی مستقیم می رود. مدلی از چگونگی تعیین نشانه ی بدافزار یاد می گیرد و سپس آن را مغلوب کرده تا موارد مبهم برای دفاع شناخته شود. که منجر به تطابق سریع با آپدیت های نشانه ها شود. در نتیجه بدافزار تطابق پذیر واکنشی نمونه ی مخفی بودن تا صرفاً نوع غیر مستقیم می باشد.

در تحقیق اخیر امکان و مؤثر بودن روش تطابق پذیر واکنشی Frankenstein را نشان داده شده است[۱۱و۱۲]. بیشتر از جهش Frankenstein خود را کاملاً از قسمت های کدها به اجرا در می آورد که پیامد حاصل از برنامه های موجود در ماشین های آلوده شده می باشد. هم چون هیولایی که توسط دانشمندان در رمان Shelley خلق شد و در نتیجه هر جهش Frankenstein، محصول جمع آوری قسمت های بدنه از طعمه های غیر مظنونش می باشد

در مقاله[۱۳] تکه های امضاء ویروسها می توانند به یکدیگر الحاق شده، جهش یافته و ویروس های جدیدی را تشکیل دهند. از اینرو امضاء ویروسها را به کروموزم هایی به طول ۴ تقسیم میکند که اندازه ژن های یک

[۱] Obfuscation

کروموزم با یکدیگر برابرند. عملگر تقاطع در این روش ژن ها را به یکدیگر الحاق می کند تا ویروس های جدید را ایجاد و تست کند از این رو تمامی ترکیب ژنی از کروموزم های مختلف می توانند با یکدیگر تلفیق شده و کروموزم های جدیدی را ایجاد نمایند. برای پیاده سازی تمام ژن های امضاء های مختلف را در استخر ژن قرار می دهیم که هر ژن می تواند در تشخیص فایل جاری مورد استفاده قرار گیرد. در نتیجه تمامی ژن های استخر ژن برای بررسی فایل جاری مورد استفاده قرار می گیرند. الگوریتم روش پیشنهادی در مقاله[۱۳] به شکل زیر پیاده سازی گردید.

۱. استخر امضاها را لود کن.
۲. هر یک از امضاها را به کروموزمی به طول چهار ژن با اندازه یکسان تبدیل کن و در مجموعه ژن قرار بده.
۳. به ازای هر یک از فایل ها
۴. به ازای هر تکرار(i)
۴.۱. به اندازه ۰.۱*i از ابتدای مجموعه ژن انتخاب کن و در استخر ژن جاری قرار بده
۴.۲. به ازای هر یک از ژن ها با مقدار select مخالف ۱ در مجموعه ژن های جاری
۴.۳. اگر امضاء فایل حاوی ژن است
۴.۳.۱. Select ژن را برابر ۱ قرار بده
۴.۳.۲. ژن را از امضا حذف کن.
۴.۴. در غیر اینصورت یک کاراکتر از ژن را به تصادف تغییر بده

۳- روش پیشنهادی

همانطور که پیش از این بیان شد با الهام از چرخه زندگی ویروس‌های کامپیوتری که روندی همانند الگوریتم ژنتیک دارند، انگیزه‌ای جهت طراحی روشی بر اساس الگوریتم ژنتیک برای کشف این ویروس‌ها ایجاد شد. از این‌رو پیش از ارائه روش پیشنهادی مروری بر الگوریتم ژنتیک خواهیم داشت.

۳-۱ الگوریتم ژنتیک

الگوریتم ژنتیک معمولاً به عنوان یک شبیه‌ساز که در آن جمعیت یک نمونه انتزاعی (کروموزوم‌ها) از نامزدهای راه‌حل یک مسأله بهینه‌سازی به راه حل بهتری منجر شود، پیاده‌سازی می‌شود. راه‌حل با جمعیتی کاملاً تصادفی منحصر بفرد آغاز می‌شود و نسل‌های بعدی با بقای بهترین کروموزم‌ها ادامه می‌یابد. در هر نسل تمام جمعیت ارزیابی می‌شود و چندین کروموزم در فرایندی تصادفی از نسل جاری انتخاب می‌شوند. لازم به ذکر است که با وجود آنکه فرایند انتخاب تصادفی است اما احتمال انتخاب هر کروموزم بر اساس شایستگی آن کروموزم تعیین می‌شود. کروموزم‌های انتخاب شده برای شکل دادن نسل جدید اصلاح می‌شوند و در تکرار بعدی الگوریتم به نسل جاری تبدیل می‌شوند. روند نسل کشی از کروموزم‌ها تا زمان احراز یکی از شرایط خاتمه ادامه می‌یابد.

۱.۱.۱. ساختار کروموزم

ساختار کروموزم در نظر گرفته شده در روش پیشنهادی ساختاری ساده دارد. ساختار کروموزم‌ها در ابتدا تنها حاوی یک ژن است که شامل امضاء ویروس است. باید توجه شود که این ساختار انعطاف لازم را برای تغییر دارد از این رو ممکن است در نسل‌های بعدی هر کروموزم به ژن‌های بیشتری توسط عملگر تقاطع تقسیم شود. شکل ۱ ساختار اولیه دو کروموزم را برای امضاء دو ویروس فرضی نشان می‌دهد.

817e0342417441bb80008b571a81c2b50081c20001891606

4c00268c1e4e00071fb804008bf581ee

شکل ۱ ساختار اولیه دو کروموزم فرضی

لازم به ذکر است که در این ساختار برخلاف ساختارهای مرسوم در الگوریتم ژنتیک اندازه کروموزم‌های مختلف می‌تواند متفاوت باشد. این مسئله از آن جهت است که امضاء ویروس‌ها دارای طول‌های مختلفی هستند

۳-۱-۱ عملگر برش

از عملگر برش برای تلفیق ژنی کروموزم‌ها در جهت ایجاد الگوهای ژنی جدید استفاده می‌شود. این عملگر با برش کروموزم‌های مختلف و تلفیق ژن‌های حاصل از برش کروموزم‌هایی جدید ایجاد می‌کند. اما با توجه به اینکه امضاء ویروس‌ها پس از تغییر نسل برگرفته از امضاء نسل قبل همراه با تغییراتی هستند از این‌رو در روش پیشنهادی از عملگر برش تنها برای تقسیم ژن‌های کروموزم استفاده می‌کنیم. در رویکرد پیشنهادی چنانچه عملگر برش برای یک ژن مورد استفاده قرار بگیرد ژن مذکور به دو یا چند ژن تقسیم می‌شود. شکل ۲ حاصل عملگر برش بر روی ژن کروموزم اول شکل ۱ را نشان می‌دهد

817e0342417441bb80008b57	1a81c2b50081c20001891606

شکل ۲ کروموزم حاصل از عملگر برش

شکل ۳ نیز حاصل عملگر برش بر روی ژن اول از کروموزم شکل ۲ است.

817e03424174	41bb80008b57	1a81c2b50081c20001891606

شکل ۳ کروموزم حاصل از عملگر برش بر روی ژن اول کروموزم شکل ۲

این شکل استفاده از عملگر برش قابلیت تقسیم ژن‌های یک کروموزم به تعداد دلخواه را می‌دهد که در ادامه به کاربرد آن در روش پیشنهادی خواهیم پرداخت.

۳-۱-۲ عملگر جهش

بررسی امضاء ویروس‌ها در نسل‌های مختلف نشان می‌دهد که کاراکترهای امضاء ویروس‌ها در نسل‌های مختلف دچار تغییراتی محدود می‌شوند به نحوی که امضاء ویروس با هر تغییر نسل در چند کاراکتر تغییر می‌کند. این موضوع شباهت عملکرد عملگر جهش با تغییرات امضاء در تغییر نسل ویروس‌ها را نشان می‌دهد. از اینرو در روش پیشنهادی چنانچه ژن‌ای از کروموزمی دچار جهش شود یک یا چند کاراکتر آن ژن تغییر می‌کند. لازم به ذکر است که بررسی امضاء ویروس‌ها نشان می‌دهد که بازه کدهای اسکی کاراکترهای امضاها بین ۴۸ تا ۱۲۲ است. از اینرو در زمان جهش یک کاراکتر، کاراکتر جایگزین نیز دارای کد اسکی‌ای بین ۴۸ تا ۱۲۲ خواهد داشت. شکل ۴ جهش سه کاراکتر را در ژن‌های کروموزم شکل ۳

نشان می‌دهد. لازم به ذکر است که علاوه بر کاراکتر جایگزین مکان جهش نیز به صورت تصادفی انتخاب می‌شود.

817e0T424174	41bb8b008b57	1a81c2b50081c2000189160k

شکل۴ کروموزم حاصل از عملگر جهش بر روی کروموزم

٤ – شناسایی ویروس با استفاده از الگوریتم ژنتیک

در این قسمت الگوریتم پیشنهادی جهت شناسایی ویروس‌ها با استفاده از الگوریتم ژنتیک ارائه شده است. لازم به یادآوری است که مقصود از شناسایی ویروس‌ها کشف امضاء آن‌هاست. امضاهایی که جهش یافته امضاء ویروس‌های نسل‌های قبل هستند. از اینرو در مجموعه امضاهای جاری وجود ندارند و در واقع قصد پیش‌بینی ویروسی بودن فایل‌ها با استفاده از الگوریتم ژنتیک داریم. در ادامه رویکرد پیشنهادی با استفاده از الگوریتم ژنتیک جهت کشف ویروس‌ها ارائه شده است.

۱. Fat و Pm را تعیین کن.
۲. استخر امضاها را لود کن.
۳. به اندازه Fat از استخر امضاء کپی ایجاد کن.
۴. به ازای هر امضاء (S_i) در استخر امضا.
۴.۱. به ازای تعداد تکرار در الگوریتم ژنتیک.
۴.۱.۱. به ازای هر یک از فایل‌ها (F_j).
۴.۱.۱.۱. اگر امضاء فایل F_j با امضاء S_i یکسان است.
۴.۱.۱.۱.۱. فایل F_j را به عنوان ویروس شناسایی کن و از مجموعه فایل‌ها حذف کن و ادامه نده.
۴.۱.۱.۲. به ازای هر یک از تقاطع‌های (C_k) از امضاء S_i
۴.۱.۱.۲.۱. اگر فایل F_j حاوی تقاطع C_k است تقاطع C_kرا از امضای فایل F_j حذف کن و برازندگی تقاطع C_kرا برابر ۱ قرار بده.
۴.۱.۱.۳. به ازای هر تقاطع C_k، با برازندگی غیر ۱
۴.۱.۱.۳.۱. قطعه C_k، را به دو قطعه تقسیم کن.
۴.۱.۱.۳.۲. یک عدد تصادفی ایجاد کن.
۴.۱.۱.۳.۳. اگر عدد تصادفی کمتر از Pm است.
۴.۱.۱.۳.۳.۱. یک کاراکتر از تقاطع C_k، را به تصادف تغییر بده.

در الگوریتم پیشنهادی ابتدا نرخ جهش (P_m) و میزان افزایش (Fat) استخر امضاها تعیین می‌شود. پس از بارگذاری استخر امضاها به اندازه Fat از امضاها کپی ایجاد می‌شود. در مراحل بعد تلاش می‌شود تا در صورتی که انطباقی میان امضاء S_i و فایل F_j وجود دارد، فایل F_j به عنوان ویروس معرفی شود. اما چنانچه انطباقی میان فایل F_j و امضاء ویروس S_i وجود نداشته باشد کروموزم حاوی امضاء Si توسط عملگر تقاطع به دو ژن تقسیم می‌شود. در ادامه الگوریتم در عوض مقایسه امضاء ویروس با امضاء فایل هر یک از ژن‌های امضاء ویروس مقایسه می‌شود. با این عمل چنانچه هر یک از برش‌های امضاء ویروس در امضاء فایل مشاهده شود برش مذکور از امضاء فایل حذف می‌شود و مسئله به یافتن تنها بخش باقیمانده امضاء ویروس کاهش می‌یابد. با وجود تطابق میان ویروس و تقاطع C_k از امضاء S_i برازندگی C_k برابر ۱ می‌شود. در ادامه هر یک از تقاطع‌ها با برازندگی غیر ۱ (که هنوز تطابقی برای آن‌ها پیدا نشده است) توسط عملگر برش به دو تقاطع تقسیم می‌شود و همچنین توسط عملگر جهش یک کاراکتر از این تقاطع به تصادف جهش می‌یابد

٥ – الگوریتم ژنتیک خودانطباقی

یکی از مشکلات الگوریتم ژنتیک مرسوم ثابت بودن نرخ برش و نرخ جهش است که اغلب بر اساس تجربه تعیین می‌شوند. در حالت کلی زمانی که نرخ برش بسیار پایین است، فرآیند تکاملی می‌تواند به سادگی در بهینه‌های محلی قرار گیرد یا اینکه دچار همگرایی زودرس شود و این امر به دلیل عدم تنوع جمعیت است. در طرف مقابل زمانی که نرخ برش بسیار بالا است؛ فرآیند بهینه سازی در مجاورت نقاط بهینه قرار می‌گیرد، کروموزم‌ها به سختی به نقطه بهینه می‌رسند و سرعت همگرایی به طور قابل ملاحظه‌ای کند می‌باشد؛ هر چند که تنوع جمعیت حفظ می‌شود. از طرفی زمانی‌که در الگوریتم ژنتیک نرخ جهش بسیار بالا است الگوریتم ژنتیک تبدیل به یک جستجوگر ساده می‌شود؛ و زمانی که نرخ جهش بسیار پایین است احتمال قرار گرفتن الگوریتم ژنتیک در بهینه‌های محلی زیاد می‌شود. در حالت کلی مقادیر مطلوب برای نرخ برش و نرخ جهش بر اساس مقادیر زیر قابل تعیین است.

۱. مسئله ای که الگوریتم ژنتیک بر روی آن اعمال می‌شود.
۲. برازندگی جمعیت جاری.

از این رو در الگوریتم ژنتیک خود انطباقی مورد استفاده در رویکرد پیشنهادی نرخ برش و نرخ جهش به صورت انطباق پذیر تنظیم شده‌اند. که در واقع نرخ برش و نرخ جهش برای هر کروموزم در هر نسل، به صورت مجزا تعیین می‌شود. و بدین منظور از روابط (۱و۲) استفاده می‌شود.

$$P_c = \frac{F_{Max} - F^C}{F_{Max} - \bar{F}} \qquad (۱)$$

$$P'_m = P_m + \left(۱ - \frac{F^C}{\bar{F}}\right) \times ۰.۰۱ \qquad (۲)$$

P'_m و P_c به ترتیب نرخ برش و نرخ جهش هستند و P_m مقدار اولیه نرخ جهش را مشخص می‌کند. F_{MAX} مقدار بیشترین برازندگی بدست آمده برای یک کروموزم است. F^c برازندگی کروموزومی است که جهت انجام عمل برش انتخاب شده است و $\bar{F}$ میانگین برازندگی جمعیت است. برازندگی هر کروموزم نیز با استفاده از معادله زیر تعیین می‌شود.

$$Fit = ۱ - \frac{\text{میزان اختلاف}}{\text{اندازه امضای باقیمانده}} \qquad (۳)$$

معادله (۳) میزان برازندگی هر کروموزم نسبت به یک امضاء را بر اساس شباهت میان آن‌ها محاسبه می‌کند. در واقع هر اندازه یک کروموزم به امضاء فایل شباهت بیشتری داشته باشد برازندگی بیشتری دریافت می‌کند. این معادله نشان می‌دهد که F_{MAX} در معادله (۱) مقداری برابر ۱ خواهد داشت. علاوه بر این بررسی معادله‌ها نشان می‌دهد که با افزایش اختلاف میان امضاء ویروس و کروموزم باعث خواهد شد تا ژن‌های کروموزم به تقاطع‌های بیشتری تقسیم شود و علاوه بر این ژن‌ها جهش بیشتری را

تجربه کنند و در حالت عکس چنانچه شباهت میان امضاء ویروس و کروموزم بیشتر باشد ژن‌ها به برش‌های کمتری تقسیم می‌شوند و عملگر جهش کاراکترهای کمتری را تغییر می‌دهد. این رویکرد منجر به ارائه الگوریتم ژنتیک خودانطباقی جهت کشف ویروس‌ها شده است.

۱. Fat و Pm را تعیین کن.
۲. استخر امضاها را لود کن.
۳. به اندازه Fat از استخر امضاء کپی ایجاد کن.
۴. به ازای هر امضاء (S_i) در استخر امضا.
۴.۱. به ازای هر تکرار.
۴.۱.۱. به ازای هر یک از فایل‌ها (F_j).
۴.۱.۱.۱. اگر فایل F_jبا امضاء یکی است.
۴.۱.۱.۱.۱. فایل F_j را به عنوان ویروس شناسایی کن و از مجموعه فایل‌ها حذف کن و ادامه نده.
۴.۱.۱.۲. به ازای هر یک از تقاطع‌های (C_k) امضاء S_i.
۴.۱.۱.۲.۱. اگر فایل F_j حاوی تقاطع C_k است تقاطع را از امضای فایل حذف کن و برازندگی تقاطع را برابر ۱ قرار بده.
۴.۱.۱.۳. به ازای هر تقاطع ($C_{k'}$) با برازندگی غیر ۱.
۴.۱.۱.۳.۱. برازندگی تقاطع $C_{k'}$ را نسبت به باقیمانده امضای فایل F_j بر اساس معادله (۳) تعیین کن.
۴.۱.۱.۳.۲. تعداد تقاطع های لازم برای قطعه $C_{k'}$ را بر اساس معادله (۱) تعیین کن.
۴.۱.۱.۳.۳. قطعه $C_{k'}$ را به تعداد قطعات مشخص شده تقسیم کن.
۴.۱.۱.۳.۴. نرخ جهش هر قطعه را بر اساس معادله (۲) تعیین کن.
۴.۱.۱.۳.۵. یک عدد تصادفی ایجاد کن.
۴.۱.۱.۳.۶. اگر عدد تصادفی کمتر از Pm است.
۴.۱.۱.۳.۶.۱. یک کاراکتر از این تقاطع را به تصادف تغییر بده

الگوریتم ژنتیک خود انطباقی پیشنهادی جهت شناسایی ویروس‌ها رویکردی همانند الگوریتم ژنتیک مرسوم دارد با این تفاوت که تعداد تقاطع‌ها و نرخ جهش کروموزم‌ها بر اساس میزان شباهت کروموزم با امضاء فایل محاسبه می‌شود.

۶ – دادگان ارزیابی

جهت ارزیابی روش پیشنهادی و روش مورد مقایسه، امضاء ۱۰۰ ویروس تحت عنوان استخر امضاء در نظر گرفته شده است. علاوه بر این ۵ مجموعه فایل که هر یک حاوی ۲۰۰ فایل هستند ایجاد شده است. سپس در مجموعه اول ۵٪ از فایل‌ها هر یک توسط یکی از ویروس‌ها آلوده شده است. لازم به ذکر است که قبل از آلوده کردن فایل توسط امضاء ویروس، امضاء جهش یافته است. این مسئله جهت شبیه سازی تغییر نسل در ویروس‌ها انجام شده است. در مجموعه فایل دوم ۲۵٪ از فایل‌ها به روش ذکر شده آلوده شده است. این روند برای مجموعه‌های بعدی به ترتیب با ۵۰٪، ۷۵٪ و ۱۰۰٪ تکرار شده است.

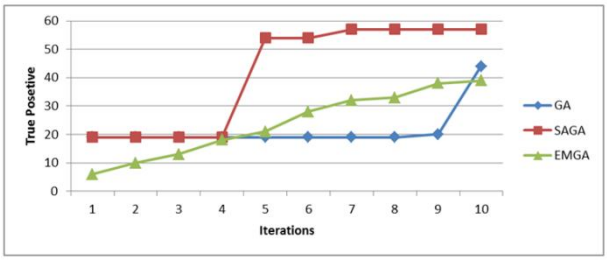
۷ – ارزیابی روش پیشنهادی

در این بخش جهت مقایسه روش‌های مختلف به معرفی پارامترهایی ارزیابی می‌پردازیم. پس از آن با استفاده از این معیارها به مقایسه روش پیشنهادی با روش‌های مورد مقایسه خواهیم پرداخت[۴].

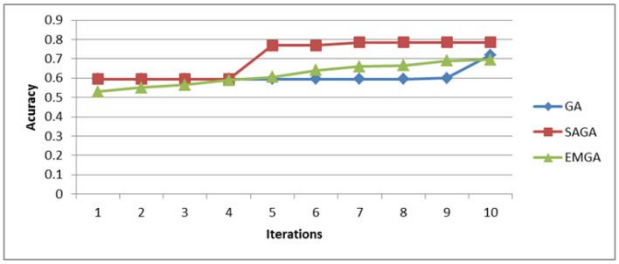
TP[۲]: در این حالت فایلی که آلوده تشخیص داده شده است در واقع به ویروس آلوده بوده است و سیستم عملکردی درست از خود بروز داده است. به این حالت در اصلاح گفته می‌شود به درستی مثبت پیش‌بینی شده است. صحت یا درستی یک سیستم شناسایی ویروس عبارتست از تعداد تصمیمات صحیح اتخاذ شده توسط سیستم به تعداد فایل‌های بررسی شده. معادله (۴) نحوه محاسبه صحت را نمایش می‌دهد[۱۴].

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

تعداد پیش بینی درست و صحت سیستم‌های مورد مقایسه در شکل‌های ۵ و۶ نمایش داده شده است.



شکل۵ تعداد پیش‌بینی مثبت درست در مجموعه فایل با ۵۰٪ فایل آلوده



شکل۶ صحت درمجموعه فایل با ۷۵٪ فایل آلوده

۸ – نتیجه گیری

با ظهور ویروس‌های کامپیوتری چالشی جدید در زمینه علم کامپیوتر پدیدار شد. مسئله کشف و از بین بردن این ویروس‌ها همواره یکی از موضوعاتی است که متخصصین و کاربران سیستم‌های کامپیوتری را درگیر خود کرده است. رایج‌ترین روش برای کشف ویروس‌ها، شناسایی امضاء آن‌هاست. این روش که به عنوان روش غالب در آنتی ویروس‌ها مورد استفاده قرار می‌گیرد دارای یک مشکل اساسی است؛ در این روش یک ویروس تنها پس از شناسایی امضاء آن قابل ردیابی است. در این حالت چرخه عمر ویروس کامل می‌شود و ویروس باعث تخریب سیستم شده

[۲] True Positive

است. در حالی که در حالت ایده‌آل یک سیستم شناسایی ویروس بایستی قبل از آن که یک ویروس فعال شود و به سیستم صدمه بزند اقدام به کشف و از بین بردن ویروس نماید.

اخیراً روش‌هایی جهت شناسایی ویروس‌ها مطرح شده‌اند که تلاش می‌کنند تا قبل از آنکه ویروس‌ها باعث صدمه زدن به سیستم شوند آن‌ها را کشف نمایند. در این مقاله روشی مبتنی بر الگوریتم ژنتیک برای کشف ویروس‌ها شناسایی کند. در روش پیشنهادی از امضاهای موجود به عنوان نسل اولیه کروموزم‌ها استفاده شده است. در ادامه با استفاده از عملگرهای تقاطع و جهش تلاش شده است تا در نسل کشی از کروموزم‌ها، فایل‌های آلوده به ویروس‌های جهش یافته شناسایی شود. سپس رویکرد پیشنهادی با استفاده از الگوریتم ژنتیک خود انطباقی بهبود یافته است.

جهت ارزیابی روش پیشنهادی علاوه بر رویکردهای ژنتیک و ژنتیک خود انطباقی روش ارائه شده در[۱۳] نیز تحت عنوان EMGA پیاده‌سازی شده است. علاوه بر این نتایج رویکرد پیشنهادی با استفاده از الگوریتم ژنتیک مرسوم تحت عنوان GA در بخش ارزیابی لحاظ شده است تا بتوان بهبود روش پیشنهادی با استفاده از الگوریتم ژنیتیک خود انطباقی را مورد سنجش قرار داد.

۹- کارهای آینده

نتایج بدست آمده در این مقاله نشان می‌دهند که با استفاده از الگوریتم‌های تکاملی می‌توان رویکردهای امیدوار کننده برای ایجاد آنتی ویروس ها در آینده متصور شد. از اینرو در نظر است تا در آینده به بررسی الگوریتم‌های تکاملی دیگری از جمله الگوریتم ازدحام ذرات به عنوان کارهای تحقیقاتی بپردازیم.

پیش از این یکی از روش‌هایی که برای تشخیص حروف و امضاها بسیار مورد توجه محققین بوده است استفاده از شبکه‌های عصبی است. خاصیت یادگیری در این شبکه‌ها کمک می‌کند تا با استفاده از رفتار گذشته به پیش بینی رفتار آینده بپردازیم. این رویکرد می‌تواند در مورد روند جهش امضاها در ویروس‌ها نیز مورد استفاده قرار گیرد. از اینرو شناسایی الگوی امضاها با استفاده از شبکه عصبی نیز به عنوان یک زمینه تحقیقاتی باز می‌تواند مورد توجه قرار گیرد.

مراجع

[۱] Reports Amount of Malware Grew by ۱۰۰% during ۲۰۰۷". Press release. Retrieved ۲۰۰۷-۱۲-۱

[۲] Symantec, Understanding Heuristics: Symantec's Bloodhound Technology, ۱۹۹۷, Symantec White Paper Series, vol. XXXIV.*

[۳] Z. Bazrafshan, H. Hashemi, S. M. H. Fard and A. Hamzeh, "A survey on heuristic malware detection techniques," *Information and Knowledge Technology (IKT), ۲۰۱۳ ۵th Conference on*, Shiraz, ۲۰۱۳, pp. ۱۱۳-۱۲۰.

[٤] Y. Zhou and X. Jiang, "Dissecting Android Malware: Characterization and Evolution," ۲۰۱۲ *IEEE Symposium on Security and Privacy*, San Francisco, CA, ۲۰۱۲, pp. ۹۵-۱۰۹.

[٥] Mikey Goldweber , Renzo Davoli , Joyce Currie Little , Charles Riedesel , Henry Walker , Gerry Cross , Brian R. Von Konsky, Enhancing the social issues components in our computing curriculum: computing for the social good, ACM Inroads, v.۲ n.۱, March ۲۰۱۱.

[٦] Sood, A. K., & Enbody, R. J. (۲۰۱۳). *Crimeware-as-a-service-A survey of commoditized crimeware in the underground market.International Journal of Critical Infrastructure Protection*, ۶(۱), ۲۸-۳۸..

[۷] A. Shalal-Esa, "Six U.S. Air Force cyber capabilities designated 'weapons'," Reuters, April ۲۰۱۳

[۸] Defense Advanced Research Projects Agency, "Foundational cyber warfare(plan X)," DARPA-BAA-۱۳-۰۲, November ۲۰۱۲.

[۹] Obama Order Sped up Wave of Cyberattacks against Iran, New York Times, ۱ June ۲۰۱۲.

[۱۰] D. Kushner, "The real story of Stuxnet," IEEE Spectrum, March ۲۰۱۳.

[۱۱] V. Mohan and K. W. Hamlen, "Frankenstein: Stitching malware from benign binaries," in Proc. ۶th USENIX Workshop on Offensive Technologies,۲۰۱۲, pp. ۷۷–۸۴.

[۱۲] Aron, J. "Frankenstein virus creates malware by pilfering code." New Scientist, August (۲۰۱۲).

[۱۳] Wu, Lihua, and Yu Zhang. "Research of the computer virus evolution model based on immune genetic algorithm." Computer and Information Science (ICIS), ۲۰۱۱ IEEE/ACIS ۱۰th International Conference on. IEEE, ۲۰۱۱.

[۱٤] Picard, Richard R., and R. Dennis Cook. "Cross-validation of regression models." Journal of the American Statistical Association ۷۹.۳۸۷ (۱۹۸۴): ۵۷۵-۵۸۳.

[۱٥] Afaneh, Suha, Raed Abu Zitar, and Alaa Al-Hamami. "Virus detection using clonal selection algorithm with Genetic Algorithm (VDC algorithm)." Applied Soft Computing ۱۳.۱ (۲۰۱۳): ۲۳۹-۲۴۶.

[۱٦] Trend Micro, "Zero-days hit users hard at the start of the year," TrendLabs Q۱ ۲۰۱۳ Security Roundup, April ۲۰۱۳.