مطالعات علوم کاربردی در مهندسی
دوره ۲، شماره ۲، تابستان ۱۳۹۵
صفحات ۶۰- ۷۰

ISSN: ۲۴۷۶-۴۵۰۷
www.irijournals.com

بومی‌سازی و پیاده‌سازی تجارب و الگوهای موفق امنیت اطلاعات در فرآیندهای چرخه عمر نرم‌افزاری مبتنی بر استاندارد ISO/IEC15504

علی‌اصغر قرایی*، ناصر مدیری

چکیده

با توجه به گسترش روزافزون برنامه‌های کاربردی و به کارگیری آنها در زمینه‌های مختلف، این برنامه‌ها روز به روز بیشتر مورد توجه قرار می‌گیرند. بروز اشکال در این برنامه‌ها می‌تواند کاربران و تولیدکنندگان این نرم‌افزارها را با چالش رو به رو نماید. یکی از این دسته اشکالات، سوء استفاده از خلاهای امنیتی موجود در نرم‌افزارهاست. هزینه برطرف نمودن این مشکلات پس از انتشار نرم افزار بسیار زیاد بوده و تولیدکنندگان برنامه‌های کاربردی را با مشکل مواجه می‌کند. برای جلوگیری از این معضل می بایست با بررسی خطرات احتمالی و مدیریت ریسکهای موجود، پیش از انتشار محصول خلاهای امنیتی را شناسایی و برطرف نمود. لحاظ نمودن امنیت در چرخه تولید نرم‌افزار سبب می‌گردد تا کاربران برنامه‌های کاربردی از امنیت برنامه مورد استفاده خود اطمینان حاصل نمایند و همچنیین تولیدکنندگان ضرر مالی کمتری را متحمل شوند. در راستای بالا بردن سطح امنیت نرم‌افزارها و مقابله با مسائلی که امنیت را نقض می‌نمایند، در این مقاله سعی بر آن است تا ابتدا الگوهای موفق چرخه تولید امن نرم افزار مورد بررسی قرار گیرند، و با توجه به اهمیت به کارگیری استانداردها در تعیین امنیت در چرخه تولید نرم افزار، الگوهای بررسی شده با این استانداردها مطابقت داده شوند. در این مقاله از استاندارد ISO/IEC15504 استفاده گردیده است و برای این که الگوهای موفق شناسایی شده را بتوان در سازمانهای ایران نیز مورد استفاده قرار داد این الگوها مطابق با نیازهای امنیتی نرم‌افزاری موجود در ایران بومی گردیده‌اند. در این مطالعه لحاظ بومی‌سازی چرخه حیات امن نرم افزار نیز بر اساس استاندارد ISO/IEC15504 می‌باشد و با استفاده از این استاندارد، اتخاذ فعالیت‌هایی امنیتی به منظور ارتقا و بومی‌سازی فعالیت‌های چرخه حیات پیشنهاد گردیده است.

واژه‌های کلیدی: چرخه تولید نرم‌افزار، چرخه تولید امن نرم‌افزار، استاندارد ISO/IEC15504، بومی‌سازی.

دوره ۲، شماره ۲، تابستان ۱۳۹۵، صفحات ۶۰- ۷۰

مقدمه و بیان مسئله

سازمان ها و شرکت های تولیدکننده نرم افزار به منظور کشف بیش ترین تعداد مشکلات امنیتی نرم افزار تولید شده در یک پروژه باید ابزار پویش امنیت را در چرخه حیات تولید نرم افزار جای دهند. سال‌ها است که برنامه‌های نرم افزاری نوشته می شوند و سال ها است که امنیت، کم و بیش، در آن ها لحاظ می‌شود شاید حتی بدون آن که بسیاری از مشتریان از کیفیت و قابلیت اعتماد آن آگاهی کافی داشته باشند. با فراگیر شدن وب و نرم افزارهای کاربردی وب محور، هم چنین هوشمندانه‌تر شدن تهاجم ها و تقلب ها، امنیت نرم افزارها بیش از پیش اهمیت یافته است.

مطالعات بی شمار و توصیه های تحلیل گران بیان گر اهمیت ارتقای امنیت نرم افزار در چرخه حیات تولید نرم‌افزار (SDLC)[1] به جای کشف و رفع آن پس از تولید و توزیع گسترده است. شرکت های تولیدکننده نرم افزار برای کشف نقایص امنیتی در نرم افزار خود، به صورت مستقیم و غیرمستقیم هزینه می کنند.(دولدی و گودینوف[2]، ۱۹۹۵)

استاندارد بین المللی ISO/IEC 15504 را می‌توان با چارچوب مدیریت امنیت اطلاعات ISO/IEC 27000 هم‌تراز کرد. در طول تحقیقات، تمام روابط موجود بین اقدامات پایه‌ی توسعه‌ی نرم افزار ISO/IEC15504-5 و کنترل‌های امنیتی ISO/IEC 27002 مورد تجزیه و تحلیل قرار گرفته و افزوده‌ی امنیتی ISO/IEC15504 توسعه یافت. تغییراتی که شرکت ها باید برای اجرای موفقیت آمیز کنترل های امنیتی مرتبط در فرایندهای چرخه‌ی عمر نرم افزار ایجاد کنند، توسط این افزوده بیان می شود.

طول مدت رفع یک خطای نرم افزاری پس از نصب صد برابر زمانی است که خطا در مراحل اولیه تولید نرم افزار کشف و اصلاح شود. در مورد نقایص امنیتی این رقم بالاتر نیز خواهد بود، چرا که علاوه بر هزینه فوق، هزینه جبران خسارت ناشی از سو استفاده از این نقایص به منظور سرقت اطلاعات، خراب کاری و سایر حملات نیز بر دوش تولیدکننده سنگینی خواهد کرد.(هاردالد و کراس[3]، ۱۹۹۹)

از مهمترین فعالیت‌ها در تولید یک نرم افزار مستقل، استخراج و تحلیل نیازمندی های آن است. لذا یک روش نیازمند برای تعیین وظایف و مسئولیت های اعضای تیم توسعه نرم‌افزار بوده و هدف آن تولید نرم افزاری با کیفیت در کمترین زمان ممکن و در محدوده بودجه مد نظر است. در چرخه عمر نرم افزارها شناسایی عوامل (تحلیل، طراحی، پیاده سازی، آزمایش، استقرار، پشتیبانی) به بالا بردن زمان حیات کمک خواهد کرد.(بنزنوسوف[4] و کروچن، ۲۰۰۵)

یکی از راهکارهای مواجهه با این چالش ها، تبعیت از چارچوب ها و استانداردهای مدیریت خدمات فناوری

اطلاعات نظیر ISO/IEC15504، بومی‌سازی و پیاده‌سازی تجارب و الگوهای موفق امنیت اطلاعات در فرآیندهای چرخه عمر نرم‌افزاری مبتنی بر استاندارد ISO/IEC15504 بهبود مدیریت خدمات فناوری اطلاعات را فراهم می‌آورند.

[1] Software Development Life Cycle
[2] Doldi and Goudenove
[3] Harold and Krause
[4] Beznosov

دوره ۲، شماره ۲، تابستان ۱۳۹۵، صفحات ۶۰- ۷۰

فرآیند تولید نرم‌افزار که با عنوان «چرخه حیات تولید نرم افزار» نیز شناخته می شود، ساختاری است که روی توسعه و تولید محصولات نرم افزاری اعمال می شود. عبارتهای مشابهی چون «چرخه حیات نرم افزار» و «فرآیند نرم افزار» در این رابطه استفاده می شود. مدلهای گوناگونی نظیر فرآیندهای (خاص) وجود دارند که هر کدام خط مشی مختص (آن فرآیندها) برای انجام کارها و فعالیت های متنوع در طول فرآیندها را مشخص می کنند. در پیاده‌سازی فرآیندهای چرخه عمر نرم‌افزار سیستم‌ها و فرایندهای متعددی وجود دارد که ما در این تحقیق قصد داریم الگوهای موفق امنیت اطلاعات در فرآیندهای چرخه عمر را برگزینیم و بهترین آنها را برای بومی‌سازی این تحقیق به اجرا درآوریم در نهایت به پیاده‌ساز ی تجارب و الگوهای موفق امنیت اطلاعات در فرآیندهای چرخه عمر نرم افزاری مبتنی بر استاندارد ISO/IEC 15504 دست یابیم.(فلرکاس و ساس[1]، ۲۰۰۸)

مبانی نظری

- امنیت و چرخه حیات نرم‌افزار

تولید نرم افزارهای امن با پیروی سازمانهای سازنده از قواعد مهندسی نرم افزار با تاکید بر طراحی و کیفیت مناسب، شامل گذراندن محصول از پروسه های بازرسی، مرور و استفاده از متدهای تست، استفاده مناسب از مفاهیم مدیریت ریسک و مدیریت پروژه میسر می‌شود. برای لحاظ امنیت در چرخه حیات نرم افزار می‌بایست استراتژیهای امنیتی را در این چرخه در نظر گرفت و در نتیجه یک چرخه حیات نرم افزار امن ایجاد نمود.(برتب[2]، ۲۰۰۹)

- امنیت محصول نرم‌افزاری

اطمینان در نرم افزار به معنی عاری بودن محصول نرم افزاری از آسیب پذیری های تعمدی و یا اتفاقی که در هر مرحله از چرخه حیات یک محصول نرم افزاری به آن تزریق می شود، می‌باشد. اطمینان از امنیت محصول نرم افزاری شامل فرآیند(فعالیتها، روشها، روالها) که طی آن نیازهای امنیتی یک محصول نرم افزاری تضمین و برآورده می شود، و همچنین شامل راهکارهایی برای هر یک ازفازهای چرخه حیات نرم افزار، از مهندسی نیازها تا نگهداری می‌باشد. .(برتب، ۲۰۰۹)

- اهمیت امنیت و لحاظ آن در چرخه تولید نرم‌افزارها

زمان مورد نیاز برای رفع یک خطای نرم افزاری پس از نصب، چندین برابر زمانی است که خطا در مراحل اولیه تولید نرم افزار کشف و اصلاح شود. هزینه جبران خسارت ناشی از سوء استفاده از این نقایص به منظور سرقت اطلاعات، خراب کاری و سایر حملات نیز بر دوش تولیدکننده نرم افزارخواهد بود. امنیت باید از ابتدای ساخت سیستم و قسمتی از مشخصات و نیازمندی های کاربردی سیستم در نظر گرفته شود. مشتریان و مدیران پروژه گاهی فکر می کنند امنیت جزئی ذاتی از سیستم است. آنها ممکن است از تاکید بر امنیت، طفره روند و آن را به دید فرآیندی که باید به مرور و در طول توسعه نرم افزار تکمیل شود، نگاه کنند. واقعیت این است که هرچه توسعه نرم افزار تکمیل‌تر شود، اعمال و در نظر گرفتن امنیت پر هزینه‌تر و زمانبرتر خواهد بود. تحلیل خودکار کد برای کشف ایرادهای امنیتی نرم افزار در مراحل اولیه تولید (ارزیابی امنیتی یک قطعه برنامه، پیش ازتکمیل به صورت یک برنامه کاربردی کامل) ، آزمون نفوذ جهت تست های مهم امنیتی که اجرای آن در مراحل نهایی و بر روی نسخه تکمیل شده

[1] Flechais and Sasse
[2] Brotb

نرم افزار کاربردی قابل اجرا است، از جمله موارد امنیتی هستند که باید در ایجاد چرخه نرم افزار لحاظ گردند. (بنزنوسوف و کروچن[1]، ۲۰۰۵)

- امنیت در پیاده‌سازی نرم‌افزار

امنیت در پیاده‌سازی نرم افزار شامل مراحل زیر می‌باشد:

تعیین استانداردهایی برای اطمینان از امنیت کد مانند استفاده از رمزنگاری یا اعتبارسنجی داده ورودی. (در واقع هر برنامه نویسی مسئول امنیت کد خود می باشد).

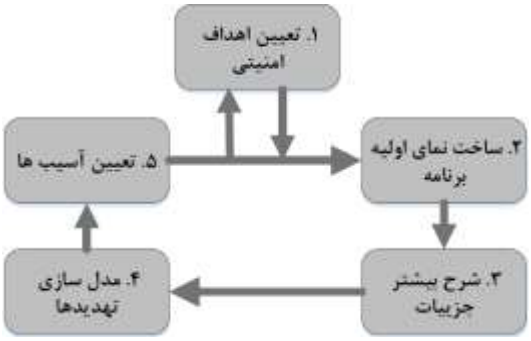
پویش کد که بسته به ابعاد برنامه کاربردی و تعداد نقیصه های امنیتی کشف شده در اولین پویش کد، احتمات آزمون امنیت به چند بار تکرار نیاز خواهد داشت، تا از صحت رفع اشکال با حفظ کارکرد برنامه اطمینان حاصل شود. گروه تضمین کیفیت وظیفه پویش آسیب پذیریها را بر عهده دارد.

تست مجدد برنامه حین ساخت آن، این مسأله از این جهت حایز اهمیت می باشد که ممکن است در تولید بخشهای جدید یا یکپارچه نمودن آنها آسیب‌پذیریهایی به وجود آید که پیشتر وجود نداشته اند. البته جهت تحقق زمان‌بندی پروژه، باید تعادل لازم را رعایت نمود.

برای مواردی که برنامه نویسان از دانش کافی در زمینه امنیت برخوردار نیستند، می توان از متخصصین بیرونی استفاده نمود. گرچه این روش خطر آگاهی یک گروه بیرونی از معماری نرم افزار را به همراه دارد.

- مراحل مدل‌سازی تهدیدها

مدل‌سازی تهدید فرآیندی برای کمک به مدل کردن امنیت برنامه و استخراج نقص‌ها و آسیب‌های بالقوه موجود در برنامه، قبل از سرمایه گذاری بر روی آنها می‌باشد. چرخه فرآیند مدل سازی تهدید شامل اضافه کردن جزئیات بیشتر با پیش رفتن در حلقه ساخت برنامه، تشخیص منابع کلیدی، بررسی و تصحیح مجدد می‌باشد. که شامل مراحل زیر است.(فلاکس[2]، ۲۰۰۵)



شکل(۱): مراحل مدل‌سازی تهدیدها

[1] Beznosov, K., Kruchten
[2] Flechais

پیشینه پژوهش

یکی از فرآیندهایی که برای تولید و توسعه امن نرم افزارها ارائه شده است توسط CLASP در سال ۲۰۰۶ است. هدف این فرآیند، فراهم کردن مجموعه ای از مؤلفه‌های فرآیندی مبتنی بر فعالیت و نقش است که هسته اصلی آن شامل تجربیاتی موفق برای افزودن امنیت به فرآیندهای موجود یا جدید توسعه نرم افزار می باشد و در قالب ۲۴ فعالیت ارائه شده است. در واقع یک فرآیند نیست بلکه مجموعه ای از فعالیتهاست که می تواند در فرآیندهای دیگر مورد استفاده قرار گیرد. شرکت مایکروسافت نیز تجربیات خود در زمینه امنیت نرم افزارهای تولیدی را جمع بندی و نهایتاً فرآیندی برای مهندسی ارائه کرده است. این فرآیند مرور گردید و آخرین نسخه آن در اوایل سال ۲۰۰۶ ارائه شد. این فرآیند از سیزده مرحله تشکیل شده است که به صورت ترتیبی اجرا شده و چرخه حیات نرم افزار را پوشش می دهد. در عین انسجام فعالیت ها در این فرآیند و قابلیت اجرای عملی آنها، فاقد مرحله ای برای تحلیل نیازمندی‌های امنیتی است، در صورتی که این موضوع، یکی از قسمت های اصلی فرآیند توسعه نرم افزار را تشکیل می دهد. همین امر باعث شده تا به جنبه های مختلف امنیتی توجهی نشود و تحلیل و بر طرف کردن بسیاری از مخاطرات و آسیب پذیری ها، و شناخت انواع تهدیدها در این فرآیند انجام نپذیرد. محصول تولید شده توسط این فرآیند، با وجود اینکه برخی از آسیب پذیری های امنیتی در آن بسیار کم است، مخاطرات دیگری را ممکن است در بر داشته باشد. برای ایمن سازی فرآیند های چابک، مایکروسافت در ضمیمه خود، رهنمودهایی را ارائه کرده است که طی آن چرخه تولید امن نرم افزار مایکروسافت شکل گرفت.

در مطالعه‌ای که توسط تعطیلی و مدیری در ۲۰۱۵ انجام شده است این موضوع مطرح گردیده است که شناخت جایگاه امنیت نرم افزار هر سازمان باید در فاز طراحی لحاظ گردد و این امنیت باید مطابق با چارچوبهای سازمان باشد. این مطالعه بر اساس ISO/IEC 27304بیان می‌دارد که توسعه دهندگان نرم افزار باید از اصول مشترک امنیتی نرم افزاری در تمامی فازها به طور مداوم استفاده نمایند و باید در طول فرآیند پیاده ساز ی به چارچوب امنیتی سازمان مراجعه نمایند.(گیلانی و مدیری[1]، ۲۰۱۵)

مدل استاندارد ISO/IEC15504

هدف این استاندارد، ارائه یک چارچوب جهت ارزیابی روند نرم افزار می‌باشد که در نسخه اول SPICE در سال ۱۹۹۵ منتشر گردید. همچنین این استاندارد برای مدیریت کردن نسخه‌های آزمایشی استاندارد در حال ظهور که بر اساس ISO/IEC15504 هستند و برای ترویج ارزیابی روند نرم افزار در صنعت نرم افزار می‌باشد.

ویژگیهای اصلی و منحصر به فرد ISO/IEC15504

مدلی برای مدیریت فرآیند می‌باشد. یک مجموعه‌ای از نیازمندیها و دستورالعمل ارزیابی فرآیند توسعه نرم افزار شرکتها را ایجاد می‌نماید. و این فرآیندها را بهبود می‌بخشد.(بنزنوسوف[2]، ۲۰۰۳)

فرآیندها می‌توانند به پنج دسته تقسیم گردند:

- مشتری/ تولیدکننده

[1] Gilani and Modiri

[2] Beznosov

- مهندسی
- پشتیبانی
- مدیریت
- سازمان (رفرنس PDF به نام COMPARING).

تحلیل مقایسات ISO/IEC15504 با سایر استانداردها

به نظر می رسد که SPICE رویکرد بسیار جامع است که در آن تمام قسمت های کسب و کار و نرم افزار پوشش داده شده است. همچنین به اندازه کافی برای ایجاد دامنه های فرعی مانند SPICE خودرو انعطاف‌پذیر می‌باشد. یکی دیگر از ویژگی‌های جالب SPICEارزیابی گسترده و ابزار بهبود برای هر دو گروه تأمین کنندگان و ارائه دهندگان است. این طرح یک رویکرد جهانی، آشنا و قابل اعتماد برای مدیریت شیوه های توسعه نرم افزار ارائه می دهد. SPICE بهبود کیفیت نرم افزار را از طریق کمک به خریداران نرم افزار برای مشخص کردن سطح توانایی تأمین کنندگان نرم افزار و میزان سازگاری تامین کنندگان با استانداردهای بین المللی، تضمین می‌کند. یک روش خوب برای مدیریت توسعه نرم افزار است. (بنزنوسوف وکروچن، ۲۰۰۵)

طرح بومی پیشنهادی

در این بخش برای بومی‌سازی روشهای شناخته شده SDLC مایکروسافت و سیسکو، به منظور ایجاد انطباق میان آنها و نیازهای امنیتی سازمانهای انتخاب شده در ایران مطابق با ISO/IEC15504 پرداخته شده است. برای این منظور چندین حالت وجود دارد، یا کنترل امنیتی مطرح در دو روش مورد نظر در ISO/IEC15504 به طور کامل وجود دارد (بدون تغییر)، یا وجود دارد و نیاز به تغییر دارد (تجدیدنظر) و یا اصلاً موجود نیست (ایجاد).

در حالت اول، برای پشتیبانی کامل از کنترل امنیتی به هیچگونه اصلاح یا تقویت نیاز نداریم و می توانیم مستقیماً از کنترل مربوطه استفاده نماییم.

در حالت دوم به منظور اصلاح یا گسترش کنترلهای مربوطه، می توان روشهای اصلی ISO/IEC15504 را گسترش داد تا تمام جنبه های کنترل امنیتی مورد نظر را در بر بگیرد.

در حالت سوم، فرآیند ISO/IEC15504 مربوطه هیچ روش اصلی ویژه ای ندارد که کنترل امنیتیی را تحت پوشش قرار دهد بنابراین، ایجاد یک کنترل جدید ضروری است.

آیین‌نامه امنیت نرم‌افزار مرکز فناوری اطلاعات و فضای مجازی دانشگاه تهران

به جهت بومی‌سازی الگوهای موفق امنیت نرم افزار، به صورت موردی مرکز فضای مجازی دانشگاه تهران و شورای عالی فضای مجازی را به عنوان سازمان مورد بررسی در نظر گرفته‌ایم.

دوره ۲، شماره ۲، تابستان ۱۳۹۵، صفحات ۶۰- ۷۰

بومی سازی SDLC سیسکو

در مدل موفق چرخه حیات توسعه امن نرم افزار سیسکو مراحل آن یک به یک بر اساس ISO/IEC15504 مطابقت داده شده و سپس بر اساس آیین نامه سازمان بومی می‌گردد:

۱- نیازمندی های امنیتی محصولات شامل نیازمندیهای داخلی و نیازمندی‌های مبتنی بر بازار با ENG.1.2.BP1: مشخص نمودن نیازمندیهای نرم افزار. تعیین و تجزیه تحلیل نیازمندی‌ها ، از اجزای نرم افزاری سیستم و مستند نمودن آن در نیازمندیهای نرم افزار به صورت خاص، مطابقت دارد .که البته برای تکمیل می بایست در آن عبارت" تعیین نیازمندی های امنیتی محصولات شاملنیازمندیهای داخلی و نیازمندیهای مبتنی بر بازار "را اضافه نمود(دلدی و گدینوف، ۱۹۹۵).

۲- امنیت نرم افزار شخص ثالث . که از طریق مخزن مرکزی با مالکیت معنوی و هشدار تهدیدات نرم افزار شخص ثالث و آسیب پذیری ایجاد میگردد را میتوان باORG.4.BP1 : شناسایی نرم افزار مورد نیاز محیط مهندسی شامل تعیین الزامات مورد نیاز برای نرم افزار مهندسی، شناسایی نقش ها و فعالیت های فرآیند، مسائل مربوط به امنیت آن، توان به اشتراک گذاری داده های مورد نیاز، تهیه پشتیبان و امکانات دسترسی از راه دور، مرتبط دانست .در این فرآیند باید عبارت"برقراری امنیت نرم افزار کمکی (جانبی)، از طریق ایجاد مخزن مرکزی با مالکیت معنوی و هشدار تهدیدات و آسیب پذیری نرم افزار کمکی"، را اضافه نمود تا فرآیند این مرحله را به طور کامل پوشش دهد (.دلدی و گدینوف، ۱۹۹۵).

۳- طراحی امن که شامل طراحی با امنیت در ذهن از طریق استفاده از استاندارد ها در محصولات امنیتی، آگاهی از روش های حمله و روشهای طراحی در برابر آنها ، بهره برداری کامل از طرح ها و کتابخانه‌هایی که به عنوان طرحها و کتابخانه‌های امن شناخته شده‌اند، در نظر گرفتن همه نقاط ورود و شناسایی نقاط ضعف در طراحی می‌باشد و همچنین استفاده از مدل سازی تهدیدات برای اعتبارسنجی امنیت طراحی، دنبال نمودن جریان داده ها از طریق سیستم، شناسایی مرز اعتماد که در آن داده ها ممکن است به خطر بیافتد، ایجاد یک لیست از تهدیدها با استفاده از یک پایگاه داده تهدیدات شناخته شده، متناسب با نوع محصول، اولویت بندی و اجرای تهدیدات شناخته شده به دست می‌آید با ENG.1.3.BP1 : توسعه طراحی معماری نرم افزار که تبدیل نیازمندیهای نرم افزار به یک معماری نرم‌افزار سطح بالا است مرتبط می‌باشد . برای طراحی امن در این معماری سطح بالا باید عبارت" استفاده از استانداردهای امنیتی، آگاهی از روش های حمله و روشهیای طراحی در برابر آنها ، بهره‌برداری طرحها و کتابخانه‌های امن، شناسایی نقاط ضعف در طراحی، استفاده از مدل سازی تهدیدات برای اعتبارسنجی امنیت طراحی، دنبال نمودن جریان داده‌ها از طریق سیستم، شناسایی مرز اعتماد داده‌ها، ایجاد لیست تهدیدها در پایگاه داده تهدیدات و اولویت بندی و اجرای تهدیدات شناخته شده"، را اضافه نمود (دلدی و گدینوف، ۱۹۹۵)

۴- برنامه نویسی امن به معنای اطمینان از مقاوم بودن کد در برابر تهدیدات با آموزش برنامه نویسی امن، استفاده از کتابخانه ها و دستورالعمل های امن، بازنگری و تحلیل کد و پیروی از استانداردهای برنامه نویسی امن است . این مرحله مربوط به ENG.1.4 فرآیند ساخت نرم افزار می‌باشد .به دلیل این که در فرآیند ساخت نرم افزار در 15514 این موارد لحاظ نشده است لذا با ایجاد : ENG.1.4.BP5 برنامه نویسی امن، اطمینان از مقاوم بودن کد در برابر تهدیدات با آموزش برنامه نویسی امن، استفاده از کتابخانه ها و دستورالعملهای امن، بازنگری و تحلیل کد و پییروی از استانداردهای برنامه نویسی امن، این مرحله نیز به طور کامل لحاظ می گردد. ENG.1.4.BP5: برنامه نویسی امن .اطمینان از مقاوم

بودن کد در برابر تهدیدات با آموزش برنامه‌نویسی امن، استفاده از کتابخانه‌ها و دستورالعملهای امن، بازنگری و تحلیل کد و پیروی از استانداردهای برنامه نویسی امن، آموزش موارد امنیتی به مدیران و نگهداران نرم افزار.

۵- تجزیه و تحلیل امن شامل بازدیدهای امنیتی، بررسی هشدارهایی که تولید می شوند، و حل مسائلی سرریز بافر، ورودی آلوده و سرریز عدد صحیح می باشید. این مرحله با ENG.1.6.BP3 : تست نرم‌افزار یکپارچه، تست نرم افزار یکپارچه در برابر معیارهای امنیتی و ثبت نتایج مطابقت دارد .در این فرآیند باید عبارت " بررسی هشدارهای امنیتی، سرریز بیافر، ورودی آلوده و سرریز عدد صحیح "اضافه گردد (دلدی و گدینوف، ۱۹۹۵)

۶- تست آسیب پذیری شامل تست پروتکل، پورتها و سرویسهایی که به طور پییش فرض فعال هستند، بررسی توانایی یک محصول و مقاومت در برابر حملات استحکام پروتکل، ابزار های هک متن باز، اسکن برنامه‌های کاربردی تحت وب می باشد .این مرحله با ENG.1.6.BP3 : تست نرم افزار یکپارچه، تست نرم افزار یکپارچه در برابر معیارهای امنیتی و ثبت نتایج، مرتبط است . فقط باید در این فرآیند عبارت " تست پروتکل، پورتها و سرویسهایی که به طور پیش فرض فعال هستند و یا یک پیکربندی معمول مشتری استفاده خواهد شد، بررسی توانایی یک محصول و مقاومت در برابر حملات استحکام پروتکل، ابزار های هک متن باز، اسکن برنامه های کاربردی تحت وب" ، را نیز اضافه نمود.

بررسی موردی: ارزیابی طرحهای بومی شده در مرکز فناوری اطلاعات دانشگاه تهران و مرکز فضای- مجازی

در مرکز فناوری اطلاعات و مرکز فضای مجازی دانشگاه تهران، در بخش امنیت، آیین نامه های امنیتی برای امنیت اطلاعات، امنیت شبکه، امنیت فیزیکی و امنیت نرم افزار وجود دارد .به دلیل این که طرح ما در خصوص امنیت نرم افزار است، طرح بومیشده را در خصوص بخش امنیت نرم افزار به اجرا در آورده ایم. این سازمان تا کنون برای ارتقا امنیت محصولات نرم افزاری خود از هیچ استاندارد خاصی پیروی ننموده است و مفهوم بومی سازی در حوزه نرم افزار مربوط به فارسی‌سازی نرم افزار و رابطه‌ای نرم افزاری بوده است.

این سازمان برای ایجاد چرخه توسعه امن نرم افزار تا کنون اقدامی انجام نداده است .اما از آن جا که بومی‌سازی نرم افزار یکی از اقدامات این سازمان است، با ارائه این طرح این سازمان قادر است تا در روند بومی‌سازی نرم‌افزار از چرخه توسعه امن نرم افزار بومی شده مطابق با آیین نامه های امنیتی خود استفاده نماید. برای این منظور مدیر فناوری اطلاعات مسئول است تابداند چه اقداماتی برای ایجاد چرخه توسعه امن نرم افزار باید صورت پذیرد تا با توصیف نقشها و مسؤلیتهای تیم توسعه دهنده نرم افزار، آنها را برای برای ارتقای امنیت محصولات بومی توجیه نماید .تیم توسعه دهنده میبایست روندهای لازم برای توصیف دقیق فعالیت خود را فرا بگیرند و مطابق با طرح پیشنهادی اقدام به بومیسازی نرم افزارهای خود بر اساس چرخه امن بومی شده بنمایند. این روندها باید ساده و قابل اجرا برای تمام انواع نرم افزارهایی باشد که این سازمان اقدام به بومیسازی آنها مینماید. برای مثال ENG.1.3.BP2: رابط طراحی. توسعه و مستندسازی طراحی برای رابط داخلی و خارجی، تنظیم یک Character Set ثابت و استفاده از UTF-8 که در طرح پیشنهادی مطرح گردیده است، می بایست در ایجاد رابطها به زبان فارسی مورد استفاده قرار گیرد و این امر فارسی‌سازی نرم افزار را مطابق با یک استاندارد امکانپذیر می‌سازد. نمونه‌ای از صفحات نرم افزار ایجاد شده با طرح بومی ارائه شده در شکل (۲) نشان داده شده است.

شکل (۲): صفحه کارمندان در نرم افزار بومیشده با استفاده از طرح پیشنهادی

تعیین شاخص‌های ارزیابی

برای ارزیابی روند امنیتی نرم‌افزار موجود و مقایسه با روند پیشین لازم است تا شاخص‌های امنیتی را تعیین نموده و بر اساس موفقییت الگوی خود را مورد تحلیل قرار دهیم شاخص‌های زیر را در نظر گرفته شده است:

۱- حفظ محرمانگی اطلاعات
۲- دسترس پذیری که شامل دسترسی کاربران مجاز به اطلاعات و سیستم میباشد.
۳- حفظ امنیت پایگاه داده
۴- کنترل دسترسی
۵- عدم دسترسی به سیستم و اطلاعات در خارج از سازمان

آزمون‌های نفوذپذیری

حال که در ساخت و بومی‌سازی نرم افزارهای سازمان، از چرخه توسعه امن نرم افزار بومی، استفاده شده است نوبت به آزمونهای نفوذ پذیری میرسد .چرا که حتی در صورت پرداخت هزینه واستقرار چنین نرم‌افزارهایی نیاز به کنترل دائمی امنیت وجود دارد. آزمون نفوذپذیری به منظور کشف و رفع آسیب پذیریهای امنیتی سیستمها، به شبیه‌سازی آنچه نفوذ گران واقعی انجام می دهند در قالبی قانونمند می پردازد. تست نفوذ را از دیدگاه میزان اطلاعاتی که در اختیار تیم نفوذ قرار دارد، می توان به سه دسته White-Box، Black-Box و Gray-Box و از دیدگاه مکان انجام تست نفوذ می توان به Internal و External تقسیم نمود.

تست نرم‌افزار

برای ارزیابی نرم افزاری که توسط چرخه‌های بومی شده توسعه امن نرم افزار ساخته شده است، از هر سه روش تست جعبه سفید، جعبه سیاه و جعبه خاکستری، برای نرم افزار بومی شده و غیر بومی استفاده نموده ایم .برای این کار نرم افزار را بدون ایجاد توسط چرخه امن بومی و پس از ایجاد نرم افزار با پیروی از چرخه‌های توسعه امن نرم افزار بومی شده مورد آزمون و تست قرار داده‌ایم.

برای ارزیابی نتایج حاصله و دسته‌بندی های امنیتی در دو نرم افزار، از روش استاندارد امتیازدهی CVSS[1] استفاده نموده‌ایم .به این ترتیب که ابتدا با توجه به شاخص‌های مطرح شده که میزان موفقیت نرم افزار در امنیت را تعیین می‌نماید، هر دو نرم افزار را بر اساس کلیه آسیب پذیری‌های مطرح شده توسط CVSS که مربوط به حفظ محرمانگی اطلاعات، دسترس پذیری، حفظ امنیت پایگاه داده، کنترل دسترسی و عدم دسترسی به سیستم و اطلاعات در خارج از سازمان بود، تست نمودیم .بر اساس نتایج حاصل از تست هر دو نرم افزار، آسیب‌پذیریهای موجود در دو نرم افزار را مشخص کردیم .

لیست آسیب پذیریها در نرم افزار اصلی

CVE-2014-4248: اجازه می دهد تا کاربران محلی بتوانند از طریق بردار ناشناخته مربوط به ورود به سیستم، محرمانگی را تحت تأثیر قرار دهند.

CVE-2014-2485: اجازه می دهد تا کاربران محلی بتوانند محرمانگی را از طریق بردارهای ناشناخته مربوط به یکپارچه سازی خدمات کسب و کار، تحت تأثیر قرار دهند.

310 CVE-2009-2752: کاربران محلی میتوانند با شکستن مکانیزم رمزنگاری به اطلاعات حساس دسترسی پیدا کنند.

200 CVE-2013-1615: امکان دسترسی مهاجمان به اطلاعات حساس از طریق web-GUI API را ایجاد مینماید.

264 CVE-2015-3179 :اجازه می دهد تا کاربران تأیید شده بتوانند از راه دور محدودیتهای login را با دسترسی به حساب کاربری تأیید نشده معلق، دور بزنند.

264 CVE-2015-1905 : اجازه می دهد تا کاربران تأییدشده محدودیت دسترسی در نظیر گرفتیه شده را از طریق بردارهای نامشخص دور بزنند.

119 CVE-2015-4390: اجازه می دهد تا کاربران محلی از طریق بردارهایی شامل مکان های حافظه به هم پیوسته اقدام به منع سرویس نمایند.

89 CVE-2015-4658: به مهاجمان اجازه میدهد از راه دور برای اجرای دستورات دلخواه SQL از طریق پارامتر USR یا PWDاقدام نمایند.

CVE-2007-1281: به مهاجمان امکان اجرای دستورات دلخواه SQL از طریق پارامتر cate_id را می‌دهد.

CVE-2007-1297: به مهاجمیان امکیان اجیر ای دستورات دلخیواه SQL از طریق پیارامتر USER_ID را میدهد.

CVE-2010-0612: تأثیر بردارهای حمله ناشناخته، در ارتباط با حق دسترسی به فایل.

284 CVE-2015-0675: عدم انجام صحیح تبدیل نوع، که موجب میشود تا کاربران تصدیق شده، از راه دور بتوانند به علت محرومیت از خدمات(عدم دسترسی به حافظه) به اجرای کد دلخواه از طریق یک فایل دستکاری شده بپردازند.

CVE-2014-2406: کاربران تأییدشده میتوانند بر روی محرمانگی، یکپیارچگی و دسترس پذیری تأثیر بگذارند.

20 CVE-2015-5386: دور زدن احراز هویت و به دست آوردن دسترسی مدیریتی از طریق درخواست‌های HTTP نامشخص.

لیست آسیب‌پذیری‌ها در نرم افزار تولیدشده با مدل بومی پیشنهادی

[1] Common Vulnerability Scoring System

CVE-2014-4248: اجازه می دهد تا کاربران محلی بتوانند از طریق بردار ناشناخته مربوط به ورود به سیستم، محرمانگی را تحت تأثیر قرار دهند.

200 CVE-2013-1615 :امکان دسترسی مهاجمان به اطلاعات حساس از طریق web-GUI API را ایجاد مینماید.

119 CVE-2015-4390: اجازه می دهد تا کاربران محلی از طریق بردارهایی شامل مکان های حافظه به هم پیوسته اقدام به منع سرویس نمایند.

89 CVE-2015-4658: به مهاجمان اجازه میدهد از راه دور برای اجرای دستورات دلخواه SQL از طریق پارامتر USR یا PWDاقدام نمایند.

CVE-2007-1281: به مهاجمان امکان اجرای دستورات دلخواه SQL از طریق پارامتر cate_id را می‌دهد.

CVE-2007-1297: به مهاجمان امکان اجرای دستورات دلخواه SQL از طریق پارامتر USER_ID را می‌دهد.

نتیجه‌گیری

این مقاله با هدف ایجاد طرحی بومی برای تولید امن نرم افزار با توجه به استانداردهای امنیت نرم افزار صورت پذیرفته استبرای تحقق هدف مقاله معرفی الگوهای چرخه توسعه امن نرم افزار ، الگوهای شناخته شده و موفق مایکروسافت و سیسکو را انتخاب نمودیم .سپس ایمنی ISO/IEC15504 را با تطبیق هریک از الگوها افزایش دادییم .درادامه سازمان مورد نظر را انتخاب نمودیم و بر اساس آیین نامه امنیت نرم افزار های سازمان، چرخه های توسعه امن نرم افزار معرفی‌شده را بر اساس ISO/IEC15504 بومی نمودیم .در انتها به ارزیابی طرح ارائه

شده پرداختیم .با توجه به بازخوردی که از بخش امنیت سازمان دریافت کردیم، و با توجه به نتیایج اعلام شده توسط این سازمان در مورد آسیب پذیری نرم افزار بومی، می توان گفت که فرآیند ساخت نرم افزار بر اساس طرح بومی‌شده سازمان موجب کاهش آسیبپذیری نرم افزار گردیده است و تعداد آسیب پذیریها از 14 به 6 رسیده است .

منابع

1. Afsaneh Gilani, Nasser Modiri2, Alireza Nikravanshalmani, “Developing a New Approach to the Architecture, Design and Secure Implementation of Web Applications”, International Journal of Computer & Information Technologies, 2015, pp 73-81.
2. Beznosov, K., 2003, "*Extreme Security Engineering: OnEmploying XP Practices to Achieve ‘Good EnoughSecurity’ without Defining It."*, First ACMWorkshop on Business Driven SecurityEngineering, pp 343-349.
3. Beznosov, K., Kruchten, P., 2005,*" Towards AgileSecurity Assurance"*,In Proceedings of the 2004Workshop on New Security Paradigms, pp 6-12.
4. Doldi, L., Goudenove, F.,1995,*"Use of SDL to specifyAirbus future air navigation systems"*, SDL95 Forum, Oslo, pp 8-16.
5. Flechais, I., 2005,*"Designing Secure and UsableSystems"*, Ph.D. Thesis, University of London,London, UK, pp 142-145.
6. Flechais, I., Sasse, M. A., Hailes, S. M. V., 2003, "*Aprocess for developing secure and usable systems"*,In Proceedings of the 2003 Workshop on NewSecurity Paradigms, pp 9-15.

7. Harold ,F.,Krause,M.,1999,*" Information Security Management"*,Handbook, 4th Edition, Vol. 3" pg. 580.
8. Krag Brotb, W., 2009, "*Information Security Management Metrics : A Definitive Guide to Effective Security Monitoring and Measurement* ", CISM / CRC Press , Taylor & Francis Group ,an informa business, pp 16-22.
9. Seacord, R. C., Plakosh, D.; & Lewis, G. A., 2003,*"Modernizing Legacy Systems: Software Technologies, Engineering Process and Business Practices"*. Addison-Wesley Longman Publishing Co.,Inc, pp 15-21.