دانشکده فنی و مهندسی

گروه مهندسی کامپیوتر و فناوری اطلاعات

گزارش سمینار کارشناسی ارشد رشته مهندسی کامپیوتر نرم‌افزار(M.Sc)

عنوان سمینار:

متدولوژی های مدیریت اطلاعات برنامه های حساس از دیدگاه مهندسی نرم‌افزار(بررسی و مرور)

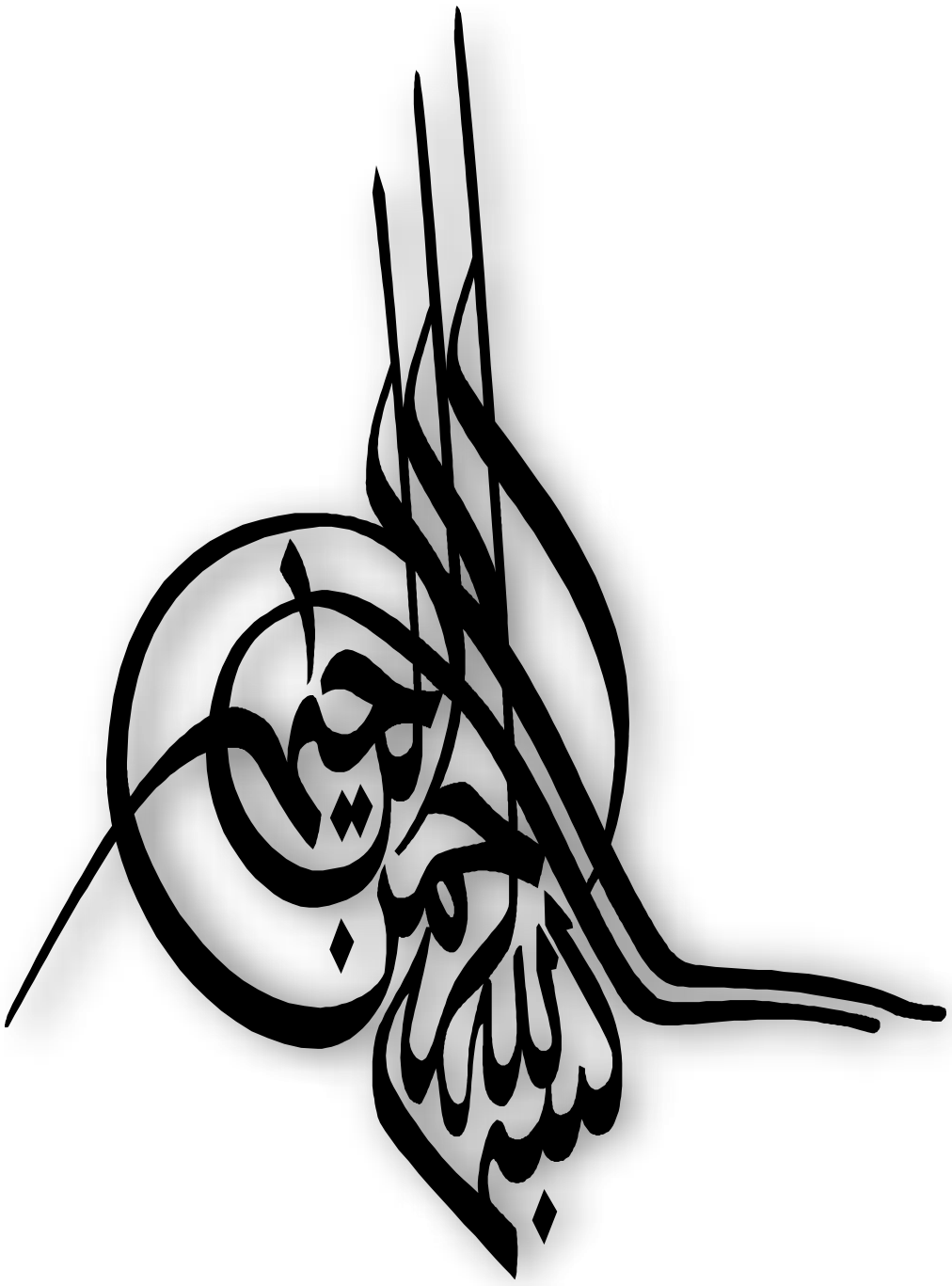
استاد راهنما:

دکتر سیدعلی رضوی

نگارنده:

سمیه کرباسی راوری

مهر ۱۴۰۰


بسم الله الرحمن الرحيم

فهرست مطالب

فهرست اشکال

فهرست جداول

فهرست علائم اختصاری

DL	deep learning	یادگیری عمیق
ML	machine learning	یادگیری ماشین
NLP	Natural language processing	رویکردهای پردازش زبان طبیعی
OOP	object-oriented programming	برنامه‌نویسی شی‌گرا
PFG	partitionable function grap	نمودار عملکرد قابل تقسیم
PII	Personally identifiable information	اطلاعات قابل شناسایی شخصی
SSVM	structured support vector machines	دستگاههای بردار پشتیبانی ساختار یافته
SPI	sensitive program information	اطلاعات حساس برنامه
TA	Trusted Applications	برنامه‌های قابل اعتماد
TCB	trusted computing base	پایگاه محاسباتی مورد اطمینان
TEE	Trusted Execution Environment	محیط اجرایی قابل اعتماد
VUSA	variable usage semantics analysis	تجزیه و تحلیل معنایی کاربرد متغیر

چکیده

برخی از بخش‌های یک برنامه کامپیوتری می‌تواند حساس باشد که به آنها اطلاعات حساس برنامه گفته می‌شود (SPI). با به خطر انداختن SPI، مهاجمان می‌توانند به امنیت و حریم خصوصی کاربران آسیب برسانند. برای توسعه‌دهندگان،شناسایی و محافظت از SPI ، به ویژه برای برنامه های بزرگ دشوار است. این گزارش روش‌ها، تکنیک‌ها و ابزارهای نرم‌افزاری جدیدی که توسعه نرم‌افزار را تسهیل می‌کنند و وظایف مربوط به مکان‌یابی و حفاظت از SPI را دارند؛ معرفی می‌کند.تجزیه و تحلیل الگوریتم‌ها و داده‌های تجاری خاص می‌تواند به امنیت و حریم خصوصی سازمان و کاربران نهایی آسیب برساند. اطلاعات برنامه حساس (SPI)، جزئی از نرم‌افزارهای مدرن سیستم‌هایی در حوزه‌های مختلف از برنامه‌های سازمانی تا تنظیمات سایبری فیزیکی هستند. از این رو ، حفاظت از SPI به یکی از چالش‌های برجسته توسعه نرم‌افزار مدرن تبدیل شده است. با این حال ، چندین مانع اساسی بر سر راه حفاظت SPI قرار دارد.

در این گزارش موارد زیر مطرح می‌شود: ۱) طراحی و توسعه تجزیه و تحلیل برنامه و پشتیبانی برنامه‌نویسی برای استنباط معناشناسی برنامه، با هدف کمک به توسعه‌دهندگان برای درک و شناسایی SPI ۲) برنامه‌نویسی قدرتمند و ابزارهایی که به طور خودکار کد را تغییر می‌دهند تا با هدف کمک به توسعه‌دهندگان به طور موثر SPI را از بقیه پایگاه کد جدا کنند. ۳) ارائه مکانیسم برنامه‌نویسی برای محیط های اجرایی مدیریت شده توزیع شده که با هدف فعال کردن اجزای برای تبادل ایمن SPI را پنهان می کنند.

متدولوژی‌ها، تکنیک‌ها و ابزارهای نرم‌افزاری پشتیبانی از برنامه‌نویسی، تجزیه و تحلیل خودکار برنامه، زمینه را برای ایجاد محیطی ایمن، قابل درک و کارآمد برای محافظت از SPI فراهم می‌کند.(liu,۲۰۲۱)

کلمات کلیدی: مهندسی نرم‌افزار[۱] ، تحلیل برنامه[۲] ، فهم برنامه[۳]، محیط اجرای مورد اعتماد[۴]، میان‌افزار[۵]

[۱] Software Engineering
[۲] Program Analysis
[۳] Program Comprehension
[۴] Trusted Execution Environment
[۵] Middleware

فصــل اول

۱. مقــدمــه

قسمت‌هایی از یک برنامه کامپیوتری می‌تواند حساس باشد که به آنها اطلاعات حساس برنامه گفته می‌شود (SPI). با به خطر انداختن SPI، مهاجمان می‌توانند به امنیت و حریم خصوصی کاربران آسیب برسانند. برای توسعه‌دهندگان،شناسایی و محافظت از SPI ، به ویژه برای برنامه های بزرگ دشوار است. این گزارش روش‌ها، تکنیک‌ها و ابزارهای نرم‌افزاری جدیدی که توسعه نرم‌افزار را تسهیل می‌کنند و وظایف مربوط به مکان‌یابی و حفاظت از SPI را دارند؛ معرفی می‌کند. در ادامه جمله حکیمانه‌ای که انگیزه طرح این موضوع در مهندسی نرم‌افزار هست؛ را بیان می‌کنم:

"من واقعاً فکر می کنم اگر ما رویکرد خود را تغییر دهیم و به آنچه در دسترس داریم فکر کنیم ، این چیزی است که توانایی ما را برای برتری واقعی در امنیت افزایش می دهد. " گریگ یورک[۱]

۱-۱ تعریف مسئله و بیان سؤال‌های اصلی تحقیق

سیستم های نرم افزاری مدرن عموما منطق تجاری حساس ، الگوریتم ها و داده هایی که حذف یا دستکاری آنها به کاربران نهایی یا کل سازمان ها آسیب می رساند؛ را با هم ترکیب می کنند. بلوک های سازنده نرم افزار در مجموع اطلاعات حساس برنامه (SPI) نامیده می شوند .نشت داده های حساس می تواند امنیت و حریم خصوصی کاربر را به خطر بیندازد. مثلا، پس از سرقت رمز عبور کاربر در بانکداری آنلاین ، مهاجم می تواند وارد حساب قربانی سیستم شود و وجوه را غیرقانونی انتقال دهید. وجود کسب و کارهای حساس می تواند رقابت یک شرکت را تضعیف کند.

داشتن مهندسی معکوس منطق تجاری یک موتور توصیه رایج ، یک مهاجم می تواند به سرعت، اطلاعات را از بین ببرد و باعث اختلال یا حتی خرابی سیستم شود. به عنوان مثال ، با دستکاری در الگوریتم ناوبری هواپیماهای بدون سرنشین ، یک دشمن می تواند پهپاد را به تغییر مسیر اشتباه دهد. SPI حفاظت یک منطقه ثابت در تحقیقات امنیتی

[۱] Greg York

است ، زمان مهندسی نرم افزار رسیده است محققان بایستند و از دید توسعه دهندگان در مورد برجسته ترین آنها تجدید نظر کنند.

سه مرحله فرایند مهندسی شامل تعریف،توسعه و نگهداری نرم‌افزار است.(جعفرنژاد و عامل ۱۳۹۶)

از دیدگاه توسعه دهنده ، SPIشامل متغیرهای برنامه است که داده ها را به عنوان توابع که منطق و الگوریتم های کسب و کار را پیاده سازی می کنند؛ ذخیره می کند. بنابراین ، از SPI با وظایف توسعه ای که از صداقت و محرمانه بودن متغیرهای حساس و کارکرد اطمینان دارند؛ محافظت می شود.

در این گزارش، براساس پایان‌نامه انتخابی و منابع مرجع، پس از طرح مباحث، در فصل چهار به سوالات زیر پاسخ داده می‌شود:

۱. استراتژی های پیشرفته درک مطلب چیست؟
۲. از کدام ابزارهای نرم افزاری می توان برای تسهیل فرایند درک استفاده کرد؟
۳. از کدام تکنیک می توان درک برای تشخیص اطلاعات مربوط به برنامه استفاده کرد؟
٤. چگونه می توان سیستم‌ها را برای تبادل ایمن *SPI* فعال کرد؟

۱-۲ ضرورت تحقیق

به طور کلی، مفاهیم جدیدی را در زمینه های تجزیه و تحلیل برنامه و بازآرایی ؛به عنوان مثال ، تحلیل معنایی متغیر ، و امکان سنجی و کاربرد آنها را می‌توان نشان داد. روش‌هایی را معرفی می‌کنیم که می‌تواند طراحی تجزیه و تحلیل تخصصی توسعه نرم‌افزار را هدایت کند و تکنیک های بازسازی می‌تواند به توسعه دهندگان نرم افزار که نقش های متفاوتی را ایفا می کنند؛ کمک کند. علاوه بر این ، با استفاده از روش‌ها، تکنیک‌ها و ابزارهایی در پروژه های خود، توسعه نرم‌افزار می‌توان بهبود بخشید. امنیت و حریم خصوصی محصولات، ضرورتی بسیار مهم برای همه افراد وابسته به نرم‌افزار مربوط است.

۱ – ۳ هدف‌ها

هدف اصلی این گزارش، ایجاد یک روش امن ، قابل درک ، و پایه ای کارآمد برای درک و مدیریت SPI است. برای برنامه نویسان تعمیر و نگهداری، شناسایی متغیرها و استفاده در کد خاص مسئله‌ای مهم است؛ برای تحلیلگران

امنیتی ، محیط امن می توانند در عملکردهای حساس، بدون متحمل شدن هزینه‌های غیرضروری عملکرد، بسیار کمک کننده باشد و با خیال راحت داده‌های حساس را ذخیره و منتقل کند، بنابراین از نشت جلوگیری می‌شود. برای زمان واقعی توسعه دهندگان سیستم ، می‌توان تلاش مورد نیاز برای سازگاری سیستم‌ها را کاهش داد و اجرای قابل اعتماد تحت محدودیت های زمان واقعی را داشت.

۱ – ٤ فرضیات تحقیق

۱) فرض می کنیم که اکثر متغیرها ، توابع و فایل های موجود در برنامه به صورت معنی دار نامگذاری شده اند به عنوان مثال ، بسیار محتمل است که متغیری به نام "رمز عبور" برخی از اطلاعات مربوط به رمز عبور را نشان می دهد. در واقع ، شـــرکت های بزرگ فناوری اطلاعات ، از جمله گوگل ، IBM و مایکروســـافت قراردادهای برنامه نویسی ایجاد کرده اند که نیاز به شناسه های برنامه دارد که به طور شهودی نامگذاری شود. مرور و تصحیح منظم کد، اغلب با پیشنهاداتی برای تغییر نام شناسه های معنادارتر همراه است.

۲) ما فرض می کنیم که کاربران دارای ســـابقه کافی در مورد معماری ســـیســـتم های نرم افزاری مورد تجزیه و تحلیل برای توصـــیف متغیرهای هدف هســـتند. انتظار می رود که کاربران در مورد چگونگی عملکرد مورد علاقه اطلاعاتی دا شته با شند. به عنوان مثال ، برای تعیین متغیرهایی که گذرواژه ها را ذخیره می کنند ، یک کاربر انتظار می رود که نحوه احراز هویت مبتنی بر رمز عبور در پروژه های تجزیه و تحلیل شده را درک کند. به عنوان مثال ، در یک عملکرد خاص "ورود" ، یک رمز عبور ، که توسـط کاربر نهایی وارد می شـود ، در مقایسـه با رمز عبور شناخته شده ، از حافظه بازیابی شده ا ست. متغیرها/توابع حـ ساس و غیر حـ ساس برای محافظت از داده های حساس را تشخیص دهدو بتواند زمان و نحوه کار مشتریان را تعیین کنند . از ویژگی های اشیاء داده حساس (به عنوان مثال ، مدت زمان د ستر سی داده ها ، نحوه د ستر سی) قابلیت پرس و جو ا ست که بارها می توان ویژگی های آن را مورد پرسش قرار داد.

۱-۵ چه کاربردهایی از انجام این تحقیق متصور است؟

- طراحی و توسعه برنامه های پشتیبانی و برنامه نویسی برای استنباط استفاده معناشناسی اجزای برنامه، با هدف کمک به توسعه دهندگان برای درک و شناسایی **SPI** یعنی تحلیل معنایی متغیر
- برنامه نویسی و ارائه ابزارهای قدرتمندی که کد را به طور خودکار تغییر می‌دهند، با هدف کمک به توسعه دهندگان که به طور موثرتر **SPI** را از بقیه پایگاه کد جدا می کند.
- ارائه مکانیسم برنامه نویسی برای محیط های اجرایی توزیع شده مدیریت شده که ساختار **SPI** را پنهان می کند ، با این هدف که اجزاء بتوانند **SPI** را به طور ایمن مبادله کنند.

۱-٦ روش و مراحل انجام تحقیق

روش انجام این تحقیق به‌صورت کتابخانه‌ای است. منابع مورداستفاده شامل پایان نامه، مقالات، تحقیقات علمی و پژوهشی، کتب و جستجوهای اینترنتی درزمینه‌ی متدولوژی های مدیریت اطلاعات برنامه های حساس از دیدگاه مهندسی نرم‌افزار است.

در این راستا یک پایان‌نامه انتخاب شد (liu,۲۰۲۱) و با بررسی ساختار پایان نامه و منابع مرجع، توانستم موضوع درک و تجزیه و تحلیل و بیان کنم.

۱-۷ سازمان پایان‌نامه مورد بررسی

فصل‌های پایان‌نامه مورد بررسی به صورت ذیل مرتب شده است:

فصل ۱ مقدمه و ضرورت تحقیق

فصل ۲ موارد فنی را معرفی می کند پیشینه و کار مربوط به این تحقیق را مورد بحث قرار می دهد.

فصل ۳ مدل ها و مفروضات اصلی را توضیح می دهد.

فصل ٤ رویکرد ما را برای استنباط معناشناسی استفاده توضیح می‌دهد.

فصل ۵ و فصل ٦ رویکردهای جدا شده را شرح می‌دهد.داده ها و توابع حساس در TEE و به ترتیب موارد غیر حساس را به خارج منتقل می کنند.

فصل ۷ رویکرد ما برای تبادل امن داده ها را توضیح می‌دهد.

فصل ۸ و فصل ۹ به ترتیب کارها و نتیجه گیری را بیان می‌کند.

۱-۸ ساختار گزارش تحقیق

فصل اول به تعریف و مقدمه و دلایل نیاز به طرح ارائه‌شده پرداخته می‌شود.

فصل دوم به پیش زمینه و کارهای وابسته پرداخته می شود.

فصل سوم مروری است بر کارهای انجام شده طرح پیشنهادی پایان‌نامه

فصل چهارم به کاربردها و مزایا و معایب روش‌های مطرح شده پرداخته می‌شود.

فصل پنجم نیز به جمع‌بندی و نتیجه‌گیری پرداخته می‌شود.

فصـــل دوم

۲. ادبیات تحقیق و پیش زمینه

در این فصل ، ابتدا تعاریف و پیشینه فنی مورد نیاز برای فهم را معرفی می کنیم. سپس درباره‌ی فعالیت‌های مربوطه بحث می کنیم.

۲-۱ تعاریف پس زمینه

در ادامه تعاریف **مربوط** و فناوری‌های اصلی که به **رویکردها** قدرت می‌دهند را شرح می‌دهیم.

۲-۱-۱ اطلاعات حساس برنامه(SPI)

SPI می‌تواند شامل منطق تجاری، الگوریتم‌ها و داده‌ها باشد. در صورتی که شامل اطلاعات حساسی باشند که تغییر آن ها به کاربران نهایی یا کل سازمان ها آسیب می رساند. "حساس" همه موارد مرتبط با امنیت را توصیف می کند مانند اشیاء (مانند گذرواژه ها ، کلیدها ، آدرس های حافظه) و عملیات (به عنوان مثال ، کنترل دسترسی ، رمزگذاری ، دسترسی به حافظه). این اشیاء و عملیات با آنچه از آن به عنوان "شرایط امنیتی"[۱] یاد می‌شود؛ مطابقت دارد. به طور خاص ، داده های حساس (یا متغیرها) اطلاعات مربوط به امنیت را ذخیره می کنند یا در رابطه با امنیت به آنها ارجاع داده می شود عملیات کد حساس (یا توابع) بر روی داده های حساس عمل می کند.(sans,۲۰۱۹)

۲-۱-۲ اطلاعات محتوایی و متنی

خصوصیات به صورت روبرو تقسیم میشوند: داخلی (به عنوان مثال ، نام نمادین ، نوع و محدوده) و بیرونی (داده‌ها و جریان کنترل) .اطلاعات متنی به نام نمادین یک متغیر، نام عملکرد، نوع توسعه دهنده تعریف می‌شود. نام (در صورت وجود) و مسیر فایل، اطلاعات زمینه به خواص دیگر آن اشاره دارد. (به عنوان مثال نوع داده، جریان داده

[۱] security terms

کنترل). توجه داشته باشید که نام نوع و نوع داده، خواص مختلفی دارند. در حالی که اولی اطلاعاتی را که متنی هستند توصیف می کند و دومی زمینه و بستر است. برای مثال متغیر دلخواهی را در نظر بگیرید struct : .type_nameنام نوع "type_name" نوع داده "struct" است.

۲-۱-۳ اطلاعات قابل شناسایی شخصی[۱]

شامل تمام اطلاعاتی است که در صورت دسترسی غیرمجاز به امنیت یا حریم شخصی افراد آسیب می رساند. نمونه های معمولی آن شامل شماره های امنیتی ، شماره کارت اعتباری و بدهی و داده های مربوط به مراقبت های بهداشتی هستند.(Schwartz, Solove ,۲۰۱۱)

۲-۱-٤ محدودیت های زمان واقعی

به طور کلی، محدودیت‌های زمان واقعی، محدودیت‌های زمان بندی، رویدادهایی هستند که باید توسط سیستم بلادرنگ تامین شوند. این محدودیت‌ها در مهلت‌های زمانی و محدودیت‌های دوره‌ای طبقه‌بندی می‌شوند. به عنوان مثال، با توجه به محدودیت دوره‌ای ۵۰ میلی ثانیه و مهلت زمانی ۲۰ میلی ثانیه، هواپیمای بدون سرنشین باید مکان GPS خود در هر دوره را در ۲۰ میلی لیتر برای هر ۵۰ میلی ثانیه بدست آورد. (liu,layland,۱۹۷۳)

در این حالت، به دلیل محدودیت حافظه، مصرف حافظه رویداد یک محدودیت دیگر است. همانطور که در بخش قبل ذکر شد؛ حافظه باید یک مقدار کوچک ردپایی با اشغال فضای محدود در حافظه را حفظ کند.

محدودیت های زمان واقعی به دو دسته سخت و نرم طبقه بندی می شوند تا محدودیت های قبلی برآورده شود. به عنوان مثال، کنترل موتور و سطح پرواز هواپیمای بدون سرنشین باید انجام شود. پاسخ به موقع (محدودیت سخت)، در حالی که هدایت آن با توجه به نقاط مورد انتظار مقاوم در برابر انحرافات ناشی از از بین رفتن موقت سیگنال GPS یا حتی وزش باد (محدودیت نرم)

۲-۲ تکنیک‌های پس زمینه

[۱] Personally identifiable information

۲-۲-۱ محیط اجرایی قابل اعتماد (TEE)

TEE یک راه حل سخت افزاری استاندارد ارائه می‌دهد که از SPI در برابر آسیب محافظت می‌کند. اولا ، TEE یک منطقه امن پردازنده (یعنی دنیای امن برای برنامه های قابل اعتماد) را از ناحیه عادی (یعنی دنیای عادی برای کاربردهای رایج) جدا می کند. یعنی ، دنیای امن دارای یک واحد محاسبه جداگانه و یک سیستم عامل مستقل است که مانع اجرای مستقیم برنامه های خارجی غیر مجاز میشود. علاوه بر این، TEE فضای ذخیره سازی قابل اعتمادی را ارائه می‌دهد که تنها از طریق برنامه‌های ارائه شده می‌توان به طور ایمن به آن دسترسی پیدا کرد. در نهایت، TEE به عنوان یک برنامه کانال ارتباطی به عنوان راهی برای ارتباط نهادهای خارجی با جهان امن ایمن ارائه می‌شود.(GlobalPlatform,۲۰۱۱)

OP-TEE: OP-TEE نمونه کامل شده مشخصات پلتفرم جهانی TEE ، یک مکانیزم جداسازی سخت افزاری است که در درجه اول به منطقه امن، با سه ویژگی های اساسی متکی است:

۱) برای محافظت از سیستم عامل مورد اعتماد از سیستم عامل غنی (به عنوان مثال ، لینوکس) برای اجرای برنامه‌های مورد اعتماد؛ جدا می‌شود و از طریق پشتیبانی سخت افزاری اساسی انجام می شود ؛

۲) به حافظه کافی نیاز دارد.

۳) می توان آن را به راحتی به انواع مختلف معماری و سخت افزار متصل کرد.

SGX : پیاده سازی دیگر TEE و شامل افزونه های نرم افزاری اینتل است که با گسترش معماری اینتل از محرمانگی و امنیت داده ها محافظت می‌کند. همانند OP-TEE، SGX از توسعه دهندگان می‌خواهد که کد اصلی را به دو قسمت تقسیم کنند؛ قطعات معمولی و قابل اعتماد در داخل منطقه حفاظت شده. قسمت اول منابع اجرا را از محیط خارجی (قسمت دوم) جدا می‌کند. علاوه بر این، اجزای معمولی فقط می‌توانند از طریق برنامه های ویژه به حوزه دسترسی داشته باشند. بنابراین، در صورت اجرا یا بارگیری در داخل محوطه، SPI در برابر حملات خارجی، آسیب ناپذیر می شود.(Costan, Devadas,۲۰۱٦)

۲-۲-۱ زبانهای برنامه نویسی و تجزیه و تحلیل

Scala، یک زبان برنامه نویسی مدرن ، ترکیبی از ویژگی های برنامه نویسی شی گرا و کاربردی است. Scala به عنوان زبان میزبان و به صورت تعبیه شده عمل می کند. Scala را به دلیل نحو انعطاف پذیر (به عنوان مثال ،

پرانتزهای اختیاری) و پشتیبانی آن انتخاب می کنیم. تعریف کلمات کلیدی جدید و نحو سفارشی و ویژگی های برنامه نویسی کاربردی (به عنوان مثال ، پشتیبانی از توابع مرتبه بالاتر) از ویژگی های Scala است که امکان آن را فراهم می کند که کتابخانه به راحتی با تکنیک های تجزیه و تحلیل جدید گسترش می یابد .سرانجام ، به عنوان یک JVMbased Scala روی زبان های مختلف اجرا می شود و محیط مستقل از پلت فرم را ایجاد می کند. (Horstmann,۲۰۱۲)

کپسوله سازی شی: یکی از مفاهیم اساسی برنامه نویسی شی گرا (OOP) کپسوله سازی است که داده‌ها و رفتارهای حساس را از کاربران شی پنهان می‌کند. علاوه بر این، جاوا اصلاح کننده‌های دسترسی را برای اطمینان از حریم خصوصی داده‌ها فراهم می‌کند. با استفاده از کلمه کلیدی، برنامه نویسان انتظار دارند که این حوزه از خارج از حوزه خود قابل دسترسی نباشد. اعلام کلاس حفاظت ارائه شده توسط اصلاح کننده های دسترسی جاوا را می‌توان با مجوز مناسب بدست آورد.. مهاجم می‌تواند با دسترسی مستقیم، ویژگی‌های خصوصی را تغییر دهد و از تابع‌های خصوصی استفاده کند. برای جلوگیری از این حمله، قابلیت مدیریت امنیت به جاوا اضافه شده است که اثربخشی آن به پیکربندی مناسب و واحد اجزا بستگی دارد.(Gong, Ellison,۲۰۰۳)

چرخه حیات شی: وقت شیء برنامه نویسی حاوی دادههای حساس خارج محدوده، در دسترس قرار می‌گیرد؛ اختیارات سیاستی رعایت می‌شود اما در برخی موارد، انتظار برای برقراری امنیت در حفظ اطلاعات حساس ممکن است کافی نباشد. در عوض، داده‌های حساس ممکن است پس از رسیدن به یک آستانه مشخص، توسط سیاست دسترسی به هر شیء به طور قابل اعتماد پاک شوند.

رویکردهای پردازش زبان طبیعی (NLP) به طور گسترده ای برای شناسایی اطلاعات متنی حساس برنامه (به عنوان مثال ، نظرات ، توضیحات) استفاده شده است. به ابزارهایی متکی است تا تجزیه و تحلیل متنی جدیدی متکی بر NLP ارائه دهد که قدرتمندتر از تحلیل متغیرهایی که باید در TEE محافظت شوند؛ باشد.(Huang, Li,۲۰۱۵)

تجزیه و تحلیل جریان داده، یک تکنیک استاندارد تجزیه و تحلیل برنامه ها، برای نتیجه ها از طریق برنامه تجزیه و تحلیل جریان داده برای تشخیص آسیب پذیری های کد است. با استفاده از تجزیه و تحلیل استاندارد جریان داده، برنامه متغیرها اطلاعات زمینه (به عنوان مثال مقادیر ورودی کاربر برای متغیرها) را شناسایی می‌کند.(Chen, Khandaker,۲۰۱۷)

۲-۳ کارهای مرتبط

این گزارش مربوط به چندین حوزه تحقیقاتی؛ از جمله درک معانی برنامه، تغییر شکل کد و نشت اطلاعات است. در این قسمت در مورد آنها بحث خواهیم کرد.

۲-۳-۱ تغییر شکل کد برای اجرای مطمئن

برای محافظت از SPI، توسعه‌دهنده باید کد برنامه را با سه مرحله تغییر شکل دهد:

۱. برنامه را به قسمتهای حساس و غیر حساس تقسیم کنید.

۲. تبدیل کد و داده‌های شناسایی شده را به برنامه ای مبتنی بر TEE برای اجرای مطمئن، که پایگاه محاسباتی مورد اطمینان (TCB) کوچک باقی بماند.

۳. اطمینان حاصل کند که کد محدودیت های اجرا را حفظ می کند (به عنوان مثال، محدودیت‌های زمان واقعی).

در ادامه درباره کارهای مربوط به پارتیشن‌بندی برنامه، تغییر کد و پروفایل اجرا و تأیید محدودیت‌ها بحث می‌کنیم.

برنامه های تقسیم بندی: کد بایت جاوا را در یک برنامه متمرکز به یک برنامه توزیع شده تقسیم می‌کند. با توجه به اعلانات برنامه‌نویس، یک برنامه وب به یک برنامه وب امن تغییر می‌کند و در قسمت سمت سرور جاوا و قسمت جاوا اسکریپت سمت مشتری از طریق HTTP با یکدیگر ارتباط برقرار می‌کنند. ZØ کامپایل اعلانی است که شامل کد C# و یک برنامه متمرکز در نسخه توزیع شده چند سطحی برای افزایش محرمانگی است. با اجرای مکانیزم پویای کنترل جریان اطلاعات، به طور خودکار و ایمن یک برنامه جاوا اسکریپت را به قسمت سرویس گیرنده و سرور تقسیم می‌کند. به طور خودکار برنامه های پشتیبانی شده از پایگاه داده را به سرور برنامه تقسیم می‌کند.(Tilevich, Smaragdakis,۲۰۰۲)

فصـــل ســـوم

۳. پیشینه تحقیق

۳-۱ مقدمه

در اوایل دهه ۱۹۶۰ معماری سرور مشتری فقط برای رایانه‌های اصلی و کلاینت مورد استفاده قرار گرفت. در آن زمان ذخیره اطلاعات بسیار گران بود. هزینه CPU نیز بسیار زیاد بود. به همین دلیل از Mainframe برای ذخیره سازی و پردازش استفاده می‌شد. برای دسترسی به داده‌ها و پردازش، از ترمینال‌های تخلیه استفاده می‌شد.

در سال ۲۰۰۶ آمازون شروع به فعالیت خود در زیر شاخه‌ای به نام خدمات وب آمازون کرد.

گوگل نسخه آزمایشی Google App Engine را در آوریل ۲۰۰۸ منتشر کرد. در همان سال ناسا OpenNebula را نیز معرفی کرد. این اولین پروژه منبع آزاد بود که برای خصوصیات ابرهای ترکیبی به کار گرفته شد.

در سال ۲۰۱۰ مایکروسافت Azure توسط مایکروسافت منتشر شد.

در سال ۲۰۱۲، موتور محاسبه Google قبل از اینکه در دسامبر ۲۰۱۳ در دسترس عمومی قرار بگیرد، در حالت پیش‌نمایش منتشر شد

در سال ۲۰۱۱ دکتر سلیمان و همکاران[۱] در مقاله‌ای، روش چندلایه‌ای را برای خدمات سلامت الکترونیکی مطابق با سند *ISO 17799* تعریف کرده و اطلاعات را به سه دسته اطلاعات سری، بسیار محرمانه و خصوصی تقسیم کرده؛ الگوریتم‌های رمزگذاری متقارن، و تابع مقدار هش را معرفی کرده اند. نویسندگان از اندازه کلید ۱۹۳ بیت برای لایه ۱ و ۱۲۹ بیت تا ۱۹۲ بیت برای لایه ۲ و ۱۱۲ تا ۱۲۸ برای لایه ۳ و ۸۰ تا ۱۱۱ بیت برای لایه ٤ استفاده کرده‌اند. کار اصلی نویسندگان روی الگوریتم‌های مختلف است. و از یک الگوریتم برای رمزگذاری و رمزگشایی استفاده می‌شود.

در سال ۲۰۱۳ کیا و همکاران[۲] در مقاله‌ای با استفاده از *SOAP / XML* داده‌ها را با *AES*رمزگذاری کرده اند.

در سال ۲۰۱٦ ژو و همکاران[۳] نویسندگان به خوبی مدل مراقبت‌های بهداشتی جدیدی را برای ذخیره داده‌های ابری در نظر گرفته‌اند. آن‌ها *RBE* (رمزنگاری مبتنی بر نقش) را اعمال کرده‌اند. ابتدا، آن‌ها مدل *PCEHR* (سوابق

[۱] R.sulaiman.D.Sharma,W.Ma and D.Tran
[۲] M.M.Kian,M.S.Nabi,B.Zaidan and A.Zaidan
[۳] L. Zhou, V. Varadharajan, and K. Gopinath

الکترونیکی کنترل الکترونیکی شخصی) را که توسط دولت استرالیا معرفی شده شرح داده‌اند. سپس *PCEHR* در *RBE* برای امنیت داده استفاده می‌شود. آن‌ها ساختار آرم داده‌ها و ویژگی‌هایش را بر اساس رمزگذاری طراحی می‌کنند و ادعا کردند که رویکرد آنها کنترل انعطاف‌پذیری در ذخیره‌سازی داده‌ها را فراهم می‌کند.

۳-۲ پیشینه مربوط به سال‌های اخیر

انتقال کد و داده‌ها به TEE، امکان راه اندازی خودکار و غیر قابل حمل پارامترهای اشاره‌گر در ارتباطات RPC را فراهم می کند.(Liu, Tan,۲۰۱۷) سنیر و همکاران یک مجموعه ابزار ارائه دادند که پروتکل های امنیتی را به چندین پارتیشن جداگانه تقسیم می‌کند تا نیازهای امنیتی را برآورده کند.(Senier, Beck,۲۰۱۷) روبینوف و همکاران از تجزیه و تحلیل برای پارتیشن بندی خودکار برنامه‌های اندروید برای اجرای قابل اعتماد، استفاده می‌کرد و به طور خودکار کد حساس را تشخیص داده و قطعات را برش می‌دهد(Mengmei, Sherman,۲۰۱۸). چارچوب تبدیل منبع به منبع لیند و همکاران زیر مجموعه هایی از آن را در برنامه‌های C برای استفاده ، استخراج می‌کند.

در سال ۲۰۱۹ سودهیپ و همکار[۱] در مقاله‌ای رمزگذاری مبتنی بر ویژگی سیاست رمزگذاری *(CP-ABE)* را معرفی کرده‌اند. کلید رمزگذاری شامل خط‌مشی‌هایی است و آن‌ها می‌گویند اگر کلید هک شده‌باشد، آن‌دسته از سوابق رمزگشایی می‌شوند که کلید آن‌ها هک می‌شود اما بقیه موارد همچنان محافظت می‌شوند.

در سال ۲۰۱۹ هما و همکار[۲]، درباره روش رمزنگاری منحنی بیضوی بحث کردند و روش‌های تولید کلید اصلی شخص ثالث را معرفی کردند. مالک داده را برای درخواست کلید و رمزگذاری سند به صورت آنلاین به بخش دیگر ارسال می‌کند. شخص ثالث رمز را رمزگذاری و به صاحب داده ارسال و مالک تاریخ را در سرور ابری بارگذاری می‌کند و کلید را برای استفاده در آینده نگه می‌دارد.

در سال ۲۰۱۹ پارا و همکاران[۳] از تکنیک‌هایی استفاده کردند که در آن، آرم داده‌ها با استفاده از درون‌یابی خطی ایجاد شده و سپس مستطیل جادویی با استفاده از الگوریتم *LSB* ایجاد و با استگانوگرافی، داده‌ها را رمزگذاری کردند.

[۱] K.Sudheep and Joseph
[۲] V.S.V Hema and R.Kesavan
[۳] S. A. Parah, A. Bashir, M. Manzoor, A. Gulzar, M. Firdous, N. A. Loan, and J. A. Sheikh

در سال ۲۰۱۹ وزید و همکاران[۱] نویسندگان در مورد مدل تهدید و احراز هویت برای دستگاه های مبتنی بر *Iot* در محیط ابر بحث کرده‌اند و سعی کرده‌اند چالش‌های فعلی امنیت و داده‌های مبتنی بر اینترنت اشیا در ابر را بررسی کنند. تمرکز اصلی آنها در تحقیق، سازوکار احراز هویت است و مفهوم مجازی تکنیک جدید را ارائه داده‌اند. در مقاله خود یک مطالعه تطبیقی در مورد هزینه‌های ارتباطی و فنی و حرفه ای انجام داده‌اند. محاسن و معایب تکنیک‌های احراز هویت موجود نیز در دست بررسی است اما راه‌حل مشخصی پیشنهاد نمی‌شود.

[۱] M. Wazid, A. K. Das, R. Hussain, G. Succi, and J. J. Rodrigues

فصـــل چهارم

۴.تجزیه و تحلیل موضوع انتخابی

٤-١ مقدمه

بیشتر بخش فنی این گزارش مربوط به بهبود امنیت و حریم خصوصی است. در این زمینه تحقیقاتی، سناریوهای خاصی تعریف می‌شود که شرح می‌دهد چگونه دشمنان می‌توانند علیه یک سیستم عمل کنند. برای افزایش امنیت سیستم و حریم خصوصی، باید از این اقدامات خصمانه جلوگیری یا حداقل مانع آن شد. این گزارش شامل سه بخش عمده است:

(۱) تجزیه و تحلیل برنامه و پشتیبانی از نتیجه گیری معناشناسی استفاده از متغیرهای برنامه

(۲) ابزارهای توسعه برای جداسازی داده ها و عملکردهای حساس

(۳)مکانیسم برنامه نویسی برای تبادل امن داده ها.

٤-٢ رفتارهای مهاجمان

(۱) آنچه انتظار می رود مهاجمان نتوانند انجام دهند:

الف) ما فرض می کنیم مهاجمان قادر به تغییر کد منبع یا نمایش کدهای میانی نیستند، تا تحلیل و تحول برنامه را مشکل همراه سازند. به طور خاص، رویکرد در فرآیندهای تجزیه و تحلیل و تحول برنامه نمی‌تواند توسط مهاجمان به خطر بیفتد.

ب) فرض می‌کنیم مهاجمان نمی‌توانند اجرا را به خطر بیاندازند، مهاجمان نمی‌توانند JVM را به خطر بیاندازند.

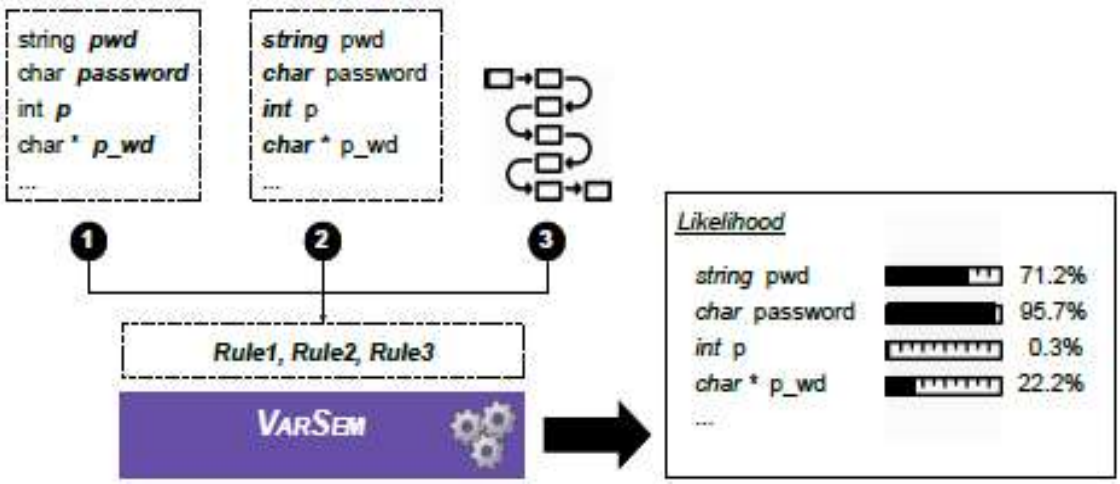
(۲) آنچه از مهاجمان انتظار می‌رود انجام دهند :

مهاجمان ممکن است سعی کنند *SPI* را در یک سیستم معین به خطر بیاندازند. در واقع، امکان نشت اطلاعات به شدت توسط توابع آسیب پذیر وجود دارد.به ویژه عملکردهای پردازش داده های حساس افزایش می یابد به عنوان

مثال ، با به خطر انداختن انتقال داده ها ، مهاجمان مکان های *GPS* فعلی را به شکل مخرب دریافت می کنند. علاوه بر این، خودسرانه افشای عملکردهای حساس برای تعامل با بازیگران خارجی می‌تواند به طور غیرقانونی مورد سوء استفاده قرار گیرد که باعث حذف فایل یا افشای اعتبار می‌شود. با قرار دادن کد *SPI* واقعی در *TEE*، از آسیب رساندن مهاجمان به *SPI* در سیستمهای بازسازی شده جلوگیری می‌شود. مهاجمان ممکن است تلاش کنند تا فرآیندهای تبادل داده اطلاعات حساس را به خطر بیاندازند. به عنوان مثال، به دلیل حمله، داده های حساس زیادی مانند مکان *GPS* را در معرض دید عموم قرار می‌دهند.؛ مکان کاربران در وب سایت افشا می شود زیرا برنامه تلفن همراه از موقعیت جغرافیایی اشتباه استفاده می‌کند. داده‌های حساس را بدون کسب مجوز کاربر ارسال می‌کنند. بدتر این است که داده‌های حساس چه به صورت درست و چه نامناسب به اشتراک گذاشته شده و می‌تواند به طور مداوم ذخیره شوند. سپس مهاجمان می‌توانند از این داده‌های حساس برای رخنه به حریم خصوصی استفاده کنند. مهاجمان ممکن است سعی کنند اشیای جاوا حاوی اطلاعات حساس را در حافظه به خطر بیاندازند؛ در واقع، دسترسی های مخرب به اشیاء جاوا تعداد برنامه های زیادی را تهدید کرده است مانند کپسوله سازی اشیا. حملات مفصل، به ویژه در محیط های توزیع شده، دسترسی خواندن و نوشتن به داده های حساس را مسدود می‌کند؛ در حالی که توسعه‌دهندگان را قادر می‌سازد از آنها استفاده کنند.

٤-٣ معناشناسی متغیرهای برنامه

در یک برنامه کامپیوتری ، متغیرها برای اهداف پیاده سازی خاص معرفی می شوند. اهدافی مانند معناشناسی استفاده متغیر. درک معانی کاربرد متغیر برای اکثر کارهای نگهداری و تکامل برای حسابرسی کد و درک برنامه مورد نیاز است. برای این منظور ، هر دو متن (طرح نامگذاری و اطلاعات زمینه و جریان داده) یک متغیر باید بررسی شوند تا کاربرد متغیر آشکار شود.



شکل ٤-١ ذخیره متغیرهای رمز کاربر

وظیفه شناسایی متغیرهایی را که رمزهای ورود کاربر را ذخیره می کنند ، در نظر بگیرید .وظیفه آن بررسی نحوه محافظت از رمزهای عبور در یک سیستم می باشد. همانطور که در شکل ٤-١ نشان داده شده است ، یک مهندس امنیت باید متغیرهایی را که نام آنها شبیه به است جستجو کند کلمه "رمز عبور" (به عنوان مثال ، "pwd"، "passwd"و "pass_word") ، ·متغیرهایی را که نوعشان می تواند اطلاعات رمز عبور (به عنوان مثال ، رشته) را ذخیره کند و متغیرهایی را که زمینه آنهاست جستجو کند.

اطلاعات با الگوهای استفاده خاصی مطابقت دارد (به عنوان مثال ، اطلاعات از ورودی کاربر به متغیر منتقل می شود . این قوانین جستجو را می توان به صورت مستقل یا به صورت یک زنجیره منطقی اجرا کرد.

برای کار ۱ ، برنامه نویسان معمولاً از امکانات جستجوی کد منبع ، مانند یونیکس استفاده می کنند .به عنوان مثال ، صدور فرمان grep "pass *word" *.c در پوسته یونیکس ، متن با پیشوند "pass" و پسوند "word" موجود در فایل های منبع C مطابقت دارد ؛ را برمی گرداند. با این حال ، این روش کاملاً شکننده است.

اسامی متغیرهای رمز عبور اغلب با کلمه "رمز عبور" تفاوت زیادی دارند .یک جستجوی ساده در GitHubنام های زیر را برای متغیرهای رمز عبور نشان می دهدpasswd" : ۱۱ میلیون مورد ، "p_wd"۹۲۵۸ مورد ، "pass_wd" ۲۸۰ مورد و "p_w_d" ١٦٤مورد .توسعه دهنده ممکن است یک متغیر رمز عبور را با یک حرف "p" یا an یا رشته بی ربط "abc" نامگذاری کند . بنابراین ، تطبیق رشته در توانایی آن کاملاً ناکافی است .در نتیجه نیاز به تلاش های اضافی دستی برای فیلتر کردن است.

وظایف ۱و۲ ، بازرسی دستی انواع متغیرها و جریان اطلاعات برای پروژه های کوچک می باشد ، اما برای پروژه های بزرگ غیر واقعی است .تجزیه و تحلیل مبتنی بر کامپایلر می تواند برای جستجوی انواع متغیرها و جریان اطلاعات به صورت خودکار استفاده شود .با این حال، استفاده صحیح از این ابزارها مستلزم آن است که برنامه نویسان تخصص کافی در کامپایلرها را کسب کنند و آن سطح انتظار از تخصص مربوط به همه برنامه نویسان برنامه یا مهندسان امنیت غیر واقعی است.

آخرین تحولات در یادگیری ماشین (ML) ، یادگیری عمیق (DL) و زبان طبیعی پردازش (NLP) ابزارهای جدیدی را برای تجزیه و تحلیل معانی کاربرد متغیر ارائه می دهد. با این حال، استفاده صحیح از این رویکردهای پیشرفته به دلیل موارد زیر کاملاً چالش برانگیز است دلایل: (الف) کاربردهای موجود این ابزارها با نام شناسه کاملاً متفاوت هستند ،. به عبارت دیگر ، این رویکردها با توجه به مدلهای متفاوت متناسب با سناریوهای خاص خود، کاربرد دارند ، بنابراین هیچ مدل و تکنیکی مخصوصی برای تجزیه و تحلیل معناشناسی کاربرد متغیر وجود ندارد. ب) دقت و فراخوانی بسیاری از این تکنیک ها برای کاربردهای متغیر عملی و وظایف تجزیه و تحلیل معناشناسی کافی نیست. به عنوان مثال توصیه می کنید نام شناسه های توصیفی استفاده شود. ج) اهداف این تکنیک ها معمولاً روی بخش هایی از کد کاربرد دارد و نه متغیرهای فردی. یعنی متنی و زمینه متغیرهای موجود اطلاعات به عنوان نشانه و ویژگی کل قطعه کد تلقی می شوند. با این حال، متغیرها اجزای برنامه متمایزی هستند. یک متغیر نه تنها اطلاعات را ذخیره می کند ، بلکه اطلاعات را نیز ذخیره می کند همچنین می تواند با سایر اجزای برنامه (به عنوان مثال ، خصوصیات کلاس ، پارامترهای تابع) و بخشی از هر زمینه برنامه (به عنوان مثال ، کنترل و جریان داده) باشد. انتخاب متغیرهایی (به عنوان مثال ، معانی و کاربرد آنها) تجزیه و تحلیل برنامه مشکلی دارد. NLP دارای قواعد اعلانی و قابل تنظیم برای تجزیه و تحلیل روال برای استنباط اطلاعات متغیر و متنی است . که می تواند هم برای توسعه دهندگان و هم تحلیلگران امنیتی مفید باشد.

٤-٤ جداسازی داده های حساس و توابع

اجرای سیستم های زمان واقعی حیاتی باید با محدودیت های زمان واقعی مطابقت داشته باشد .بسیاری از چنین سیستم هایی همچنین حاوی اطلاعات حساس برنامه (SPI) هستند. الگوریتم ها و داده ها باید محافظت شوند. عدم رعایت هریک از این الزامات می تواند منجر به عواقب فاجعه بار شود. استفاده از تحویل خودکار را در نظر

بگیرید حمل بسته ها ، حاوی غذا ، آب ، دارو یا واکسن ، از راه دور و مکانهای سخت دسترسی پرسنل اورژانس حرفه ای هنگام مواجهه با بحران های مختلف است .

ناوبری پهپاد دارای محدودیت های زمان واقعی است ؛ اگر نتواند دستورالعمل تنظیم خلبان خودکار را محاسبه کند جهت پرواز یا سرعت پرواز به موقع ، پهپاد ممکن است نتواند تنظیم شود که مسیر آن را به درستی تغییر دهد و از مسیر تحویل برنامه ریزی شده منحرف شود. از آنجا که بار اغلب باید تحت شرایط سخت زمانی تحویل داده شود و از کوتاهترین مسیر می تواند منحرف شود و باعث شکست کل ماموریت تحویل شود. علاوه بر این ، ماژول کنترل نرم افزار (به عنوان مثال ، ناوبری) اطلاعات حساس برنامه (SPI) را تشکیل می دهد. اگر یک عامل مزاحم ، کنترل عملکرد ماژول را مختل کند ، کل هواپیمای بدون سرنشین را می توان به اشتباه مسیریابی کرد و باعث شکست در تحویل شود.

با جداسازی توابع SPI در یک اجرای امن می توان آسیب پذیری از محیطی که تعاملات آنها را با جهان خارج نیز کنترل می کند را کاهش داد. به عنوان یک راه برای تحقق این ایده ، تولید کنندگان سخت افزار شروع به ارائه اجرای قابل اعتماد در محیط امن می کنند. پردازنده های خاص که می توانند برای اجرای وابسته به SPI برای عملکرد امن استفاده شوند. می تواند با اطمینان کد قابل اعتماد را از حالت عادی جدا کند ؛ محیط امن با سخت افزار قابل اعتماد ، ذخیره سازی و سیستم عامل ارتباطی ویژه تنها راه برای تعامل با کد مبتنی بر TEE است . برای حل مشکل سازش TEE ها ، جدا کردن SPI در امنیت محیط به طور موثر با حملات خصمانه مقابله می کند و از سرقت مالکیت معنوی جلوگیری می کند .با این حال ، برای بهره مندی از اجرای مطمئن ، سیستم ها باید طراحی و پیاده سازی مختلفی از TEE انجام شوند . سیستم های زمان واقعی برای استفاده از TEE نیاز به تغییر و تطبیق برنامه ها مستعد خطا دارند .

به طور خاص ، توسعه دهنده یک سیستم را تغییر می دهد تا از مزایای آن استفاده کند. ماژول TEE به انجام کارهای زیر نیاز دارد:

۱) جدا کردن کد وابسته به SPI

۲) هدایت فراخوانی توابع SPI به در محیط ارتباط TEE

۳) بررسی این که سیستم تغییر یافته همچنان با محدودیت های اصلی زمان واقعی مطابقت دارد.

توجه کنید که همه انجام این کارها دشوار است .

برای تکمیل وظیفه ۱ ، توسعه دهنده نه تنها باید به درستی کد وابسته به SPI را استخراج کند بلکه باید همه وابستگی ها را نیز به درستی شناسایی کند. به دلیل پتانسیل پیچیدگی این وابستگی ها ، برخی از کد های وابسته به SPI را نمی توان از TEE جدا کرد .

برای تکمیل وظیفه ۳ ، توسعه دهنده باید مایل به توسعه موارد آزمایشی دیگری باشد که می تواند بررسی کنید که آیا سیستم تبدیل شده محدودیت های اصلی زمان واقعی را برآورده می کند یا خیر.

رویکردها از ابزارهای پروفایل ، از جمله ابزار Pin استفاده می‌کنند که مستلزم آن است که پروفایل بهبود یابد. برای تسهیل فرایند تطبیق سیستم های زمان واقعی برای محافظت از کد وابسته به SPI آنها این مقاله با استفاده از TEE، یک مجموعه ابزار تجزیه و تحلیل برنامه و ارائه می دهد که از توسعه دهندگان برای تقسیم بندی سیستم های زبان C در زمان واقعی پشتیبانی می کند.

توسعه دهنده می تواند پیاده سازی TEE به عنوان یک گزینه کامپایلر ، یا برای تعیین خودکار موجود ، اعمال کند. پیاده سازی با تحلیل سیستم از طریق یک مدل برنامه نویسی با تک تک عملکردها برای جدا شدن در محیط امن اعمال شود ، بنابراین می توان آن را با موفقیت تقسیم کرد .در نهایت ، سیستم را به قسمتهای معمولی و قابل اعتماد ، تبدیل می کند. در صورت عدم موفقیت کد تبدیل شده برای برآوردن محدودیت های زمان واقعی ، محیط را اصلاح می کند.

برای ادامه ٤ حوزه را مطرح می کنیم :

۱- یک مدل برنامه نویسی فراگیر برای تقسیم بندی سیستم های زمان واقعی برای استفاده از TEE ها که با زبان C نوشته شده است. این مدل به صورت خاص الزامات سناریوهای مختلف پارتیشن بندی را در بر می گیرد.

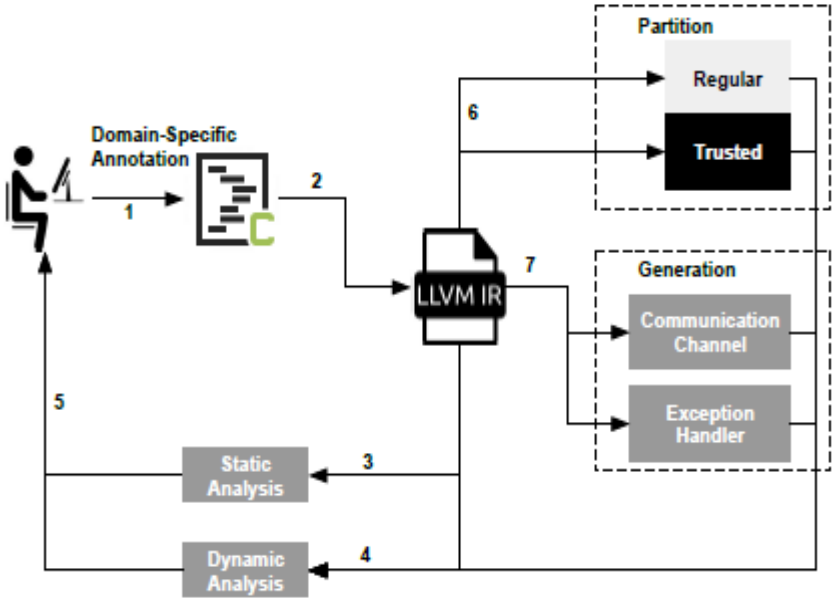
۲- مکانیزم های بررسی استاتیک و پویا که مشخص می کند آیا یک سیستم می تواند برای پیاده سازی TEE مشخص شده باشد یا خیر و احتمال تقسیم بندی آن چقدر است. هدف برآوردن محدودیت های اصلی در زمان واقعی است. مکانیزمی که به توسعه دهندگان اطلاع می دهد چگونه می توانند سیستم های خود را بازسازی کنند ، بنابراین آنها را می توان با موفقیت تقسیم کرد.

۳- تغییر برنامه مبتنی بر کامپایلر برای برنامه های زبان C کار می کند ، در حالی که کانال های ارتباطی سفارشی را نیز ایجاد می کند و زمان واقعی ماسب را دارد.

٤- معیار مستقل از محیط برای سنجش میزان تابع SPI که انتظار می رود هنگامی که به TEE منتقل می شود ، عملکرد آن را پایین بیاورد و چنین مواردی را مقایسه کند تخریب بین TEE های مختلف ؛ ما کاربرد این معیار را مورد ارزیابی قرار می دهیم.

٤-٤-١ مرور راه حل

در این بخش ، ما زنجیره ابزار تجزیه و تحلیل مبتنی بر کامپایلر و بازسازی کد را معرفی می کنیم. ابزار و سپس ورودی و خروجی محیط امن را شرح می دهیم.

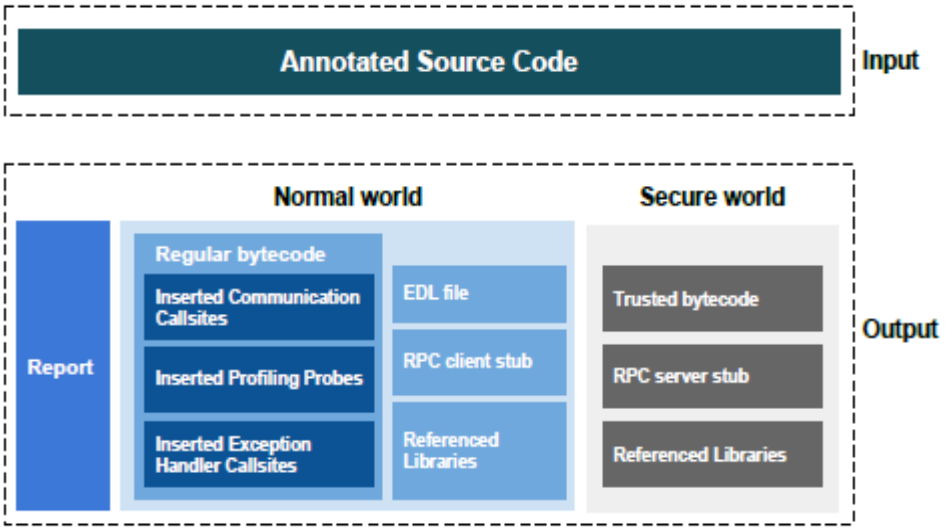


شکل ٤-٢ مراحل پردازش محیط امن

فرایند توسعه نرم افزار شکل ٤-٢ نشان داده شده است که فرایند توسعه نرم افزار برای پارتیشن بندی بلادرنگ را تشریح می کند. استفاده از سیستم های TEE با توجه به سیستم زمان واقعی به این گونه است که ابتدا توسعه دهنده توابع وابسته به SPI را در کد منبع با استفاده از دامنه امن مشخص می کند (مرحله ١)
کد منبع به محل مورد نظر منتقل میشود. (مرحله ٢)

پس از آن ،محیط امن تعیین می کند که آیا TEE یا SGX با بررسی محیط اجرا یا پیکربندی ساخت به صورت درست پیاده سازی شده است یا خیر همچنین آیا سناریوی پارتیشن بندی مشخص می تواند محقق شود یا خیر. محیط امن به طور آماری، نمودار تماس سیستم تجزیه و تحلیل می کند. (مرحله ۳)

با توجه به نمودار فراخوانی سیستم و پارتیشن بندی، نمودار عملکرد قابل تقسیم (PFG) را که شامل تمام اطلاعات مورد نیاز برای تعیین صحت مشخصات است ؛ می سازد. در حالی که تحلیل استاتیک اعتبار معنایی مشخصات پارتیشن بندی ، یک تحلیل پویا جداگانه را تعیین می کند. در این مرحله مشخص می شود که آیا سیستم پارتیشن بندی شده مطابق محدودیت های زمان واقعی عمل می کند یا خیر. (مرحله ٤)

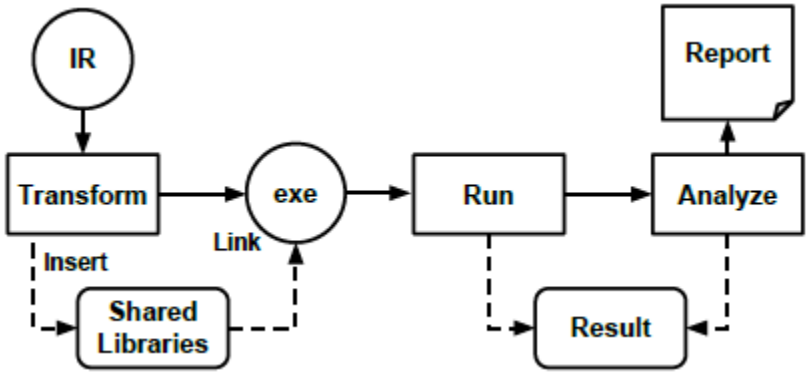


شکل ٤-۳ ورودی و خروجی RT-Trust

٤-٤-۲ تولید کد

شکل ٤-۳ ورودی‌ها و خروجی‌های کد RT-Trust را نشان می‌دهد. به عنوان ورودی، RT-Trust ، کد منبع حاشیه نویسی شده را دریافت می کند و به عنوان خروجی ، مشخصات منبع ورودی را تغییر می دهد و همچنین کد اضافی تولید می کند که کامپایل شده و در حالت عادی و امن ادغام می شود. پارتیشن های محیط عادی ، RT-Trust را با درج پروفایل در کاوشگرها ، کنترل کننده های استثنا و کانال های ارتباطی تغییر می دهد .

تجزیه و تحلیل پویا RT-Trust به شناسایی احتمال تقسیم بندی مشخص کمک می کند تا محدودیت های اصلی زمان واقعی را برآورده کند .از آنجا که تضمین اینکه آیا اجرای پروفایل بدترین حالت را دارد؛ تجزیه و تحلیلهای ما فقط برای سیستم های زمان واقعی، قابل اجرا است.



شکل ٤-٤ مراحل تحلیل RT-Trust

شکل ٤-٤ نشان می دهد که چگونه RT-Trust قابلیت تجزیه و تحلیل پویا را فراهم می کند .تحلیل ها با تغییر برنامه اصلی یعنی درج RT-Trust شروع می شود. کد پروفایل در توابع محدودیت های زمان واقعی به جای وارد کردن کل کد نمایه وارد می شود؛ RT-Trust تماس هایی را با توابع پروفایل ویژه ، برقرار می کند. به عنوان بخشی از کتابخانه های مشترک در دسترس قرار می دهد. اکنون RT-Trust توابع را به تنهایی ارائه می دهد. عملکردهای مشابهی را می توان توسط کتابخانه ها ارائه داد. این طراحی انعطاف پذیر توسعه دهندگان را قادر می سازد تا خدمات سفارشی خود را ارائه دهند نمایه کردن کتابخانه ها یا افزودن ویژگی های جدید به کتابخانه هایی که توسط RT-Trust ارائه شده است منطق پروفایل را تقویت می کند. پس از پیوند دادن کتابخانه های مشترک با محیط تبدیل شده برنامه ، توسعه دهندگان با فراخوانی تابع درج شده، عملکرد مناسب را انجام می دهند.

عملکردهای پروفایل در کتابخانه های مشترک توابع، محدودیت های زمان واقعی را اندازه گیری می کنند و داده های نتیجه را برای تجزیه و تحلیل آینده حفظ می کند .در نهایت ، RT-Trust داده ها را تخمین زده و مشخص می کند آیا توابع حاشیه نویسی می توانند نیازهای اصلی زمان واقعی را برآورده کنند و گزارش مجدد نتایج به توسعه دهنده داده می‌شود.

جدول ٤-١ : تلاش های برنامه نویس ها (ULOC)

Algorithm	RTTAs	Generate and Transform		Adjust	
		OP-TEE	SGX	OP-TEE	SGX
CRC٣٢	٥	٣٨٨	٨٧	٠	٠
PC١	٤	٣٤٤	٧٣	٦	٦
RC٤	٣	٢٩٢	٦١	٣	١
MD٥	٣	٣٦٤	٨٦	٣	١
DES	٢	٢٤٤	٤٦	١٥	٣

٤-٤-٣ بررسی نتایج تولید کد

صحت RT-Trust را با اعمال تمام شرایط (یعنی تجزیه و تحلیل کد ، تبدیل ، و تولید) را به شرح زیر ارزیابی می‌شود. با استفاده از RT-Trust تعداد کل ULOC به طور خودکار توسط RT-Trust تولید و تبدیل می شود. به طور مثال ٣٨٤ ~ ٢٤٤ برای OPTEE ؛ ٤٦ ~ ٨٧ برای SGX . نتایج کامل در جدول ٤-١ نشان داده شده است.

جدول ٤-٢ – سربار پروفایل RT-Trust برحسب میلی ثانیه

Algorithm	Execution Time		Invocation Intervals		Memory	
	Normal	Secure	Normal	Secure	Parameter	Local
OP-TEE	٠,٤٤٢	١٤٤	٠,٤١٨	١٣٩	٠,٠٥١	٠,٠٥٣
SGX	٠,٢	٥٢	٠,٢١٢	٢٥,٥		

عملکرد: جدول ٤-٢ گزارش مربوط به سربار پروفایل RT-Trust و زمان اجرا را نشان می دهد. فواصل فراخوانی و مصرف حافظه را محاسبه می کند. RT-Trust سیستم ها را قبل و بعد از بازسازی آنها مشخص می کند. برآورد حالت قبل که آیا سیستم بازسازی شده همچنان به محدودیت های زمان واقعی پاسخ می دهد ، در حالی که حالت بعد ویژگی های اجرایی برآورد شده را با مواردی که در آن اجرا شده است مقایسه می کند. سخت‌افزار محیط‌ها به شدت تفاوت زیادی در سربار پروفایل دارند:

برای OP-TEE، ٠,٤ میلی ثانیه در دنیای عادی در مقابل ١٤٠ میلی ثانیه در دنیای امن.

برای SGX ، در دنیای عادی ٠,٢ میلی ثانیه در مقابل ٥٠ میلی ثانیه در دنیای امن.

این تفاوت عمدتا به دلیل تفاوت بین کارایی سیستم استاندارد لینوکس است.

جدول ٤-٣ – محدودیت های TEE

Limitations	OP-TEE	SGX
Language	C	c/c++
Memory	No limit	Hard limit
Threading	No	No
Sys./Lang.APIs	Sepecial version	Sepecial Version

محدودیت های TEE: جدول ٤-٣ محدودیت های OP-TEE و SGX را نشان می دهد. برای پشتیبانی زبان ، قسمت مورد اعتماد OP-TEE فقط به زبان C قابل نوشتن است. که برای SGX را می توان در دو زبان C و C ++ نوشت ، در حالی که کانال ارتباطی بین قطعات مورد اعتماد و غیرقابل اعتماد را فقط می توان با C نوشت. برای تخصیص حافظه ، OP-TEE محدودیت اندازه ثابت ندارد و با محدوده بالایی به مقدار حافظه فیزیکی تبدیل می‌شود. در مقابل، حداکثر اندازه حافظه محافظت شده SGX توسط سیستم محدود می شود.

جدول ٤-٤ – FPI مربوط به OP-TEE و SGX

Algorithm	OP-TEE	SGX
CRC٣٢	٠,٩٨٢	٠,٩٧٣
PC١	٠,٥٨١	٠,٦
RC٤	٠,١٤٢	٠,٤٣٦
MD٥	٠,٢١٨	٠,٢٠٥
DES	٠,٧٦٦	٠,٨٠٣

٤-٥ انتخاب بین OP-TEE یا SGX

جدول ٤-٤ شاخص عملکرد (FPI) هر یک از معیارهای خرد را برای OPTEE و SGX را نشان می دهد . به طور کلی ، ارزش FPI برای هر دو TEE در همه معیارها قابل مقایسه است. هرچه سرعت اجرا قبل از انتقال به TEE بیشتر باشد ، مقدار FPI بزرگتر است. (یعنی بیشتر از کاهش عملکرد). دلیل آن این است که اگر یک تابع سریع اجرا شود ؛ هزینه های اضافی کانال ارتباطی وجود دارد.

به طور خلاصه ، توسعه دهندگان همیشه باید از TEE با استفاده کنند کخ کوچکترین مقدار FPI را دارد . با این حال ، اگر زمان اجرای یک عملکرد SPI بسیار کمتر از زمان صرف شده توسط کانال ارتباطی باشد. سپس OP-TEE و SGX قابلیت تخریب بالا دارد.

RT-Trust که یک برنامه نویسی فوق العاده کامل را ارائه می دهد؛ مطرح . مدل با تجزیه و تحلیل استاتیک و پویا برای تعیین اینکه آیا استراتژی پارتیشن بندی منطقی است و آیا سیستم پارتیشن بندی مطابق با محدودیت های اصلی در زمان واقعی است و یک بازسازی خودکار که نسخه اصلی را هنگام ایجاد ارتباط RPC سفارشی و کد مدیریت استثنا تغییر می دهد. ما رویکرد به طور خودکار سیستم های زمان واقعی را با عملکردهای وابسته به SPI برای موارد قابل اعتماد بازآرایی می کند. اجرا تحت محدودیت های زمان واقعی نتایج ارزیابی استفاده از RT-Trust به نشانهای کوچک نشان دهنده برنامه نویسی امن به منظور کاهش تلاش برنامه نویس مورد نیاز برای جداسازی SPI تحت محدودیت های زمان واقعی است .

٤-٦ شناسایی و مهاجرت کد غیر حساس در TEE

تعداد زیادی از دستگاه های محاسباتی به طور مداوم حجم عظیمی از داده ها را جمع آوری می کنند (به عنوان مثال، شناسه های بیومتریک ، موقعیت جغرافیایی و تصاویر) ، که بسیاری از آنها حساس هستند. داده‌های حساس و پردازش کد آنها هدف بسیاری از افشای داده ها و حملات تغییر کد است. مکانیسم حفاظتی که به طور فزاینده ای محبوب می شود ، کد و داده های دنیای خارج در یک محیط اجرای مطمئن (TEE) حساس را جدا می‌کند.

کد و داده های دنیای خارج ۱ در یک محیط اجرای مطمئن (TEE) : با افزایش حجم کد در TEE ،همه کد حساس نیست ، بنابراین پایگاه محاسباتی مورد اعتماد (TCB) بی جهت رشد می کند و باعث می شود تا مسائل مربوط به عملکرد و امنیت وقتی نوبت به عملکرد می رسد ، تحقیقات قبلی آن را مشخص می‌کند.

ارتباط بین TEE و جهان خارج به عنوان یک تنگنای عملکردی که می تواند بیشتر زمان اجرا را مصرف می‌کنند. به عنوان مثال ، فراخوانی تابع متعدد ، ورود یا خروج از TEE ، باعث ایجاد حجم زیادی از ارتباطات ورودی/ خروجی و کند شدن سرعت کل سیستم ارتباط می‌شود.

٤-٦-١ مبادله داده های حساس

موبایل ، اینترنت اشیا و دستگاه های پوشیدنی به طور مداوم حجم بیشتری از داده های کاربر را جمع آوری می‌کند؛ مانیتورهای بهداشتی علائم حیاتی صاحبان خود را ردیابی می کنند. تلفن های هوشمند داده های شخصی حسی ، می خوانند و از جمله مکان GPS ، سرعت ، جهت و غیره. دستگاه های اینترنت اشیا به دست می آورند. وقتی صحبت از داده های حساس می شود ، یک اصل اساسی وجود دارد : تضاد بین نیازهای کاربر نهایی و آرزوهای توسعه دهنده برنامه.

کاربران نهایی می‌خواهند مطمئن شوند که اطلاعات حساس آنها خصوصی و غیرقابل دسترسی برای افراد غیرقابل اعتماد است. توسعه دهندگان برنامه می خواهند از خواص داده های حساس برای ارائه استفاده کنند. برنامه های کاربردی هوشمندی که تجربیات شخصی کاربر را ارائه می دهند و هوشمند و حساس به زمینه خدمات. هستند؛ این دو هدف ظاهراً آشتی ناپذیرند.

سه نمونه زیر را از برنامه های کاربردی با حجم بالا که به طور بالقوه کار می کنند در نظر بگیرید

(۱) برنامه های انتقال تلفن همراه اطلاعات ترافیک را در زمان واقعی به کاربران خود ارائه می دهند.همچنین هنگام استفاده از برنامه های کاربردی ، این اطلاعات را به اشتراک بگذارید. یک برنامه انتقال به طور مداوم GPS فعلی دستگاه خود را روی ابر بارگذاری می کند. مکان و سرعت ، که سپس برای تخمین اطلاعات ترافیک در زمان واقعی استفاده می شود. با اینکه این اطلاعات به صورت ناشناس بارگذاری می شوند ، با توجه به داده های GPS و سرعت کافی ، این کار قابل اعتماد نیست. ممکن است طرف بتواند با روال روز مالک دستگاه آشنا شود و از این اطلاعات برای اهداف نادرست، سوء استفاده کند. بنابراین ، اگرچه مشارکت کنندگان ممکن است مایل به تخمین اطلاعات ترافیکی در زمان واقعی ، باشند و نمی خواهند مکان GPS آنها فاش و ضبط شود.

(۲) گروهی از دوستان به دنبال رستورانی هستند که با هم شام بخورند. تلفن های هوشمند آنها، تاریخ ناهارخوری صاحبان خود را حفظ کنند. بر اساس تاریخ غذاخوری هر فرد ، سرور در یک میدان خرید می تواند پیشنهاد کند که کدام رستوران ها برای این مکان مناسب تر هستند. ممکن است افراد مایل نباشند که تاریخچه ناهار خوری خود را با یک گروه غیر قابل اعتماد به اشتراک بگذارند.

(۳) یک ساختمان هوشمند ممکن است درجه حرارت و سطح روشنایی خود را مطابق با حرارت تنظیم کند. ترجیحات ساکنان فعلی آن کاربرانی که دارای ردیاب سلامت شخصی هستند و مجهز به تلفن های هوشمند هستند ؛ می توانند حیات و مکان ساختمان صاحبان خود را گزارش کنند. این اطلاعات یکی از عناصر کلیدی است که ساختمان را "هوشمند" می کند. افراد ممکن است مایل نباشند مواد حیاتی و مکان فعلی آنها را به یک شخص غیرقابل اعتماد فاش کنند.

لطفاً توجه داشته باشید که در هر سه مثال بالا ، هنوز خدمات هوشمند قابل ارائه است. در حالی که داده های حساس کاربران نامرئی است. برای محاسبه میانگین سرعت گزارش شده ، سیستم نظارت بر ترافیک نباید از سرعت واقعی هر وسیله نقلیه عبوری آگاه باشد. این اطلاعات حساس می تواند مخفی بماند و فقط میانگین آماری آنهامحاسبه و در پیش بینی شرایط فعلی ترافیک استفاده می شود. برای توصیه یک رستوران ، سرور نباید نیاز داشته باشد که افراد درگیر چه رستوران های خاصی دارند و می تواند یک رستوران قابل قبول پیشنهاد کند. در مورد محبوب ترین غذاهای هر فرد ، اطلاعات آماری را می توان به دست آورد.

٤-٧ پاسخ به سوالات تحقیق

برای درک تاثیر معنایی برنامه، باید پاسخ‌هایی به سوالات مطرح شده در فصل اول داشته باشیم. :

١. استراتژی های پیشرفته درک مطلب چیست؟
٢. از کدام ابزارهای نرم افزاری می توان برای تسهیل فرایند درک استفاده کرد؟
٣. از کدام تکنیک می توان درک برای تشخیص اطلاعات مربوط به برنامه استفاده کرد؟
٤. چگونه می توان سیستم‌ها را برای تبادل ایمن *SPI* فعال کرد؟

جواب سوال١-استراتژی های درک معناشناسی: استراتژی های درک معنایی متکی بر تکنیک ها هستند. تکنیک ها را می توان در گروه های زیر طبقه بندی کرد:

الف) داده کاوی: وایمر و همکاران، کد و برنامه ای برای اشکال زدایی برنامه ها استخراج کردند(Weimer, Necula,٢٠٠٥). هاست و همکاران از داده کاوی برای استخراج قوانین از برنامه های کاربردی جاوا برای شناسایی نام توابع غیر معمول و غیرقابل قبول استفاده می کنند(Host, ostvold,٢٠٠٩).

(ب) تحلیل استاتیک: جستجوی معنایی مبتنی بر تحلیل استاتیک را برای درک استفاده از API اعمال می‌شود.

(ج) ML/NLP: از طریق مدل زبان و سبک برنامه نویسی برای توصیه نام شناسه و قراردادهای قالب بندی استفاده می‌شود. از دستگاههای بردار پشتیبانی ساختار یافته (SSVM) برای پیش بینی نام و نوع شناسه ها برای اعلانات پروژه های جاوا اسکریپت استفاده می‌شود. از یادگیری عمیق برای یادگیری کد جاسازی و پیش بینی نام تابع ها استفاده می‌شود.

(د) : الگوریتم خاصی تشخیص می دهد که آیا فراخوانی یک تابع پارامترهای صحیح را از نام یک شناسه ارسال می کند یا خیر. الگوریتم اسناد را برای روش های جاوا با انتخاب دستورات مهم کد و بیان آنها به عنوان زبان طبیعی نشان می‌دهد. مدل توصیفی اندازه گیری خوانایی کد را مشخص می‌کند.

این راه حل ها برای کاربردهای خاص در نظر گرفته شده است و بر نظریه های متفاوت متکی است و الگوریتم ها آنها همچنین بر اطلاعات زمینه کد منبع ، تمرکز می کنند ، درک معنایی استراتژی – تجزیه و تحلیل معنایی کاربرد متغیر (VUSA) – هم متنی و هم زمینه ای اطلاعات برای استنباط معناشناسی متغیر را در نظر می گیرد.

جواب سوال ۲ – ابزارهای کنونی برای حمایت از درک معناشناسی: PQL یک زبان درخواست برنامه برای جستجوی الگوهای کد منبع خاص است و VFQL یک زبان درخواست برنامه برای بیان و جستجوی نقص کد از طریق نمودارهای جریان مقدار است . Wiggle یک سیستم پرس و جو است که برای جستجوی کد منبع با ویژگی های متنی یا زمینه مشخص شده توسط کاربر ، از استفاده می شود. طراحی زبانی برای جستجوی الگوهای خاص در کد منبع جاوا را مطرح کردند. برخی ابزارهای آنلاین و افزونه های می توانند الگوهای کد داده شده را پیدا کنند. با این حال، هیچ کدام این زبانها بر استنباط معناشناسی متغیر تمرکز نمی کنند.

جواب سوال ۳و ٤ – درک معنایی برای تشخیصSPI : از آنجا که اطلاعات متنی داده های حساس ، می تواند افشا شود؛ خطرات احتمالی امنیت و حریم خصوصی را می توان با راه حل مناسب تشخیص داد. معناشناسی داده که به طور مستقل اجرا می شود؛ به طور خودکار ورودی کاربر حساس با استفاده از تکنیک های استخراج شده برای شناسایی کلمات کلیدی مشکوک شناسایی می کند.

فصـــل پنجم

۵. جمع‌بندی و پیشنهادها

۵-۱ مقدمه

در این بخش نتایج حاصل از تحقیق با توجه به مباحث پژوهش‌شده پرداخته شده و پیشنهادات ارائه می‌شود و مزایا و معایب و نکات اصلاحی از دیدگاه شخصی و با توجه به تجربه‌ام در زمینه تحقیقاتی در خصوص بهبود طرح مطرح می‌شود.

۵ – ۲ نتایج حاصل از تحقیق

ظهور دستگاه های تلفن همراه ، پوشیدنی و IoT حجم زیادی از داده ها را ایجاد کرده است که دارای برخی محدودیت های حریم خصوصی است. کاربران نهایی می خواهند حساسیت خود را حفظ کنند. داده ها خصوصی هستند ، در حالی که شرکت مایل است از این داده ها برای ارائه هوشمند و حساس به زمینه خدمات و برنامه های کاربردی استفاده کند.

در این کار ، ما استدلال می کنیم که نوآوری در فناوری برنامه نویسی می تواند به حل این موارد کمک کند. دو دستور کار متضاد: داده های حساس را می توان خصوصی نگه داشت ، در حالی که شرکت ها هنوز می توانند ادغام شوند اطلاعات تجاری ارزشمند از داده ها ، نحوه طراحی ، پیاده سازی را ارزیابی کرد.

در حالی که چرخه حیات خود را کنترل می کند و با سیاست پرس و جو ، رابط های برنامه نویسی را برای انجام محاسبات آماری ، ارائه می دهد ؛ به طوری که توسعه دهندگان قادر به ساخت تلفن همراه هوشمند و برنامه های کاربردی اینترنت اشیا هستند.

علیرغم مزایای حفظ حریم خصوصی ObEx که در بالا نشان داده شد ، باید توسط JVM بومی پشتیبانی شود. این پشتیبانی بومی می تواند:

(۱) پیچیدگی افزایش امتیازات و حملات جایگزینی کتابخانه را پیچیده کند.

(۲) ادغام کردن مدیریت چرخه توسعه نرم‌افزار

۳) از وابستگی جلوگیری کند.

پیچیدگی افزایش امتیاز و حملات جایگزینی کتابخانه: از آنجا که زمان اجرا محلی ObEx یک کتابخانه مشترک است که توسط JVM بارگیری شده است ، افزایش امتیاز (به عنوان مثال ، root حساب در لینوکس) می تواند مستقیماً به داده های حساس موجود در حافظه دسترسی پیدا کند. علاوه بر این ، یک حمله می تواند زمان اجرای ObEx را با یک نسخه مخرب در زمان بارگذاری جایگزین کند. ادغام با JVM می تواند یکپارچگی مکانیسم حفاظت از حریم خصوصی ObEx را افزایش دهد.

۵-۳ بررسی معایب پایان‌نامه مورد بررسی و بیان پیشنهاد

۵-۳-۱ مشکلات مربوط به نوآوری‌ها

تحقیقات پایان‌نامه مورد بررسی، نوآوری‌هایی را در زمینه در نرم افزار و فضای مهندسی برای درک و مدیریت SPI با استفاده از رویکردها، نشان می‌دهد. تکنیک‌های و ابزارها، یک توسعه دهنده می‌تواند توابع و متغیرهای SPI را شناسایی کرده و همچنین آنها را در برابر آنها نشت داده ها محافظت کند.

با این وجود، انتظار این که تمام مشکلات مربوط به حفاظت از SPI را برطرف شود؛ غیر واقعی است. به عنوان مثال ، هر زمان که برنامه های تلفن همراه مبادله داده ها را انجام دهند ، این فرآیند ممکن است همچنان خطرات نشت داده احتمالی مانند حملات علیه کانال ارتباطی ، از جمله رهگیری ، استراق سمع را داشته باشند. برای حل این مشکلات پیشنهاد می کنیم که در زمینه طراحی وسایل واسط نیز نوآوری داشته باشیم تا بتوانیم موارد جدیدی را ارائه دهیم طرح هایی که می توانند خطرات نشت داده را کاهش دهند:

(۱) به سوی ارتباطات مبتنی بر پیام ایمن: در سیستم عامل های تلفن همراه مدرن ، تبادل اطلاعات بین مولفه ها تحت تأثیر حملات نشت داده قرار می گیرد که از طریق آنها غیرقابل اعتماد است. برنامه ها به داده های منتقل شده دسترسی پیدا می کنند. دفاعیات موجود نیز بیش از حد محدود کننده هستند. همه مبادلات مشکوک داده ها را مسدود کنید ، بنابراین از دریافت هرگونه برنامه به برنامه جلوگیری می کنید. برای بهتر شدن تبادل اطلاعات بین مولفه های ایمن ، هدف ما طراحی و پیاده سازی مدلی است که امنیت را تقویت می کند ، در حالی که به برنامه های غیرقابل اعتماد اما غیر مخرب اجازه می دهد تا برنامه های خود را در زمینه منطق کسب و کار اجرا کنند. و

تحویل داده های حساس در یک پاکت رمزگذاری شده پنهان منتقل می شوند. تحویل آنها چند شکلی است: با توجه به قابلیت اطمینان مقصد ، می تواند باشد. بدون داده ، داده خام یا داده رمزگذاری شده باشد.
(۲) خصوصی سازی اشتراک گذاری داده‌های دستگاه: برای به حداکثر رساندن رضایت مشتری ، خدمات مدرن تلفن همراه (به عنوان مثال ، تبلیغات ، انتقال ، مراقبت های بهداشتی) باید شخصی باشد. کاربران برای شخصی سازی خدمات خود و ارائه دهندگان برنامه به طور مداوم، موارد لازم را انجام داده و داده ها را به صورت محلی یا از طریق ابر با دیگران به اشتراک می گذارند. با این حال ، مکانیسم های موجود برای به اشتراک گذاری داده های حسگر می تواند با حملات نشت داده مورد سوء استفاده قرار گیرد ، بنابراین حریم خصوصی می تواند به خطر بیفتد. برای حل این مشکل ، در نظر داریم که چارچوبی برای اشتراک گذاری خصوصی داده های حسگر روی دستگاه که از طریق آن برنامه خصوصی سازی می شود؛ را طراحی و پیاده سازی کنیم.

۵-۳-۲ مشکلات موجود در ساختار پایان‌نامه مورد بررسی

- مشکل که مربوط به نگارش پایان‌نامه است؛ استفاده از شیوه ارجاع به منابع می‌باشد. به نظر بنده، شیوه ارجاع بهتر است مطابق راهنماری نگارش پایان‌نامه معاونت آموزشی و تحصیلات تکمیلی دانشگاه پیام‌نور باشد. شیوه ارجاع در این پایان‌نامه به صورت لینک به منابع بود و شخصا کار مشکلی در درک مفاهیم آن داشتم.
- همچنین در پایان‌نامه مورد بررسی، قسمتی تحت عنوان کلمات اختصاری و معرفی آن‌ها وجود نداشت که با تحقیق و پژوهش مجموعه کلمات اختصاری را در قسمت ویژه‌ای گزارش درج کردم.

۵-٤ ارائه ایده برای پایان‌نامه‌های جدید تکمیلی

- نوآوری‌هایی در زمینه توسعه نرم افزار و فضای مهندسی برای درک و مدیریت SPI با استفاده از رویکردها
- حفاظت بهتر از SPI
- ارتباطات مبتنی بر پیام ایمن
- ساخت برنامه امنیتی Open Source خصوصی سازی اشتراک گذاری داده‌های دستگاه ها

۵-۵ جمع‌بندی و نتیجه‌گیری

نشت داده ها و حملات تغییر اطلاعات، انگیزه تلاش های تحقیقاتی با هدف ایجاد مکانیسم هایی برای حفاظت از منطق حساس تجارت ، الگوریتم ها و داده ها است .اگرچه حفاظت SPI از منظر امنیت به طور گسترده مورد مطالعه قرار گرفته است ، این پایان نامه تحقیقات این مشکل را از زاویه مهندسی نرم افزار ، با هدف ارائه فناوری های جدید نرم افزاری که می تواند توسعه دهندگان را بهتر پشتیبانی کند. این پایان نامه تحقیقات در زمینه مهندسی نرم افزار برای درک و مدیریت SPI نوآوری را به طور مشخص، بیان می کند.

(۱) تجزیه و تحلیل برنامه و پشتیبانی برنامه نویسی برای نتیجه گیری را طراحی و توسعه می دهیم. معناشناسی کاربرد سازه های برنامه ، با هدف کمک به توسعه دهندگان برای درک SPI.

(۲) ابزارهای برنامه نویسی قدرتمندی را ارائه می دهند که کد را به طور خودکار ، تغییر شکل می دهند. با هدف کمک به توسعه دهندگان به طور موثر SPI را از بقیه کد جدا می کنیم.

(۳) ارائه مکانیسم برنامه نویسی برای توزیع محیط های اجرایی مدیریت شده که SPI را با هدف فعال کردن اجزا برای تبادل ایمن و مطمئن SPI پنهان می کند .

هدف اصلی این تحقیق این است که به توسعه نرم افزار و تجزیه و تحلیل ، در نتیجه ایجاد امنیت ، قابل درک و کارآمد پایه ای برای محافظت از SPI کمک کند.

مراجع

- جعفرنژادقمی،عین‌اله.عامل محرابی،ابراهیم(مترجمان). (۱۳۹۶). «مهندسی نرم‌افزار جلد ۲» انتشارات دانش‌نگار
- Yue Chen, Mustakimur Khandaker, and Zhi Wang. Pinpointing vulnerabilities. In Proceedings of the ۲۰۱۷ ACM on Asia conference on computer and communications security, pages ۳۳٤–۳٤٥, ۲۰۱۷.
- Victor Costan and Srinivas Devadas. Intel SGX explained. IACR Cryptology ePrint Archive, ۲۰۱٦(۰۸٦):۱–۱۱۸, ۲۰۱٦.
- GlobalPlatform. GlobalPlatform, TEE system architecture, technical report, ۲۰۱۱.https://www.globalplatform.org/specificationsdevice.asp
- V. S. V. Hema and R. Kesavan, "Ecc based secure sharing of healthcare data in the health cloud environment," Wireless Personal Communications, vol. ۱۰۸, no. ۲, pp. ۱۰۲۱–۱۰۳٥, ۲۰۱۹.
- Cay S Horstmann. Scala for the Impatient. Pearson Education, ۲۰۱۲. Li Gong and Gary Ellison. Inside Java (TM) ۲ Platform Security: Architecture, API *Design, and Implementation*. Pearson Education, ۲۰۰۳.
- Jianjun Huang, Zhichun Li, Xusheng Xiao, Zhenyu Wu, Kangjie Lu, Xiangyu Zhang, and Guofei Jiang. {SUPOR}: Precise and scalable sensitive user input detection for Android apps. In ۲٤th USENIX Security Symposium (USENIX Security ۱٥), pages ۹۷۷–۹۹۲, ۲۰۱٥.
- M. M. Kiah, M. S. Nabi, B. Zaidan, and A. Zaidan, "An enhanced security solution for electronic medical records based on aes hybrid technique with soap/xml and sha-۱," Journal of medical systems, vol. ۳۷, no. ٥, p. ۹۹۷۱, ۲۰۱۳.
- Shen Liu, Gang Tan, and Trent Jaeger. Ptrsplit: Supporting general pointers in automatic program partitioning. In Proceedings of the ۲۰۱۷ ACM SIGSAC Conference on Computer and Communications Security, pages ۲۳٥۹–۲۳۷۱. ACM, ۲۰۱۷.
- Chung Laung Liu and James W Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. Journal of the ACM (JACM), ۲۰(۱):٤٦–٦۱, ۱۹۷۳.
- S. A. Parah, A. Bashir, M. Manzoor, A. Gulzar, M. Firdous, N. A. Loan, and J. A. Sheikh, "Secure and reversible data hiding scheme for healthcare system using magic rectangle and a new interpolation technique," in Healthcare Data Analytics and Management. Elsevier, ۲۰۱۹, pp. ۲٦۷–۳۰۹
- SANS. Glossary of security terms, ۲۰۱۹. https://www.sans.org/security-
- resources/glossary-of-terms/.
- Paul M Schwartz and Daniel J Solove. The pii problem: Privacy and a new concept of personally identifiable information. NYUL rev., ۸٦:۱۸۱٤, ۲۰۱۱.

- Alexander Senier, Martin Beck, and Thorsten Strufe. Prettycat: Adaptive guaranteecontrolled software partitioning of security protocols. arXiv preprint arXiv:١٧٠٦,٠٤٧٥٩,
- K. Sudheep and S. Joseph, "Review on securing medical big data in healthcare cloud," in *2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS)*. IEEE, ٢٠١٩, pp. ٢١٢–٢١٥
- R. Sulaiman, D. Sharma, W. Ma, and D. Tran, "A new security model using multilayer approach for e-health services," Journal of Computer Science, vol. ٧, no. ١١, pp. ١٦٩١–١٧٠٣, ٢٠١١.
- Eli Tilevich and Yannis Smaragdakis. J-orchestra: Automatic Java Application Partitioning. In European conference on object-oriented programming, pages ١٧٨–٢٠٤. Springer, ٢٠٠٢.
- M. Wazid, A. K. Das, R. Hussain, G. Succi, and J. J. Rodrigues, "Authentication in cloud-driven iot-based big data environment: Survey and outlook," Journal of Systems Architecture, vol. ٩٧, pp. ١٨٥-١٩٦, ٢٠١٩.
- L. Zhou, V. Varadharajan, and K. Gopinath, "A secure role-based cloud storage system for encrypted patient-centric health records," *The Computer Journal*, vol. ٥٩, no. ١١, pp. ١٥٩٣–١٦١١, ٢٠١٦.

واژه‌نامه

واژه‌نامه فارسی به انگلیسی

تحلیل برنامه	Program Analysis
توسعه‌دهنده	Developer
شی‌گرا	Object-Oriented
فهم برنامه	Program Comprehension
محدودیت‌ها	Constraints
هزینه	Cost
محیط اجرای مورد اعتماد	Trusted Execution Environment
منطقی	Logic
مهندسی نرم‌افزار	Software Engineering
میان‌افزار	Middleware

واژه‌نامه انگلیسی به فارسی

Constraints	محدودیت‌ها
Cost	هزینه
Developer	توسعه‌دهنده
Logic	منطقی
Middleware	میان‌افزار
Object-Oriented	شی‌گرا
Program Analysis	تحلیل برنامه
Program Comprehension	فهم برنامه
Software Engineering	مهندسی نرم‌افزار
Trusted Execution Environment	محیط اجرای مورد اعتماد

Abstract

Exfiltrating or tampering with certain business logic, algorithms, and data can harm the security and privacy of both organizations and end users. Collectively referred to as sensitive program information (SPI), these building blocks are part and parcel of modern software systems in domains ranging from enterprise applications to cyberphysical setups. Hence, protecting SPI has become one of the most salient challenges of modern software development.
However, several fundamental obstacles stand on the way of effective SPI protection: (١) understanding and locating the SPI for any realistically sized codebase by hand is hard; (٢) manually isolating SPI to protect it is burdensome and error-prone; (٣) if SPI is passed across distributed components within and across devices, it becomes vulnerable to security and privacy attacks. To address these problems, this dissertation research innovates in the realm of automated program analysis, code transformation, and novel programming abstractions to improve the state of the art in SPI protection. Specifically, this dissertation comprises three interrelated research thrusts that: (١) design and develop program analysis and programming support for inferring the usage semantics of program constructs, with thegoal of helping developers understand and identify SPI; (٢) provide powerful programming abstractions and tools that transform code automatically, with the goal of helping developers effectively isolate SPI from the rest of the codebase; (٣) provide programming mechanism for distributed managed execution environments that hides SPI, with the goal of enabling components to exchange SPI safely. (liu,٢٠٢١)

Keywords
Software Engineering, Program Analysis, Program Comprehension, Trusted Execution Environment, Middleware

Payam Noor University

Department of Computer Engineering and Information Technology

Seminar Report (M.Sc)

Title:

Methodologies for Managing Sensitive Program Information (Review)

Supervisor:

Dr. Ali Razavi

By:

Somayeh Karbasy

September ۲۰۲۱