

روش آرایه رمزگذاری شده در محافظت از عامل در برابر حملات میزبان در سیستم‌های تجارت الکترونیک مبتنی بر عامل

حمیدرضا افتخاری

عضو هیات علمی دانشگاه ملایر

Eftekhari@malayeru.ac.ir

چکیده: عامل‌های متحرک به عنوان یکی از مدل‌های بکارگرفته شده در برنامه‌های توزیع شده مانند تجارت الکترونیک و بازیابی اطلاعات معرفی شده و در چنین برنامه‌هایی موفق تر از مدل‌های سنتی انتقال کد و داده همچون سرویس دهنده/ مشتری عمل نموده است. با این حال بدلیل چالش امنیتی که در این مدل وجود دارد، نتوانسته است آنگونه که باید گسترش یابد. این چالش بیشتر در حوزه تهدیدهای میزبان برای عامل است که هنوز یک راه حل کامل برای این موضوع وجود ندارد. آنچه در این مقاله به آن اشاره شده است، روشی است جدید جهت تعامل امن یک عامل با میزبانها بر اساس رمزگذاری نامتقارن که موجب می گردد داده‌ها و مسیر حرکت عامل از تهدیدهای میزبان بدخواه محفوظ بماند.

کلمات کلیدی: عامل‌های متحرک، تجارت الکترونیک، امنیت در سیستم‌های مبتنی بر عامل، محافظت از عاملها، حملات میزبان به عامل

۱. مقدمه

۱- عدم وجود یک زبان برنامه نویسی خاص

۲- عدم وجود استاندارد کامل برای محیط‌های مبتنی بر عامل

۳- نبود یک راه کار استاندارد و کامل امنیتی

غالب محیط‌های مبتنی بر عامل از زبان جاوا به دلیل مستقل از سکو^۱ بودن آن استفاده می نمایند. اما در واقع این زبان نمی تواند به عنوان زبان کاملاً مناسب برای محیط‌های مبتنی بر عامل عمل نماید خصوصاً آنکه از انتقال قوی^۲ پشتیبانی نمی نماید.

همچنین در مورد دوم، اگرچه دو استاندارد MASIF و FIPA تعریف شده، اما هنوز در تعامل میان سیستم‌های مبتنی بر عامل فقدان استاندارد مشاهده می شود.

اما سومین نقطه ضعف، موضوعی است که بیش از دیگر موارد موجب جلوگیری از گسترش محیط‌های مبتنی بر عامل شده است.

عاملها به عنوان قطعه کدی که در میزبان‌ها اجرا می شوند می توانند رفتارهایی شبیه ویروس و کرم را پیاده سازی کنند و از این طریق بتوانند امنیت میزبانهای پذیرنده خود را تهدید

عامل‌های متحرک^۱ به عنوان یکی از مدل‌های انتقال کد و داده در کنار مدل‌های سرویس دهنده/ مشتری^۲، اجرای دور^۳ و تقاضای کد^۴ مطرح می باشد و با توجه به ویژگی‌های خاصی که در افزایش کارایی نسبت به مدل‌های سنتی خصوصاً سرویس دهنده/ مشتری دارد، جایگزین مناسبی به جای مدل مذکور است. عاملها با مهاجرت از یک میزبان به میزبان دیگر و جمع آوری نتایج اجرای برنامه‌های خود باعث کاهش بار شبکه می گردند. همچنین قابلیت تصمیم گیری یک عامل مستقل، موجب می گردد تا نیاز به ارتباط همزمان با فرستنده خود مرتفع گردد. ازاینرو اجرای غیرهمزمان و کاهش درگیری فرستنده عامل را به همراه دارد، لذا امکان بهره گیری از عاملها را در سیستم‌های بلادرنگ^۵ به عنوان یک مدل مناسب فراهم می گردد [۱]. این مزایا باعث شده است تا از عامل‌های متحرک بتوان در کاربردهایی همچون تجارت الکترونیک، بازیابی اطلاعات، مدیریت شبکه، سیستم‌های توزیع شده و پردازش‌های موازی استفاده کرد. اما در کنار مزایای عنوان شده نقاط ضعف سیستم‌های مبتنی بر عامل مانعی برای عمومی شدن این مدل شده است. در یک دسته بندی این نقاط ضعف را می توان به سه دسته ذیل تقسیم نمود [۲]:

حملات دیگر، حملات میزبان به عامل می باشد. این حملات از مهمترین چالش های امنیتی محیط های مبتنی بر عامل است. به این دلیل که در خصوص حملات عامل به میزبان با توجه به شباهت آن به حملات ویروس ها و کرم ها؛ همچنین روشهای مختلف اجرای مطمئن کد بارگذاری شده در یک میزبان همچون Aplet ها، ActiveX ها و موارد دیگر توسط روشهای مختلفی که تحقیقات کافی در طول زمان بر روی انجام شده است قابل کنترل و مدیریت است. از سوی دیگر ارسال پیام میان عاملها و تهدیدهای عامل برای عامل نیز اولاً به دلیلی که هیچ کدام از دو طرف بستر اجرایی دیگری نمی باشند حساسیت کمتری نسبت به حملات میزبان و عامل دارد، ثانیاً استانداردهای لازم برای این ارتباط خصوصاً از طریق امضا دیجیتال فراهم شده است. اما حملات یک میزبان به یک عامل موضوعی جدید است که هنوز راه کار کامل و جامعی درباره آن ارائه نشده است. هر چند تلاشهای زیادی در مقابله با این حملات صورت گرفته و می گیرد [۴]، لیکن هر کدام از روشها بخشی از تهدیدهای این حوزه را مورد بررسی قرار می دهند.

بدلیل اینکه یک میزبان بستر اجرایی عامل می باشد، لذا بالاترین سطح دسترسی را به کدها و داده های عامل دارد. این سطح دسترسی در صورتی که کنترل نشود می تواند تا جایی پیش رود که یک عامل سالم با ورود به میزبان بدخواه به یک عامل بدخواه جهت حمله به میزبان های دیگر تبدیل گردد. این حملات می تواند یکی از حملات ذیل باشد: [۳]

تقابگذاری: میزبان بدخواه با معرفی خود به عنوان یک میزبان امن، عامل را قانع می کند تا اطلاعات حساس خود را در اختیارش بگذارد. عامل فریب خورده خود می تواند یک واسطه ای جهت دریافت اطلاعات از میزبان های دیگر امن باشد. با این عمل سرویس امنیتی احراز هویت نقض می گردد. این حمله مقدمه ای برای حملات سرقت و تغییر می باشد که در ادامه اشاره خواهد شد.

عدم پذیرش سرویس: میزبان بدخواه می تواند با آزاد نمودن برخی منابع مورد نیاز یک عامل که در آن میزبان درحال اجراست، فرآیند اجرای آنرا به بن بست بکشاند. حتی می تواند آن عامل را به گونه ای هدایت کند که عامل بدون اطلاع برنامه اش پایان پذیرد و نابود گردد.

سرقت: هر گونه دسترسی بدون مجوز به داده ها و حالت برنامه عامل یک سرقت می تواند محسوب گردد. گاهی اوقات داده های حمل شده توسط عامل دارای اهمیت بالایی هستند، به

نمایند. از سوی دیگر میزبانهای بدخواه^۱ نیز می توانند سرقت کننده اطلاعاتی باشند که توسط عاملها در سطح شبکه منتقل می گردند.

در بخش دوم به مشکلات امنیتی سیستم های مبتنی بر عامل و خصوصاً حملات میزبان به عاملها اشاره خواهد شد و در بخش سوم کارهای انجام شده در موضوع حفاظت از عاملها اشاره می شود و در بخش چهارم و پنجم ایده جدید و مدل کارایی آن و نهایتاً در بخش ششم تحلیل امنیتی و کارایی و در بخش هفتم نتایج عنوان می گردد.

۲. چالشهای امنیتی سیستم مبتنی بر عامل

چالشهای امنیتی موجود در سیستم های مبتنی بر عامل را می توان در سه دسته گنجانند: حملات عامل به میزبان، حملات میزبان به عامل و حملات عامل به عامل [۳]. اگرچه در برخی منابع حمله میزبان به میزبان نیز آورده شده است، اما این نوع حمله بیش از آنکه یک چالش امنیتی سیستم های مبتنی بر عامل محسوب شود یک مساله امنیتی سنتی است که در مدل سرویس دهنده/ مشتری تا کنون به تفصیل به آن پرداخته شده است. در حملات عامل به میزبان، عامل به عنوان یک برنامه اجرایی در میزبان می تواند نقش بدخواه و تهدید کننده منابع میزبان را ایفا نماید. اگرچه در هر میزبانی یک واسطه^۲ مجری کدهای عامل است و این واسطه می تواند دسترسی های مختلف به منابع میزبان را کنترل نماید، اما عامل بدخواه نیز می تواند با نقض سرویس امنیتی احراز هویت^۳ خود را به عنوان یک عامل امن معرفی نماید و اهداف شوم خود را دنبال کند. حملات DoS که موجب اختلال در سرویس دهی میزبان می شود نیز در این دسته می گنجد.

حملات دیگر، حملات عامل به عامل می باشد. ارتباط و ارسال پیام در محیط های مبتنی بر عامل یک ضرورت است تا جایی که استاندارد ACL به منظور نحوه تعامل عاملها با یکدیگر ارائه شده است. اما از آنجایی که در کنار هر ارتباط امن می توان ارتباط غیر مطمئن از عاملهای بدخواه را نیز متصور شد، لذا تهدیدهایی از ناحیه هر عامل برای عامل دیگر وجود دارد که از جمله آنها می توان به نقاب گذاری^۴ اشاره نمود که در این حمله عامل بدخواه خود را به عنوان همکار و یا سرویس دهنده ای که عامل به دنبال آن می گردد، معرفی می نماید. همچنین حملات DoS که می تواند از طریق ارسال پی در پی و بی امان پیام به عامل هدف انجام پذیرد و باعث گردد تا دسترسی پذیری عامل از بین برود.

از *مداخله*، راه کارهایی به منظور عدم امکان وقوع حمله ارائه می گردد و در محدود سازی شبکه، تعامل و مهاجرت عامل صرفا با میزبانهای انجام می پذیرد که از امنیت آنها اطمینان حاصل می گردد و اجازه داده نمی شود که عامل به یک میزبان بدخواه مهاجرت نماید.

در روش *ردیابی/اجر*^{۱۴} تمامی فعالیت های اجرایی یک عامل که در یک میزبان انجام می شود، ثبت^{۱۵} می گردد و فایل رمزگذاری شده ثبت اجرا برای یک مرکز کنترل ارسال می گردد [۶]. عیب این روش حجم بالای فایل ثبت و امکان دستکاری فایل ثبت از سوی میزبان ها می باشد.

در روش تولید کلید *محیطی*^{۱۶} کد عامل رمزگذاری می گردد و فقط با محقق شدن شرایطی خاص در یک میزبان رمزگشایی می گردد. این شرایط خاص می تواند یافتن یک رشته در میزبان باشد. برای عدم دسترسی میزبان های بدخواه و شرایط رمزگشایی، این شرایط به صورت یک طرفه رمزگذاری می شود و امکان رمزگشایی آن وجود ندارد [۷]. مشکل اصلی این روش این است که تضمینی وجود ندارد که بعد از رمزگشایی در یک میزبان که شرایط مذکور در آن محقق شده است، عامل مورد تغییر یا سرقت واقع نشود.

در روش *تابع محاسباتی رمزگذاری* شده^{۱۷} زمانی که عامل می خواهد تابع f را اجرا نماید، تابع $E(f)$ را به میزبان ارائه می دهد که خروجی تابع $E(f(x))$ برابر خروجی $f(x)$ می باشد، از اینرو میزبان بدون اطلاع از تابع واقعی f و ماهیت آن نتیجه مورد نظر را بدست آورده و رمز گذاری می کند برای فرستنده عامل ارسال می نماید [۸]. اگرچه ظاهرا این مدل در ایجاد امنیت کد عامل و عدم سرقت کد مدل خوبی به نظر می رسد، اما دارای محدودیت برای اعمال کلیه توابع می باشد. به بیان دیگر یک مدل عمومی به منظور رمزگذاری توابع مختلف و دلخواه ارائه نداده است. دیگر مشکل این روش عدم محافظت از عامل از نظر تغییر می باشد.

در روش *ارجاع حالت*^{۱۸}، حالات مختلف یک عامل در طول اجرا در میزبانهای امنی که عامل در برنامه سفر خود مرتبا به آنها مهاجرت می کند، بررسی می گردد [۹]. نقطه ضعف مشهود این روش تغییر برنامه سفر عامل توسط یک میزبان بدخواه است که موجب خواهد شد تا در مهاجرت بعدی آن بجای یک میزبان امن به سوی یک میزبان بدخواه دیگر حرکت نماید. از سوی دیگر مهاجرت مرتب یک عامل به سوی میزبان های امن، بیش از آنکه مزایای مدل عملهای متحرک - کاهش بار شبکه و

عنوان مثال در یک تجارت الکترونیک شماره کارت اعتباری و رمز عبور توسط عامل حمل می گردند و هر زمان که عامل به این نتیجه رسید که کالای مورد نیاز با قیمت کمتر را پیدا نموده است، اقدام به خرید آن می نماید. در صورتی که میزبان بدخواه بتواند به هر دلیلی به این اطلاعات حمل شده توسط عامل دسترسی پیدا کند، یک سرقت صورت پذیرفته است. اما این تنها مشکل نیست، بلکه ممکن است حالت عامل که مجموعه متغیرهای داخلی عامل باشند نیز حاوی اطلاعاتی باشد که با استنتاج آنها بتوان به اهداف شومی رسید. به عنوان مثال میزبان بدخواه می تواند مسیر حرکت یک عامل را سرقت نماید، از اینرو خواهد دانست که کدام میزبان ها در طی این برنامه سفر^{۱۹} عامل ملاقات خواهند شد، این مساله می-تواند شرایط یک تقلب را در یک مناقصه الکترونیک فراهم نماید. تمامی این نوع حملات موجب می گردد تا در محرمانگی^{۲۰} عامل اختلال ایجاد گردد.

تغییر یکی دیگر از حملات مخربی که در سیستم های مبتنی بر عامل مطرح می باشد تغییر در داده ها، حالت برنامه یا کد برنامه عامل است. در کنار سرقت اطلاعات حساس یک عامل، ممکن است میزبان بدخواه با تغییر این اطلاعات موجب بروز مشکلات گردد. به عنوان مثال تغییر قیمت های جمع آوری شده در یک مناقصه الکترونیک مبتنی بر عامل باعث خواهد شد که میزبان بدخواه به عنوان برنده مناقصه شناخته شود و یا تغییر در برنامه سفر عامل به صورتی که عامل به سمت میزبان های بدخواه دیگر سفر نماید یا مستقیما به سوی فرستنده روانه شود و ادامه سفر را انجام ندهد. حتی میزبان می تواند محصول مورد نظر را یا تعداد مورد درخواست را تغییر دهد که خود نوعی اختلال در خرید ایجاد می نماید [۵]. مساله بدین جا ختم نمی شود، چرا که تغییر می تواند حتی در کد عامل نیز انجام پذیرد به گونه ای که عامل به عنوان یک جاسوس در میزبان های امن دیگر و خصوصا صاحب عامل مبدل شود و شرایط نفوذ در دیگر میزبان ها را فراهم سازد.

۳. کارهای انجام شده

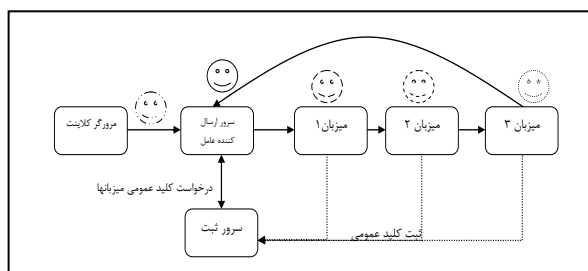
به طور کلی به منظور مقابله با حملات میزبان به عامل سه استراتژی وجود دارد که تمامی راه کارهای ارائه شده در قالب این سه گنجانده می شود. کشف *مداخله*، *جلوگیری از مداخله* و محدود سازی شبکه [۵]

کشف *مداخله* شامل روشهایی است که مداخله میزبان بدخواه را پس از وقوع حمله مشخص می کنند. در روش های *جلوگیری*

است ابتدا به این مساله بپردازیم که اجرای عامل در میزبان با چه هدفی انجام می پذیرد.

در صورتی که خرید الکترونیک را شامل دو بخش بررسی کمترین قیمت و سپس خرید بدانیم می توانیم این دو را که ترتیب اجرایی دارند از هم جدا کنیم [۵]. لذا در یک خرید الکترونیک آنچه انجام می پذیرد فعال سازی یک عامل به منظور گردآوری اطلاعات - قیمت ها- از کالاهای مورد نظر است و در گام بعد برقراری تعامل با میزبان برنده استعلام و خرید از آن توسط خریدار می باشد.

از اینرو در واقع عامل با مهاجرت به یک میزبان ابتدا درخواست خود را مطرح می نماید که این درخواست در ساده ترین شکل خود شامل نوع کالا و تعداد آن می باشد و در مقابل قیمت پیشنهادی میزبان را گرفته و به میزبان بعدی مهاجرت می کند. آنچه یک سیستم مبتنی بر عامل را در این شکلش تهدید می کند، اولاً: تغییر در نوع کالا یا تعداد آن - یا مشخصات دیگر استعلام- است ثانیاً: سرقت یا تغییر نتایج جمع آوری شده از میزبان های دیگر است که می تواند به سوءاستفاده از این اطلاعات منجر شود. ثالثاً: سرقت یا تغییر در برنامه سفر یک عامل است که می تواند موجب شود که میزبان بدخواه از وضعیت و نتایج نهایی به طور ضمنی آگاه گردد یا با تغییر مسیر از ادامه سفر عامل به میزبان های دیگر جلوگیری نماید؛ که این خود نوعی تغییر در حالت عامل می باشد.



شکل شماره ۱- معماری ارتباط مدل ارائه شده

در مدل ارائه شده که از رمزگذاری نامتقارن جهت ایجاد امنیت داده ها و مسیر حرکت عامل استفاده شده است. در این مدل، تمامی میزبان هایی که به عنوان ارائه دهنده خدمات تجارت الکترونیک هستند ابتدا در سروری ثبت می شوند و کلید عمومی خود را در اختیار سرور ثبت قرار می دهند (شکل شماره ۱) عامل برای درخواست یک کالا از ایجاد کننده خود به سمت سرور ارسال کننده عامل مهاجرت می نماید تا اولاً مسیر حرکت خود را تعیین نماید و ثانیاً درخواست هایش در

عدم وابستگی - را پیاده سازی نماید، مدل مشتری/سرورس دهنده با تعدد سرورها را تداعی می نماید.

در روش بعدی تعدادی عامل همکار با مسیر های مختلف به سوی شبکه ارسال می گردند، و اطلاعات بین این عاملها توزیع می شود، از اینرو میزبان داده معتبری را نمی تواند از یک عامل سرقت نماید یا تغییر دهد [۱۰]. با این وجود ارسال متعدد عاملها که موجب افزایش بار شبکه می گردد، جز نقاط ضعف این روش است و از سوی دیگر تضمین و اثبات خاصی جهت حفاظت قطعی از اطلاعات در این مدل وجود ندارد.

در روشی دیگر که حالت عمومی تر روش تابع رمزگذاری شده است، کد عامل بعد از سپری شدن زمان مشخصی مبهم و غیر قابل خواندن می شود، از این طریق میزبان های بدخواه فرصت لازم را جهت انجام عملیات تخریب خود ندارند [۱۱]. اما این روش نیز فقط به ازای زمان کوتاهی محرمانگی کد را تضمین می نماید و از سوی دیگر از طریق مهندسی معکوس امکان بازیابی کد اولیه امکان پذیر خواهد بود. در حالیکه این روش به ازای انواع داده مختلف قابل اعمال نیست و نهایتاً محاسبه و تخمین زمان مبهم شدن کد بسیار مشکل است [۱۲]. علاوه بر این تحقیقاتی نشان داده است که این مدل آنچنان هم مدل موفقی نیست [۱۳].

روش دیگر مدل سرپرست/کارگر است که در این روش عاملهای سرپرست یا ناظر در میزبان های امن مستقر هستند و عاملهای کارگر در میزبان های ناامن فعالیت می کنند و اطلاعات و وظایف خود را از سرپرست ها دریافت می کنند [۱۴]. مشکل این روش نیز مشابه با روش ارجاع حالت، ثابت بودن مکان عاملهای سرپرست و ارسال پیام های مکرر میان کارگرها و سرپرست ها و در نتیجه پیاده سازی عملی مدل مشتری/سرورس دهنده به جای عاملهای متحرک است.

۴. مدل ارائه شده

آنچه که کمتر در این روش ها اشاره شده است نوع فعالیتی است که عامل در میزبان انجام می دهد. این فعالیت خصوصاً در بسترهای مختلف متفاوت و قابل احصا می باشد. به بیان دیگر بسته به نوع کاربرد، نحوه فعالیت قابل تعریف است. با توجه به اینکه یکی از پرکاربردترین حوزه های بهره گیری از عاملها در تجارت الکترونیک و E-Shopping است و از سوی دیگر مباحث امنیتی عامل ها عملاً در این حوزه اهمیت دارند، بهتر

اندازه یک عامل که بخش داده ای آن در سرور ارسال کننده عامل به آن ملحق می گردد شامل دو بخش کد و آرایه است که به تعداد نودهای ملاقاتی خواهد بود، D_{mig} اندازه عامل انتقالی است.

$$D_{mig} = D_{code} + D_{data} \quad (۱)$$

با توجه به اینکه نوع درخواست و مشخصات آن - مانند تعداد - میزبان بعدی عامل و نتیجه بازگشتی در یک رشته متناظر هر میزبان ذخیره می گردد، لذا اندازه داده ها وابستگی مستقیم به اندازه رشته ها دارد.

$$D_{mig} = D_{code} + N * D_h \quad (۲)$$

عامل در سرور ارسال کننده - پس از ارسال از کلاینت به سرور ارسال کننده - مستقر می گردد و تمامی رشته های آن توسط کلیدهای عمومی میزبان ها رمزگذاری می شوند. این کلیدها در سرور ثبت موجود است که سرور ارسال کننده از آن درخواست می نماید.

$$T_{TS} = T_{rqPK} + T_{rpPK} + N * D_h \left(\frac{1}{R_s} + \frac{1}{R_e} \right) \quad (۳)$$

زمان T_{TS} کل زمانی است که صرف رمزگذاری و آماده سازی عامل در سرور ارسال کننده می شود. زمان ارسال یک عامل در شبکه نیز بسته به اندازه آن می باشد که در این مدل اندازه عامل در طول سفر تغییر نمی کند.

$$T_{mig} = \left(\frac{D_{mig}}{R_{th}} \right) \quad (۴)$$

در هر میزبان ابتدا رشته مربوطه توسط کلید خصوصی بازگشایی شده و سپس امضای دیجیتالی آن بررسی می گردد. جستجوی در بانک اطلاعاتی و نهایتاً ذخیره اطلاعات به صورت رمزگذاری شده و امضا شده در همان آرایه صورت می پذیرد. T_h شامل تمامی فعالیت هایی است که در یک میزبان انجام می شود.

(۵)

$$T_h = D_h \left(\frac{1}{R_d} + \frac{1}{R_v} \right) + \frac{1}{R_{se}} + D_h \left(\frac{1}{R_e} + \frac{1}{R_s} \right)$$

کل زمانی که صرف مسافرت عامل تا بازگشت به سرور ارسال کننده می شود برابر مقدار ذیل است:

$$T_{TripMA} = T_{mig} + T_{TS} + N * T_{mig} + N * T_h \quad (۶)$$

پس از رسیدن عامل به ارسال کننده کافیت تمامی رشته های آرایه رمزگذاری شده توسط کلید خصوصی بازگشایی شده و امضای میزبان ها تایید شود. از اینرو زمان بازگشایی و تایید امضا نیز به زمان (۶) اضافه می گردد. بر این اساس با جایگذاری فرمولهای بالا زمان ذیل برای مدل ارائه شده بدست می آید:

$$T_{SecMA} = T_{rqPK} + T_{rpPK} + 2 * N * D_h * \quad (۷)$$

$$\left\{ \frac{1}{R_{se}} + \frac{1}{2 * R_{th}} + \left(\frac{1}{R_s} + \frac{1}{R_e} \right) + \left(\frac{1}{R_d} + \frac{1}{R_v} \right) \right\}$$

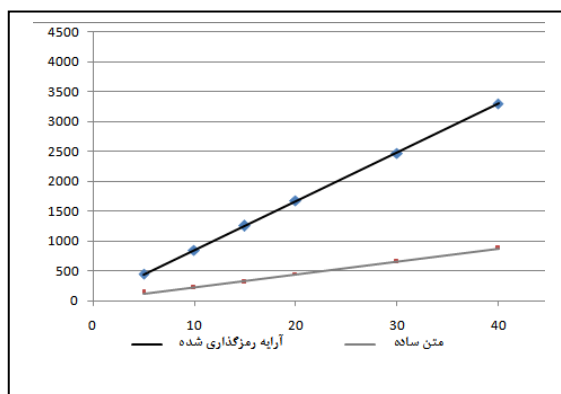
۶. تحلیل مدل:

الف- تحلیل امنیتی: با توجه به دسته بندی حوزه تهدیدهای میزبان علیه عامل که در [۱۷] بیان شده است، مدل ارائه شده را می توان بررسی و تحلیل نمود.

حمله به جامعیت^{۲۰} عامل: حملاتی را که منجر به تغییر کد، داده یا حالت عامل می گردند، شامل می شود و بر دو دسته تقسیم می گردند:

مدخله در جامعیت: که در این حالت در کد و داده تغییر داده نمی شود اما میزبان بدخواه عامل را برای میزبان دیگری ارسال می کند که در برنامه سفر عامل نیست. در این مدل با توجه به رمزگذاری داده ها و حالت برنامه امکان تغییر در برنامه سفر وجود ندارد و میزبان جاری پس از بازگشایی رشته خود از آرایه فقط از میزبان بعدی اطلاع می یابد.

تغییر در اطلاعات: در این حالت کد و داده های عامل در معرض تغییر قرار می گیرند. در این مدل با توجه به اینکه کد عامل شامل یک درخواست از قبل مشخص برای هر میزبان پذیرنده عامل است، امکان تغییر آن وجود ندارد و در خصوص داده نیز آرایه داده بدون رمزگشایی هر رشته با استفاده از کلید خصوصی میزبان مربوطه قابل تغییر نمی باشد. طبیعتاً تغییر بایستی به گونه ای انجام پذیرد که امکان کشف آن توسط پردازش کننده نتایج عامل وجود نداشته باشد؛ که در این صورت در این مدل امکان این مساله وجود ندارد.



شکل شماره ۳- نمودار زمان اجرای الگوریتم خرید

این موضوع باعث می گردد که بیشتر زمان اجرای الگوریتم در میزبان ها سپری شود و کمترین بار بر روی شبکه تحمیل شود که باعث می گردد مدل مشخصه های مدل مبتنی بر عامل به خوبی پیاده شود. بر اساس این فرمول سرعت رمزگذاری، رمزگشایی می تواند عوامل تعیین کننده ای باشد که با توجه به عمومیت بهره گیری از الگوریتم کلید عمومی در بسیاری از روش های ارائه شده در این حوزه مانند [۱۸] [۱۹] [۷] [۱۱] اثرات آن به منظور نیل به امنیت بالاتر پذیرفته شده است. با این وجود بهره گیری از روش رمزگذاری لغزشی باعث افزایش کارایی مدل خواهد شد.

۷. نتایج:

الگوریتم مذکور در محیط مبتنی بر عامل Jade که از معروفترین محیط های عامل است پیاده سازی شد، سیستم های بررسی شده با مشخصات پردازنده دو هسته ای و حافظه ۱GB و سیستم عامل ویندوز xp اجرا شده اند.

خرید الکترونیک را برای ۵ تا ۴۰ میزبان اجرا نمودیم و در هر مرحله درخواست و پاسخ عامل را به صورت متن ساده (رمز نشده) و رمز شده با استفاده از الگوریتم آرایه های رمزگذاری شده اجرا شد.

الگوریتم رمزگذاری کلید عمومی RSA بوده است و با توجه به کوچک بودن داده های عامل (درخواست عامل و پیشنهادات میزبان ها) اندازه متن ساده کم حجم می باشد. از اینرو برای کلید عمومی میزبان ها از RSA ۱۲۸ بیتی استفاده شده است و برای کلید عمومی سرور ارسال کننده عامل از RSA ۲۵۶ بیتی بهره گرفته شده است. نتایج بدست آمده در جدول شماره ۲ ارائه شده است. از نتایج مشخص می گردد، اگرچه زمان رمزگذاری و رمزگشایی تاثیر خود را بر روی کارایی داشته

محرمانگی عامل: شامل تهدیدهایی که به هرشکلی امکان دسترسی به داده های عامل را فراهم می نماید. که شامل سه دسته حملات می شود:

استراق سمع: که هر گونه دسترسی به اطلاعات در حال انتقال با عامل را شامل می گردد. در مدل ارائه شده رمزگذاری اجازه دسترسی به برنامه سفر، پیشنهادهای ارائه شده میزبان ها دیگر و حتی درخواست عامل از میزبان های دیگر را نمی دهد. بدین شکل یک عامل می تواند برای رد گم کردن و عدم اطلاع میزبان ها برنامه سفر خود را از میزبان هایی عبور دهد که اصلا با موضوع درخواست اصلی عامل ارتباطی ندارند؛ لذا هیچ اطمینانی برای میزبان وجود ندارد که درخواست عامل مذکور، درخواستی است که از تمامی میزبان های دیگر می شود.

سرقت: تفاوت آن با استراق سمع، برداشتن یا حذف نمودن همان اطلاعاتی که میزبان به آن دسترسی داشته است. که شاید بتوان این نوع تهدید را در نوعی تغییر در اطلاعات دانست که به آن اشاره شد.

نقابگذاری^۱: که میزبان بدخواه خود را به جای یک میزبان دیگر معرفی می نماید تا بتواند اطلاعات مورد نظر را از عامل دریافت نماید. با توجه به اینکه آرایه صرفا با کلید خصوصی هر میزبان قابل دسترسی می باشد، حتی در صورتی که میزبان بدخواه خود را به عنوان مقصد بعدی عامل معرفی نماید و میزبان جاری، عامل را برای میزبان بدخواه ارسال کند؛ امکان هیچ سو استفاده ای از عامل وجود نداشته و میزبان بدخواه بدلیل نداشتن کلید خصوصی امکان بازگشایی هیچ رشته ای از آرایه را ندارد.

ب- تحلیل کارایی: فرمول (۷) نشان می دهد مدل ارائه شده قابلیت گسترش بالایی دارد و بجز درخواست کلید میزبان ها و پاسخ آن از سرور ثبت $(TrqPK + TrspPK)$ هیچ ارتباطی دیگر میان عامل با سرور ارسال کننده یا سرورهای دیگر وجود ندارد.

تعداد میزبان	متن ساده	آرایه رمزگذاری شده
5	140	453
10	219	843
15	313	1250
20	453	1672
30	650	2470
40	880	3300

جدول شماره ۲- زمان جمع آوری پیشنهادات در خرید الکترونیک

3. *Classification of Malicious Host Threats in Mobile Agent Computing*. ELMARIE BIERMAN, Technikon Pretoria, ELSABE CLOETE. s.l.: Proceedings of SAICSIT, 2002.

4. *MobileTrust: a trust enhanced security architecture for mobile agent systems*. Ching Lin, Vijay Varadharajan. s.l. s.l.: International Journal of Information Security, 2010.

5. A. Kannammal, N.Ch.S.N. Iyengar. *A Model for Mobile Agent Security in E-Business Applications*. s.l.: International Journal of Business and Information, December 2007.

6. Vigna, G. *Protecting Mobile Agents through Tracing*. Workshop on Mobile Object Systems, 1997.

7. Riordan J., B. Schneier. *Environmental Key Generation towards Clueless Agent*. s.l.: In LNCS, pages 15–24. Springer, 1998.

8. Hyungjick, Lee. *The use of encrypted functions for mobile agent security*. Hawaii : 37th Hawaii International Conference on System Sciences, 2004. 0-7695-2056-1/04.

9. Hohl, F. *A Protocol to Detect Malicious Hosts Attacks by Using Reference States*. s.l.: , Universität Stuttgart, Fakultät Informati 1999/09 , 1999. 1999/09 .

10. Volke, R. *Mutual Protection of Co-operating Agents*. s.l.: Secure Internet Programming , pages 275-285, 1999.

11. F., Hohl. *An Approach to Solve the Problem of Malicious Hosts*. s.l.: Technical Report, Universität Stuttgart, , March 1997. 1997/03.

12. Giansiracusa, Michelangelo. *Mobile Agent Protection Mechanisms, and the Trusted Agent Proxy Server (TAPS) Architecture*. s.l.: Information Security Research Centre, 2003. Brisbane, Australia.

13. BARAK, B., GOLDREICH, O., IMPAGLIAZZO, R., RUDICH, S., SAHAI, A, VADHAN, S., and YANG. *On the (im)possibility of obfuscating programs*. Berlin : , 21st Annual International Cryptology Conference, Springer-Verlag, 2001.

14. S., Fishmeister. *Building Secure Mobile Agents: The Supervisor Worker Framework*. s.l.: Diploma Thesis, February 2000.

15. *Sliding encryption: A cryptographic tool for mobile agents*. YOUNG, A. AND YUNG, M. s.l.: In Fast Software Encryption—FSE'97, 1997.

16. *Classification of Malicious Host Threats in Mobile Agent Computing*. ELMARIE BIERMAN, ELSABE CLOETE. s.l.: Proceedings of SAICSIT, 2002.

17. *Implementing Mobile Agent Security In An Untrusted Computing Environment*. Wagner, Ambrus. Zagreb, Croatia : 8th International Conference on

است. اما میزان این تاثیر به حدی کم است که در یک خرید الکترونیک این اختلاف زمانی بی اهمیت تلقی می گردد. در حالیکه امنیت تامین شده داده ها و برنامه سفر عامل، مزیتی است که سلامت خرید را به همراه خواهد داشت. نمودار شماره ۱ رشد خطی زمان اجرا نسبت به افزایش تعداد میزبان را نشان می دهد.

این نمودار مشخص می نماید که الگوریتم ارائه شده روند خطی را دارد. در حالیکه این میزان در مدل [۵] بستگی کاملی به زمان پاسخگویی سرور تایید دارد. این نتایج با بهره گیری از الگوریتم رمزگذاری لغزشی بهبود می یابد چرا که زمان رمزگذاری و رمزگشایی کاهش قابل توجهی خواهد داشت.

۸. جمع بندی و کارهای آینده:

یکی از مشکلات عمده سیستم های مبتنی بر عامل چالش امنیتی آن خصوصا در حوزه حملات میزبان به عامل می باشد. اگرچه راه حل های مختلفی برای مشکل ارائه شده است اما هنوز یک پاسخ جامع و کامل به این مشکل داده نشده است. مدل ارائه شده در این مقاله با نگاه ویژه به بحث بهره گیری از عامل در تجارت الکترونیک، راه کار مناسب و امنی برای تعامل عامل و جلوگیری از تهدیدهای مختلف علیه عامل ارائه کرده است. این راه کار مبتنی بر ذخیره اطلاعات عامل در آرایه ای رمزنگاری شده است به نحوی که بجز میزبان مورد نظر امکان دسترسی به اطلاعات برای میزبان های دیگر وجود ندارد. مدل مذکور در مقایسه با مدل های مشابه مانند [۵] از جهت امنیتی جامع تر و از جهت کارایی بهتر عمل می نماید، خصوصا اینکه به ایده اصلی عاملها متحرک که هدف آن کاهش بار شبکه و غیرهمزمانی با ارسال کننده آن است مطابقت دارد، در حالیکه برخی مدل های ارائه شده قبلی این خصوصیت را ندارند. در ادامه می توان این مدل را برای دیگر کاربردهای سیستم های مبتنی بر عامل سفارشی نمود. همچنین این مدل را در برابر تهدیدات دیگر امنیتی سیستم های مبتنی بر عامل نیز مقاوم ساخت.

۹. مراجع

. Lange, Danny B., and Oshima, Mitsuru. *Seven good reasons for mobile agents*. s.l.: Communications of the ACM, 1999. pp. pp.88-89.

2. Wagner, Ambrus. *Implementing Mobile Agent Security In An Untrusted Computing Environment*. Zagreb, Croatia : 8th International Conference on Telecommunications - ConTEL 2005, 2005. ISBN: 953-184-081-4, June 15-17.



Telecommunications - ConTEL 2005, 2005. ISBN: 953-184-081-4.

18. Sander T., C. F. Tschudin. *Protecting Mobile Agents Against Malicious Hosts*. Heidelberg, Germany : Springer-Verlag, 1998. Mobile Agents and Security, LNCS, pages 44-60.

19. Borselius, N. *Mobile agent security*. s.l. : ELECTRONICS & COMMUNICATION ENGINEERING JOURNAL, OCTOBER 2002.

20. Yee, Bennet S. *A Sanctuary for Mobile Agents*. San Diego : University of California , April 28, 1997. Technical Report CS97-537.

21. Eung-Gu You, Keum-Suk Lee. *A Mobile Agent Security Management*. s.l. : 18th International Conference on Advanced Information Networking and Application, 2004.

22. Hyungjick Lee, Jim Alves-Foss and Scott Harrison. *The Use of Encrypted Functions for Mobile Agent Security*. Hawaii : 37th International Conference on System Sciences, 2004.

23. *Implementing Mobile Agent Security In An Untrusted Computing Environment*. Wagner, Ambrus. Zagreb, Croatia : 8th International Conference on Telecommunications - ConTEL 2005, 2005. ISBN: 953-184-081-4, June 15-17.

-
- ¹ Mobile Agent
 - ² Client/Server
 - ³ Remote Evaluation
 - ⁴ Code on Demand
 - ⁵ Real-time
 - ⁶ Platform Independent
 - ⁷ Strong Mobility
 - ⁸ Malicious
 - ⁹ Agency
 - ¹⁰ Authentication
 - ¹¹ Masquerade
 - ¹² Itinerary
 - ¹³ Confidentiality
 - ¹⁴ Execution Tracing
 - ¹⁵ Log
 - ¹⁶ Environmental Key Generation
 - ¹⁷ Encrypted function computing
 - ¹⁸ Reference state
 - ¹⁹ Sliding Encryption
 - ²⁰ Integrity
 - ²¹ Masquerading