هفتمین کنفرانس بین المللی انجمن رمز ایران

۲۴ و ۲۵ شهریورماه ۱۳۸۹، دانشگاه صنعتی خواجه نصیرالدین طوسی

انجمن رمزایران

دانشگاه صنعتی خواجه نصیرالدین طوسی

!

استفاده از فرایندکاوی در سیستم‌های تشخیص نفوذ مبتنی بر میزبان

هانیه جلالی[۱]، دکتر احمد براآنی[۲]

دانشجوی کارشناسی ارشد، اصفهان، دانشگاه اصفهان، گروه مهندسی کامپیوتر[۱]
jalali@eng.ui.ac.ir

عضو هیئت علمی، اصفهان، دانشگاه اصفهان، گروه مهندسی کامپیوتر [۲]
ahmadb@eng.ui.ac.ir

چکیده

امروزه بسیاری از سازمان‌ها برای بهبود کارایی خود از سمت سیستم‌های داده-محور به فرایند-محور حرکت کرده و از سیستم‌های اطلاعاتی مبتنی بر فرایند استفاده می‌کنند. از نکات مهم در چنین سیستم‌هایی، همانند دیگر نمونه‌ها، نیاز به برقراری امنیت است. یکی از بهترین مکانیسم‌های برقراری امنیت استفاده از سیستم‌های تشخیص نفوذ مبتنی بر میزبان است. به همین منظور در این مقاله با استفاده از تکنیک‌های فرایندکاوی یک مدل تشخیص نفوذ مبتنی بر میزبان فرایندمحور ارائه می‌شود که میزان هشدار اشتباه را کاهش داده و دقت تشخیص را بالا ببرد. در این مدل از دو تکنیک تشخیص ناهنجاری و تشخیص سوءاستفاده بصورت ترکیبی برای ابعاد جریان کار و سازمانی فرایندکاوی استفاده شده است. نتایج حاصل از تست و ارزیابی مدل، کارا و مناسب بودن آن را نشان می‌دهند.

واژه های کلیدی

فرایندکاوی، سیستم تشخیص نفوذ مبتنی بر میزبان، سیستم اطلاعاتی مبتنی بر فرایند.

۱- مقدمه !

اخیراً پیشرفت‌های حوزه مدیریت، سازمان‌ها را به سمت استفاده از سیستم‌های اطلاعاتی مبتنی بر فرایند[۱] سوق داده‌اند. استفاده از PAISها منجر به حرکت از سیستم‌های داده-محور[۲] به سیستمهای فرایند-محور[۳] می‌شود، چرا که در اصل این فرایندهای حاکم بر یک سیستم هستند که داده‌ها را حرکت داده و تغییر می‌دهند. در این سیستم‌های اطلاعاتی، نیاز مبرم به وجود تکنیک‌هایی برای بیرون کشیدن دانش از اطلاعات ثبت شده در سیستم‌های اطلاعاتی توسط مکانیسم‌های حسابرسی و همچنین رشد سریع داده‌های لاگ[۴] در قالب ردپای بررسی[۵]، لاگ‌های تراکنش و انبارهای داده منجر به توسعه تکنیک‌هایی در زمینه فرایندکاوی[۶] شده است. هدف فرایندکاوی بدست آوردن دانش از لاگ‌های وقایع ثبت شده در سیستم‌های اطلاعاتی است.

هنگام صحبت از هرگونه سیستم اطلاعاتی، امنیت نقش مهمی را ایفا می‌کند. به طوری که هر کاربر کامپیوتر اصطلاح "امنیت" و نیاز به محافظت از اطلاعات موجود بر روی کامپیوترها را شنیده است و دلیل آن است که در واقع، بسیار سخت است که سیستم‌های اطلاعاتی کاملاً امنی را ایجاد کرده و آن‌ها را در طول مدت حضور و استفاده در همان وضعیت امنیتی نگاه داشت. از این رو در شبکه‌ها و سیستم‌های کامپیوتری به مکانیسم‌هایی برای برقراری امنیت احتیاج است. یکی از مهم‌ترین این مکانیسم‌ها، سیستم‌های تشخیص نفوذ[۷] هستند [۱و ۲].

سیستم‌های تشخیص نفوذ وظیفه نظارت بر استفاده از سیستم‌های اطلاعاتی را برعهده دارند تا هر وضعیت ناامنی را تشخیص دهند. آنها تلاش برای سوء استفاده یا انجام آن را توسط

[1] Process Aware Information Systems (PAIS)
[2] Data-centric
[3] Process-centric
[4] Log
[5] Audit trail
[6] Process mining
[7] Intrusion Detection System (IDS)

کاربران قانونی سیستم‌های اطلاعاتی یا افراد بیرونی که قصد استفاده نادرست از امتیازات خود یا بهره‌بری از آسیب‌پذیری‌های امنیتی موجود را دارند، تشخیص می‌دهند.

در زمینه تشخیص نفوذگران یک شبکه کامپیوتری، دو نوع سیستم تشخیص نفوذ وجود دارد که بر روی دو مجموعه داده مختلف کار می‌کنند: سیستم تشخیص نفوذ مبتنی بر میزبان[1] و سیستم تشخیص نفوذ مبتنی بر شبکه[2]. تشخیص نفوذ مبتنی بر میزبان اولین حوزه کشف شده در زمینه تشخیص نفوذ است. HIDS اغلب یک برنامه کاربردی است که برای اهداف نظارتی بر روی یک میزبان نصب می‌شود، رخدادهای حاصل از برنامه‌های کاربردی، سیستم‌عامل، بسته‌های شبکه و لاگ‌ها را تجزیه و تحلیل کرده و اگر نفوذی رخ داده باشد هشداری به مسئول امنیت می‌فرستد. NIDS اغلب یک محصول تجاری است که بر روی سخت‌افزار خاصی نصب شده و بر روی یک نود شبکه قرار می‌گیرد. NIDS بسته‌های شبکه را که از نود عبور می‌کنند دریافت کرده و تجزیه و تحلیل می‌کند، بسته‌های دریافت شده به طور محلی بررسی شده و اگر حمله‌ای تشخیص داده شود هشداری به مسئول امنیت فرستاده می‌شود [۱-۳]. در دهه گذشته، سیستم‌های تشخیص نفوذ مبتنی بر شبکه به وضوح بر سیستم‌های مبتنی بر میزبان غلبه داشتند. اما امروزه برای بار دیگر سیستم‌های تشخیص نفوذ به سرورهای میزبان بازگشته‌اند، چرا که محتویات داده‌ها به وضوح قابل رویت هستند و رمزنگاری نشده‌اند، تکنیک‌هایی مانند حمله به شبکه برای حمله به میزبان وجود ندارد و همچنین کیفیت داده‌هایی که در سیستم عامل و برنامه‌های کاربردی وجود دارند بسیار بالا است. در واقع سیستم‌های تشخیص نفوذ مبتنی بر شبکه با دو مشکل روبرو هستند، آن‌ها نمی‌توانند دقت لازم برای تشخیص نفوذ را فراهم کرده و تضمین کنند که تشخیص حملات با نرخ بالای هشدار اشتباه[3] همراه نیست.

از آنجایی که سیستم‌های تشخیص نفوذ مبتنی بر میزبان یکی از کاراترین روش‌های موجود برای برقراری امنیت در سیستم‌های کامپیوتری هستند، در این مقاله سعی می‌شود با استفاده از تکنیک‌های فرایندکاوی یک سیستم تشخیص نفوذ مبتنی بر میزبان برای سیستم‌های اطلاعاتی مبتنی بر فرایند ارائه شود.

کارهای قبلی انجام شده به منظور تشخیص نفوذ در PAISها با استفاده از تکنیک تشخیص ناهنجاری[4]، فرایندهای غیرمعمول انجام شده در سیستم را به عنوان حمله (ناهنجاری) کشف می‌کنند. در [۴] با استفاده از الگوریتم α در فرایندکاوی، دو روش برای تشخیص ناهنجاری ارائه شده است که در آن‌ها لازم است یک لاگ "نرمال" شناخته شده وجود داشته باشد تا براساس آن فرایندهای ناهنجار تشخیص داده شوند. در نتیجه این روش‌ها برای حوزه‌های کاربردی که نیاز به انعطاف‌پذیری دارند مناسب نیستند، چرا که معمولاً قبل از اجرا یک لاگ یا مدل "نرمال" از سیستم وجود ندارد. دیگر تکنیک‌های ارائه شده [۵-۸] هستند. این تکنیک‌ها برای حوزه‌های کاربرد انعطاف‌پذیر قابل استفاده هستند. اما [۵و ۶] برای لاگ‌های با حجم زیاد مناسب نیستند و [۷] به یک معیار تناسب دقیقتر احتیاج داشته و همچنین روشی خودکار برای کشف ناهنجاری‌ها نیست. در [۸] برای تشخیص ناهنجاری روشی با استفاده از فرایندکاوی ژنتیکی[5] ارائه شده است که خودکار بوده و به معیار دقیق تناسب احتیاج نداشته و محدودیتی در حجم لاگ ندارد. تکنیک تشخیص ناهنجاری استفاده شده در این کارها حملات ناشناخته را تشخیص می‌دهد ولی معمولاً دارای نرخ بالای هشدار اشتباه است. از طرف دیگر در این کارها برای یافتن حملات فقط بعد جریان‌کار[6] فرایندها در نظر گرفته شده است. یعنی ممکن است حمله مسیر نرمالی را طی کند اما توسط نقش‌ها یا کاربران غیرمجاز انجام شده باشد.

در این کار، هدف این است که سیستم تشخیص نفوذ مبتنی بر میزبانی ارائه شود که برای حوزه‌های کاربرد انعطاف‌پذیر مناسب بوده و در تشخیص ناهنجاری خودکار عمل کند. از طرف دیگر سعی بر اینست که از مزایای هر دو تکنیک تشخیص ناهنجاری و تشخیص سوءاستفاده[7] امکان پذیر در سیستم‌های تشخیص نفوذ استفاده شود تا نرخ هشدار اشتباه کاهش یافته و در عین حال حملات ناشناخته نیز کشف شده و دقت تشخیص هم بالا رود. همچنین علاوه بر بعد جریان کار، بعد سازمانی فرایندکاوی نیز مورد بررسی قرار گیرد.

با توجه به این اهداف، یک مدل تشخیص نفوذ مبتنی بر میزبان فرایند محور[8] ارائه شده است که برای تشخیص حملات در سیستم‌های اطلاعاتی مبتنی بر فرایند استفاده می‌شود. این مدل دارای سه قسمت پیش پردازش، تشخیص و ترکیب نتایج است. در قسمت تشخیص، از دو تکنیک تشخیص ناهنجاری و تشخیص سوءاستفاده بصورت ترکیبی برای یافتن ناهنجاری‌ها در بعد جریان‌کار و سوءاستفاده‌ها در بعد سازمانی[9] استفاده می‌شود. سپس در قسمت ترکیب، حملات بدست آمده از دو مرحله قبل ادغام شده و فرایندهایی که مورد حمله قرار گرفته‌اند به همراه نوع حملات مشخص می‌شوند.

[1] Host-based Intrusion Detection System (HIDS)
[2] Network-based Intrusion Detection System (NIDS)
[3] False positive
[4] Anomaly detection
[5] Genetic Process Mining
[6] Control-flow
[7] Misuse detection
[8] Process Aware Host-based Intrusion Detection Model
[9] Organizational perspective

در ادامه در بخش ۲ کارهای مشابهی که در حوزه امنیت سیستم‌های مبتنی بر فرایند انجام شده‌اند با توضیح بیشتر آورده شده‌اند. در بخش ۳ مدل تشخیص نفوذ مبتنی بر میزبان فرایندمحور پیشنهادی معرفی شده و جزئیات هر مرحله همراه با نحوه پیاده‌سازی شرح داده شده است. نحوه انجام ارزیابی و نتایج حاصل از آن در بخش ۴ ذکر شده‌اند. در نهایت در بخش ۵ نتیجه‌گیری کار آمده است.

۲-کارهای مشابه!

هدف فرایندکاوی بیرون کشیدن اطلاعات در زمینه‌های مختلف از لاگ‌های ثبت شده در سیستم‌های اطلاعاتی است و به عنوان راهی برای تجزیه و تحلیل سیستم‌ها و رفتار واقعی آن‌ها براساس لاگ وقایعی که تولید می‌کنند، بکار می‌رود [۹].

در حوزه تشخیص نفوذ در سیستم‌های اطلاعاتی مبتنی بر فرایند، چندین کار برای تشخیص دنباله‌های[1] ناهنجار فرایندها انجام شده است [۴-۸]. در [۴] با استفاده از الگوریتم α در فرایندکاوی، دو روش برای تشخیص ناهنجاری ارائه شده است. در این دو روش با استفاده از یک لاگ قابل قبول (نرمال) یک مدل فرایند قابل قبول (مرجع) بدست می‌آید. سپس بصورت بلادرنگ وقایع جدید رخ داده شده، که در یک لاگ جداگانه ثبت می‌شوند، با مدل مرجع مقایسه شده و هر کجا عدم تطابقی وجود داشته باشد، یک حمله در نظر گرفته می‌شود. این روش با توجه به رفتار غیر قابل پیش بینی کاربران می‌تواند دارای هشدار اشتباه زیاد بوده و برای کاربردهای انعطاف پذیر مناسب نیست، چرا که یک لاگ نرمال قبل از اجرا وجود ندارد. در چهار کار دیگر، مدل نرمال مرجع در طول فرایند تشخیص ناهنجاری تولید می‌شود و در نتیجه برای کاربردهای انعطاف پذیر مناسب هستند. در [۵و ۶] سه الگوریتم برای تشخیص ناهنجاری ارائه و مقایسه شده‌اند: نمونه-برداری[2]، آستانه[3] و تکراری[4]. در نهایت الگوریتم نمونه‌برداری به عنوان روش بهتر معرفی شده است. این روش‌ها به دلیل تغییر و تطابق الگوریتم فرایندکاوی افزایشی دارای محدودیت بوده و برای لاگ‌های با حجم بالا قابل استفاده نیستند. در [۷] روشی چهار مرحله‌ای ارائه شده است که مبتنی بر تعریفی صوری از یک دنباله ناهنجار طبق پارامترهای درجه تناسب مدل[5] (p%) و مناسب بودن مدل[6] (a) است. با استفاده از دو پارامتر ذکر شده یک مدل مناسب منطبق بر لاگ وقایع پیدا شده و دنباله‌هایی که بر آن منطبق نیستند به عنوان حمله در نظر گرفته می‌شوند. اما همانطور که در مقاله نیز ذکر شده است، معیار مناسب بودن مدل چندان دقیق نبوده و همچنین روش بصورت خودکار مدل مناسب را پیدا نکرده و نیازمند بررسی‌های دستی مسئول امنیت است. در [۸] با استفاده از فرایندکاوی ژنتیکی روشی سه مرحله‌ای برای تشخیص ناهنجاری ارائه شده است. به دلیل استفاده از الگوریتم ژنتیک، مدل ارائه شده خودکار بوده و از نظر اجرایی محدودیتی در حجم لاگ ندارد و برای تشخیص ناهنجاری در PAIS راه مناسبی است. اما این کار فقط محدود به بعد جریان کار فرایندها است و کاربران انجام‌دهنده آن‌ها را بررسی نمی‌کند. به این معنی که ممکن است رفتار انجام شده مسیر نرمالی را طی کند اما توسط کاربر یا نقشی غیرمجاز انجام شود و به عنوان حمله تشخیص داده نشود. همچنین روش ارائه شده در این مقاله پیاده‌سازی نشده و مورد ارزیابی قرار نگرفته است.

در مقایسه با کارهای قبلی، روش پیشنهاد شده در این مقاله سعی دارد علاوه بر داشتن ویژگی‌های مثبت کارهای انجام شده پیشین مانند انعطاف پذیر و خودکار بودن و نداشتن محدودیت اجرایی، از مزایای هر دو تکنیک تشخیص ناهنجاری و تشخیص سوءاستفاده امکان‌پذیر در سیستم‌های تشخیص نفوذ استفاده کرده و نتایج حاصله را ترکیب کند. همچنین بعد سازمانی که در هیچ یک از کارهای قبل در نظر گرفته نشده‌اند را برای تشخیص حملات بیشتر و دقیقتر مورد استفاده قرار دهد.

۳- مدل تشخیص نفوذ مبتنی بر میزبان فرایندمحور!

تشخیص نفوذ، فرایند نظارت بر وقایع رخ داده در سیستم یا شبکه کامپیوتری و تجزیه و تحلیل آن‌ها برای آشکارسازی نشانه‌های نفوذ است. نفوذ، به عنوان تلاش در جهت به خطر انداختن محرمانگی[7]، جامعیت[8]، قابلیت دسترسی[9] یا عبور کردن از مکانیسم‌های امنیتی کامپیوتر یا شبکه در نظر گرفته می‌شود. نفوذ، توسط مهاجمینی که از طریق اینترنت به سیستم دسترسی پیدا می‌کنند، کاربران مجاز سیستم که تلاش می‌کنند به امتیازاتی فراتر از آنچه اجازه دارند دست پیدا کنند و کاربران مجازی که از امتیازات خود سوءاستفاده می‌نمایند، بوجود می‌آید. سیستم‌های تشخیص نفوذ محصولات نرم افزاری و سخت‌افزاری هستند که این فرایند نظارت و تجزیه و تحلیل را خودکار می‌سازند [۱۰].

HIDSها بر روی اطلاعاتی که از یک سیستم کامپیوتری منفرد جمع‌آوری شده است، کار می‌کنند. این نکته به IDSهای مبتنی بر میزبان اجازه می‌دهد که فعالیت‌ها را با دقت و قابلیت اعتماد بالا تجزیه و تحلیل کرده و دقیقاً تعیین کنند که چه فرایندها و کاربرانی در یک حمله شرکت داشته‌اند. علاوه بر این بر خلاف NIDSها ، HIDSها می‌توانند نتیجه یک تلاش برای حمله

[1] Trace
[2] Sampling
[3] Threshold
[4] Iterative
[5] Fitness model degree
[6] Appropriateness of model
[7] Confidentiality
[8] Integrity !
[9] Availability

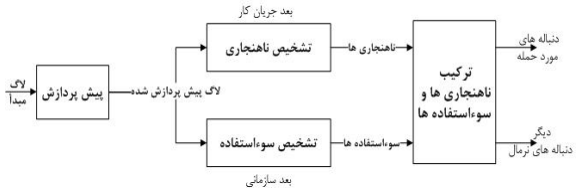
را مشاهده کنند، چرا که آن‌ها مستقیماً به فایل‌های داده‌ای و فرایندهایی که معمولاً مورد هدف مهاجمان هستند دسترسی دارند [۱۰].

دو روش اصلی برای تجزیه و تحلیل رخدادها به منظور تشخیص حملات وجود دارد: تشخیص ناهنجاری و تشخیص سوءاستفاده. تشخیص‌دهندگان ناهنجاری رفتارهای غیرنرمال و غیرمعمول (ناهنجاری‌ها) را روی یک میزبان یا شبکه شناسایی می‌کنند. آن‌ها طبق این فرض کار می‌کنند که حمله‌ها با فعالیت‌های "نرمال" (قانونی) متفاوت هستند و در نتیجه توسط سیستمی که این تفاوت‌ها را شناسایی می‌کند قابل تشخیص هستند. تشخیص‌دهندگان ناهنجاری پروفایل‌ها یا مدل‌هایی را می‌سازند که رفتار نرمال کاربران، میزبان‌ها یا اتصالات شبکه را نمایش می‌دهند. این پروفایل‌ها از داده‌های قبلی که در طول یک دوره عمل نرمال جمع‌آوری شده‌اند، ساخته می‌شوند. سپس تشخیص‌دهنده‌ها داده‌های رخدادها را جمع‌آوری کرده و از روش‌های مختلفی استفاده می‌کنند تا تعیین کنند چه زمانی فعالیت‌های مورد نظارت بر حالت نرمال منطبق نیستند. تشخیص‌دهندگان ناهنجاری و IDSهای مبتنی بر آن‌ها اغلب تعداد زیادی هشدار اشتباه تولید می‌کنند، چرا که الگوهای نرمال رفتار کاربر و سیستم می‌توانند متنوع و غیر قابل پیش بینی باشند. با این وجود IDSهای مبتنی بر تشخیص ناهنجاری می‌توانند اشکال جدید حمله‌ها را کشف کنند و نشانه‌های آن‌ها را بدون داشتن دانش خاصی از جزئیاتشان تشخیص دهند. تشخیص‌دهندگان سوءاستفاده فعالیت‌های سیستم را تجزیه و تحلیل می‌کنند تا رخدادها یا مجموعه‌هایی از رخدادها را بیابند که بر یک الگوی از پیش تعریف شده از یک حمله شناخته شده، منطبق هستند. به الگوهای حملات شناخته شده امضا[1] نیز گفته می‌شود. در تشخیص‌دهندگان سوءاستفاده، هر امضا یک الگو از رخدادهای مربوط به یک حمله را مشخص می‌کند. به این ترتیب تشخیص سوءاستفاده فقط می‌تواند حملاتی را شناسایی کند که آن‌ها را می‌شناسد و راجع به آن‌ها اطلاعات دارد و در نتیجه دانش این روش مرتباً باید توسط امضاهای جدید بروز شود. به طورکلی، تشخیص سوءاستفاده می‌تواند به شکلی سریع و قابل اعتماد حمله را شناسایی کند و به همین دلیل به عنوان روشی کارا برای تشخیص حملات، بدون تولید تعداد زیادی هشدارهای اشتباه شناخته شده است. همانطور که ملاحظه شد، همراه با هر روش نقاط قوت و ضعفی وجود دارد. به همین دلیل در کاراترین IDSها سعی می‌شود از روش‌های تشخیص سوءاستفاده همراه با مؤلفه‌هایی از تشخیص ناهنجاری استفاده شود[2]. تا بدین وسیله از مزایای هر دو روش استفاده شده و معایب هر کدام توسط دیگری پوشانده شود.

در این مقاله نیز سعی بر آنست روشی برای تشخیص نفوذ مبتنی بر میزبان ارائه شود که حاصل ترکیب این دو تکنیک است [۱۰].

همانگونه که در بخش ۱ ذکر شد، فرایندکاوی تکنیکی است برای بیرون کشیدن دانش از لاگ‌های ثبت شده در سیستم‌های اطلاعاتی مبتنی بر فرایند. در نتیجه برای برقراری امنیت توسط یک HIDS در چنین سیستم‌هایی از تکنیک‌های فرایندکاوی استفاده می‌شود.

در تکنیک‌های فرایندکاوی سه بعد مختلف در نظر گرفته می‌شوند: ۱) بعد فرایند (جریان‌کار)، ۲) بعد سازمانی و ۳) بعد مورد[3] (داده). بعد فرایند بر جریان‌کار تمرکز می‌کند، یعنی ترتیب فعالیت‌ها. هدف کاوش این بعد آنست که ویژگی‌های مناسبی از همه مسیرهای ممکن فهمیده شود که برای مثال به صورت شبکه پتری[4] نمایش داده می‌شود. بعد سازمانی بر آغازگرها تمرکز می‌کند، یعنی کدام اجراکننده‌ها حاضر هستند و چگونه به هم مربوط می‌شوند. هدف اینست که از طریق گروه‌بندی افراد در قالب نقش‌ها و واحدهای سازمانی به سازمان ساختار داده شود یا رابطه بین اجراکننده‌های فردی نمایش داده شود. بعد مورد بر خصوصیات نمونه‌های فرایند تمرکز می‌کند. موردها می‌توانند از طریق مسیرهایشان در فرایند یا آغازگرهایی که بر روی مورد کار می‌کنند، توصیف شوند. با این حال، مورد می‌تواند از طریق مقادیر مربوط به عناصر داده‌ای نیز توصیف شود. به طور خلاصه، بعد فرایند به سؤال "چگونه؟"، بعد سازمانی به سؤال "چه کسی؟" و بعد مورد به سؤال "چه چیز؟" مربوط هستند [۹]. در این مقاله سعی بر آنست که برخلاف کارهای قبل که فقط بعد فرایند را در نظر گرفته‌اند، از بعد سازمانی نیز برای تشخیص نفوذ استفاده شود.



شکل ۱- مدل تشخیص نفوذ مبتنی بر میزبان فرایندمحور

در شکل ۱، شمایلی از مدل تشخیص نفوذ مبتنی بر میزبان فرایندمحور پیشنهادی ارائه شده است. این مدل شامل مراحل پیش پردازش، دو مرحله موازی تشخیص ناهنجاری و تشخیص سوءاستفاده و ترکیب نتایج است. در ادامه جزئیات هر یک از مراحل و نحوه پیاده سازی آن‌ها شرح داده شده است. مدل ارائه شده با استفاده از چارچوب ProM قابل پیاده سازی است. ProM یک چارچوب متن باز است که طیف وسیعی از تکنیک‌های

[1] Signatures
[2] Hybride system
[3] Case perspective
[4] Petri net

فرایندکاوی را در قالب plugin فراهم کرده و امکان توسعه و اضافه کردن plugin را نیز دارد[1].

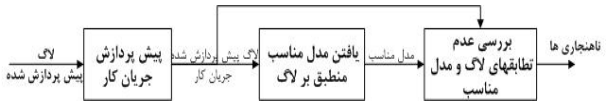
۳-۱- پیش پردازش!

اولین مرحله از مدل تشخیص نفوذ پیشنهادی مربوط به حذف یا نگه داشتن فعالیت‌ها یا دنباله‌ها از لاگ مبدأ است براساس اهمیت و مناسب بودن آن‌ها برای تجزیه و تحلیل‌های بعدی. در این مرحله سه کار مختلف بنا به تصمیم مسئول امنیت قابل انجام است: حذف دنباله‌های ناکامل از لاگ، تکمیل و نگه داشتن دنباله‌های ناکامل، حذف فعالیت/دنباله‌های نامربوط از لاگ. از آنجا که در هر بار اجرای سیستم تشخیص نفوذ، لاگ مبدأ مربوط به یک برهه زمانی مشخص است، بعضی از دنباله‌های اجرا ممکن است در این برهه آغاز و یا تمام نشده باشند. چنین دنباله‌هایی ناکامل هستند و با فعالیت‌های خاصی شروع نشده یا پایان نمی‌یابند. مسئول امنیت برای تجزیه و تحلیل قادر است این دنباله‌ها را حذف کند. اما از آنجایی که خود این دنباله‌ها هم می‌توانند یک حمله باشند و حذف آن‌ها موجب بررسی نشدن آن‌ها می‌شود، می‌توان چنین دنباله‌هایی را تکمیل کرده و در لاگ نگه داشت. برای این کار باید فعالیت شروع یا پایان مصنوعی را به ابتدا یا انتهای آن‌ها اضافه کرد. پیش پردازش دیگری که می‌توان بر روی لاگ مبدأ انجام داد، حذف فعالیت/دنباله‌های نامربوط از لاگ است. این فعالیت/دنباله‌ها در جریان تجزیه و تحلیل مهم نبوده و تنها زمان اجرا را بالا می‌برند.

به منظور پیاده سازی این مرحله، از ابزارهای فیلتر موجود در نرم‌افزار ProM استفاده می‌شود. برای حذف دنباله‌های ناکامل و فعالیت‌های نامربوط از Simple Log Filtering Tools و برای تکمیل دنباله‌های ناکامل از Advanced Filtering Tools استفاده می‌شود.

۳-۲- تشخیص ناهنجاری

تشخیص دهنده ناهنجاری در ابتدا مدلی می‌سازد که استفاده نرمال از سیستم را نشان می‌دهد. سپس با استفاده از این مدل هرگونه عدم تطابق احتمالی که در داده‌های جاری با مدل مرجع وجود دارد را پیدا کرده و به عنوان حمله تشخیص می‌دهد. برای تولید مدل مرجع نرمال، به یک مجموعه آموزشی از رفتار نرمال و قانونی سیستم احتیاج است. چنین مجموعه نرمالی معمولاً براساس رفتار غیر قابل پیش بینی کاربران همه حالات را در نظر نگرفته و یا بسیار پیچیده است. همچنین در کاربردهای انعطاف-پذیر چنین مجموعه‌ای از قبل وجود ندارد. به همین دلیل در این مقاله روشی ارائه شده است که در آن فرض می‌شود مدل و لاگ نرمالی در اختیار نداریم. به این ترتیب مدل مرجع در طول فرایند تشخیص ناهنجاری از روی داده‌های لاگ واقعی پیدا می‌شود. این مدل بیشترین تناسب را با داده‌ها، ترتیب فعالیت‌ها و رفتار لاگ دارد. چنین مدلی، مدل مناسب[2] خوانده می‌شود. پس از یافتن پویای مدل مناسب در سیستم، از این مدل به عنوان مرجع استفاده شده و رفتار جاری سیستم با آن مقایسه می‌شود و هرگونه عدم تطابق به عنوان حمله در نظر گرفته می‌شود.



شکل ۲- قدم‌های تشخیص ناهنجاری

در شکل ۲ جزئیات کار مرحله تشخیص ناهنجاری نشان داده شده است، که در ادامه هر قدم شرح داده خواهد شد.

۳-۲-۱- پیش پردازش جریان کار

این قدم لاگ پیش پردازش شده حاصل مرحله قبل را به عنوان ورودی دریافت کرده و در صورت لزوم پیش پردازش بیشتری در بعد جریان کار دنباله‌ها انجام می‌دهد. برای مثال می‌توان فعالیت-هایی که در مرحله تشخیص سوءاستفاده بررسی خواهند شد را از این مرحله حذف کرد تا کار یافتن مدل مناسب سریعتر و راحتتر شود.

۳-۲-۲. یافتن مدل مناسب

برای جداسازی دنباله نرمال و غیر نرمال در لاگ، به یک مدل فرایند مناسب احتیاج است. این مدل باید کامل و دقیق بوده و به خوبی بر رفتار لاگ منطبق باشد. فرایندکاوی را در بعد جریان کار می‌توان به عنوان روش جستجویی در نظر گرفت که بهترین مدل فرایند را از بین فضای جستجوی مدل‌های داوطلب پیدا می‌کند. از میان تکنیک‌های فرایندکاوی موجود در بعد جریان کار، روش فرایندکاوی ژنتیکی برای کشف مدل مناسب انتخاب شده است.

فرایندکاوی ژنتیکی در مقایسه با دیگر تکنیک‌های فرایندکاوی، می‌تواند تمام ساختارهای ممکن در یک لاگ را بررسی کند (توالی، همزمانی، انتخاب، حلقه، non-free-choice، فعالیت‌های پنهان و تکراری) و نسبت به نویز مقاوم است. به دلیل همزمانی موجود در الگوریتم ژنتیک، این الگوریتم می‌تواند یک فضای جستجوی بزرگ را در زمان مناسب بررسی کند. همچنین یک سیستم مبتنی بر الگوریتم ژنتیک قابل آموزش مجدد است و در نتیجه برای کاربردهای انعطاف پذیر مناسب است.

الگوریتم ژنتیک استفاده شده در این قدم توسط Genetic Algorithm Plugin موجود در چارچوب ProM حمایت می‌شود. لاگ پیش پردازش شده قدم قبل به عنوان ورودی به plugin داده می‌شود و مدل مناسب پیدا شده در طول کاوش که بهترین توصیف از رفتار لاگ را دارد در خروجی بدست می‌آید.

[1] Plugable

[2] Appropriate model

خواننده برای یافتن جزئیات بیشتر راجع به فرایندکاوی ژنتیکی به [۱۱-۱۳] ارجاع داده می‌شود.

۳-۳-۳- یافتن عدم تطابق‌های لاگ و مدل مناسب

در آخرین قدم مرحله تشخیص ناهنجاری، پس از یافتن مدل مناسب منطبق بر لاگ در قدم قبل، جداسازی دنباله‌های نرمال و ناهنجار از یکدیگر انجام می‌شود. یک دنباله ناهنجار[1]، دنباله‌ای در نظر گرفته می‌شود که بر مدل فرایند مناسب پیدا شده در قدم قبل منطبق نباشد. چنین دنباله‌ای یک حمله در نظر گرفته می‌شود (ناهنجاری).

این قدم توسط *Conformance Checker Plugin* موجود در چارچوب ProM قابل پیاده سازی است. این plugin، لاگ و یک مدل مرجع را به عنوان ورودی دریافت کرده و هر یک از دنباله‌های موجود در لاگ را به منظور یافتن عدم تطابق با مدل مرجع مقایسه می‌کند. لاگ پیش پردازش شده جریان کار و مدل مناسب پیدا شده در قدم‌های قبل به عنوان وروردی‌های این قدم هستند. به این ترتیب، دنباله‌های موجود در لاگ به این شکل تقسیم می‌شوند: ۱) دنباله منطبق بر مدل مرجع به عنوان نرمال و ۲) دیگر دنباله‌ها به عنوان دنباله‌های ناهنجار [۸].

خواننده برای یافتن جزئیات بیشتر راجع به مرحله تشخیص ناهنجاری به [۸] ارجاع داده می‌شود.

۳-۳- تشخیص سوءاستفاده

تشخیص‌دهنده سوءاستفاده، رخدادهای سیستم را نظارت و تجزیه و تحلیل کرده و به دنبال یک واقعه یا دنباله‌ای از وقایع می‌گردد که یک حمله را نشان می‌دهند. این واقعه یا دنباله‌ای از وقایع در فرم یک امضا ذخیره شده‌اند. هر امضا در واقع الگوی رفتار یک حمله را نشان می‌دهد. الگوی حملات می‌توانند بصورت قوانین[2] نوشته شوند و بر روی لاگ سیستم مورد بررسی قرار گیرند. هر دنباله‌ای از لاگ که بر الگو منطبق باشد، یک دنباله مورد سوءاستفاده را نشان می‌دهد. برای افزایش دقت تشخیص و کاهش نرخ هشدار اشتباه تشخیص ناهنجاری از تشخیص سوءاستفاده نیز در این مرحله بر روی لاگ استفاده می‌شود. تشخیص سوءاستفاده این مرحله در بعد سازمانی فرایندکاوی انجام گرفته و حملات مربوط به این بعد را کشف می‌کند.

چهار نوع حمله برای پیاده سازی این مرحله در نظر گرفته شده‌اند: **حدس کلمه عبور**[3]، کاربری بیش از سه بار برای ورود به سیستم از طریق وارد کردن نام کاربری و کلمه عبور تلاش کرده باشد. **کاربر به ریشه**[4]، نوعی ارتقاء امتیاز کاربران مجاز است. به طوریکه یک کاربر مجاز دسترسی غیرمجازی به هر بخشی از سیستم داشته باشد. **کاربر راه دور به ریشه**[5]، دسترسی غیرمجاز یک کاربر راه دور غیرمجاز به هر بخشی از سیستم است. **سوءاستفاده مدیر**، مدیر سیستم از امتیازات خود سوءاستفاده کرده و در فرایندهایی که مربوط به دیگر کاربران است دخالت کرده باشد. لازم به ذکر است چهار حمله فقط به منظور سرعت و دقت بیشتر پیاده سازی در نظر گرفته شده اند و هر تعداد و نوع حمله دیگری نیز قابل پیاده سازی است.

به منظور نوشتن قوانین مربوط به حملات در مرحله تشخیص سوءاستفاده، از زبان LTL[6] استفاده شده است. زبان LTL ویژگی‌های لاگ‌های وقایع ثبت شده از فرایندها را بررسی می‌کند. در لاگ یک فرایند، نمونه‌های مختلف اجرای آن فرایند وجود دارد که هر کدام از نهادهای فعالیت تشکیل شده‌اند. در بررسی یک لاگ توسط زبان LTL، نمونه‌های فرایند یک به یک و در هر نمونه، فعالیت‌ها به ترتیب بررسی شده و قوانین نوشته شده چک می‌شوند. قوانین در قالب فرمول نوشته می‌شوند و هر فرمول قانون مربوط به یک حمله است. فرمول‌ها با استفاده از عبارت‌های منطقی، سوری[7]، منطق linear temporal و ترکیب آن‌ها نوشته می‌شوند [۱۴]. به این ترتیب چهار فرمول برای چهار نوع حمله ذکر شده نوشته شده است. این فرمول‌ها توسط *LTL Checker* نرم‌افزار ProM بررسی می‌شوند. ورودی این مرحله لاگ پیش پردازش شده است که این چهار فرمول یک به یک بر روی آن اجرا می‌شوند و سوءاستفاده‌های موجود در لاگ در خروجی مشخص می‌شوند.

۴-۳- ترکیب نتایج

در دو مرحله موازی قبل، ناهنجاری‌های بعد جریان کار و سوءاستفاده‌های بعد سازمانی موجود در لاگ پیش پردازش شده کشف شده‌اند. وظیفه این مرحله دریافت این نتایج به عنوان ورودی و ادغام آن‌ها است. به طوریکه در خروجی، هر یک از فرایندهای مورد حمله به همراه نوع حملات مشخص می‌شوند. حملات شامل ناهنجاری‌های موجود در ترتیب فعالیت‌ها و چهار نوع حمله مرحله تشخیص سوءاستفاده هستند. فرایندهای مورد حمله حداقل یک تا پنج نوع حمله را دارا هستند.

برنامه *Combinator* که وظیفه ادغام نتایج مراحل قبل را دارد، به زبان جاوا نوشته شده و پیاده سازی این مرحله را بر عهده دارد.

۴- ارزیابی!

برای ارزیابی مدل تشخیص نفوذ مبتنی بر میزبان فرایندمحور پیشنهادی از یک مجموعه-داده[8] مصنوعی استفاده شده است. چرا که اولاً، پیدا کردن حملات در یک لاگ واقعی سخت است. چون

[1] Anomalous trace
[2] Rules
[3] Password guessing
[4] User to Root
[5] Remote to Root
[6] Linear Temporal Language
[7] Quantificational
[8] Data-set

برای خود افراد همه حملات موجود در لاگ واقعی بطور کامل شناخته شده نیستند که بعد بتوانند برای ارزیابی مدل تشخیص نفوذ استفاده شوند. دوماً، یک لاگ واقعی برای استفاده در دسترس نیست.

در ادامه مراحل تولید مجموعه-داده ارزیابی، نحوه انجام ارزیابی و نتایج حاصله آورده شده است.

۴-۱- تولید مجموعه- داده ارزیابی

به منظور تولید مجموعه-داده ارزیابی، فرایند فروش اجناس در یک کارخانه سازنده کاشی و سرامیک به عنوان فرایند اصلی در نظر گرفته شده است. این فرایند شامل نه فعالیت و پنج نقش که هر کدام یک یا چند کاربر دارند، است. در جریان فروش اجناس در این کارخانه، این فرایند فروش توسط افراد و با استفاده از نرم‌افزار نصب شده بر روی شبکه کارخانه طی می‌شود و همه وقایع انجام شده توسط افراد مختلف بطور کامل در میزبان شبکه ثبت می‌شوند. از مشخصات این سیستم واقعی استفاده شده و لاگ تولیدی توسط نرم‌افزار CPN Tools شبیه سازی شده است.

CPN Tools نرم‌افزاری است برای ساخت و تجزیه و تحلیل مدل‌های CPN[1]. یک مدل CPN از یک سیستم، مدلی قابل اجراست که حالات سیستم و رخدادهایی که باعث تغییر حالات سیستم می‌شوند را نشان می‌دهد. این مدل توسط زبان CPN که یک زبان مدلسازی اتفاقات گسسته توسط ترکیب شبکه پتری و زبان استاندارد ML است، بصورت گرافیکی طراحی و پیاده سازی می‌شود [۱۵]. مدل CPN طراحی شده از فرایند فروش توسط CPN Tools شبیه‌سازی شده و نتایج حاصل توسط نرم‌افزار $ProM_{Import}$ ترکیب و به یک لاگ آماده استفاده تبدیل شده است [۱۶].

۴-۲- نحوه انجام ارزیابی

برای انجام ارزیابی بر روی داده‌های تولید شده، از روش stratified 10-fold Cross Validation[2] استفاده شده است. در این روش مجموعه داده‌ها به ۱۰ قسمت[3] مساوی و با پراکندگی یکسان از داده‌های نرمال و مورد حمله تقسیم می‌شوند و ده بار مدل اجرا می‌شود. در هر بار ۹ قسمت داده به عنوان مجموعه آموزشی[4] و یک قسمت دیگر به عنوان مجموعه تست[5] در نظر گرفته می‌شوند، که در هر بار اجرا مجموعه‌های آموزشی و تست تغییر می‌کنند. پس از پایان کار از نتایج حاصل از هر اجرا میانگین گرفته می‌شود.

در این کار، لاگ تولید شده توسط CPN Tools دارای ۵۰۰ نمونه اجرای مختلف و تصادفی از فرایند فروش اجناس است. از این ۵۰۰ نمونه، ۲۰٪ نمونه‌ها دارای حمله هستند (هر نمونه ممکن است یک یا چند حمله از انواع مختلف را دارا باشد). داده‌ها به ۱۰ قسمت ۵۰ تایی تقسیم شده است، هر قسمت نمونه‌ای مناسب از کل داده‌ها بوده و دارای ۲۰٪ دنباله مورد حمله (۱۰ تا از ۵۰ نمونه) است. ۱۰ بار مدل تشخیص نفوذ اجرا شده است و در هر بار اجرا ۹ قسمت مجموعه آموزشی ادغام شده و مدل مناسب طبق آن پیدا می‌شود. یک قسمت باقیمانده مجموعه تست بوده و برای انجام تشخیص سوءاستفاده و تشخیص ناهنجاری طبق مدل مناسب، استفاده شده است.

۴-۳- نتایج

برای ارزیابی نتایج بدست آمده از اجرا، از سه متغیر نرخ هشدار اشتباه (FPR[6])، نرخ تشخیص درست (TPR[7]) و دقت تشخیص (Accuracy) استفاده می‌شود. فرمول مربوط به نحوه محاسبه هر یک از متغیرها در فرمول ۱ آورده شده است.

$$FPR=\frac{FP}{N},TPR=\frac{TP}{P},Accuracy=\frac{TP+TN}{P+N} \quad (۱)$$

FPR، نسبت تعداد دنباله‌هایی که مورد حمله نیستند ولی به اشتباه در دسته حملات قرار گرفته‌اند (FP) به تعداد کل دنباله-های نرمال (N) است. TPR، نسبت تعداد دنباله‌هایی که مورد حمله هستند و درست تشخیص داده شده‌اند (TP) به تعداد کل دنباله‌های مورد حمله (P) است. Accuracy، نسبت تعداد دنباله-هایی (نرمال یا حمله) که درست تشخیص داده شده‌اند (TP+TN) به تعداد کل دنباله‌ها (P+N) است.

مقادیر محاسبه شده متغیرها پس از اجرای مدل تشخیص نفوذ در جدول ۱ آورده شده است. این مقادیر مربوط به مرحله تشخیص ناهنجاری مدل ارائه شده هستند و با نتایج حاصل از روش نمونه‌برداری در [۵] که به عنوان روش بهتر معرفی شده است، مقایسه شده‌اند. مرحله تشخیص سوءاستفاده به این دلیل در مقایسه قرار نگرفته است که در هیچ یک از کارهای قبل از این تکنیک استفاده نشده است.

جدول ۱- نتایج ارزیابی مرحله تشخیص ناهنجاری

	Accuracy	TPR	FPR
تشخیص نفوذ فرایندمحور	٪۹۴,۴	٪۸۶,۲۵	٪۳,۵
نمونه برداری	٪۸۵,۹۶	٪۷۴,۰۴	٪۱۱,۰۱

همانطور که از نتایج مشخص است، در مدل تشخیص نفوذ مبتنی بر میزبان فرایندمحور ارائه شده در این مقاله نرخ هشدار اشتباه کاهش یافته و دقت تشخیص بالا رفته است. چرا که در روش نمونه‌برداری، مدل مناسب براساس نمونه‌ای حاوی نیمی از

[1] Coulered Petri Net

[2] طبق مقایسه‌هاي انجام شده، این روش در بین انواع روش‌هاي Cross Validation براي محاسبه دقت تشخیص بهترین روش است [۱۷].

[3] Fold

[4] Training set

[5] Test set

[6] False Positive Rate

[7] True Positive Rate

دنباله‌های موجود در لاگ ساخته شده و سپس هر دنباله‌ای که بر آن منطبق نباشد، ناهنجار در نظر گرفته می‌شود. در نتیجه نرخ هشدار اشتباه بالا بوده و دقت تشخیص کاهش می‌یابد. اما در تشخیص نفوذ مبتنی بر میزبان فرایندمحور، مدل مناسب با استفاده از الگوریتم ژنتیک براساس همه دنباله‌های موجود در لاگ تولید می‌شود و هر دنباله با این مدل مناسب کامل مقایسه می‌شود. در نتیجه میزان هشدار اشتباه کاهش یافته و نرخ تشخیص درست بالا می‌رود.

مقادیر محاسبه شده متغیرها پس از اجرا برای کل مدل که حاصل ترکیب تشخیص ناهنجاری و سوءاستفاده است، در جدول ۲ آورده شده‌اند.

جدول ۲- نتایج ارزیابی مدل تشخیص نفوذ فرایندمحور!

	Accuracy	TPR	FPR
تشخیص نفوذ فرایندمحور	٪۹۴,۸	٪۸۹	٪۳,۷

همانطور که ملاحظه می‌کنید، نرخ هشدار اشتباه در این مدل به دلیل استفاده از الگوریتم ژنتیک در تشخیص ناهنجاری و ترکیب با تشخیص سوءاستفاده بسیار پایین است و دقت تشخیص نیز مقدار خوبی را دارا می‌باشد. در حالت ترکیبی مقدار TPR نسبت به حالت تشخیص ناهنجاری افزایش پیدا کرده است، چرا که در تکنیک تشخیص سوءاستفاده به دلیل استفاده از قوانین مشخص، حملات با دقت بسیار بالایی کشف می‌شوند.

۵- نتیجه‌گیری!

امروزه، سازمان‌ها به منظور حمایت بهتر از فرایندهای حاکم بر خود به سوی استفاده از سیستم‌های مبتنی بر فرایند روی آورده‌اند. اما یکی از نکات مهم در هر سیستمی برقراری امنیت در آن است و یکی از بهترین مکانیسم‌های فراهم کردن امنیت، استفاده از سیستم‌های تشخیص نفوذ مبتنی بر میزبان است. به همین دلیل در این مقاله سعی شده است با استفاده از تکنیک‌های فرایندکاوی به عنوان یک روش فرایندمحور مناسب، مدلی برای تشخیص نفوذ مبتنی بر میزبان ارائه شود که نرخ هشدار اشتباه را کاهش داده و دقت تشخیص را بالا ببرد. به همین منظور در این مدل از دو تکنیک تشخیص ناهنجاری و تشخیص سوءاستفاده بصورت ترکیبی برای تشخیص حملات استفاده شده تا از مزایای هر دو تکنیک استفاده شده و هر یک عیب دیگری را بپوشاند. همچنین در این روش هر دو بعد جریان کار و سازمانی در فرایندکاوی مورد بررسی قرار گرفته‌اند.

نتایج بدست آمده از اجرا و ارزیابی مدل در بعد تشخیص ناهنجاری در مقایسه با کارهای قبل (جدول۱)، ۸٪ کاهش در نرخ هشدار اشتباه و ۹٪ افزایش در دقت تشخیص را نشان می‌دهد. و نتایج حاصل از اجرای کامل مدل بصورت ترکیب دو تکنیک نیز نشان دهنده نرخ هشدار اشتباه مناسب ۳,۷٪ و دقت تشخیص ۹۴,۸٪ است، همچنین نرخ تشخیص درست نیز به دلیل دقت بالای تشخیص سوءاستفاده ۳٪ افزایش یافته است.

مراجع

[1] Muscat Andre', "A Log Analysis based Intrusion Detection System for the creation of a Specification Based Intrusion Prevention System", Department of Computer Science and AI, University of Malta, Jul 2003.

[2] Debar Hervé, "An Introduction to Intrusion-Detection Systems", Proceedings of Connect'2000, Doha, Qatar, April 2000.

[3] Ehret Christoph, and Ultes-Nitsche Ulrich, "Immune System Based Intrusion Detection System", Department of Computer Science, University of Fribourg, 2008.

[4] Aalst W.M.P. van der, and de Medeiros A.K.A., "Process mining and security: Detecting anomalous process executions and checking process conformance," Electronic Notes in Theoretical Computer Science, vol. 121(4), pp. 3–21, 2005.

[5] Bezerra F., and Wainer J., "Anomaly detection algorithms in logs of process aware systems," SAC 2008: Proceedings of the 2008 ACM symposium on Applied computing, ACM Press, pp. 951–952, New York, 2008.

[6] Bezerra F., and Wainer J., "Anomaly detection algorithms in business process logs," ICEIS 2008: Proceedings of the Tenth International Conference on Enterprise Information Systems, AIDSS, pp. 11–18, Barcelona, Spain, June 2008.

[7] Bezerra Fabio, Wainer Jacques, and Aalst W. van der, "Anomaly detection using process mining," Springer-Verlag, pp. 149–161, Berlin Heidelberg, 2009.

[8] Jalali Hanieh, and Baraani Ahmad, "Genetic-based Anomaly Detection in Logs of Process Aware Systems", Proceedings of WASET 2010: International Conference of Computer Science and Systems Security, pp. 251-6, Rome, April 2010. !

[9] Aalst W.M.P. van der, and Weijters A.J.M.M., "Process Mining", Department of Technology Management, Eindhoven University of Technology. !

[10] Bace Rebecca, and Mell Peter, "Intrusion Detection Systems", NIST Special Publication on Intrusion Detection System. !

[11] Aalst W.M.P. van der, de Medeiros A.K. Alves, and Weijters A.J.M.M., "Genetic process mining", Applications and theory of Petri nets, Springer, 2005.

[12] De Medeiros A.K.A., Weijters A.J.M.M., and Aalst W.M.P. van der, "Using genetic algorithms to mine process models: Representation, operators and results," BETA Working Paper Series, WP 124, Eindhoven University of Technology, Eindhoven, 2004.

[13] De Medeiros Ana Karla Alves, "Genetic Process Mining," Eindhoven University of Technology, ISBN 978-90-386-0785-6, 2006.

[14] De Beer HT, and Brand PCW van den, "The LTL Checker Plugins a (reference) manual", Eindhoven University of Technology, 2007. !

[15] Jensen Kert, Kristensen Lars Micheal, and Wells Lisa, "Coloured Petri Nets and CPN Tools for Modelling and Validation of Concurrent Systems", International Journal on Software Tools for Technology Transfer (STTT), Volume 9, Numbers 3-4, June, 2007.

[16] De Medeiros A.K. Alves, and Gunther C.W., "Process Minning: Using CPN Tools to Create Test Logs for

Mining Algorithms", Department of Technology Management, Eindhoven University of Technology.
[17] Rafeilzadeh Payam, Tang Lei, and Liu Huan, "Cross Validation", Arizona State University.