مدیریت شرایط نرم افزار امنیتی به عنوان یک سرویس محاسبات (رایانش) ابری در حال ظهور

چکیده

کاربردهای در حال ظهور ابر به سرعت در حال رشد هستند و نیاز به شناسایی و شرایط مدیریت خدمات نیز در حال حاضر بسیار مهم و حیاتی است. سیستم های مهندسی نرم افزار و اطلاعات, تکنیک ها، روش ها و فن آوری های را در طول دو دهه برای کمک به کسب شرایط سرویس ابری، طراحی، توسعه، و تست ایجاد کرده اند. با این حال، با توجه به عدم شناخت ما از آسیب پذیری های امنیتی نرم افزار که باید شناسایی شوند و در طول فاز مهندسی شرایط مدیریت شوند، ما در استفاده از مهندسی نرم افزار، مدیریت اطلاعات و اصول مدیریت شرایط که برای حداقل 25 سال گذشته در حین توسعه سیستم های نرم افزاری امن ایجاد شده اند، بسیار موفق بوده ایم. بنابراین، امنیت نرم افزار را نمی توان درست بعد از ساخت و تحویل یک سیستم به مشتریان اضافه نمود, همانطور که در کاربردهای نرم افزار امروزی دیده می شود. در این مقاله, روش های مختصر، تکنیک ها، و بهترین شرایط عمل مهندسی و مدیریت به عنوان یک سرویس ابری در حال ظهور (SSREMaaES) و نیز دستورالعمل هایی در مورد امنیت نرم افزار به عنوان یک سرویس ارائه شده است. این مقاله همچنین یک مدل یکپارچه امن SDLC (IS-SDLC) را مورد بحث قرار می دهد که به پزشکان، محققان، زبان آموزان و آموزگاران سود خواهد رساند. در این مقاله رویکرد ما برای خدمات سیستم ابر AMAZON EC2 بزرگ نشان داده شده است.

کلید واژه ها :خدمات ابر در حال ظهور، امنیت نرم افزار، مهندسی شرایط امنیتی نرم افزار، توسعه نرم افزاری امن، روش مربع، BSI، نقطه تماس، SDL، مهندسی و مدیریت شرایط به عنوان یک سرویس در حال ظهور (SSREMaaES)

1. مقدمه

شکی نیست که محاسبات ابری, زندگی انسانی، ارتباطات، اقتصاد دیجیتال، اجتماعی و سرگرمی را متحول کرده است. در عین حال, خواسته ها برای کاربردهای اینترنت-محور به سرعت در حال رشد است. تقریبا همه کسب و کارها، کاربردها، دستگاه های سرگرمی، دستگاه های تلفن همراه، روبات ها، سیستم های مقیاس بزرگ (هواپیماها، سیستم های کنترل ماموریت)، سیستم های ایمنی-بحرانی، سیستم های پزشکی، کاربردهای اینترنت دستگاه ها, به دلایل مختلف از جمله ارتقاء آنلاین، برنامه های توزیع شده، پروژه های تیمی، و اتصال به سرور اینترنت-محور هستند. بنابراین، همیشه تقاضای در حال رشد برای کاربردهای امن و اعتماد وجود دارد. حملات سایبری به طور مداوم از هرزنامه ها، فیشینگ، سرقت هویت، و دیگر موارد در حملات مقیاس بسیار بزرگتر مانند پول شویی و تروریسم سایبری در حال افزایش است. یک امکان واقعی وجود دارد که یک حمله شبکه می تواند سیستم های فرمان را غیر فعال کند، شبکه های توان را تضعیف کند، سیل گیرهای سد را باز کند، ارتباطات و سیستم های حمل و نقل را فلج نماید، تشنجات جمعی را ایجاد نماید: که هر یک یا همه آنها می توانند پیش نیاز ابتدایی حمله تروریستی و یا نظامی باشند. اینها برخی از تهدیدات هستند, زیرا برای ارتباطات و مدیریت, ما (شخصی، دولت, سازمان ها، شرکت ها، و کسب و کار) بیشتر به رایانه ها و تلفن همراه وابسته هستیم.

خدمات ابر در حال ظهور در حال افزایش هستند از جمله eHealthCloud، eHealthCloud, E-Learning, E-Manufacturing. Kostoska، Gusev، و Ristov (2014) یک پلت فرم (سیستم عامل) قابلیت حمل ابر جدید (PaaS) را به عنوان یک سرویس توصیف نمودند که می تواند یک پلت فرم ابر را به پلت فرم دیگر بپذیرند و تبادل نمایند و به طور کاملاً خودکار نصب شوند. Han و همکاران, تثبیت VM آگاه از-انرژی را بر اساس الگوریتم

آگاه از استفاده-باقیمانده (RUA) پیشنهاد کرده اند. Xu (2013) یک سیستم تولید ابر سازگار (ICMS) را پیشنهاد داده است که منابع محصور را در یک سرویس ابری برای تولید به اشتراک می گذارد که در آن قابلیت های ساخت و فرصت های کسب و کار ادغام می شوند و در یک حوزه منابع بزرگتر محصور پخش می شوند. Srivastava و همکاران (2015) توضیح می دهند که چگونه ابتکارات سلامت الکترونیک و فن آوری, توسط اتصال و مشاوره تخصصی در سراسر جهان برای جراحی های بسیار پیچیده، خدمات استفاده از ابر، خدمات اینترنت اشیا برای جمع آوری و نظارت بر داده، و ادغام های مختلف فن آوری, به دستیابی به خدمات درمانی با یک هزینه بسیار کم کمک می کنند. هدف این مقاله, ارائه مهندسی امنیت نرم افزار به عنوان یک سرویس در حال ظهور که هدف آن, پیشنهاد ابزارها و تکنیک ها در مورد شرایط امنیتی، طراحی برای امنیت نرم افزار، توسعه نرم افزار امن، تجزیه و تحلیل آسیب پذیری، تست امنیت، و معیارهای امنیتی است.

فروشندگان ابر شامل کسب و کارهای شناخته شده مانند AMAZON EC2، IBM, HP. Google, Microsoft Azure می شود. علاوه بر این، کسب و کارهای ابر جدید در بازار مانند 2nd Watch که با AWS (Amazon EC2 Web Service) همکاری نموده است, مهاجرت ابر، مدیریت حجم کار را ارائه می دهد و دارای 75000 نمونه از صدها نفر از مشتریان تحت مدیریت آن است. به طور مشابه، مدیریت خودکار BetterCloud و امنیت داده ها برای سیستم عامل های دفتر ابر، از جمله Google Apps و Microsoft Office 365 ,Whiting) 365(2015), Hsu and Cheng (2015) یک عامل معنایی به عنوان یک سرویس (SAaaS) را توصیف می کند که دانش را از اطلاعات معنایی جمع آوری و کشف می کند و با خدمات اصلی ابر کار می کند: SaaS ، PaaS و IaaS. همه فروشندگان ابر با نیاز قوی به خدمات ابر امن و مهاجرت کسب و کار موافق هستند.

هدف این مقاله, مطرح نمودن اهمیت توسعه خدمات ابری امن با استفاده از یک رویکرد منظم شناخته شده به عنوان مهندسی امنیت نرم افزار است که به عنوان توسعه نرم افزار امن نیز شناخته می شود. به طور خاص، این مقاله, روش ها و تکنیک های کلیدی در مورد الزامات امنیتی نرم افزار را شناسایی می کند زیرا قلب در حال توسعه خدمات ابری امن است. این مقاله, بهترین رهنمودهای عمل واضح در مورد امنیت نرم افزار را مورد بحث قرار می دهد و مدل

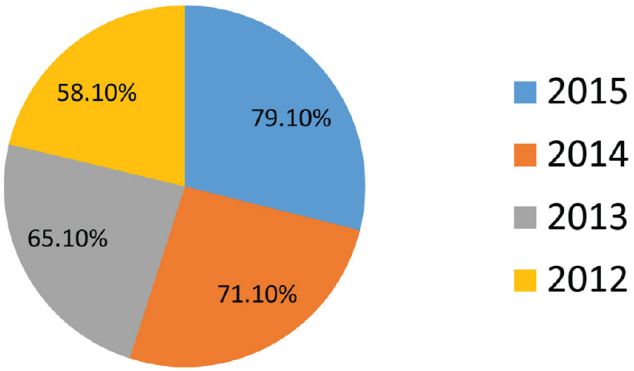
Integrated-Secure SDLC (IS-SDLC) را مورد بحث قرار می دهد ما بر مشکلات فعلی در شناسایی و نمایش بصری، فرایند امنیتی که از شرایط امنیتی برگرفته شده است غلبه می کند. علاوه بر این، همه ما از فن آوری های جدید بر اساس محاسبات خدمات، مانند برنامه های تلفن همراه، ذخیره سازی های ابر، شبکه های رسانه های اجتماعی، و خدمات ابر مانند نرم افزار به عنوان خدمات (SaaS), سیستم عامل به عنوان یک سرویس (PaaS) و زیرساخت به عنوان سرویس (IaaS) استفاده می کنیم. نیاز به محاسبات ابری با کارایی بالا و صحت و دقت داده های بزرگ, نیاز به شناسایی کامل الزامات خدمات ابر و مشخصات آن را تقویت می کند. علاوه بر این، امنیت، حریم خصوصی و اعتماد, کلیدهای موفقیت محاسبات ابری هستند که باید شناسایی و به عنوان بخشی از فرایند مدیریت الزامات مشخص شوند. تقسیم بندی این مقاله عبارتست از: بخش 1, مقدمه کار ما، بخش 2, منطق حوزه موضوع، بخش 3, معرفی مهندسی و مدیریت شرایط امنیتی نرم افزاری، و بخش 4, معرفی یک رویکرد یکپارچه برای SDLC ، و بخش نهایی, یک مطالعه موردی در مقیاس بزرگ با استفاده از Microsoft Security Development Lifecycle در مورد تکنیک های مدلسازی تهدید برای خدمات ابر AMAZON EC2 است.

2. چرا مهندسی امنیت نرم افزار؟

مهندسی نرم افزار (SE) روش ها، رویکردها و فن آوری هایی را در بیش از دو دهه ایجاد کرده است. SE نیز تکنیک ها و ابزارهای غنی در مورد شرایط نرم افزار مدل سازی، طراحی، توسعه، تست، مدیریت پیکربندی، و معیارهای نرم افزاری را فراهم می کند. با این حال، مسائل امنیتی, ویژگی های مستقیم نرم افزار های مختلف از قبیل کاربردها، رابط کاربری، شبکه، توزیع، انتقال داده های فشرده، و ابزارهای ارتباطی، و غیره هستند. کاربردهای کنونی در حال توسعه و تحویل هستند. توسعه دهندگان تجاری اولیه با استفاده از فایروال (در سطح برنامه)، و تست نفوذ و مدیریت گروهی از عهده مشکلات امنیتی برآمده اند.

ما همچنین با جنگ افزار اطلاعات، جرایم اینترنتی، سایبر تروریسم در حال رشد سریع، سرقت هویت، هرزنامه و دیگر تهدید های مختلف مواجه هستیم. بنابراین، درک نگرانی های امنیتی با شروع از الزامات، طراحی، و تست برای

کمک به ما در ساخت امنیت (BSI) مهم است. McGraw (2006) می گوید: یک جنبه مرکزی و حیاتی از مشکل امنیت کامپیوتر, مشکل نرم افزار است. در این مقاله, مهندسی امنیتی نرم افزار را به عنوان یک رشته تعریف می کنیم که ضبط و مدل سازی برای امنیت، طراحی برای امنیت، اتخاذ بهترین شیوه ها، تست امنیت، مدیریت، و آموزش های امنیتی نرم افزار برای تمام سهامداران را در نظر می گیرد.



شکل. 1. هزینه سالانه برای امنیت اطلاعات.

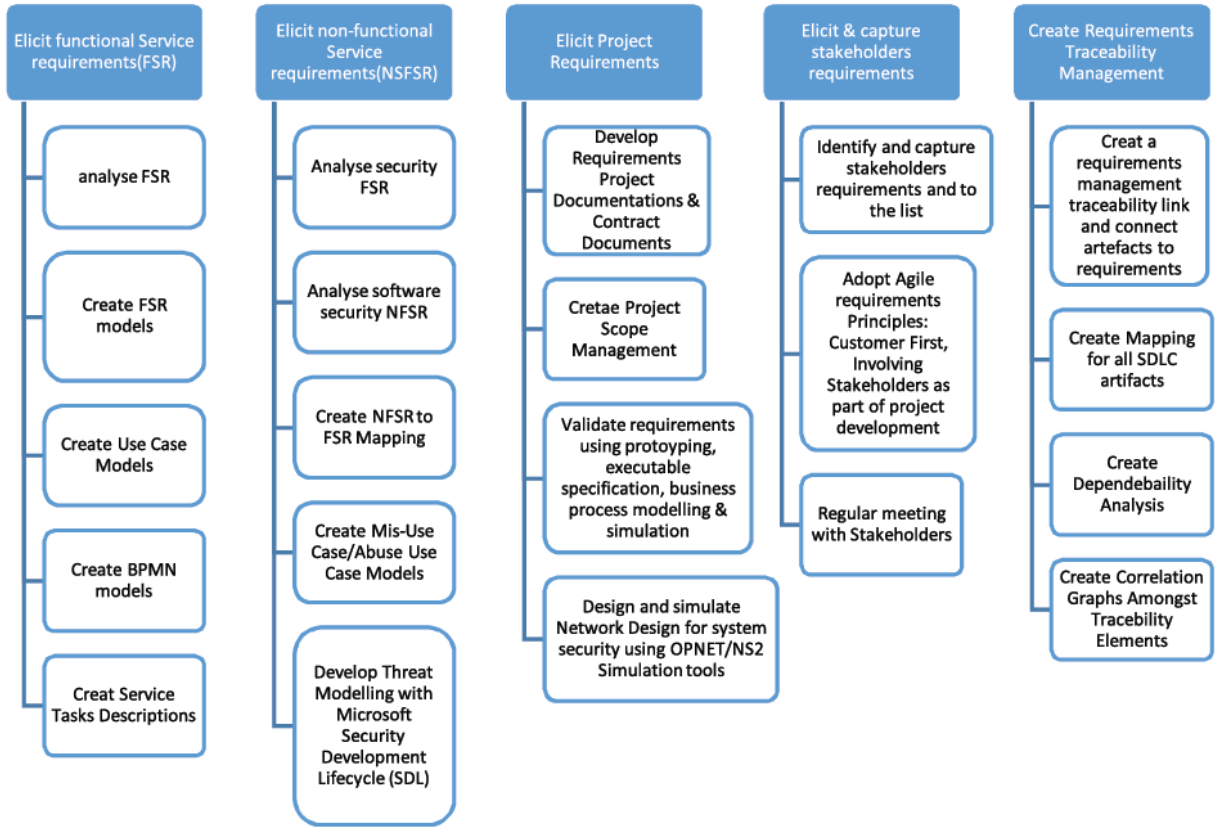
مهندسی نرم افزار به خوبی چارچوب روش ها، تکنیک ها و فرآیندهای غنی را ایجاد نموده است و محصولات و سازمانهای مقیاس کوچک تا بزرگ (CMM، CMMI، SPICE، و غیره)، و فن آوری مرتبط از جمله مدل سازی (UML)، ابزارهای CASE، و ابزار CAST، و دیگران از آن استفاده نمایند. مهندسی نرم افزار نیز به خوبی مدل ها و روش های کیفی، مدل ها و روش های استفاده مجدد، مدل های و روش های قابلیت اطمینان، و لیست های متعددی از روش های دیگر را ایجاد نموده است. مهندسی نرم افزار به مدت طولانی به عنوان بخشی از ویژگی های کیفیت (کیفیت، آزمون پذیری، نگهداری، امنیت، قابلیت اطمینان، قابلیت استفاده مجدد) کمک کرده است. این ویژگی ها را نمی توان فقط به سیستم اضافه نمود, زیرا آنها باید در بخش اولیه مراحل چرخه عمر (یک چرخه عمر توسعه نرم افزار معمولی شامل شروع از مهندسی شرایط (RE)، مشخصات نرم افزار، نرم افزار و طراحی معماری، توسعه نرم افزار (برنامه نویسی)، تست نرم افزار و تعمیر و نگهداری ساخته شوند. امنیت از زمان توسعه کاربردهای مبتنی بر آنلاین, به یک ویژگی بسیار مهم تبدیل شده است. مدیریت پروژه نرم افزار به خوبی تکنیک ها و وسعت

دانش از جمله توسعه جهانی (به علت بروز انقلاب اینترنت و مهارت های افراد در سراسر جهان)، تکنیک های کاهش هزینه، تکنیک های مدیریت ریسک، و دیگران را ایجاد کرده است. امروزه، بسیاری از سیستم های و دستگاه های فعلی, مبتنی بر وب هستند و از این رو امنیت باید درست از همان ابتدا به دست آید: باید شناسایی، ضبط، طراحی، توسعه یافته و تست شود. Ashford (2009) گزارش نموده است که 75 درصد خرج کسب و کار UK در بودجه انکشافی نرم افزار در مورد رفع نقص های امنیتی پس از تحویل کالاست. این یک هزینه بزرگ است و عدم اعتماد را در میان مشتریان نیز ایجاد می کند. شکل. 1 چارت برای خرج سالانه در مورد امنیت IT را نشان می دهد که افزایش بیش از 10٪ در هر سال را (Gartner، 2016) نشان می دهد.

با توجه به اهمیت توسعه نرم افزاری امنیت محور ، رشته های مهندسی امنیتی نرم افزار و توسعه نرم افزار امن در سال های اخیر (Ramachandran، 2012) پدید آمده است. مهندسی امنیت نرم افزار با ابزارها و تکنیک های غنی مدل سازی در شرایط امنیتی نرم افزار مانند سوء استفاده و موارد سوء استفاده، مدل سازی تهدید، طراحی برای امنیت، تحلیل آسیب پذیری، برنامه نویسی امن، تست، و معیارها سرو کار دارد. علاوه بر این، مسائل اخیر با استراتژی های امنیت ابر، تکنیک ها توسط Chang and Ramachandran (2015، 2016) تشریح می شوند.

Allen et al. (2008) اظهار داشتند که یکی از اهداف اصلی مهندسی نرم افزار امنیتی, پرداختن به بهترین شیوه های مسئله امنیت نرم افزار، فرآیند، فنون و ابزارها در هر فاز و فعالیت های هر چرخه عمر توسعه نرم افزار استاندارد (SDLC) است. هدف اصلی از ساخت نرم افزار های امن اینست که عاری از نقص شود و با موارد زیر بهتر ساخته شود:

- ادامه عملیات نرمال در هر مورد حملات و تحمل هر گونه شکست
- محدود کردن خسارات در حال ظهور به عنوان یک نتیجه از هر گونه حمله تحریک شده
- ساخت اعتماد و تاب آوری در (BTRI)
- حفاظت از داده ها و دارایی



شکل. 2. وظایف و فعالیت های مدیریت الزامات

به عبارت دیگر، نرم افزار امن به طور معمول باید در صورت هر گونه حمله به کار خود ادامه دهد. علاوه بر این، این مورد شامل فرآیند استخراج شرایط امنیتی از الزامات کلی سیستم (شامل سخت افزار، نرم افزار، کسب و کار، بازاریابی، و شرایط زیست محیطی) و در نتیجه شرایط امنیتی اصلاح شده و استخراج شده و امنیت نرم افزار از شرایط نرم افزار و کسب و کار می شود. سپس شرایط امنیتی نرم افزاری تصفیه شده را می توان تعبیه نمود و در سراسر مراحل چرخه حیات توسعه نرم افزار (SDLC) از جمله شرایط، طراحی، توسعه، و تست پیش بینی کرد. این مورد به خوبی در نوشته های مرتبط با ناامنی توضیح داده نشده است. این یک تعریف روشن از استخراج شرایط امنیتی نرم افزار را فراهم می کند.

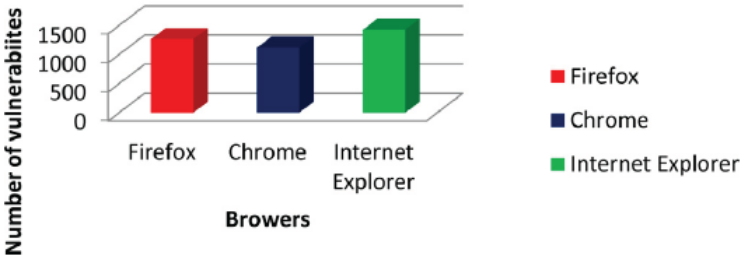
نمونه هایی از حملات سایبری اخیر شامل حملات به انبار Carphone که منجر به از دست دادن اطلاعات شخصی تا 2.4 میلیون مشتریان انبار Carphone می شود در حملات شبکه دیده می شود. اخبار مشابه در اخبارهای اخیر

مانند Talktalk، و غیره به طور بسیار مکرر دیده شده است. بنابراین، نتیجه گیری می توان کرد که آنچه که ما کشف کرده ایم, تنها تعداد انگشت شماری از حملات سایبری و آسیب پذیری نرم افزار است.

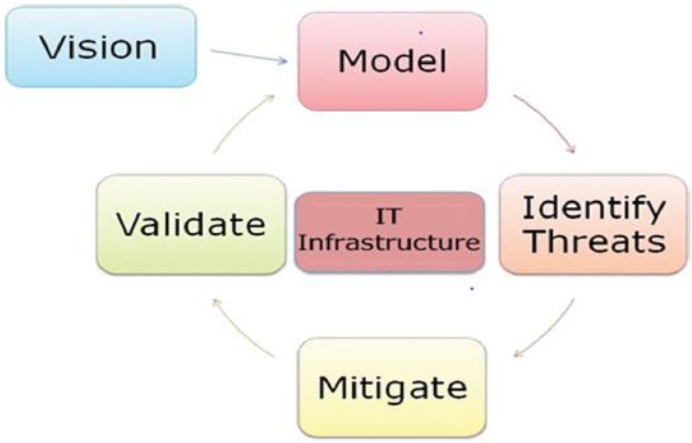
3. مهندسی و مدیریت شرایط امنیتی نرم افزار به عنوان یک سرویس ابری در حال ظهور (SSREMaaES)

در عصر ابر رایانه، در حال ظهور به معنای تکنیک ها و روش های نوآورانه مورد استفاده برای حوزه های سنتی (اقتصاد، بهداشت و درمان، آموزش و پرورش، مهندسی نرم افزار، امنیت و غیره) و حوزه های جدید (برنامه های تلفن همراه، شبکه های اجتماعی، تجسم آب و هوا، بزرگ پردازش داده ها و غیره) است. تحقیقات قبلی ما در این زمینه شامل چند برنامه ابر نوآورانه به عنوان یک سرویس (Chang, Ramachandran, Wills, Walters, Kantere, ، Kantere، و Li، 2015) می شود. با این حال، تا کنون، تحقیقات زیادی در حوزه شیوه های مهندسی نرم افزار و مهندسی نرم افزار امنیتی به عنوان یک سرویس ارائه نشده است. بنابراین، هدف این مقاله, پیشنهاد مهندسی و مدیریت شرایط امنیتی نرم افزار ها به عنوان یک سرویس در حال ظهور (SSREMaaES) است. الزامات, نقطه شروع مسئول هر سیستم، مسائل حقوقی و قراردادی، حکومتداری است و یک چشم انداز کامل کاربردی از سیستم در حال توسعه را ارائه می دهد. مهندسی شرایط در جایگاه خود یک رشته است که فرایند، ابزارها، تکنیک ها، مدل سازی، برآورد هزینه، برنامه ریزی پروژه، و موافقت نامه های قراردادی را فراهم می کند. میزان زیادی از روش های مهندسی شرایط، تکنیک ها، و رهنمودهای بهترین عمل و ابزارها وجود دارد (Jacobson، 1992؛ Kotonya و Sommerville، 1992؛ Lamsweerde، 2009؛ Sommerville و Sawyer، 1998). با این حال، با توجه به ماهیت افزایش تقاضا برای کاربردهای امنیت-محور، تکنیک های فعلی در مورد داشتن شرایط موثر مربوط به امنیت, ناکافی می باشند. Firesmith (2007، 2003) گزارش می دهد که الزامات ضعیف, دلایل اصلی برای هزینه و اجراهای بیش از حد زمانبندی، قابلیت ضعیف و سیستم های تحویل داده شده هستند که هرگز استفاده نمی شوند. الزامات به دو بخش عمده طبقه بندی می شوند از جمله شرایط عملکردی که با قابلیت سیستم و

نیازهای غیر کارکردی سرو کار دارند که مرتبط با محدودیت ها، کیفیت، داده ها، استانداردها، مقررات، رابط، کارایی، قابلیت اطمینان، و سایر الزامات پیاده سازی است. مطالعات (Jacobson 1992؛ Kotonya و Sommerville، 1992؛ Lamsweerde، 2009؛ Sommerville و Sawyer، 1998) نشان داده اند که هزینه نقایص مهندسی شرایط, 10-200 برابر اصلاح سیستم پس از پیاده سازی است. بنابراین، دریافت الزامات درست، دقیق و بدون ابهام, برترین شیوه کار است



شکل. 3. آسیب پذیری شناخته شده در مرورگرها.



شکل. 4. فرایند مدل سازی تهدید Microsoft.

ضبط شرایط امنیتی کسب و کار, یک تلاش همکارانه است که شامل سهامداران مانند بسیاری از تحلیلگران کسب و کار، مهندس الزامات نرم افزار ، معمار نرم افزار ، و مدیران آزمون می شود. شرایط امنیتی باید مجموعه ای روشن از نیازهای خاص امنیت و رفتار مورد انتظار یک سیستم را فراهم کنند. هدف اصلی, حفاظت از دارایی های سیستم (داده ها و فایل) و دسترسی غیر مجاز به سیستم از حملات عمدی تا سیستم های نرم افزاری کاربردی و سایر اشکال حملات امنیتی مبتنی بر اینترنت مانند هرزنامه، محرومیت از خدمات، سرقت هویت، ویروس ها، و بسیاری از دیگر

اشکال حملات عمدی است که هر روز پدیدار می شوند. امنیت, همچنان یک مشکل نرم افزار است, همانطور که تعدادی از تهدیدات و آسیب پذیری ها برایCERT-SEI و CERT-UK (تیم واکنش اضطراری رایانه ای) از 2493٪ افزایش بین سال های 1997 (311 مورد گزارش شده)، 2006 (8064 مورد گزارش شده)، و در 30 آوریل 2015 (3192 مورد گزارش شده) گزارش شده اند.

در RE سنتی، شرایط امنیتی به عنوان بخشی از خواص غیر تابع و یک جنبه از جمله راهکارهای پیاده سازی مانند حفاظت از رمز عبور، ورود و خروج، فایروال ها، تشخیص ویروس، محرومیت از خدمات حملات، و غیره در نظر گرفته می شوند. بنابراین نیازهای امنیتی باید به عنوان مجموعه ای بسیار خاص از شرایط مورد نیاز برای هر شرایط عملکردی که شناسایی می شود در نظر گرفته شوند و باید در طول چرخه زندگی اعمال شوند, به طوری که می توانیم به امنیت (BSI) داخلی دست یابیم. علاوه بر این، روش های RE فعلی عمدتاً به عنوان چیزی در نظر گرفته می شوند که سیستم باید انجام دهد، نه آنچه که سیستم نباید انجام دهد. این موضوع کلیدی است که در هنگام انتخاب روش های RE برای امنیت نرم افزار در نظر گرفته خواهد شد. علاوه بر این، در روش های امنیتی RE نرم افزار، سهامداران بیشتری نسبت به روش های سنتی RE مانند مهندسان اجتماعی، متخصص امنیت، فرایند کسب و کار کارشناسان مدل سازی، متخصصان محاسبات خدمات و کاربران وجود دارند. اغلب مهاجمان به دنبال نقص در سیستم هستند، و نه ویژگی ها و کارکردهای سیستم.

در این بخش به روش های مختلف برای استخراج شرایط مورد نیاز در نرم افزار امنیتی نگاه خواهیم کرد. بهترین شیوه های معمول عبارتند از:

1. استخراج صریح شرایط مورد نیاز برای نرم افزار امنیتی
2. اولویت بندی شرایط امنیتی
3. ارزیابی ریسک برای شرایط امنیتی
4. طراحی و پیاده سازی شرایط امنیتی

قبل از شروع جزئیات روشهای مختلف برای استخراج شرایط امنیتی، ما سه مفهوم مهم پایه را امتحان و درک خواهیم کرد: موارد استفاده، موارد سوء استفاده، و موارد استفاده نامناسب. مورد استفاده, شکل موثر از ارائه الزامات کاربر (کاربری که با سیستم تعامل پیدا می کند به عنوان عامل شناخته شده است) از نظر بصری (؛ Kotonya و Sommerville 1998؛ Jacobson 1992 Lamsweerde، 2009) بوده است. این فرم بصری نشان دهنده یک داستان شرایط (الزامات) کاربر است که به عنوان سناریو نیز شناخته می شود. بنابراین، یک مورد استفاده = سناریو (داستان کاربر) + عاملان (کسانی که با سیستم تعامل پیدا می کنند). ترکیبی از یک سناریو و عاملان به عنوان یک مورد استفاده شناخته شده است. با این حال، مورد استفاده برای نشان دادن جنبه غیرکارکردی سیستم ها مورد بهره برداری قرار نگرفته است. از اینرو، دلیلی برای ظهور روش های جدید مانند موارد سوء استفاده و استفاده نامناسب وجود دارد. مورد سوء استفاده می تواند به ما کمک کند تا شرایط امنیتی بصری را از نقطه نظر مهاجم نشان دهیم (یک مثال می توان تلاس برای ورود به سیستم با استفاده از رمز عبور های مختلف باشد) در حالی که مورد تخلف باید شرایط امنیتی را از جنبه های تخریب بسیار قوی تر از سیستم نشان دهد (یک مثال می تواند تلاش برای از بین بردن با تغییر کلید ثبت نام، پیکربندی و دیگر فایل های DLL باشد). تعدادی از تکنیک های برای پرداختن به RE از دیدگاه یک مهاجم پدید آمده اند:

• الگوهای حمله مشابه با الگوهای طراحی هستند که برای مطالعه حملات از حالت مخرب، Allen و همکاران (2008) و ایجاد امنیت در (BSI) طراحی می شوند.

• موارد سوء استفاده و تخلف, مجموعه ای از موارد استفاده از دیدگاه یک مهاجم هستند، McGraw (2006)

• درختان حمله یک مکانیزم رسمی را برای تجزیه و تحلیل و توصیف شیوه های مختلف را ارائه می دهند که در آن حملات می توانند از دیدگاه یک مهاجم رخ دهد. به سادگی نشان دهنده حملات علیه یک سیستم در یک ساختار درختی، با هدف گره ریشه و روش های مختلف دستیابی به این هدف به صورت گره های برگ، Schneier (1999، 2000) و Ellison و Moore (2003) است

• Microsoft SDL (SDL، 2016 و TAM، 2016) در مورد مدل سازی تهدید را پشتیبانی می کند که مجموعه ای از جنبه های امنیتی را با تعریف مجموعه ای از حملات امنیتی ممکن توصیف می کند. این بخشی جدایی ناپذیر از روش Microsoft SDL، Howard و LEBLANC (2002) است.

• امنیت ساختمان در (BSI) (2013)، فرآیند، اصول طراحی، و تکنیک های ارائه شده توسط Mcgraw (2006) و دیگران است که در حال حاضر رسماً توسط وزارت امنیت خانه-زمین پشتیبانی می شود. برخی از اصول طراحی عبارتند از:

الف) صحت توسط ساخت (CbyC)

ب) تامین امنیت ضعیف ترین لینک

ج) دفاع عمیق

د) کوتاهی در تامین امنیت

ه) حداقل امتیاز

ج) جداسازی امتیاز

g) اقتصاد مکانیزم

h) حداقل مکانیزم مشترک

i) بی میلی به اعتماد

j) عدم این فرض که اسرار شما, امن هستند

K) میانجیگری کامل

L) پذیرش روانی

m) ترویج حریم خصوصی

• روش SEI (موسسه مهندسی نرم افزار) یک روش شناخته شده را به عنوان SQUARE (مهندسی شرایط تضمین کیفیت) شرح داده شده توسط Mead و همکاران SQUARE (2008) را شناسایی کرده است که برای استخراج و اولویت بندی الزامات است و شامل نه مرحله به شرح زیر می شود:

الف) توافق بر سر تعریف

ب) شناسایی اهداف امنیتی

ج) توسعه مصنوعات

د) انجام ارزیابی های ریسک

ه) انتخاب یک روش استخراج

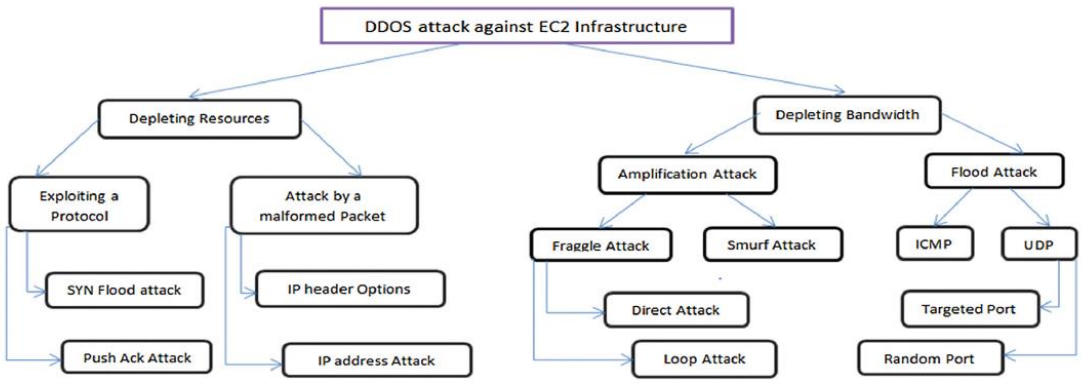
ج) استخراج شرایط امنیتی

g) دسته بندی شرایط امنیتی

H) اولویت بندی شرایط امنیتی

i) بازرسی شرایط امنیتی

j) شناسایی واضح شرایط سیستم کلی نرم افزار و استخراج شرایط امنیتی. تعامل با سهامداران برای روشن نمودن شرایط امنیتی و فن آوری که آنها می خواهند استفاده نمایند، و پیامدهای هزینه.



شکل. 5. حمله DDoS درخت.

• روش OCTAVE توسط Caralli et al. (2007), Alberts and Dorofee و Dorofee (2002) و Woody and Alberts (2007) فعالیت های روشن را در شرایط امنیتی فراهم می کند:

الف) شناسایی دارایی های حیاتی

ب) تعریف اهداف امنیتی

ج) شناسایی تهدیدات

د) تجزیه و تحلیل ریسک

ه) تعریف شرایط امنیتی

• روش های دیگر شامل CLASP (2006) و S-SDLC می شوند و شرح مفصلی توسط Ramachandran، 2012 داده شده اند.

Chen (2004) تفاوت کلیدی بین مهندسی امنیت نرم افزار را از نیرومندی برای مهندسی ایمنی نرم افزار متمایز می کند. مهندسی امنیتی نرم افزار با رویکرد مهندسی برای توسعه نرم افزار با هدف مهندسی و پیاده سازی ویژگی های امنیتی سروکار دارد در حالی که استحکام (نیرومندی) به نرم افزار مهندسی برای سیستم های بحرانی ایمنی می پردازد. بنابراین، ما نیاز به شناسایی، تجزیه و تحلیل، و ترکیب شرایط امنیتی به عنوان بخشی از فرآیند شرایط عملکردی داریم. Belapurkar و همکاران (2009) لیستی از برخی حوزه ها در سطح بالا را برای شرایط عملکردی خاص هر امنیتی به شرح زیر را شناسایی کرده اند:

• شناسایی باید به این مورد بپردازد که چگونه یک سیستم، عاملان / اشخاص (انسان یا سیستم) را که با سیستم ارتباط برقرار می کنند شناسایی می کند

• تایید باید به این مورد بپردازد که چگونه یک سیستم، هویت اشخاص را تایید می کند

• مجوزدهی باید به این مورد بپردازد که چه امتیازاتی باید برای یک نهاد دارای تعامل با یک سیستم تنظیم شود

• انکارناپذیری باید به این مورد بپردازد که چگونه یک سیستم مانع این می شود که نهادها تعاملات خود را با قابلیت سیستم انکار نمایند

• صداقت باید به این مورد بپردازد که چگونه یک سیستم اطلاعات را از هر گونه تغییرات عمدی یا غیر عمدی و دستکاری محافظت نماید.

• حسابرسی باید به این مورد بپردازد که چگونه یک سیستم به حسابرسان اجازه می دهد تا وضعیت کنترل های امنیتی در محل را ببینند

• حریم خصوصی باید به این مورد بپردازد که چگونه یک سیستم مانع از افشای غیر مجاز اطلاعات حساس می شود

• در دسترس بودن باید به این مورد بپردازد که چگونه یک سیستم به خودی خود از اختلالات عمدی برای خدمات محافظت می نماید به طوری که زمانی که کاربران نیاز دارند برایشان دردسترس باشد.

شرایط امنیتی نرم افزار نه تنها مجموعه ای از محدودیت ها روی سیستم های نرم افزاری است، بلکه اداره امور مورد نیاز آنها را برآورده سازد و حفاظت و اعتماد را فراهم می کند. این به این معنی است که ما به تکنیک های جدیدتر مانند الگوهای حمله، سوء استفاده و تخلف نیاز داریم. وظایف و فعالیت های مرتبط با مدیریت مهندسی شرایط برای امنیت نرم افزار در شکل 2 نشان داده شده است.

همانطور که در شکل 2 نشان داده شده است، مدیریت قابلیت ردیابی شرایط (RTM) به عنوان مدیریت ماتریس قابلیت ردیابی شرایط تعریف می شود که در طی فاز توسعه شرایط, ایجاد و اعتباردهی می شود. RTM اغلب با استفاده از یک فایل ساده و یا توسط ابزارهای CASE ایجاد شده است. RTM مرتبط با مصنوعات شرایط در سراسر آثار دیگر مانند طراحی، کد، مورد آزمون برای یک RTM کامل است. موضوع این است که روش های سنتی موجود, امنیت نرم افزاری را که با استفاده از شرایط فعلی شناسایی شده است، روش های مورد بحث در این بخش, (موارد سوء استفاده، مدل سازی تهدید، و غیره) در نظر نگرفته اند. بنابراین شکل 2, اهمیت یکپارچه سازی مصنوعات امنیتی نرم افزار را با سیستم های سنتی RTM برجسته می کند. روش RTM پیشنهادی متشکل از پنج وظایف به عنوان بخشی از مرحله امنیتی نرم افزار RE است:

• استخراج شرایط خدمات عملکردی (FSR) و فعالیت ها, FSR تجزیه و تحلیل ، ایجاد مدل کاربردی، ایجاد مدل های مورد استفاده هستند

• استخراج شرایط غیر کارکردی خدمات (NFSR) و فعالیت های مرتبط, FSR امنیت آنالیز، NFSR امنیتی نرم افزار آنالیز، خلق NFSR برای نگاشت FSR، ایجاد مورد سوء استفاده و مدل های مورد تخلف, توسعه مدلسازی تهدید با چرخه عمر توسعه امنیت Microsoft (SDL)

• استخراج شرایط پروژه و فعالیت های مرتبط و پیوند خورده, توسعه مستندات پروژه شرایط و اسناد قرارداد، ایجاد مدیریت محدوده پروژه، اعتبارسنجی شرایط با استفاده از نمونه اولیه، مشخصات اجرایی، مدل سازی و شبیه سازی فرایند کسب و کار، طراحی و شبیه سازی طراحی شبکه برای امنیت سیستم با استفاده از ابزارهای شبیه سازی OPNET / NS2

• استخراج و ثبت شرایط سهامداران و فعالیت های مرتبط, مبتنی بر Agile هستند: مانند شناسایی و ثبت شرایط سهام دارنده و برای لیست، اتخاذ شرایط اصول چابک: نخستین مشتری، درگیر شدن سهامداران به عنوان بخشی از توسعه پروژه، جلسه گذاشتن منظم با سهامداران

• ایجاد مدیریت قابلیت ردیابی شرایط (RTM) و فعالیت های مرتبط شامل خلق یک مدیریت شرایط، لینک لینک قابلیت ردیابی و اتصال مصنوعات به شرایط، ایجاد نقشه برداری برای همه مصنوعات SDLC، خلق تحلیل قابلیت اعتماد، ایجاد نمودارهای همبستگی در میان عناصر قابلیت ردیابی

این ایده کلیدی, اتخاذ یک رویکرد جامع برای مدیریت مهندسی شرایط در امنیت نرم افزار است که شامل تکنیک ها و شیوه های مدرن می شود. این به ما اجازه خواهد داد تا یک شرایط امنیتی نرم افزار موجز را ایجاد نماییم که با فن آوری های شبیه سازی مانند BPMN (نمادگذاری های مدلسازی فرآیند کسب و کار) اعتباردهی می شوند که اعتباردهی شرایط توسط شبیه سازی فرآیندهای پیشنهادی را در برابر عملکرد مورد انتظار و سطح امنیت که داخلی است (BSI) میسر می سازد.

بنابراین، یکی از اهداف اصلی این مقاله, ارائه SSRE-MaaES به عنوان یک سرویس ابری در حال ظهور است که تمام ویژگی ها، تکنیک ها و ابزارهای در دسترس به عنوان یک سرویس را ارائه خواهد داد. این اجازه خواهد داد تا خدمات توزیع شوند، برای توسعه و مدیریت شرایط توزیع شده توسط سهامداران که در مکان های توزیع شده قرار

دارند دردسترس قرار گیرند و در نتیجه از توسعه نرم افزار جهانی و مدیریت پروژه های نرم افزار در مقیاس بزرگ و محصولاتی که برای امنیت نرم افزار طراحی در دسترس قرار می گیرند حمایت نمایند. این سرویس در حال ظهور جدید, تجزیه و تحلیل خودکار و ارزیابی شرایط امنیتی، مدل سازی تهدید، و درختان حمله را ارائه می دهد.

4. شناسایی آسیب پذیری امنیتی و یکپارچه سازی برای حملات DDoS

این بخش در مورد تعدادی از آسیب پذیری ها بحث می کند و SDLC در مدل سازی تهدید را برای شناسایی آسیب پذیری در مرحله RE نیز ارائه می دهد. این مورد با استفاده از حملات DDoS (انکار توزیع شده سرویس) نشان داده می شود که در سرویس های ابری و سایر بخش های حیاتی کسب و کار مانند انبارCarphone ، Talktalk mobile، و غیره رایج بوده اند. این به دلیل مرورگرهای موجود است که با BSI توسعه یافته اند. بنابراین، ثابت شده است که اضافه کردن امنیت از طریق تکه ها که در شیوه های فعلی دیده می شود, آسیب پذیر به حملات امنیت سایبری می باشد. این مورد در شکل 3 در مورد تجزیه و تحلیل آسیب پذیری برای مرورگرهای مختلف موجود با استفاده از داده های موجود نشان داده شده است.

تجزیه و تحلیل ریسک، بخش مهمی از هر گونه شرایط و فرآیند مدل سازی است. Ashbaugh (2006) می گوید 90 درصد از آسیب پذیری های امنیتی نرم افزار از انواع نقص نرم افزار شناخته شده ایجاد می شوند و از این رو یکی از راههای کاهش این آسیب پذیری، انجام تجزیه و تحلیل خطر برای امنیت نرم افزار در سراسر SDLC سیستماتیک است.

در رویکرد ما، با توجه به روش های RTM ما، ما استفاده از ابزار مدلسازی تهدید Microsoft را پیشنهاد می کنیم که به عنوان SDL Microsoft (چرخه عمر توسعه امنیت) شناخته می شود. این مورد در شکل 4 نشان داده شده است که متشکل از (شرایط امنیتی نرم افزار) چشم انداز منجر به ایجاد مدل های تهدید است و به ما اجازه می دهد تا تهدید های مختلف برای هر شرایط عملکردی را در نظر بگیرید و برای هر تهدید شناسایی شده, تعدادی از کاهش ها را در نظر می گیریم. در طول این مرحله، یک گزارش مدل تهدید با استفاده از ابزار مدل سازی تهدید

Microsoft (TAM، 2016) را تولید خواهد کرد. در مطالعه موردی ما، ما زیرساخت های AMAZON EC2 را استفاده کرده ایم و برای تهدید DDoS بالقوه و کاهش, آنها را تحلیل می نماییم.

گام اول, ساخت یک چشم انداز روشن با برنامه ریزی محل برای هر یک از تجهیزات است. در این مرحله، تولید یک بیانیه که دلیل و نقش هر دارایی انتخاب شده برای به کارگیری را توضیح می دهد، ترجیح می دهد و این عملیات به سازمان در جداسازی بین فعالیت ها کمک خواهد کرد.

پس از داشتن یک چشم انداز روشن، یک نمودار جریان داده ها می تواند به منظور درک جریان داده ها بین دارایی های داخلی و خارجی مدلسازی شود. به طور کلی، چهار عنصر، که می تواند برای مرزهای برچسب زدن استفاده شود:

1. جریان داده ها: برای ردیابی جریان داده ها از منبع به مقصد.

2. نقاط ورود: هر دارایی است که داده را از خارج دریافت می کند، و می تواند یک کاربر، سخت افزار و یا نرم افزار باشد. در این سطح، ما باید فرصت های حملات DDoS بالقوه را که شاید ایجاد شوند حذف نمایند.

3. مرزهای اعتماد: که همه شبکه ها و حوزه های مورد اعتماد را تعیین می کند؛ به عنوان مثال، دستگاه هایی مانند روترها، فایروال و متوازن کننده های بار.

4. دارایی های حفاظت شده: که دارایی های حیاتی را تعریف می کند که باید برای پایگاه داده و یا ذخیره سازی مشتری نمونه محافظت شوند.

از آنجا که حملات DDoS, فقط در دسترس بودن منابع را هدف قرار می دهند؛ شناسایی تهدیدها نباید فقط بر طبقه بندی های حمله DDoS (کاهش منابع و پهنای باند) تمرکز نمایند و بنابراین شدت هر یک از آنها را نشان دهند.

مرحله چهارم پس از شناسایی تهدید, توسعه اقدامات متقابل ممکن برای کاهش همه تهدیدات شناخته شده است. رده های مختلف برای مشخص کردن تمام مکانیسم های کاهش در دسترس هستند. در این مرحله، هر مکانیسم پیشنهاد شده باید عمیقاً از طریق مقابله با حمله از همه ابعاد تحلیل شوند.

آخرین مرحله از فرایند مدل سازی تهدید, اعتباردهی تمام تجزیه و تحلیل و یافته های ما است. علاوه بر این، توسط اندازه گیری تاثیر هر حمله در صورت قصور در کاهش؛ برای مثال، اگر یک حمله DDoS با موفقیت متعادل کننده بار داخلی را هدف قرار دهد؛ عواقبی که ممکن است منابع را تحت تاثیر قرار دهند چیست؟

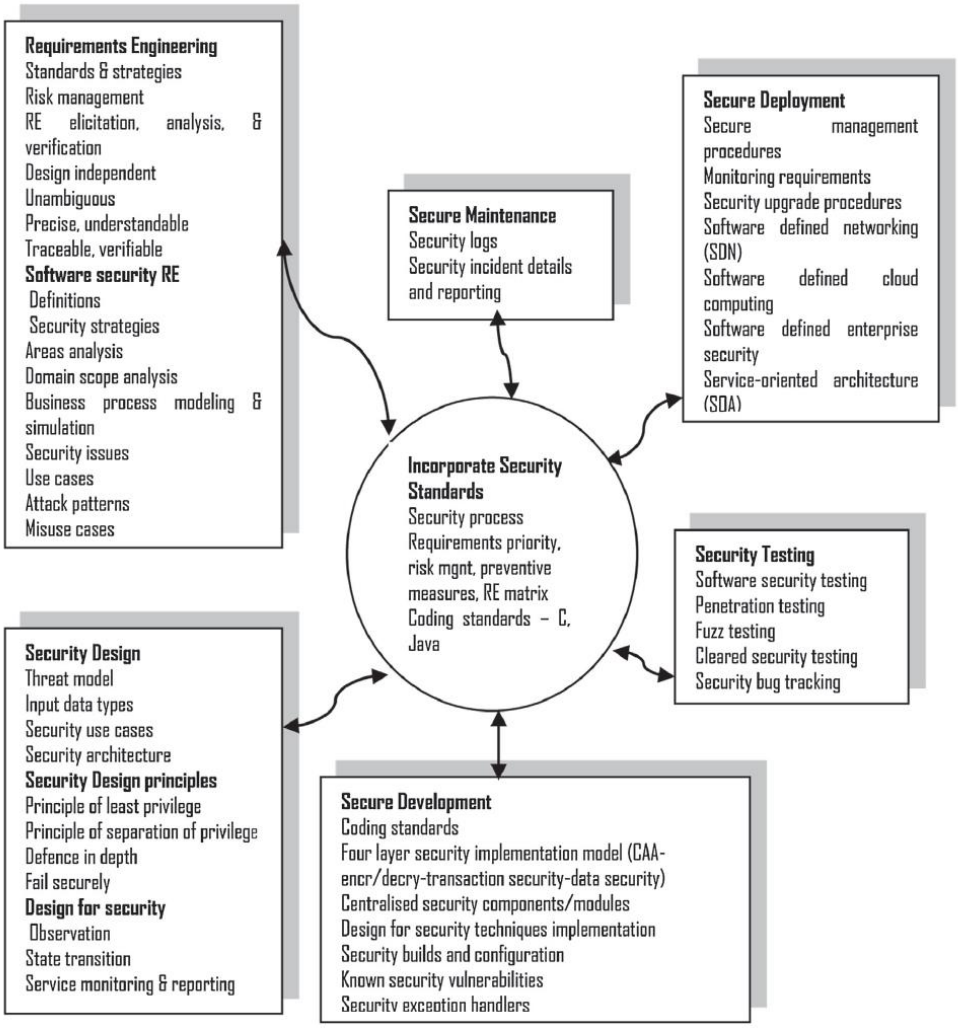
زمانی که آمازون AWS, با معماری داخلی خود برای اهداف امنیتی مواجه نمی شود. ما تصمیم گرفته ایم تا بر برخی از مقالات پژوهشی و وب سایت های آنلاین برای درک بهتر معماری تکیه نماییم، و با توجه به مقاله معماری های مرجع آمازون AWS؛ دستگاه های مختلف در داخل قلمروی مراکز داده های خود مستقر می شوند، مانند؛ فایروال، تفکیک DNS، نرم افزار و وب سرورها، مناطق افزونه، متعادل کننده های بار الاستیک، سیستم پیشگیری از نفوذ (IPS)، سرویس ذخیره سازی ساده آمازون (S3)، Amazon CloudFront. به منظور سازماندهی شبیه سازی؛ این پروژه به دو مرحله اصلی تقسیم خواهند شد:

فاز اول, مطالعه رفتار معماری AMAZON EC2 تحت حملات DDoS است. در این مرحله, دو سناریو توسعه خواهند یافت. اولی, تولید یک ترافیک قانونی از منابع مختلف، و سپس نظارت بر واکنش موارد آمازون است. سناریوی دوم, این نمونه ها را با طبقه بندی های DDoS مختلف هدف قرار خواهد داد و نتایج را با یافته های قبلی ما مقایسه خواهد کرد. حمله DDoS شکل 5, درخت حمله توسعه داده شده برای هدایت توسعه حمله DDoS ما است.

Scheneier (1999، 2000) یک رویکرد روشمند برای توصیف تهدیدات در برابر یک سیستم محافظت شده را اختراع کرده است. این فرایند از یک ساختار درختی برای نمایش هدف به صورت گره ریشه استفاده می کند و گره های برگ تعیین خواهد کرد که چگونه به این هدف برسیم. به دلیل پیچیدگی بالای شبکه های مقیاس بزرگ؛ درختان حمله باید توجه بالایی را دریافت نمایند چرا که اگر یک ایده در مورد چگونکی به خطر افتادن یک دارایی از دست رفته از دست برود؛ درخت حمله بی فایده خواهد بود. شکل بالا حملات مختلف را نشان می دهد که در دسترس عاملان برای هدف قرار دادن زیرساخت های آمازون AWS توسط انکار توزیع شده حمله به سرویس با استفاده از منابع و یا کاهش پهنای باند قرار می گیرد.

تجزیه و تحلیل دلایل وقوع حمله DDoS, ما را قادر می سازد تا برای پروژه خود یک مکانیزم جامع را توسعه دهیم که به طور گسترده با آسیب پذیری سیستم ها و پروتکل های حال حاضر ما مقابله می کند.

لازم است که تحقیقات ما از یک رویکرد سیستماتیک به منظور توسعه محصول پیروی نماید و همه روش ها و رویه های انتخاب شده باید برای الزامات محصول مناسب باشند، همچنین رویه ها باید دارای استانداردهای بالا از نظر کارایی و دقت باشند. هر یک از گره و زیر گره ها در درخت باید با شرایط امنیتی مرتبط باشند که عبارتند از:



شکل. 6. چرخه عمر مهندسی توسعه نرم افزار ایمن یکپارچه (IS-SDLC).

جدول 1 تهدیدات و ویژگی های امنیتی.

Threat	Security Property
Spoofing	Authentication
Tampering	Integrity
Repudiation	Non-Repudiation
Information Disclosure	Confidentiality
Denial of Service	Availability
Elevation of Privilege	Authorization

4.1 شرایط امنیتی برای اندازه گیری حملات DDoS

این یکی از روش های اصلی ما به عنوان بخشی از مدل یکپارچه برای فرایند مهندسی نرم افزار امنیتی است.

4.2 شناسایی تهدیدات

در این مرحله، تهدیدات را می توان شناسایی و بر اساس مدل STRIDE طبقه بندی نمود که توسط Microsoft توسعه داده شده است و شامل طبقه بندی تهدیدات به شش دسته می شود: حقه بازی، دستکاری، طلاق، محرومیت از خدمات و بالا بردن امتیاز (STRIDE، 2002). رویکرد STRIDE شامل تجزیه و تحلیل هر یک از مؤلفه ها در برابر تهدیدات بالقوه با استفاده از تجزیه فرآیند و برای هر تهدید، کاهش های ممکن را می توان پیشنهاد داد که تحکیم امنیت قوی در زیرساخت ابر را میسر می سازد. این مورد در جدول 1 نشان داده شده است.

زمانی که حملات DDoS, تنها در دسترس بودن منابع را هدف قرار می دهند، ما دامنه کاربرد خود را تنها روی یک دسته بندی DOS تمرکز متمرکز می نماییم که این کار با شناسایی دارایی هایی صورت می گیرد که می توانند به صورت زیر هدف قرار گیرند:

- با ذخیره سازی و بازرسی هر جلسه ارتباطات ذخیره شده بر روی میز ورودی آن, فایروال و IPS برای مقابله با تهدیدهای هدف قرار دادن تمامیت شبکه و محرمانگی طراحی می شوند. در طول یک DDoS؛ یک مهاجم می تواند منابع خود را با پر کردن جدول ورود با اطلاعات بسته ساختگی تخلیه نماید.

- همانطور که می دانیم، متعادل کننده بار سرور را می توان برای تحمل خرابی سرور با توزیع وظیفه در سراسر مراکز داده های مختلف برای دستیابی به در دسترس بودن بالای برنامه و همچنین مقیاس پذیری سرور استفاده

نمود. با این حال، اثربخشی الگوریتم های متعادل کننده بار (گرد رابین، تصادفی، آستانه، مدیر مرکزی) با چالش های مختلف از نظر توان عملیاتی، مقیاس پذیری زمان پاسخ و بهره برداری از منابع در طی سیل شبکه مواجه است همانطور که مقاله های تحقیقاتی مختلف, کارایی الگوریتم های متعادل کردن بار ایستا و پویا را ارزیابی کرده اند.

- لینک های مستقر در زیرساخت ابر را می توان با حمله حجمی 3/4 تنگ تر نمود، به ویژه هنگامی که باتنت قادر به تولید حدود 300 گیگابایت ترافیک است که به کارگیری آن با لینک های فعلی شبکه در دسترس مشکل است.

رتبه بندی, بخشی از فرآیند شناسایی تهدید است؛ به هر یک از اجزای مهم که قصور در آن می تواند یک تنگنای جدی را ایجاد نماید باید بالاترین رتبه اختصاص داده شود که بعد از آن در طول فرایند تعدیل, این جزء توجه بیشتر و همچنین کنترل های امنیتی پیشرفته را دریافت خواهد کرد. با این حال، مدل رتبه بندی خطر تهدید DREAD Microsoft را می توان برای رتبه تهدیدات بر اساس موارد زیر به کار گرفت: آسیب های بالقوه، تکرار پذیری، قابلیت بهره برداری، کاربران تحت تاثیر قرار گرفته و قابلیت کشف. ما مدل DREAD را برای سناریوی زیرساخت ابر خود به منظور مقابله با تکنیک های دفاع در برابر حملات DDoS استفاده می نماییم, همانطور که در جدول 2 نشان داده شده است.

همه تهدیداتی که به عنوان تهدیدات دارای خطر بالا طبقه بندی می شوند, باید کنترل اضافی را توسط تبدیل آنها را به یک فرآیند کاهش قوی و یک طرح واکنش حادثه دریافت نماییم. به طریقی دیگر، می توانیم از فرمول های زیر برای انجام ارزیابی ریسک خطر استفاده نماییم:

ریسک = تهدید × آسیب پذیری × نتیجه

درخت حمله به عنوان یکی از موثرترین روش ها برای فهرست کردن تمام فرصت های موجود برای عاملان به منظور دستیابی به اهدافشان در نظر گرفته می شود و می توان آن را توسط تفکر از دیدگاه دشمن در مورد نحوه تنزل عملکرد یک زیرساخت ابر انجام داد، و یا اگر این امکان وجود داشته باشد, منابع در دسترس از بین می روند. ما یک درخت حمله DDoS را برای ارائه یک تصویر واضح در فرصت های مختلف در دسترس برای عاملان ساخته ایم و این

یک درخت کامل نیست چرا که هر زیرساخت ابر دارای طراحی خاص، تجهیزات، خدمات، سیاست های امنیتی و توان عملیاتی شبکه است. این باید شبیه به نمودار نشان داده شده در شکل 5 انجام شود.

4.3 ساخت درخت حمله و فرآیند کاهش (تعدیل)

حملات DDoS, رایج ترین نوع حملات امنیتی هستند که اغلب در سال های اخیر دیده شده است, همانطور که قبلاً آنها را مورد بحث قرار دادیم. حملات DDoS می توانند یک ویژگی کامل را غیر فعال سازند و منابع را غیر قابل دسترس سازند. حمله DDoS, بزرگترین تهدید برای محاسبات ابری و برای موفقیت آن است. بنابراین، نشان دادن رویکرد سیستماتیک برای شرایط امنیتی بخش مهمی از رویکرد ماست که می تواند مانع تاثیر DDoS از طریق طراحی و توسعه شود. رویکرد ما, استفاده از ابزار مدل سازی تهدید Microsoft برای ساخت درخت حمله و استفاده از موارد سوء استفاده برای شناسایی شرایط امنیتی است. ما همچنین استفاده از فرآیند کسب و کار، شبکه، و ابزارهای شبیه سازی ابر را برای مطالعه و ارزیابی کامل طراحی پیشنهادی پیشنهاد می نماییم. پس از ساخت یک نمودار جریان داده ها؛ می توانیم نتایج ارائه شده را توسط درخت حمله استفاده کنیم تا تعیین نماییم که کدام اقدامات متقابل باید برای در نظر گرفتن هر حمله به کار گرفته شوند و هر اقدام متقابل باید توسط یک وضعیت مشخص شود. DDoS مدرن, وظیفه ارائه دهندگان امنیت برای ساخت یک حفاظت جامع را پیچیده نموده است که می تواند ترافیک DDoS را در زمان واقعی بدون تغییر با ترافیک قانونی پیچیده نماید. با این حال، فرمول های زیر برای ارائه بینشی در مورد بهترین رویکرد توسعه یافته است که به طور کامل می تواند این تهدید را شکست نماید و همچنین ارائه دهندگان ابر و همچنین شرکت های آماده را قبل از وقوع حمله را هدایت خواهد کرد. بنابراین، می توانیم فرآیند حفاظت DDoS را به صورت زیر تعریف کنیم:

شرایط امنیتی با حفاظت DDoS = طرح احتمالی + DDoS بهترین اقدامات امنیتی + استفاده از موارد سوء استفاده هنگامی که یک حمله DDoS توسط ابزارهای نظارت آشکار می شود؛ ارائه دهنده ابر باید یک طرح پاسخ حادثه (IR) داشته باشد که فرآیندهایی را تفصیل نماید که باید در طول کاهش دنبال شود. طرح IR شامل: اسناد و مدارک

حوادث، استراتژیهای مهار، تشدید حادثه و بازیابی حادثه می شود. برخی از نمونه های رویه های پاسخ حادثه اشاره می کنیم:

- استفاده از دستگاه های تخصصی برای جلوگیری از حمله در نزدیکی شبکه ابری.
- استفاده از حفاری سیاه برای هدایت ترافیک حملات DDoS به یک پورت تهی.
- سوئیچینگ به شبکه های متناوب.
- اتصالات ناخواسته در روترها (مسیریاب ها) باید فسخ شوند.
- تماس با تیم داخلی برای قابلیت دید به حمله (ستاد امنیت + عملیات شبکه).
- تسکین فشار بر زیرساخت با استفاده از فیلترینگ بالادست.

Deshmukha و Devadkarb (2015) بررسی نموده اند که قصد اصلی یک حمله DDoS, ناتوان نمودن قربانی در استفاده از منابع است. در بسیاری از حالات، اهداف می تواند وب سرورها، CPU، ذخیره سازی، و دیگر منابع شبکه 4 باشند. در محیط ابر نیز DDoS را می تواند عملکرد خدمات ابر را به طور قابل توجهی توسط آسیب رساندن به سرور های مجازی کاهش دهد.

5. فرآیند چرخه عمر توسعه نرم افزار امنیت یکپارچه

اشکالات و شرایط بحث شده در بالا برای یک روش مختصر، به توسعه یک مدل منجر می شود که فعالیت های مختلف شناسایی و تجزیه و تحلیل مهندسی نرم افزار امنیتی را در فرآیند توسعه نرم افزار ادغام می کند و این فرآیند جدید و فعالیت های آن در شکل6 نشان داده شده است. با این حال، این مقاله فقط بر فعالیت های خاص شرایط امنیتی نرم افزار متمرکز می کند. بر اساس این مدل، SSRE (مهندسی شرایط امنیتی نرم افزار) شامل شناسایی استانداردها و استراتژی های سازمان با توجه به استخراج شرایط (از جمله تجزیه و تحلیل، اعتبار سنجی، تایید)، انجام مدیریت ریسک و کاهش می شود و شناسایی الزامات (شرایط) امنیتی نرم افزار متشکل از یک فرایند فرعی تعریف امنیت، شناسایی استراتژی های امنیتی، انجام تجزیه و تحلیل محدوده های کاربرد و حوزه ها، مدل

سازی و شبیه سازی فرایند کسب و کار، شناسایی مسائل امنیتی، استفاده از موارد استفاده و موارد سوء استفاده، الگوهای حمله می شود.

به همین ترتیب، این مدل فرایندهای امنیتی خاص را برای شناسایی تهدیدات امنیتی در طول طراحی، توسعه، آزمایش، استقرار، و تعمیر و نگهداری فراهم می کند. تعدادی متعددی از اصول طراحی خوب وجود دارد که می توانند در اکثریت قریب به اتفاق از نوشته های نرم افزار طراحی یافت شوند. با این حال، در زیر لیستی از برخی از اصول طراحی کلیدی بیان شده است که بسیار مربوط به طراحی نرم افزار امنیتی هستند و بخشی از مدل IS-SDLC ماست:

• اصول حالات حداقل امتیاز برای ایجاد تنها یک مجموعه حداقل از حقوق (امتیازات) برای یک فرد که دسترسی به یک منبع را درخواست می کند. این به جلوگیری از خسارت عمدی یا غیرعمدی که می تواند برای یک منبع در صورت حمله ایجاد شود کمک می کند.

• اصول جداسازی حالات امتیاز می گوید که یک سیستم نباید دسترسی به منابع را بر اساس یک شرط مجاز سازد.

• طراحی توسط ترکیب CVE شناخته شده

• طراحی برای انعطاف پذیری که به خاطر توسعه یک مدل انعطاف پذیری که از پایداری سیستم در کنار اعتماد سازی و امنیت در (BTSI) پشتیبانی می کند, با IBM یک تیم شده ایم.

• انتخاب شرایط امنیتی نرم افزار بعد از شبیه سازی عملکرد با استفاده از BPMN (نشانه گذاری مدل سازی فرآیند کسب و کار) و با جزئیات توسط Ramachandran، 2014 و Chang و Ramachandran، 2016 است.

Threat	D	R	E	A	D	Total	Rating
Exhausting the resource of a cloud server using a flood based Internet Group Management Protocol	8	9	7	6	5	35	Medium risk
Targeting a cloud server with DNS amplification attack (Receiving high load of recursive queries)	9	9	9	9	9	45	Medium risk
A cloud server will be targeted with an ICMP amplification attack (Spoofing the source address of ping request)	4	7	8	3	4	26	Medium risk
An end system will be crashed by a Land attack (Receiving packets with same IP source and destination address)	4	7	8	3	4	26	Medium risk
Consuming the resources of a system using SYN flood attack	4	7	8	3	4	26	Medium risk
A SIP Proxy server could be targeted by a high load of SIP invite messages	8	8	7	7	7	37	High risk
The attacker congest the uplink with ICMP flood	9	9	9	9	9	45	High risk
The attacker congests the uplink with UDP flood	9	9	9	9	9	45	High risk
Crashing Cisco routers with an unexpected header (DOS attack)	9	9	9	9	9	45	High risk
Crashing an end system by ping of death attack (Sending oversized ping request packets which violates the Internet Protocol)	5	6	7	3	3	24	Medium risk
Crashing a firewall with unexpected header (DOS attack)	9	8	8	0	8	33	Medium risk
Overflowing the IPS/Firewall entry Table with bogus packets	9	8	8	0	8	33	Medium risk

D: damage potential E: exploitability D: discoverability R: reproducibility A: affected users.

جدول 2 اولویت بندی خطر بر اساس DREAD

فعالیت های SSRE در Is-SDLC ما از امنیت در شبکه تعریف شده نرم افزاری (SDN)، خدمات رایانش ابری (نرم افزار به عنوان سرویس (SaaS)، بستر نرم افزاری (سیستم عامل) به عنوان یک سرویس (PaaS) و زیرساخت به عنوان یک سرویس (IaaS) حمایت می کند. امنیت شرکت شامل ارائه دهندگان خدمات ابر و مصرف کنندگان خدمات، و طراحی برای اصول و تکنیک های امنیتی می شود. این یکی از کمک های منحصر به فرد ما به بدنه دانش در پژوهش امنیتی نرم افزار است.

6. روش مهندسی شرایط امنیتی نرم افزار به عنوان بخشی از IS-SDLC

مهندسی نرم افزار امن, به درک گسترده از اهمیت یکپارچه سازی امنیت در سراسر مراحل چرخه عمر توسعه نیاز دارد. Oueslati و همکاران (2016), چالش های توسعه نرم افزار امن را با استفاده از رویکرد مبتنی بر چابک بودن مورد بحث قرار داده اند. به طور مشابه، در طول فعالیت های مهندسی شرایط، روش های موجود, بسیاری از تغییرات شرایط را مورد بحث قرار می دهد و یکپارچه سازی امنیت را به عنوان بخشی از تکامل شرایط در نظر نمی گیرد. Mead, Morales, and Alice (2015) افزایش SDLC کنونی با موارد سوء استفاده به دست آمده از تجزیه و تحلیل بدافزار بر اساس حملات را پیشنهاد کرده اند. Beckers و همکاران (2014)، یک روش مبتنی بر الگوی ساختاریافته را با حمایت از استخراج شرایط امنیتی و انتخاب اقدامات امنیتی پیشنهاد کرده اند و رویه آنها, مشتریان بالقوه ابر را برای مدلسازی کاربرد مورد کسب و کار خود در زمینه رایانش ابری با استفاده از یک رویکرد مبتنی بر

الگو که همچنین به عنوان الگوی تجزیه و تحلیل سیستم ابری شناخته می شود هدایت می کند. با این حال، این روش ها بسیار جالب هستند, اما فاقد یکپارچه سازی شرایط امنیتی از سهامداران مختلف می شوند. هدف این مقاله, برآورده سازی این شکاف با یکپارچه سازی سهامداران مختلف است و آنها اغلب شرایط امنیتی نرم افزار متفاوت را که باید توسط مهندسان شرایط تحکیم شوند فراهم می کنند. توسعه نرم افزار و توسعه نرم افزار امن شامل بسیاری از سهامداران و رهبران کسب و کار می شود و هماهنگی آنها برای ارائه سیستم های نرم افزاری امن, حیاتی است. سهامداران مختلف در شکل 7 نشان داده شده اند.

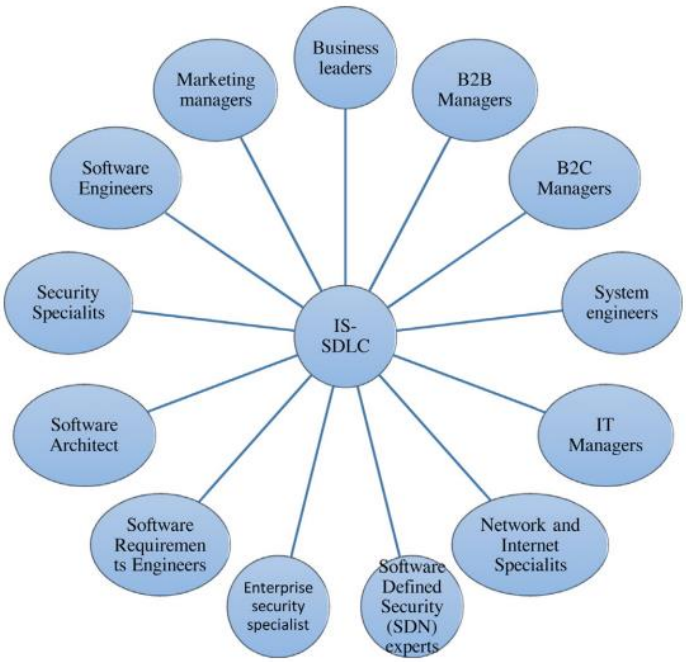


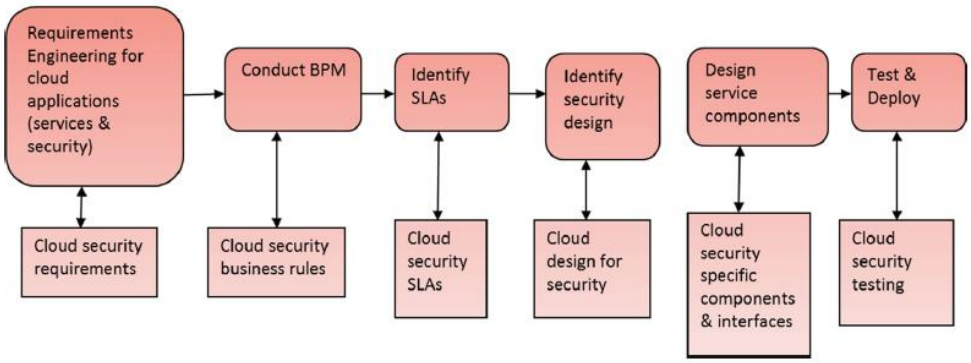
Fig. 7. Stakeholders in integrated secure-SDLC (IS-SDLC).

بخش قبلی شرح مختصری از روش های مختلف برای استخراج شرایط مورد نیاز برای نرم افزار امنیتی را فراهم کرده است. بهترین شیوه های معمول عبارتند از:

1. استخراج و انتخاب شرایط مورد نیاز برای نرم افزار امنیتی با نمادهای بصری صریح
2. اولویت بندی شرایط امنیتی نرم افزار
3. ارزیابی ریسک و خفیف کردن شرایط امنیتی نرم افزار
4. طراحی و پیاده سازی شرایط امنیتی نرم افزار

5. ارائه پشتیبانی از چرخه عمر SDLC

روش های موجود فاقد ترکیب شدید مهندسی اجتماعی با مطالعه شرایط امنیتی نرم افزار (یادگیری از تجربیات واقعی)، مدل سازی فرایند کسب و کار امنیتی خاص، شبیه سازی عملکرد فرآیندهای کسب و کار امنیتی خاص، محاسبات خدمات، فن آوری های فعلی و در حال ظهور مانند محاسبات ابری، معماری شبکه بندی نرم افزاری-تعریف شده و امنیت شرکت نرم افزار-تعریف شده، و آسیب پذیری های در حال ظهور، و حملات سایبری هستند. این ما را به توسعه یک مدل توسعه نرم افزار یکپارچه-امن با حمایت از شرایط امنیتی نرم افزار مورد ارزیابی و پیاده سازی صریح هدایت می کند که در این مقاله ارائه شده است. Ramachandran (2012), یک تجزیه و تحلیل مقایسه ای از مهندسی شرایط امنیتی نرم افزار های مختلف (SSRE)، طراحی، و روش های آزمون مبتنی بر معیارهای ارزیابی استفاده شده ما فراهم می کند و این به سازمان و مهندسان کمک خواهد کرد که روش مناسب را برای توسعه مناسب سیستم انتخاب نمایند. بر اساس تجربه با مدل IS-SDLC در پروژه های مختلف، این مقاله مجموعه ای از بهترین رهنمودهای عمل و توصیه ها در مورد SSRE و SSE را به طور کلی شناسایی کرده است.



شکل. 8. ابر فرآیند توسعه خدمات IS-SDLC.

به عنوان بخشی از IS-SDLC، این مقاله, یک مدل فرایند توسعه برای خدمات ابر را با BSI (ایجاد امنیت داخلی) در سراسر چرخه حیات توسعه سرویس ابری توسعه داده ایم. این نیز به خوبی بهترین عمل شناخته شده برای استخراج و اعتباردهی به الزامات سطح خدمات به موقع است می تواند به اندازه 70٪ در هزینه آزمون و توسعه هزینه های

کلی صرفه جویی ایجاد کند. همانطور که در نمودار نشان داده شده است، مدل فرایند توسعه ابر شامل تعدادی از مراحل مانند RE برای ابر، انجام مدل سازی BPM و مشخصات (با استفاده از BPMN 2 استاندارد و BPEL)، شناسایی و تعیین SLA ها، ساخت امنیت نرم افزار، خدمات طراحی، و تست و استقرار می شود. این مورد در شکل 8 نشان داده شده است.

مرحله RE متشکل از تعدادی از توالی های موازی مانند شناسایی روشنی از شرایط خدمات همراه با شرایط امنیتی و انجام / توسعه BPMN (نشانه گذاری مدل سازی فرآیند کسب و کار) برای هر سناریوی خدمات است که همچنین به ما اجازه خواهد داد تا اثربخشی عملکرد خود را قبل از شروع واقعی طراحی و پیاده سازی شبیه سازی و مطالعه نماییم. این برای فرآیند سنتی مهندسی شرایط جدید است. گام بعدی در مدل IS-SDLC ما, شناسایی SLA های مشخص (قرارداد سطح خدمات و اداره امور برای هر سرویس ابر است). گام بعدی, توسعه یک طراحی برای امنیت ابر با استفاده از تمام اصول طراحی شناخته شده و طراحی خدمات با استفاده از SoaML یا مدل جزء خدمات با استفاده از UML 2 است. همچنین می توانیم مدل های SDLC کلاسیک مانند آبشار، مارپیچی، و دیگران را برای در برداشتن ویژگی های عملکردی و غیر عملکردی برای هر خدمات SaaS و خدمات وب اعمال نماییم. به عنوان مثال، به منظور توسعه خدمات ابری سلامت الکترونیک، IS-SDLC توصیه روش مشخصات رسمی و کلاسیک. در مرحله طراحی، ما می تواند مجموعه ای از دلایل طراحی خوب استخدام به سرودن SaaS به عنوان اجزاء. در مدل سازی فرایند کسب و کار و خدمات مشخصات سطح، ما می توانیم BPMN برای مدل سازی و BPEL برای مشخص گردش سطح خدمات استفاده کنید. سپس می توان به راحتی این مصنوعات به مجموعه ای از خدمات SaaS تبدیل نمود. مدل نشان داده شده در این بخش, یک فرایند توسعه ابر امنیت-محور است. از این رو، دلیل اصلی برای شناسایی، مشخص سازی، و طراحی موضوعات خاص امنیتی نرم افزار سطح خدمات است که در تمام مراحل فرایند توسعه است.

7. بهترین رهنمودهای عمل

برای سیستم های امن، این مقاله مجموعه ای از دستورالعمل های رایج را شناسایی می کند که برای توسعه نرم افزار امن قابل اجرا می باشند:

1. ایجاد یک لیست از چک لیست های شرایط امنیتی و طبقه بندی آنها به صورت: مهم، میانه و متوسط .

2. گرد هم آوردن یک تیم بازرسی شرایط برای انجام فرایند اعتبار سنجی شرایط امنیتی

3. شناسایی، استخراج، تجزیه و تحلیل، و مدیریت شرایط امنیتی

4. تعیین و مدلسازی موارد سوء استفاده و استخراج شرایط امنیتی از موارد سوء استفاده

5. چک کردن الزامات عملیاتی و کاربردی در برابر شرایط امنیتی

6. ایجاد یک فرهنگ امنیت سازمانی (مثلاً, بررسی اطمینان از استفاده مناسب از سیستم های ایمیل بایدها و نبایدها).

7. کاربرد مدل DREAD و توسعه چرخه عمر امنیت Microsoft برای شناسایی تهدید و تخفیف

8. درخواست مدلسازی و شبیه سازی فرایند کسب و کار با استفاده از ابزارهای BPMN مانند Benita soft که عملکرد مشخصی را فراهم می کند که به همه فرایندهای امنیتی خاص انتخاب شده نسبت داده می شود.

9. توسعه طراحی برای امنیت ابر

10. طراحی خدمات ابر با SoaML یا مدل UML 2. 0 component

11. به کار بردن ابزارهای شبیه سازی شبکه به منظور مطالعه و شناسایی تهدیدات شبکه

8. نتیجه گیری

مهندسی امنیت نرم افزار, چند شیوه عالی, تکنیک, و روش ها برای توسعه سیستم ها و خدمات را که برای امنیت, انعطاف پذیری, پایداری ساخته می شوند ارائه می دهد. هرچند, امنیت نرم افزار را نمی توان بعد از اینکه یک سیستم ساخته می شوند و به مشتریان تحویل داده می شود اضافه نمود, همانطور که در کاربردهای نرم افزاری امروزی دیده می شود. این مقاله, تکنیک های مختصر و رهنمودهای شرایط بهترین شیوه را در مورد امنیت نرم افزار

ارائه نمود و همچنین یک مدل SDLC-امن-یکپارچه (IS-SDLC) را مورد بحث قرار داد که به دست اندرکاران, محققان, یادگیرندگان و زبان آموزان سود خواهد رساند. این مقاله, کاربرد چرخه عمر توسعه امنیتی مایکروسافت را نشان داده شده است و یک مدل فرآیند توسعه نرم افزاری امن یکپارچه را برای محاسبه ابری توسعه داده است. این مقاله, موضوعات مرتبط با ضبط, تحلیل و مدلسازی شرایط امنیتی نرم افزار را ارائه نمود و مهندسی و مدیریت شرایط امنیتی نرم افزار را به عنوان یک خدمات در حال ظهور پیشنهاد نموده است (SSREMaaES). SSREMaaES دارای قابلیت تغییر رو توسعه نرم افزار است و ابزارها و تکنیک ها را به عنوان یک خدمات ارائه می دهد که می توان با سهامداران توزیع شده به توسعه و تسهیم آنها پرداخت.

References

Alberts, C., & Dorofee, A. (2002). *Managing information security risks: the OCTAVE℠ approach*. Addison Wesley.

Allen, J. H., et al. (2008). *Software security engineering: a guide for project managers*. Addison Wesley.

Ashbaugh, D. A. (2006). *Assessing information security risks in the software development life cycle, september*. www.stsc.hill.af.mil

Ashford, W. (2009). *On-demand service aims to cut cost of fixing software security flaws*. http://www.computerweekly.com/Articles/2009/07/14/236875/on-demand-service-aims-to-cut-cost-of-fixing-software-security.htm

BSI. (2013). *Attack patterns articles*. https://buildsecurityin.us-cert.gov/articles/knowledge/attack-patterns

Beckers, K., et al. (2014). A structured method for security requirements elicitation concerning the cloud computing domain. *International Journal of Secure Software Engineering, 5*(2), 20–43.

Belapurkar, A., et al. (2009). *Distributed system security: issues, processes and solutions*. Wiley.

Caralli, R. A., et al. (2007). Introducing OCTAVE Allegro: Improving the Information Security Risk Assessment Process, TECHNICAL, REPORT, CMU/SEI-2007-TR-012.

CERT-SEI, www.cert.org.

CERT-UK, https://www.cert.gov.uk/.

CLASP. (2006). *OWASP CLASP version 1.2*. http://www.lulu.com/items/volume_62/1401000/1401307/3/print/OWASP_CLASP_v1.2_for_print_LULU.pdf

Chang, V., & Ramachandran, M. (2016). Towards achieving Data Security with the Cloud Computing Adoption Framework. *IEEE Transactions on Services Computing, 9*(1), 138–151.

Chang, V., Ramachandran, M., Wills, G., Walters, R. J., Kantere, V., & Li, C. S. (2015). ESaaSA 2015-Proceedings of the International Workshop on Emerging

Chang, V., & Ramachandran, M. (2016). Towards achieving Data Security with the Cloud Computing Adoption Framework. *IEEE Transactions on Services Computing, 9*(1), 138–151.

Chang, V., Ramachandran, M., Wills, G., Walters, R. J., Kantere, V., & Li, C. S. (2015). ESaaSA 2015-Proceedings of the International Workshop on Emerging Software as a Service and Analytics, Lisbon, Portugal, May 20–22, 2015.

Chen, A. J., (2004). Security engineering for software (SES), CS996-CISM, isis.poly.edu/courses/cs996-management/Lectures/SES.pdf.

Deshmukha, R. V., & Devadkarb, K. K. (2015). Understanding DDoS attack & its effect in cloud environment. *Procedia Computer Science, 49*(2015), 202–210.

Ellison, R. J., & Moore, A. P. (2003). *Trustworthy refinement through intrusion-aware design (CMU/SEI-2003-TR-002, ADA414865)*. pp. 2003. Pittsburgh, PA: Software Engineering Institute, Carnegie Mellon University.

Firesmith, D. (2003). Engineering security requirements. *Journal of Object Technology, 2*(1), 2003.

Firesmith, D. (2007). *Engineering safety- & security-related requirements ICCBSS tutorial*. SEI Carnegie Mellon University., 27 February

Gartner (2016). http://www.gartner.com/newsroom/id/2828722 (Accessed 15.03.16).

Han, G., et al. (2016). An efficient virtual machine consolidation scheme for multimedia cloud computing. *Sensors, 16*, 246. www.mdpi.com/journal/sensors

Howard, M., & LeBlanc, D. C. (2002). *Writing secure code* (2nd ed.). Redmond, WA: Microsoft Press.

Hsu, I.-C., & Cheng, F.-Q. (2015). SAaaS: a cloud computing service model using semanticbased agent. *Expert Systems, 32*(February (1)) http://dx.doi.org/10.1111/exsy.12063

Jacobson, I. (1992). *Object oriented software engineering: use case driven approach*. Addison Wesley.

Kostoska, M., Gusev, M., & Ristov, S. (2014). A new cloud services portability platform. In *24th DAAAM international symposium on intelligent manufacturing and automation*.

Kotonya, G., & Sommerville, I. (1998). *Requirements engineering: processes and techniques*. Wiley.

vana Lamsweerde, A. (2009). *Requirements Engineering: from system goals to UML models to software specifications*. UK: Wiley.

McGraw, G. (2006). *Software security: building security in*. USA: Addison Wesley.

Mead, N. R., et al. (2008). *Incorporating security quality requirements engineering (SQUARE) into standard life-cycle models, SEI technical note CMU/SEI-2008-TN-006*. http://www.sei.cmu.edu

Mead, N. R., Morales, A., & Alice, J. (2015). A method and case study for using malware analysis to improve security requirements. *International Journal of Secure Software Engineering, 6*(1). IGI Global.

Oueslati, H., et al. (2016). Evaluation of the challenges of developing secure software using the agile approach. *International Journal of Secure Software Engineering, 7*(1). IGI Global.

Ramachandran, M. (2012). *Software security engineering: design and applications*. New York, USA: Nova Science Publishers. ISBN: 978-1-61470-128-6, https://www.novapublishers.com/catalog/product_info.php?products_id=26331

Ramachandran, M. (2014). Enterprise Security Framework for Cloud Data Security, Book chapter Delivery and Adoption of Cloud Computing Services in Contemporary Organizations. In Chang, V. & Graham, D. (Eds.) (2006) *Building Security In*, IGI Global https://buildsecurityin.us-cert.gov/bsi/articles/best-practices/requirements/548-BSI.html.

SDL. (2016). *Microsoft security development lifecycle*. https://www.microsoft.com/en-us/sdl/adopt/threatmodeling.aspx

S-SDLC. Introducing Secure Software development Life Cycle (S-SDLC), Infosec Institute, http://resources.infosecinstitute.com/intro-secure-software-development-life-cycle/.

STRIDE. (2002). *The STRIDE threat model*. https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx

Schneier, B. (1999). Attack Trees: modelling security threats. *Dr Dobbs Journal*, http://www.schneier.com/paper-attacktrees-ddj-ft.html

development-life-cycle/.

STRIDE. (2002). *The STRIDE threat model*. https://msdn.microsoft.com/en-us/library/ee823878(v=cs.20).aspx

Schneier, B. (1999). Attack Trees: modelling security threats. *Dr Dobbs Journal*, http://www.schneier.com/paper-attacktrees-ddj-ft.html

Schneier, B. (2000). *Secrets and lies: digital security in a networked world*. New York, NY: John Wiley & Sons.

Sommerville, I., & Sawyer, P. (1998). *Requirements engineering: a good practice guide*. Wiley.

Srivastava, S., et al. (2015). The technological growth in eHealth services. *Computational and Mathematical Methods in Medicine*, 18. http://dx.doi.org/10.1155/2015/894171. Article ID 894171, Hindawi

TAM. (2016). *Mocrosoft threat modelling tool*. https://www.microsoft.com/en-us/download/confirmation.aspx?id=49168

Whiting, R. (2015). *10 emerging cloud vendors you need to know about july 2015*. http://www.crn.com/slide-shows/cloud/300077581/10-emerging-cloud-vendors-you-need-to-know-about.htm/pgno/0/1

Woody, C., & Alberts, C. (2007). Considering operational security risk during system development. *IEEE Security & Privacy*, 30–35.

Xu, X. (2013). Cloud manufacturing: a new paradigm for manufacturing businesses. *Australian Journal of Multi-Disciplinary Engineering, 9*(2), 105–116. http://dx.doi.org/10.7158/N13-GC08.2013.9.2