دانشگاه صنعتی خواجه نصیرالدین طوسی

دانشکده مهندسی کامپیوتر

گزارش گروهی دوم درس تکامل نرم افزار

نویسندگان:

محمدسینا اله‌کرم	**محمد اکبرپورجنت**	**محمد شمس الدینی**
رضا لشنی زند	**ارسلان واثق**	**علیرضا پرستار**
مهدوی اصل	**رضا قنبرزاده**	**معراج سوسن**
بهنام کاظمی	**فاطمه عاصی آتشکاهی**	**محمد شیبانی**
نیما گمرکیان		

استاد راهنما: دکتر محمدهادی علائیان

۷ آذر ۱۴۰۴

چکیده

تحولات مهندسی نرم‌افزار از روش‌های ابتدایی بدون ساختار به مدل‌های خطی مانند آبشاری و سپس به رویکردهای تکرارشونده و چابک، نشان‌دهنده تلاش برای مدیریت پیچیدگی و افزایش کیفیت سیستم‌ها بوده است. با این حال، چالش‌هایی مانند ارتباطات ناکارآمد، مستندسازی ضعیف، انباشت بدهی فنی و ناسازگاری با فناوری‌های نوین، همچنان تهدیدی برای پایداری نرم‌افزارها محسوب می‌شوند.

DevOps به‌عنوان فرهنگی نوین و مجموعه‌ای از ابزارها، با هدف یکپارچه‌سازی تیم‌های توسعه و عملیات، افزایش سرعت تحویل و ارتقای کیفیت معرفی شد. خودکارسازی فرآیندهای CI/CD و استفاده از ابزارهایی مانند ،Kubernetes Docker و ،Jenkins امکان تحویل سریع، کاهش خطا و افزایش پایداری استقرار را فراهم می‌کند. با این حال، چالش‌هایی مانند پیچیدگی زیرساخت و مقاومت فرهنگی نیازمند آموزش، مستندسازی و فرهنگ‌سازی هستند.

بازطراحی نرم‌افزار برای سیستم‌های قدیمی ضروری است و دلایل آن شامل ضعف معماری، فناوری‌های منسوخ، تغییر نیازمندی‌ها و انباشت بدهی فنی است. تکنیک‌های بازآرایی ، مهندسی معکوس و مهاجرت افزایشی (Incremental Migration) ضمن افزایش قابلیت نگهداری، هزینه اصلاح بدهی فنی را کاهش می‌دهند و ریسک تغییرات را مدیریت می‌کنند. مطالعات موردی مانند بازطراحی اپلیکیشن PayPal و نئوبانک فوربیکس، نشان‌دهنده تاثیر مستقیم بازطراحی بر تجربه کاربری، یکپارچگی خدمات و مزیت رقابتی هستند.

در نهایت، ترکیب رویکردهای DevOps و بازطراحی، پایداری و تکامل نرم‌افزارها را تضمین می‌کند. توصیه می‌شود تیم‌های مهندسی بر فرهنگ همکاری، مدیریت فعال بدهی فنی، مستندسازی همگام با توسعه، خودکارسازی CI/CD و آموزش مستمر تمرکز کنند. همچنین، مسیرهای تحقیقاتی آینده شامل امنیت یکپارچه در DevOps ،(DevSecOps) کاربرد هوش مصنوعی در مهندسی معکوس، مدیریت پیچیدگی ابزارها، آموزش مهارت‌های نرم و توسعه الگوهای پیشرفته مهاجرت و بازطراحی خواهد بود.

فهرست مطالب

فهرست تصاویر

فصل ۱

فرایندهای مهندسی نرم‌افزار و چرخه‌های تکامل تا پیدایش

۱.۱ مقدمه‌ای بر ضد دیس اسمبلی

۲.۱ استفاده از تکنیک‌های تغییر ساختار کد

۳.۱ استفاده از دستورات و ساختارهای غیرمعمول

۴.۱ پنهان‌سازی از طریق تجزیه و تحلیل پویا

۵.۱ روش‌های ضد دیس اسمبلی مبتنی بر خود تغییر

۶.۱ مقاومت در برابر دیس اسمبلی با استفاده از تکنیک‌های رمزگذاری

با تبدیل فایل اجرایی به کد اسمبلی، تحلیل‌گران می‌توانند منطق داخلی، باگ‌ها و بدافزارهای احتمالی را شناسایی کنند[۱]. به‌عبارت دیگر، دیس‌اسمبلی امکان افشای عملکرد برنامه و پیدا کردن آسیب‌پذیری‌ها یا کدهای مخرب را فراهم می‌کند. به همین دلیل، نویسندگان نرم‌افزارهای محافظت‌شده تلاش می‌کنند

با روش‌هایی مانند رمزگذاری و ابهام‌زایی، دیس‌اسمبلی را دشوار کنند.

۱.۶.۱ کد پلی‌مورفیک (Polymorphic Code)

تعریف و مکانیسم:

کد پلی‌مورفیک نوعی کد خودتغییرده است که در هر نسخه ظاهر باینری متفاوتی دارد، ولی منطق و الگوریتم اصلی‌اش حفظ می‌شود[۲]. معمولاً در این روش، بخش اصلی کد («payload») با یک کلید تصادفی رمزگذاری می‌شود و یک بخش کوتاه کوچک در ابتدای برنامه (بخش رمزگشا) وجود دارد که با کلید مربوطه، payload را در حافظه رمزگشایی می‌کند[۲، ۳]. به‌عنوان مثال، ویروس‌های پلی‌مورفیک مشهور، از رمزنگاری XOR با یک کلید تصادفی برای مخفی کردن بدنه‌ی اصلی خود استفاده می‌کنند، به‌طوری که هر نمونه از لحاظ باینری کاملاً با نمونه‌های دیگر متفاوت است[۳].

مزایا و کاربردها:

مزیت اصلی پلی‌مورفیسم مقاومت در برابر شناسایی مبتنی بر امضاست؛ چون هر بار که کد اجرا می‌شود، الگویی یکتا ایجاد می‌کند و امضای ثابتی ندارد[۳]. این روش عمدتاً در ویروس‌ها، کرم‌ها و بدافزارها به کار می‌رود تا برنامه‌های ضدویروس را فریب دهد. همچنین در برخی محافظت‌کننده‌های تجاری کد (پکرها) دیده می‌شود که با تولید کلیدهای مختلف یا تغییر الگوریتم رمزگشایی، کد را در هر توزیع متفاوت می‌کنند. پلی‌مورفیسم نسبتاً ساده است و تأثیر زیادی بر عملکرد اجرایی برنامه ندارد.

نحوه عملکرد:

معمولاً هر نمونه پلی‌مورفیک شامل دو بخش است: یک لایه رمزنگاری (برای پنهان‌سازی payload) و یک بخش رمزگشای کوتاه. در زمان اجرا، ابتدا بخش رمزگشا فعال شده و بدنه اصلی را در حافظه بازمی‌کند، سپس کنترل به کد اصلی داده می‌شود[۲]. به خاطر این ساختار، دیس‌اسمبلی ایستا مؤثر نیست زیرا کد اصلی در فایل رمزیافته و فقط با اجرای برنامه آشکار می‌شود[۲].

معایب و روش‌های تحلیل:

گرچه پلی‌مورفیسم شناسایی ساده مبتنی بر امضا را دشوار می‌کند، اما شکستن آن آسان است. تحلیل‌گران می‌توانند برنامه را اجرا کنند و با قرار دادن نقطه‌نگاری روی بخش رمزگشا یا استفاده از دیباگر، بدنه اصلی را در زمان اجرای برنامه مشاهده و استخراج نمایند[۲]. افزون بر این، چون الگوریتم رمزگشایی معمولاً ثابت یا ساده است، می‌توان با نوشتن اسکریپت‌های کوچک یا استفاده از ابزارهای خودکار، کد را رمزگشایی کرد. به همین علت، اکثر روش‌های مقابله با کد پلی‌مورفیک بر تحلیل پویا (اجرا در محیط کنترل‌شده و خاموش کردن رمزنگاری) و یا نرمال‌سازی کد بر مبنای الگوریتم‌های آماری

متمرکزند.

۲.۶.۱ کد متامورفیک (Metamorphic Code)

تعریف و مکانیسم:

کد پلی‌مورفیک نوعی کد خودتغییرده است که در هر نسخه ظاهر باینری متفاوتی دارد، ولی منطق و الگوریتم اصلی‌اش حفظ می‌شود[۲]. معمولاً در این روش، بخش اصلی کد («payload») با یک کلید تصادفی رمزگذاری می‌شود و یک بخش کوتاه کوچک در ابتدای برنامه (بخش رمزگشا) وجود دارد که با کلید مربوطه، payload را در حافظه رمزگشایی می‌کند[۳][۲]. به‌عنوان مثال، ویروس‌های پلی‌مورفیک مشهور، از رمزنگاری XOR با یک کلید تصادفی برای مخفی کردن بدنه‌ی اصلی خود استفاده می‌کنند، به‌طوری که هر نمونه از لحاظ باینری کاملاً با نمونه‌های دیگر متفاوت است[۳].

مزایا و کاربردها:

مزیت اصلی پلی‌مورفیسم مقاومت در برابر شناسایی مبتنی بر امضاست؛ چون هر بار که کد اجرا می‌شود، الگویی یکتا ایجاد می‌کند و امضای ثابتی ندارد[۳]. این روش عمدتاً در ویروس‌ها، کرم‌ها و بدافزارها به کار می‌رود تا برنامه‌های ضدویروس را فریب دهد. همچنین در برخی محافظت‌کننده‌های تجاری کد (پکرها) دیده می‌شود که با تولید کلیدهای مختلف یا تغییر الگوریتم رمزگشایی، کد را در هر توزیع متفاوت می‌کنند. پلی‌مورفیسم نسبتاً ساده است و تأثیر زیادی بر عملکرد اجرایی برنامه ندارد.

معایب و روش‌های تحلیل:

گرچه پلی‌مورفیسم شناسایی ساده مبتنی بر امضا را دشوار می‌کند، اما شکستن آن آسان است. تحلیل‌گران می‌توانند برنامه را اجرا کنند و با قرار دادن نقطه‌نگاری روی بخش رمزگشا یا استفاده از دیباگر، بدنه اصلی را در زمان اجرای برنامه مشاهده و استخراج نمایند[۲]. افزون بر این، چون الگوریتم رمزگشایی معمولاً ثابت یا ساده است، می‌توان با نوشتن اسکریپت‌های کوچک یا استفاده از ابزارهای خودکار، کد را رمزگشایی کرد. به همین علت، اکثر روش‌های مقابله با کد پلی‌مورفیک بر تحلیل پویا (اجرا در محیط کنترل‌شده و خاموش کردن رمزنگاری) و یا نرمال‌سازی کد بر مبنای الگوریتم‌های آماری متمرکزند.

۳.۶.۱ بسته‌سازی کد (Packers)

تعریف و مکانیسم:

کد پلی‌مورفیک نوعی کد خودتغییرده است که در هر نسخه ظاهر باینری متفاوتی دارد، ولی منطق و الگوریتم اصلی‌اش حفظ می‌شود[۲]. معمولاً در این روش، بخش اصلی کد («payload») با یک کلید تصادفی رمزگذاری می‌شود و یک بخش کوتاه کوچک در ابتدای برنامه (بخش رمزگشا) وجود دارد که با کلید مربوطه، payload را در حافظه رمزگشایی می‌کند[۳][۲]. به‌عنوان مثال، ویروس‌های پلی‌مورفیک مشهور، از رمزنگاری XOR با یک کلید تصادفی برای مخفی کردن بدنه‌ی اصلی خود استفاده می‌کنند، به‌طوری که هر نمونه از لحاظ باینری کاملاً با نمونه‌های دیگر متفاوت است[۳].

مزایا و کاربردها:

مزیت اصلی پلی‌مورفیسم مقاومت در برابر شناسایی مبتنی بر امضاست؛ چون هر بار که کد اجرا می‌شود، الگویی یکتا ایجاد می‌کند و امضای ثابتی ندارد[۳]. این روش عمدتاً در ویروس‌ها، کرم‌ها و بدافزارها به کار می‌رود تا برنامه‌های ضدویروس را فریب دهد. همچنین در برخی محافظت‌کننده‌های تجاری کد (پکرها) دیده می‌شود که با تولید کلیدهای مختلف یا تغییر الگوریتم رمزگشایی، کد را در هر توزیع متفاوت می‌کنند. پلی‌مورفیسم نسبتاً ساده است و تأثیر زیادی بر عملکرد اجرایی برنامه ندارد.

معایب و روش‌های تحلیل:

گرچه پلی‌مورفیسم شناسایی ساده مبتنی بر امضا را دشوار می‌کند، اما شکستن آن آسان است. تحلیل‌گران می‌توانند برنامه را اجرا کنند و با قرار دادن نقطه‌نگاری روی بخش رمزگشا یا استفاده از دیباگر، بدنه اصلی را در زمان اجرای برنامه مشاهده و استخراج نمایند[۲]. افزون بر این، چون الگوریتم رمزگشایی معمولاً ثابت یا ساده است، می‌توان با نوشتن اسکریپت‌های کوچک یا استفاده از ابزارهای خودکار، کد را رمزگشایی کرد. به همین علت، اکثر روش‌های مقابله با کد پلی‌مورفیک بر تحلیل پویا (اجرا در محیط کنترل‌شده و خاموش کردن رمزنگاری) و یا نرمال‌سازی کد بر مبنای الگوریتم‌های آماری متمرکزند.

۴.۶.۱ ابهام‌زایی مبتنی بر مجازی‌سازی (Virtualization-Based Obfuscation)

تعریف و مکانیسم:

کد پلی‌مورفیک نوعی کد خودتغییرده است که در هر نسخه ظاهر باینری متفاوتی دارد، ولی منطق و الگوریتم اصلی‌اش حفظ می‌شود[۲]. معمولاً در این روش، بخش اصلی کد («payload») با یک کلید تصادفی رمزگذاری می‌شود و یک بخش کوتاه کوچک در ابتدای برنامه (بخش رمزگشا) وجود دارد که با کلید مربوطه، payload را در حافظه رمزگشایی می‌کند[۳][۲]. به‌عنوان مثال، ویروس‌های پلی‌مورفیک مشهور، از رمزنگاری XOR با یک کلید تصادفی برای مخفی کردن بدنه‌ی اصلی خود استفاده می‌کنند،

به‌طوری که هر نمونه از لحاظ باینری کاملاً با نمونه‌های دیگر متفاوت است[۳].

مزایا و کاربردها:

مزیت اصلی پلی‌مورفیسم مقاومت در برابر شناسایی مبتنی بر امضاست؛ چون هر بار که کد اجرا می‌شود، الگویی یکتا ایجاد می‌کند و امضای ثابتی ندارد[۳]. این روش عمدتاً در ویروس‌ها، کرم‌ها و بدافزارها به کار می‌رود تا برنامه‌های ضدویروس را فریب دهد. همچنین در برخی محافظت‌کننده‌های تجاری کد (پکرها) دیده می‌شود که با تولید کلیدهای مختلف یا تغییر الگوریتم رمزگشایی، کد را در هر توزیع متفاوت می‌کنند. پلی‌مورفیسم نسبتاً ساده است و تأثیر زیادی بر عملکرد اجرایی برنامه ندارد.

معایب و روش‌های تحلیل:

گرچه پلی‌مورفیسم شناسایی ساده مبتنی بر امضا را دشوار می‌کند، اما شکستن آن آسان است. تحلیل‌گران می‌توانند برنامه را اجرا کنند و با قرار دادن نقطه‌نگاری روی بخش رمزگشا یا استفاده از دیباگر، بدنه اصلی را در زمان اجرای برنامه مشاهده و استخراج نمایند[۲]. افزون بر این، چون الگوریتم رمزگشایی معمولاً ثابت یا ساده است، می‌توان با نوشتن اسکریپت‌های کوچک یا استفاده از ابزارهای خودکار، کد را رمزگشایی کرد. به همین علت، اکثر روش‌های مقابله با کد پلی‌مورفیک بر تحلیل پویا (اجرا در محیط کنترل‌شده و خاموش کردن رمزنگاری) و یا نرمال‌سازی کد بر مبنای الگوریتم‌های آماری متمرکزند.

۵.۶.۱ رمزگذاری کد در زمان اجرا (Runtime Code Encryption)

تعریف و مکانیسم:

کد پلی‌مورفیک نوعی کد خودتغییرده است که در هر نسخه ظاهر باینری متفاوتی دارد، ولی منطق و الگوریتم اصلی‌اش حفظ می‌شود[۲]. معمولاً در این روش، بخش اصلی کد («payload») با یک کلید تصادفی رمزگذاری می‌شود و یک بخش کوتاه کوچک در ابتدای برنامه (بخش رمزگشا) وجود دارد که با کلید مربوطه، payload را در حافظه رمزگشایی می‌کند[۲، ۳]. به‌عنوان مثال، ویروس‌های پلی‌مورفیک مشهور، از رمزنگاری XOR با یک کلید تصادفی برای مخفی کردن بدنه‌ی اصلی خود استفاده می‌کنند، به‌طوری که هر نمونه از لحاظ باینری کاملاً با نمونه‌های دیگر متفاوت است[۳].

مزایا و کاربردها:

مزیت اصلی پلی‌مورفیسم مقاومت در برابر شناسایی مبتنی بر امضاست؛ چون هر بار که کد اجرا می‌شود، الگویی یکتا ایجاد می‌کند و امضای ثابتی ندارد[۳]. این روش عمدتاً در ویروس‌ها، کرم‌ها و بدافزارها به کار می‌رود تا برنامه‌های ضدویروس را فریب دهد. همچنین در برخی محافظت‌کننده‌های

تجاری کد (پکرها) دیده می‌شود که با تولید کلیدهای مختلف یا تغییر الگوریتم رمزگشایی، کد را در هر توزیع متفاوت می‌کنند. پلی‌مورفیسم نسبتاً ساده است و تأثیر زیادی بر عملکرد اجرایی برنامه ندارد.

معایب و روش‌های تحلیل:

گرچه پلی‌مورفیسم شناسایی ساده مبتنی بر امضا را دشوار می‌کند، اما شکستن آن آسان است. تحلیل‌گران می‌توانند برنامه را اجرا کنند و با قرار دادن نقطه‌نگاری روی بخش رمزگشا یا استفاده از دیباگر، بدنه اصلی را در زمان اجرای برنامه مشاهده و استخراج نمایند[۲]. افزون بر این، چون الگوریتم رمزگشایی معمولاً ثابت یا ساده است، می‌توان با نوشتن اسکریپت‌های کوچک یا استفاده از ابزارهای خودکار، کد را رمزگشایی کرد. به همین علت، اکثر روش‌های مقابله با کد پلی‌مورفیک بر تحلیل پویا (اجرا در محیط کنترل‌شده و خاموش کردن رمزنگاری) و یا نرمال‌سازی کد بر مبنای الگوریتم‌های آماری متمرکزند.

۷.۱ نقاط ضعف روش‌های ضد دیس اسمبلی

فصل ۲

مشکلات مطرح در چرخه‌های توسعه و تکامل نرم‌افزار

۱.۲ روش‌های Disassembly Anti Anti

۲.۲ استفاده از دیس اسمبلرهای پیشرفته

۳.۲ استفاده از تحلیل رفتار اجرایی

۴.۲ تکنیک‌های رمزگشایی و تحلیل کدهای رمزگذاری‌شده

۵.۲ شبیه‌سازی و اجرای داینامیک

۶.۲ استفاده از روش‌های شکست کدهای خود تغییر

۷.۲ مقاومت در برابر تحلیل‌های ضد ضد دیس اسمبلی

منابع

[۱] Twingate Team. What is Disassembly? ۲۰۲۴. URL: https://www.twingate.com/blog/glossary/disassembly.

[۲] Liviu PETREAN. POLYMORPHIC AND METAMORPHIC CODE APPLICATIONS IN PORTABLE EXECUTABLE FILES PROTECTION. ۲۰۱۰. URL: https://users.utcluj.ro/~atn/papers/ATN%5C_1%5C_2010%5C_1.pdf.

[۳] What is Polymorphic Malware? URL: https://www.threatdown.com/glossary/what-is-polymorphic-malware/.