نام دانشجو: امیرمحمد آقاجانی نام درس: تحلیل و طراحی الگوریتم پروژه پایانی درس

شماره‌ی دانشجویی: 400521063 نام استاد: دکتر آرش عبدی هجراندوست تاریخ: 1402/04/16

برای شروع کار ابتدا می بایست تعدادی الگو پیدا شود که درون فایل های benign نیستند ولی درون حداقل بخشی از فایل های malware یافت می شوند.

با توجه به اینکه تعداد پوشه های مربوط به فایل های malware ، 20 تا بود تصمیم گرفته شد که برای هر پوشه یک الگو به دست بیاید که بخش قابل توجهی از فایل های malware آن پوشه را شناسایی کند (تا حد امکان همه ی فایل های آن پوشه را).

## الگوریتم اول:

ایده ی اولیه برای شروع، همان الگوریتم ارائه شده در داک پروژه بود یعنی Longest Common Substring . که البته این الگوریتم برای مطابقت دو رشته طراحی شده است پس آن را برای تعداد n رشته ارتقا داده تا بتوان همزمان زیررشته ای را پیدا کرد که درون همه ی فایل های malware یک پوشه یافت شود. که خب برای این کار فایل اول پوشه را به عنوان رشته ی معیار در نظر می گیرد و تمام زیررشته های آن را به دست آورده و چک می کند که آیا این زیررشته، زیررشته ی سایر رشته ها نیز هست یا خیر و در این کار آن زیررشته ای که دارای بزرگترین طول هست را به خروجی می دهد.

به این شکل که ابتدا پوشه ی اول فایل های malware در نظر گرفته شد. چون حجم فایل ها به میزان قابل توجهی زیاد بود تصمیم گرفته شد که تنها بخشی از هر فایل بررسی شود. مثلا 100 خط اول هر فایل با 100 خط اول فایل های دیگر (هر خط به منزله ی 32 بایت است). جواب بدست آمده همچنان قابل قبول نیست. وقتی می توان آن را پذیرفت که این الگو درون هیچ یک از فایل های benign یافت نشود. چون بزرگترین زیررشته ی مشترک را پیدا کردیم پس خیلی بعید بود که درون 900 فایل benign یافت شوند و همین طور هم بود. پس تصمیم گرفته شد که این الگو را تا حد امکان کوچک کرد تا به یک الگوی معتبر و البته راحت تر برای تطبیق، بدل شود. برای این کار آن قدر از سر و ته این الگو را زدیم تا جایی که به کوچکترین حد آن رسیدیم و البته درون هیچ یک از فایل های benign نیز نبود. به این ترتیب ما به یک الگوی 11 کاراکتری برای پوشه ی اول، یک الگوی 6 کاراکتری برای پوشه ی دوم (البته با درصد تطابق 64.5) و یک الگوی 12 کاراکتری برای پوشه ی سوم رسیدیم.

این روش برای پوشه ی اول و دوم و سوم خوب پیش رفت و حتی برای پوشه ی اول و سوم به میزان 100 درصد تطابق با فایل های malware پوشه شان رسید. اما برای پوشه چهارم به خوبی پیش نرفت. الگو های پیدا شده

برای این پوشه هیچ یک ارزشی نداشتند. زیرا یا دارای مقدار زیادی 0 بودند (که درون غالب فایل های benign نیز به راحتی موجود بود) یا این الگو ها حداکثر دارای طول 3 کاراکتر بودند!!! (که بدون تردید درون همه ی فایل های benign وجود داشت).

نتایج این الگوریتم در جدول زیر قابل مشاهده است:

| شماره پوشه | الگوی مربوط به پوشه | تعداد فایل های malware شناسایی شده با الگو | درصد تطابق با فایل های malware پوشه |
|---|---|---|---|
| 1 | ‘65F33C05EC3’ | 400/400 | 100.00 |
| 2 | ‘B8AC07’ | 258/400 | 64.50 |
| 3 | ‘A5400085C075’ | 400/400 | 100.00 |

## الگوریتم دوم:

در این جا بود که به این فکر افتادم که چرا اصلا دنبال الگو بگردیم. بیاییم خودمان یک سری الگوی کاندید تولید کنیم و آن وقت آن ها را بررسی کنیم که اولا درون هیچ فایل benign ای نباشند و ثانیا در تعداد زیادی فایل malware یافت شوند. بنابراین تصمیم گرفتم تمام جایگشت های 0 تا 9 و A تا F را با اندازه ی حداکثر 5 کاراکتر را تولید کنم و در حین تولید، آن ها را بررسی کنم که درون هیچ فایل benign ای نباشند و آنها را به عنوان الگوهای کاندید استخراج کنم و درون فایل GeneratedPatterns.txt بنویسم. پس درون فایل GeneratePattern.py ، کدی نوشتم که این کار را انجام دهد. مدت زیادی طول کشید تا نتیجه حاصل شد. اما نتیجه شگفت انگیز بود. اول اینکه تمام زیررشته های با طول حداکثر 4 کاراکتر بالاخره درون حداقل یک فایل benign وجود داشت و هیچ یک از آن ها به عنوان کاندید پذیرفته نشد. اولین الگویی که پذیرفته شد و درون هیچ فایل benign یافت نشد الگوی ’011ED‘ بود و آخرینِ آنها که البته میشد دویست و بیست و یک هزار و هفتصد و هشتاد و هفتمینِ آن، الگوی ’FFED3‘ بود. یعنی بیش از یک پنجم همه ی زیررشته های 5 کاراکتری کاندید شدند. این تعداد که البته کم هم نیستند تبدیل به الگوهای پیشنهادی من شدند که هر کدام باید برای فایل های یک پوشه بررسی میشدند تا آن الگویی که 100 درصدِ (یا خیلی نزدیک به این مقدارِ) فایل های malware پوشه اش را شناسایی کرد انتخاب شود.

اما مسئله ی دیگر این بود که تعداد الگوهای کاندید بسیار زیاد بود (221787 تا) و برای اکثر پوشه های malware نیز تعداد فایل ها به 400 تا می رسید و این زمانی، حداکثر به میزان 221787 * 400 تراکنش

صرف می کرد که طبق تخمین من به طور میانگین برای هر پوشه چیزی حدودِ 5 ساعت زمان می برد. من نیز تا مهلت پروژه یک شبانه روز بیشتر وقت نداشتم!!! پس تصمیم بر این شد که از این تعداد الگوی کاندید، بخشی از آن، یعنی حدود یک دهمِ آن (برای پوشه های 400 فایله)، بررسی شود و برای پوشه های با میزان فایل malware کمتر مقدار بیشتری از الگوها مثلا یک پنجمشان.

طبق این الگوریتم نتایج خوب و البته تا حد زیادی دقیقی حاصل شد که در جدول زیر قابل مشاهده است:

| شماره پوشه | الگوی مربوط به پوشه | تعداد فایل های malware شناسایی شده با الگو | درصد تطابق با فایل های malware پوشه |
|---|---|---|---|
| 1 | ‘015CD’ | 400/400 | 100.00 |
| 2 | ‘139FE’ | 393/400 | 98.25 |
| 3 | ‘01DAE’ | 400/400 | 100.00 |
| 4 | ‘0ADB7’ | 400/400 | 100.00 |
| 5 | ‘1C61F’ | 322/322 | 100.00 |
| 6 | ‘FAFED’ | 265/400 | 66.25 |
| 7 | ‘FAFED’ | 211/319 | 66.14 |
| 8 | ‘FAFED’ | 263/400 | 65.75 |
| 9 | ‘D7FF2’ | 294/319 | 82.13 |
| 10 | ‘C1FB2’ | 192/358 | 53.63 |
| 11 | ‘9E413’ | 349/400 | 87.25 |
| 12 | ‘026AB’ | 193/193 | 100.00 |
| 13 | ‘1451A’ | 400/400 | 100.00 |
| 14 | ‘019D6’ | 25/25 | 100.00 |
| 15 | ‘119D5’ | 70/70 | 100.00 |
| 16 | ‘01B5A’ | 400/400 | 100.00 |
| 17 | ‘01DEC’ | 400/400 | 100.00 |
| 18 | ‘55FA5’ | 399/400 | 99.75 |
| 19 | ‘011ED’ | 400/400 | 100.00 |
| 20 | ‘012FD’ | 400/400 | 100.00 |
| مجموع | 18 الگو | 6176/6806 | 90.74 |

نام دانشجو: امیرمحمد آقاجانی نام درس: تحلیل و طراحی الگوریتم پروژه پایانی درس

شماره‌ی دانشجویی: 400521063 نام استاد: دکتر آرش عبدی هجراندوست تاریخ: 1402/04/16

حال که این 20 الگو که هر کدام تنها دارای 5 کاراکتر هستند را پیدا کردیم کافیست برای هر یک Finite Automata مربوط به آن را تولید کرده و سپس و در مواجهه با هر فایلی که قرار است بررسی شود، کافیست این 20 الگو را برای آن چک کنیم و اگر حداقل یکی از آنها مطابقت یافت آنگاه آن فایل را به عنوان فایل malware در پوشه ی Malwares می ریزد.

جدول زیر نتایج حاصله از اعمال تمام 18 الگوی بدست آمده بر روی تمام فایل های malware را نشان می دهد:

| شماره پوشه | تعداد فایل های malware شناسایی شده با الگو | درصد تطابق با فایل های malware پوشه |
|---|---|---|
| 1 | 400/400 | 100.00 |
| 2 | 400/400 | 100.00 |
| 3 | 400/400 | 100.00 |
| 4 | 400/400 | 100.00 |
| 5 | 322/322 | 100.00 |
| 6 | 387/400 | 96.75 |
| 7 | 311/319 | 97.49 |
| 8 | 387/400 | 96.75 |
| 9 | 307/319 | 96.24 |
| 10 | 282/358 | 78.77 |
| 11 | 373/400 | 93.25 |
| 12 | 193/193 | 100.00 |
| 13 | 400/400 | 100.00 |
| 14 | 25/25 | 100.00 |
| 15 | 70/70 | 100.00 |
| 16 | 400/400 | 100.00 |
| 17 | 400/400 | 100.00 |
| 18 | 400/400 | 100.00 |
| 19 | 400/400 | 100.00 |
| 20 | 400/400 | 100.00 |
| مجموع | 6657/6806 | 97.81 |

نام دانشجو: امیرمحمد آقاجانی نام درس: تحلیل و طراحی الگوریتم پروژه پایانی درس

شماره‌ی دانشجویی: 400521063 نام استاد: دکتر آرش عبدی هجراندوست تاریخ: 1402/04/16

## تحلیل مرتبه زمانی:

با توجه به اینکه از Finite Automata Algorithm برای تطابق الگو استفاده شده است و ساخت Transition Function برای هر یک از الگوها در پیش پردازش رخ می دهد؛ کل زمان تطابق، به تعداد فایل های مورد بررسی (k)، تعداد الگوها (18) و میانگین تعداد کاراکترهای موجود در هر فایل (n) بستگی دارد. پس مرتبه ی زمانی آن از O(kn) می باشد. (صرفا کافی است که به ازای هر الگو، روی همه کاراکترهای موجود در هر فایل پیمایش صورت گیرد.)

(لازم به ذکر است که این الگوریتم بر روی تمام فایل های malware ، یک بار اجرا شد و در مجموع 30 دقیقه به طول انجامید)