شماره:

تاریخ:

فرم پروژه تحصیلات تکمیلی ۱

پیشنهاد پروژه تحصیلات تکمیلی

(رساله کارشناسی ارشد و دکترا)

دانشگاه صنعتی
امیرکبیر
(پلی تکنیک تهران)

۱- مشخصات دانشجو

نام و نام خانوادگی: سید محمدمهدی احمدپناه
شماره دانشجویی: ۹۴۱۳۱۰۸۶
رشته تحصیلی: مهندسی فناوری اطلاعات – امنیت اطلاعات
دانشکده: مهندسی کامپیوتر و فناوری اطلاعات
آدرس: دانشکده مهندسی کامپیوتر و فناوری اطلاعات، آزمایشگاه امنیت صوری
تلفن : ۰۹۳۹۰۳۶۱۳۰۳ مقطع: کارشناسی ارشد

۲-مشخصات استاد راهنما

نام و نام خانوادگی: دکتر مهران سلیمان‌فلاح
سمت، مرتبه علمی و محل خدمت: دانشیار، دانشگاه صنعتی امیرکبیر
آدرس: دانشگاه صنعتی امیرکبیر، دانشکده مهندسی کامپیوتر و فناوری اطلاعات
تلفن: ۰۲۱۶۴۵۴۲۷۱۸

۳-مشخصات استاد مشاور

نام و نام خانوادگی:
سمت، مرتبه علمی:
تلفن :

۴-عنوان پایان نامه یا رساله

فارسی: بهبود مکانیزم‌های مبتنی بر چنداجرایی برای اِعمال خط‌مشی‌های جریان اطلاعات

انگلیسی: Improving Multi-Execution-based Mechanisms for Enforcing Information Flow Policies

نوع پروژه: کاربردی ☐ بنیادی ■ توسعه‌ای ■ **تعداد واحد** ۶

۵- خلاصه پایان‌نامه: (مسئله، فرضیات، هدف از اجرا، توجیه ضرورت انجام طرح)

خط‌مشی امنیتی، تعریفی از امن‌بودن یک سامانه یا برنامه را ارائه می‌دهد که رفتارهای مجاز و غیرمجاز، در آن مشخص می‌شود. خط‌مشی‌های جریان اطلاعات، خط‌مشی‌های محرمانگی[۱] و صحت[۲] هستند که انتشار داده‌ها را در برنامه کنترل می‌کنند. یکی از خط‌مشی‌های محرمانگیِ مهم برای امنیت جریان اطلاعات، عدم تداخل[۳] است. یک برنامه عدم تداخل را برآورده می‌کند اگر هیچ دو اجرایی با مقادیر ورودی عمومی یکسان، که ممکن است در مقادیر ورودی محرمانه متفاوت باشند، خروجی‌های عمومی متفاوتی نداشته باشند.

خط‌مشی‌های امنیتی را می‌توان به دو دسته خاصیت[۴] و فوق خاصیت[۵] تقسیم‌بندی کرد. همان‌طور که می‌دانیم، یک سامانه، شامل مجموعه‌ای از اجراها است. یک خط‌مشی امنیتی را خاصیت می‌نامند اگر بتوان آن را با مجموعه‌ای از اجراهای دارای رفتار مجاز بیان کرد. برای نمونه، می‌توان به خط‌مشی‌های کنترل دسترسی اشاره کرد. خط‌مشی‌هایی مانند عدم تداخل، خاصیت نیستند؛ یعنی آن‌ها را باید با مجموعه توانیِ مجموعه اجراها بیان کرد. به این‌گونه خط‌مشی‌ها، فوق خاصیت گفته می‌شود [۱]. بنابراین، روش اِعمال خاصیت‌ها با نحوه اعمال فوق خاصیت‌ها متفاوت است.

اِعمال خط‌مشی‌های جریان اطلاعات، یک مسئله چالش‌برانگیز است. مکانیزم‌های اِعمال، به دنبال دست‌یابی به این اهداف هستند [۲]: ۱) درستی[۶]: اجازه وقوع جریان غیرمجاز اطلاعات در طول اجرا داده نشود. ۲) دقت[۷]: از اجرای امن برنامه‌ها جلوگیری نشود. ۳) عملی‌بودن[۸]: هزینه اعمال مکانیزم قابل قبول باشد. هزینه‌ها ممکن است در زمان توسعه، استقرار[۹] یا اجرای برنامه باشد. گرچه تلاش‌های بسیاری در دهه‌های اخیر برای پاسخ به این مسئله شده است، اما کماکان مکانیزم‌های اعمالی که به طور همزمان به همه این اهداف دست یابند، مطرح نشده است.

دو دسته کلی برای مکانیزم‌های اعمال خط‌مشی‌های جریان اطلاعات وجود دارد. از جمله رویکردهای ایستا می‌توان به مکانیزم‌های مبتنی بر نوع[۱۰] [۳] و مکانیزم‌های مبتنی بر راستی‌آزمایی[۱۱] [۴] اشاره کرد. این‌گونه مکانیزم‌ها، دارای درستی هستند و هزینه‌ای در زمان اجرا یا استقرار

[1] Confidentiality
[2] Integrity
[3] Non-interference
[4] Property
[5] Hyperproperty
[6] Soundness
[7] Precision
[8] Practicality
[9] Deployment
[10] Type-based

تحمیل نمی‌کنند. با این حال، مکانیزم‌های مبتنی بر نوع دقیق نیستند و ممکن است برنامه‌های امن زیادی توسط آن‌ها پذیرفته نشوند و محافظه‌کار هستند. اما مکانیزم‌های مبتنی بر راستی‌آزمایی، برحسب کامل‌بودن[12] منطق برنامه، ممکن است از دقت کامل برخوردار باشند [۲]. همچنین، هم مکانیزم‌های مبتنی بر نوع و هم مکانیزم‌های مبتنی بر راستی‌آزمایی، هزینه زمان توسعه زیادی دارند.

رویکردهای پویا، که در سال‌های اخیر توجه بیشتری به آن‌ها شده است، شامل ناظرهای زمان‌اجرا[13] [۵، ۶] و تکنیک چنداجرایی امن[14] (SME) [۷، ۸] می‌شود. مکانیزم‌های ذکرشده درستی دارند و می‌توانند نسبت به بعضی از مکانیزم‌های ایستا، برای خط‌مشی‌های بیشتری دقت را فراهم کنند. به عنوان نمونه، ناظرهای زمان‌اجرا، برنامه‌های کمتری نسبت به مکانیزم‌های مبتنی بر نوع را رد می‌کنند. در نظارت زمان‌اجرا، برخلاف مکانیزم‌های ایستا، حالت‌های برنامه‌ی در حال اجرا توسط ناظر بررسی شده و در صورت امکان ورود به حالت ناامن ادامه اجرا متوقف خواهد شد یا با اعمال تغییراتی به اجرای امن تبدیل می‌شود. البته اثبات شده است [۱۱] که عدم تداخل، با توجه به این‌که یک خاصیت ایمنی[15] نیست، توسط ناظرهای اجرا قابل اعمال نیست.

مفهوم اصلی تکنیک چنداجرایی امن [۷] آن است که به ازای هر سطح امنیتی، یک اجرا از برنامه انجام شود. به این ترتیب که ورودی‌ها، در اجراهای مربوط به سطح امنیتی خود یا بالاتر، مقدار می‌گیرند و در غیر این صورت، با مقادیر پیش‌فرض جایگزین می‌شوند. خروجی‌ها نیز فقط در اجرای مربوط به سطح امنیتی خود تولید می‌شوند. ضمناً با توجه به این‌که اجراهای سطح بالا از ورودی‌های سطح پایین هم استفاده می‌کنند، اثرات جانبی ورودی‌ها نیز در نظر گرفته می‌شوند. با توجه به تفکیک یک اجرای برنامه به چندین اجرا به ازای هر سطح امنیتی، استراتژی زمان‌بندی[16] این اجراها از نکات مهم این روش به شمار می‌رود. نشان داده می‌شود که این روش درستی را تضمین می‌کند.

چنداجرایی امن هزینه توسعه ندارد، اما چنداجرایی امن را نمی‌توان به سادگی اعمال کرد زیرا همه پیاده‌سازی‌های چنداجرایی امن، نیازمند ایجاد اصلاح‌هایی در زیرساخت محاسباتی مانند سیستم عامل، مرورگر وب و یا ماشین مجازی است [۲]. نکته دیگر آن که در تکنیک چنداجرایی، امکان تشخیص تغییر معناشناخت[17] برنامه وجود ندارد و هیچ تضمینی برای ترتیب نسبی خروجی‌های سطوح مختلف امنیتی نمی‌دهد [۱۱].

در این پژوهش، با بررسی مزایا و معایب مکانیزم‌های موجودِ اعمال خط‌مشی‌های جریان اطلاعات، می‌خواهیم دقت مکانیزم‌های مبتنی بر چنداجرایی را افزایش دهیم. به دنبال آن هستیم که برای یک زبان برنامه‌نویسی مدل، با به کارگیری تکنیک چنداجرایی و استفاده از روش‌های دیگر مانند خودترکیبی [۴] و نظارت زمان اجرا، مکانیزم بهتری در این خصوص ارائه کنیم. با انجام این پژوهش، از حیث دقت مکانیزم، اعمال مبتنی بر تکنیک چنداجرایی برای فوق خاصیت محرمانگی جریان اطلاعات را بهبود خواهیم داد و با اثبات صوری درستی و دقت مکانیزم بهبودیافته، به ارزیابی و مقایسه آن با سایر مکانیزم‌ها تحت خط‌مشی‌های مختلف می‌پردازیم. به این ترتیب، توسعه‌دهندگان برنامه‌های کاربردی، هزینه کم‌تری برای تضمین امنیت نرم‌افزار متحمل خواهند شد.

۶-کلمات کلیدی فارسی:

جریان اطلاعات؛ چنداجرایی امن؛ فوق خاصیت امنیتی؛ اِعمال پویا؛ نظارت زمان‌اجرا

کلمات کلیدی انگلیسی:

Information Flow, Secure Multi-Execution, Security Hyperproperty, Dynamic Enforcement, Run-time Monitoring

۷- مدت زمان اجرای پایان‌نامه به ماه: ۱۲ ماه **تاریخ شروع:** مرداد ۱۳۹۵

۸- مراحل اجرای پایان‌نامه	۱۲	۱۱	۱۰	۹	۸	۷	۶	۵	۴	۳	۲	۱
مطالعه فوق خط‌مشی‌های جریان اطلاعات											■	■
مطالعه انواع مکانیزم‌های پویا، شامل چنداجرایی و نظارت زمان‌اجرا								■	■	■		
بیان صوری خط‌مشی جریان اطلاعات و زبان انتخاب‌شده						■	■					
ارائه مکانیزم مبتنی بر چنداجرایی بهبودیافته				■	■							
اثبات درستی و دقت مکانیزم ارائه‌شده و مقایسه آن با مکانیزم‌های موجود		■	■									
نگارش پایان‌نامه	■											

[11] Verification-based
[12] Completeness
[13] Execution Monitor
[14] Secure Multi-Execution
[15] Safety Property
[16] Scheduling Strategy
[17] Semantics

۹- روش پژوهش و تکنیک‌های اجرایی:

برای پاسخ‌گویی به سوال پژوهش، ابتدا باید با بررسی خط‌مشی‌های مطرح‌شده در سابقه علمی این حوزه، خط‌مشی جریان اطلاعات مورد نظر را تعیین کرد. انتخاب این خط‌مشی، بستگی به سطح انتزاع و موازنه[18] کاربردی‌بودن-مدل‌بودن خط‌مشی دارد. ضمناً باید محدودیت‌های تکنیک چنداجرایی برای اعمال خط‌مشی‌ها را نیز در نظر داشت. در ادامه، متناسب با خط‌مشی انتخاب‌شده، یک زبان برنامه‌نویسی مدل برای بیان صوری آن خط‌مشی معرفی می‌شود. بیان صوری نحو[19] و معناشناخت آن زبان، گام بعدی است. پس از آن، با بررسی مکانیزم‌های پویا، مقایسه مکانیزم‌های اعمال مختلف و دسته‌بندی نقاط قوت و ضعف هر یک، با توجه به عناصر زبان برنامه‌نویسی مطرح‌شده، مکانیزمی مبتنی بر تکنیک چنداجرایی، برای اعمال فوق خاصیت امنیتی جریان اطلاعات ارائه خواهد شد که در برابر مکانیزم‌های موجود، از منظر دقت، برتری داشته باشد. همان‌طور که قبل‌تر اشاره شد، اصلی‌ترین معیارهای مقایسه مکانیزم‌های اعمال، همان درستی و دقت روش ارائه‌شده خواهد بود. به همین منظور، برای راستی‌آزمایی مکانیزم ارائه‌شده، از رویکرد اثبات درستی و دقت استفاده خواهد شد.

۱۰- سابقه علمی و فهرست منابع:

در پیوست آمده است.

[1] Clarkson, Michael R., and Fred B. Schneider. "Hyperproperties." Journal of Computer Security 18.6 (2010): 1157-1210.

[2] Barthe, Gilles, et al. "Secure multi-execution through static program transformation." Formal Techniques for Distributed Systems. Springer Berlin Heidelberg, 2012. 186-202.

[3] Volpano, Dennis, Cynthia Irvine, and Geoffrey Smith. "A sound type system for secure flow analysis." Journal of computer security 4.2-3 (1996): 167-187.

[4] Barthe, Gilles, Pedro R. D'argenio, and Tamara Rezk. "Secure information flow by self-composition." Mathematical Structures in Computer Science 21.06 (2011): 1207-1252.

[5] Le Guernic, Gurvan. "Confidentiality enforcement using dynamic information flow analyses". PhD thesis, Kansas State University, 2007.

[6] Austin, Thomas H., and Cormac Flanagan. "Permissive dynamic information flow analysis." Proceedings of the 5th ACM SIGPLAN Workshop on Programming Languages and Analysis for Security. ACM, 2010.

[7] Devriese, Dominique, and Frank Piessens. "Noninterference through secure multi-execution." 2010 IEEE Symposium on Security and Privacy. IEEE, 2010.

[8] Capizzi, Roberto, et al. "Preventing information leaks through shadow executions." Computer Security Applications Conference, 2008. ACSAC. Annual. IEEE, 2008.

[9] Hamlen, Kevin W., Greg Morrisett, and Fred B. Schneider. "Computability classes for enforcement mechanisms." ACM Transactions on Programming Languages and Systems (TOPLAS) 28.1 (2006): 175-205.

[10] Hamlen, Kevin. "Security policy enforcement by automated program-rewriting." PhD thesis, Cornell University, 2006.

[11] Zanarini, Dante, Mauro Jaskelioff, and Alejandro Russo. "Precise enforcement of confidentiality for reactive systems." Computer Security Foundations Symposium (CSF), 2013 IEEE 26th. IEEE, 2013.

[12] Kashyap, Vineeth, Ben Wiedermann, and Ben Hardekopf. "Timing-and termination-sensitive secure information flow: Exploring a new approach." Security and Privacy (SP), 2011 IEEE Symposium on. IEEE, 2011.

[13] Jaskelioff, Mauro, and Alejandro Russo. "Secure multi-execution in haskell." Perspectives of Systems Informatics. Springer Berlin Heidelberg, 2011. 170-178.

[14] Bielova, Nataliia, et al. "Reactive non-interference for a browser model." Network and System Security (NSS), 2011 5th International Conference on. IEEE, 2011.

[15] Austin, Thomas H., and Cormac Flanagan. "Multiple facets for dynamic information flow." ACM SIGPLAN Notices. Vol. 47. No. 1. ACM, 2012.

[16] Bielova, Nataliia, and Tamara Rezk. "A Taxonomy of Information Flow Monitors." Principles of Security and Trust. Springer Berlin Heidelberg, 2016. 46-67.

[17] Russo, Alejandro, and Andrei Sabelfeld. "Dynamic vs. static flow-sensitive security analysis." Computer Security Foundations Symposium (CSF), 2010 23rd IEEE. IEEE, 2010.

[18] Rafnsson, Willard, and Andrei Sabelfeld. "Secure multi-execution: fine-grained, declassification-aware, and transparent." Computer Security Foundations Symposium (CSF), 2013 IEEE 26th. IEEE, 2013.

[19] Agrawal, Shreya. "Monitoring and Enforcement of Safety Hyperproperties." (2015), Master Thesis, Waterloo University.

[20] Khoury, Raphaël, and Nadia Tawbi. "Which security policies are enforceable by runtime monitors? a survey." Computer Science Review 6.1 (2012): 27-45.

۱۱- وسایل و تجهیزات مورد نیاز:

یک دستگاه کامپیوتر با قابلیت اتصال به اینترنت

[18] Trade-off

[19] Syntax

۱۲- اعتبار اجرای پایان‌نامه و نحوه تامین آن (ریالی و ارزی)

عنوان هزینه	ریالی	ارزی
هزینه پرسنلی	۱,۰۰۰,۰۰۰	-
وسایل و مواد	-	-
مسافرت (داخل و خارج)	-	-
سایر هزینه ها	۱,۰۰۰,۰۰۰	-
جمع کل (هزینه ها تا سقف ۲/۰۰۰/۰۰۰ ریال قابل پرداخت می باشد)	**۲,۰۰۰,۰۰۰**	-

۱۳- نظریه استاد راهنما:

امضاء

۱۴- نظریه مسئول تحصیلات تکمیلی دانشکده:

امضاء

۱۵- رئیس دانشکده :

امضاء

۱۶-تعهدنامه دانشجو:

اینجانب دانشجوی پروژه متعهد می‌شوم که در مدت اجرای پروژه به طور تمام‌وقت انجام وظیفه نموده و بدون اطلاع معاونت پژوهشی دانشگاه از مرخصی تحصیلی استفاده ننمایم و همچنین اطلاع دارم که کلیه نتایج و حقوق حاصله از این پروژه، متعلق به دانشگاه بوده و مجاز نیستم بدون موافقت دانشگاه اطلاعاتی را در رابطه با پروژه به دیگری واگذار نمایم.

نام و امضاء دانشجو

۱۷-نظریه شورای تحصیلات تکمیلی دانشگاه :

امضاء

تاریخ

۱۸-سایر توضیحات :

سابقه علمی

مکانیزم‌های پویا مانند نظارت اجرا [۵، ۶] و چنداجرایی امن به اطلاعات زمان‌اجرا دسترسی دارند، و به همین دلیل، نسبت به مکانیزم‌های ایستا آسان‌گیر[۲۰]تر هستند. مفهوم اصلی چنداجرایی امن توسط پژوهشگران متفاوتی، به طور مستقل، مطرح شده است. در [۸]، اجراهای در سایه[۲۱] معرفی شده است که در آن، برای تضمین محرمانگی قوی، دوبار اجرای پردازه[۲۲]ها برای سطح امنیتی بالا و پایین را پیشنهاد کرده‌اند. مقاله [۷]، برای اولین بار به اثبات درستی و دقت تکنیک چنداجرایی امن پرداخته است. گرچه در این مقاله بیان شده است که تکنیک چنداجرایی ارائه‌شده، دارای درستی کامل، حتی برای کانال‌های نهان خاتمه و زمانی، و دقت خوبی است، اما از نظر موازنه زمان-حافظه[۲۳] بهینه نیست. در کار [۱۲]، تعمیم این تکنیک به خانواده‌ای از روش‌ها مطرح شده است که با نام رویکرد زمان‌بندی برای عدم تداخل[۲۴] معرفی می‌شوند. در این مقاله، به تحلیل استراتژی زمان‌بندی و تأثیر آن در اعمال امنیت می‌پردازد. ضمناً با ارائه یک استراتژی زمان‌بندی جدید، تضمین امنیتی قوی‌تری از کارهای قبل‌تر فراهم می‌کند. جسکلیوف و روسو [۱۳] یک کتابخانه برای چنداجرایی امن در زبان هسکل[۲۵] ارائه کرده‌اند. مقاله‌های [۱۴] و [۱۵]، پیاده‌سازی‌های این تکنیک را برای کاربردهای خاص، مانند مرورگر، بررسی کرده‌اند. همچنین، در مقاله [۲]، مکانیزمی بر اساس ترکیب مزایای تکنیک‌های تغییر برنامه و چنداجرایی امن ارائه شده است که با استفاده از این مکانیزم، می‌توان بدون تغییر در زیرساخت محاسباتی، اثر تکنیک چنداجرایی امن را به دست آورد. در [۱۶]، به بررسی و مقایسه پنج مکانیزم اعمال پویا، از جمله چنداجرایی امن و ناظرهای اجرایی مختلف، برای انواع عدم تداخل، مانند حساس یا غیرحساس به خاتمه، پرداخته است که از برتری نسبی چنداجرایی امن نسبت به سایرین خبر می‌دهد. در [۱۱]، علاوه بر ارائه زمان‌بندی برای حفظ ترتیب خروجی‌های برنامه، روش جدیدی حاصل از ترکیب نظارت و چنداجرایی امن، به نام ناظر چنداجرایی[۲۶] ارائه کرده است.

از دیگر رویکردهای اعمال خط‌مشی‌ها، روش خودترکیبی [۴] است. منظور از خودترکیبی آن است که بتوان مسئله امنیت جریان اطلاعات در یک برنامه را به یک خاصیت ایمنی[۲۷] برای برنامه دیگری که از برنامه اولیه حاصل می‌شود، به کمک ترکیب برنامه اصلی با خودش، با نام‌گذاری جدید، کاهش داد. با استفاده از روش خودترکیبی، می‌توان مسئله اعمال فوق خاصیت‌ها را به اعمال خاصیت‌ها تبدیل کرد. ممکن است بتوان از تکنیک چنداجرایی برای ایجاد خودترکیبی استفاده کرد، که این خود یکی از سوالات پژوهش محسوب می‌شود.

ضمناً اثبات شده است که هیچ مکانیزم کاملاً پویایی برای اعمال خط‌مشی عدم‌تداخل حساس‌به‌جریان وجود ندارد [۱۷]. حساس به جریان به این معنا که ترتیب گزاره‌های برنامه حائز اهمیت است و برای هر گزاره باید یک تحلیل جداگانه محاسبه شود. این موضوع باعث می‌شود که پروژه‌هایی که محدودیت‌های نَحوی[۲۸] بر روی کد دارند، از اطلاعات ایستا در ناظری بر اجراهای برنامه‌ها استفاده کنند.

---

[20] Permissive
[21] Shadow Executions
[22] Process
[23] Time-Memory Trade-off
[24] Scheduling Approach to Non-interference
[25] Haskell
[26] Multi-Execution Monitor
[27] Safety Property
[28] Syntactic