گزارش

اسم درس

۱۴۰۱/۴/۱۱

- **نام و نام خانوادگی :**
- **نام استاد :**
- **گروه :**

گزارش کار

فاز ۱:

در ابتدا به دنبال راه بهینه برای پیدا کردن رشته های مخرب در فایل های مخرب بودم. برای این کار ابتدا به صورت brute force تمامی فایل های یک فولدر (یک ویروس) را بررسی میکردم. اینکار بسیار زمان بر و غیر بهینه بود. همچنین برای بررسی فایل ها از الگوریتم Longest Common Subsequence استفاده میکردم. تمامی مثال های اینترنت از این تابع برای رشته ها نوشته شده بود در حالی که برای رشته های باینری به آن نیاز داشتم. برای همین به صورت دستی این تابع را نوشته و دستکاری کردم. این تابع بزرگترین رشته ی مشترک در دو فایل را به دست میاورد.

سپس برای پیدا کردن نمونه های مخرب ، از یک روش جدید استفاده کردم به این صورت که کوچک ترین فایل هر فولدر را در نظر میگرفتم و بقیه ی فایل ها را با آن مقایسه میکردم. چون همه ی فایل های یک فولدر برای یک نوع ویروس خاص هستند پس حتما با همدیگر اشتراکاتی دارند و راه سریع تر برای پیدا کردن آن ، استفاده از کوچک ترین فایل و مقایسه آن با بقیه بود. مشکلی که این روش داشت ، این بود که خیلی دقیق نبود و فایل های مخرب کمی با آن پیدا میشد. پس برای حل این مشکل یک متغییر accuracy تعریف شد و تعداد فایل هایی را نشان میداد که بقیه ی فایل ها برای اشتراکات مقایسه میشد.

فاز ۲:

سپس نیاز به بررسی و پیدا کردن نمونه های تکراری و جایگزینی آن ها بود. پس یک تابع برای تمیز کردن و پاک کردن نمونه های تکراری نوشتم که خب از الگوریتم KMP برای آن استفاده کردم ، به گونه ای که هر بار یک نمونه از الگوهای مخرب شناسایی شده با بقیه ی الگو ها مقایسه میشد و اگر اشتراک داشتند ، کوچک ترین آن ها (به دلیل اینکه رشته ی کوچک تر احتمال

وجودش در فایل ها بیشتر از رشته ی بزرگ تر است) را نگه میداشتم و بقیه ی مشابه ها را پاک میکردم.

فاز ۳:

بعد از آن ، نیاز بود که الگو های مخرب شناسایی شده ، چک شوند که با فایل ها سالم برخوردی نداشته باشند. پس یک تابع جدا برای اینکار طراحی کردم که هر الگو را با همه ی فایل های سالم بررسی کند و اگر در یکی از فایل های سالم موجود بود ، اون الگو را حذف کند. مشکل این روش نبود فایل های زیاد سالم بود که خب ممکن است این الگو ها در این فایل ها وجود نداشته باشد ولی در فایل های سالم دیگری موجود باشد.

فاز ۴:

در فاز بعدی ، نیاز بود که الگو های شناسایی شده با خود فایل های مخرب نیز بررسی شوند و هر کدام از الگو ها که در فایل های مخرب پیدا نشد (غیر از خود همان فایلی که از آن این الگو ساخته شده) ، باید پاک شوند که الگوی مخربی نیستند و فقط یک رشته ای بودند که در فایل چک شده و بقیه ی فایل ها اشتراک داشته اند. بنابراین در اینجا حجم دیتابیس ذخیره شد کم شده و دقت آن بسیار بالاتر میرود.

فاز ۵:

به هر حال ، بعد از آن ، یک تابع scanner نوشتم که فایل های فولدری که آدرس آن را میدهیم را بررسی کند و اگر الگوی فایل مخرب در آن پیدا کرد ، آن را نمایش دهد.

بعد از آن ایده ای به ذهنم رسید که فایل های مخرب ممکن است با هم اشتراکاتی داشته باشند (دو فولدر مجزا) ، پس با تابع scanner فولدر های malware را اسکن کردم و در بعضی فولدر ها الگو های آشنا و قبلی پیدا شد که به معنی این بود که ویروس ها میتوانند با هم اشتراکاتی داشته باشند. ایده این بود که برای الگوها هم priority در نظر بگیرم و هر کدام که در فایل های

بیشتری پیدا میشد یا دقت بالاتری داشت و حجم کمتری ، الویت بیشتری نیز داشته باشد ولی خب به دلیل اینکه تعداد الگو ها زیاد بود و حجم رم و cpu ای که برای کار مصرف میکرد بالا بود ، صرفا در حد یک ایده باقی ماند.