动静结合的污点分析方法,既可以弥补动态分析由于路径覆盖不足引起的信息丢失的问题,又能够提供某些静态分析无法获得的精确运行信息,提高污点分析的精度.例如:针对 Android 隐私泄露检测工具 Appaudit[90],首先利用轻量级的静态分析缩小了可能的分析范围(指出一些可疑的函数);之后,结合近似执行技术动态验证这些可疑的函数是否具有威胁.

3) 最后,除了解决应用程序中的安全问题这一传统职责外,污点分析技术也逐渐被应用在程序分析领域其他问题(如性能分析、软件缺陷定位、错误检测等)的研究和实践中.RETracer[98]提出了一种针对二进制代码的动态后向污点分析技术来定位引起软件系统崩溃的浅层原因.ConfAid[99]基于通用污点分析技术提出了一种专门用于解决由于配置文件中的错误而导致性能下降的错误定位技术.X-ray[100]在 ConfAid 的基础上,利用污点分析技术逆向追溯导致性能瓶颈的根本原因.如何将污点分析技术与程序分析领域的其他技术相结合、扩展污点分析技术的应用范畴,也是一个值得探索的研究方向.

References:

[1] Sabelfeld A, Myers AC. Language-Based information-flow security. IEEE Journal on Selected Areas in Communications, 2003, 21(1):5−19. [doi: 10.1109/JSAC.2002.806121]

[2] Foundation TO. Top ten most critical Web application vulnerabilities. 2013. https://www.owasp.org/index.php/Top_10_2013-Top_10

[3] McAfee. Mobile Threat Report. 2016. http://www.mcafee.com/us/resources/reports/rp-mobile-threat-report-2016.pdf

[4] Citi credit card data breached for 200000 customers. 2011. https://www.wired.com/2011/06/citi-credit-card-breach/

[5] Dennis JB, Van Horn EC. Programming semantics for multiprogrammed computations. Communications of the ACM, 1966,9(3): 143−155. [doi: 10.1145/365230.365252]

[6] Sandhu RS, Samarati P. Access control: Principles and practice. Communications Magazine, 1994,32(9):40−48. [doi: 10.1109/35.312842]

[7] Oppliger R. Internet security: Firewalls and beyond. Communications of the ACM, 1997,40(5):92−102. [doi: 10.1145/253769.253802]

[8] Bellare M, Boldyreva A, Micali S. Public-Key encryption in a multi-user setting: Security proofs and improvements. In: Proc. of the Int'l Conf. on the Theory and Applications of Cryptographic Techniques. Berlin, Heidelberg: Springer-Verlag, 2000. 259−274. [doi: 10.1007/3-540-45539-6_18]

[9] Myers AC, Liskov B. A decentralized model for information flow control. ACM SIGOPS Operating Systems Review, 1997,31(5): 129−142. [doi: 10.1145/268998.266669]

[10] Livshits VB, Lam MS. Finding security vulnerabilities in Java applications with static analysis. In: Proc. of the Conf. on Usenix Security Symp. USENIX Association, 2005. 262−266. https://www.usenix.org/legacy/event/sec05/tech/full_papers/livshits/livshits_html/

[11] Rasthofer S, Arzt S, Bodden E. A machine-learning approach for classifying and categorizing android sources and sinks. In: Proc. of the Network and Distributed System Security Symp. (NDSS). 2014. [doi: 10.14722/ndss.2014.23039]

[12] Gordon MI, Kim D, Perkins JH, Gilham L, Nguyen N, Rinard MC. Information flow analysis of Android applications in DroidSafe. In: Proc. of the NDSS 2015. 2015. [doi: 10.14722/ndss.2015.23089]

[13] Arzt S, Rasthofer S, Fritz C, Bodden E, Bartel A, Klein J, Le Traon Y, Octeau D, McDaniel P. Flowdroid: Precise context, flow, field, object-sensitive and lifecycle-aware taint analysis for Android apps. ACM SIGPLAN Notices, 2014,49(6):259−269. [doi: 10.1145/2594291.2594299]

[14] Yang Z, Yang M. Leakminer: Detect information leakage on Android with static taint analysis. In: Proc. of the Software Engineering. IEEE, 2012. 101−104. [doi: 10.1109/WCSE.2012.26]

[15] Lu L, Li Z, Wu Z, Lee W, Jiang G. Chex: Statically vetting Android apps for component hijacking vulnerabilities. In: Proc. of the 2012 ACM Conf. on Computer and Communications Security. ACM Press, 2012. 229−240. [doi: 10.1145/2382196.2382223]

[16] Zhao Z, Osono FC. "TrustDroid™": Preventing the use of SmartPhones for information leaking in corporate networks through the used of static analysis taint tracking. In: Proc. of the 7th Int'l Conf. on Malicious and Unwanted Software (MALWARE). IEEE, 2012. 135−143. [doi: 10.1109/MALWARE.2012.6461017]

[17] Klieber W, Flynn L, Bhosale A, Jia L, Bauer L. Android taint flow analysis for app sets. In: Proc. of the ACM Sigplan Int'l Workshop on the State of the Art in Java Program Analysis. ACM Press, 2014. 1−6. [doi: 10.1145/2614628.2614633]

[18] Octeau D, McDaniel P, Jha S, Bartel A, Bodden E, Klein J, Le Traon Y. Effective inter-component communication mapping in android with EPICC: An essential step towards holistic security analysis. In: Proc. of the Usenix Conf. on Security. 2013. 543−558. https://www.usenix.org/conference/usenixsecurity13/technical-sessions/presentation/octeau

[19] Enck W, Gilbert P, Han S, Tendulkar V, Chun BG, Cox LP, Jung J, McDaniel P, Sheth AN. TaintDroid: An information-flow tracking system for realtime privacy monitoring on smartphones. ACM Trans. on Computer Systems, 2014,32(2):393−407. [doi: 10.1145/2619091]

[20] Rastogi V, Chen Y, Enck W. AppsPlayground: Automatic security analysis of smartphone applications. In: Proc. of the ACM Conf. on Data and Application Security and Privacy. 2013. 209−220. [doi: 10.1145/2435349.2435379]

[21] Zhang Y, Yang M, Xu B, Yang Z, Gu G, Ning P, Wang XS, Zang B. Vetting undesirable behaviors in Android apps with permission use analysis. In: Proc. of the 2013 ACM SIGSAC Conf. on Computer & Communications Security. 2013. 611−622. [doi: 10.1145/2508859.2516689]

[22] Tripp O, Rubin J. A Bayesian approach to privacy enforcement in smartphones. In: Proc. of the Usenix Conf. on Security Symp. USENIX Association, 2014. 175−190. https://www.usenix.org/node/184428

[23] Hornyack P, Han S, Jung J, Schechter S, Wetherall D. These aren't the droids you're looking for: Retrofitting Android to protect data from imperious applications. In: Proc. of the ACM Conf. on Computer and Communications Security (CCS 2011). Chicago, 2011. 639−652. [doi: 10.1145/2046707.2046780]

[24] Sridharan M, Artzi S, Pistoia M, Guarnieri S, Tripp O, Berg R. F4F: Taint analysis of framework-based Web applications. ACM SIGPLAN Notices, 2011,46(10):1053−1068. [doi: 10.1145/2048066.2048145]

[25] Tripp O, Pistoia M, Fink SJ, Sridharan M, Weisman O. TAJ: Effective taint analysis of Web applications. ACM SIGPLAN Notices, 2009,44(6):87−97. [doi: 10.1145/1542476.1542486]

[26] Papagiannis I, Migliavacca M, Pietzuch P. PHP ASPIS: Using partial taint tracking to protect against injection attacks. In: Proc. of the Usenix Conf. on Web Application Development. USENIX Association, 2011. 2. https://www.usenix.org/conference/webapps11/php-aspis-using-partial-taint-tracking-protect-against-injection-attacks

[27] Balzarotti D, Cova M, Felmetsger V, Jovanovic N, Kirda E, Kruegel C, Vigna G. Saner: Composing static and dynamic analysis to validate sanitization in Web applications. In: Proc. of the 2012 IEEE Symp. on Security and Privacy. IEEE, 2008. 387−401. [doi: 10.1109/SP.2008.22]

[28] Halfond WG, Orso A, Manolios P. Using positive tainting and syntax-aware evaluation to counter SQL injection attacks. In: Proc. of the ACM Sigsoft Int'l Symp. on Foundations of Software Engineering (FSE 2006). Oregon, 2006. 175−185. [doi: 10.1145/1181775.1181797]

[29] Vogt P, Nentwich F, Jovanovic N, Kirda E, Kruegel C, Vigna G. Cross site scripting prevention with dynamic data tainting and static analysis. In: Proc. of the NDSS 2007. 2007. 12. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.72.4505

[30] Møller A, Schwarz M. Automated detection of client-state manipulation vulnerabilities. ACM Trans. on Software Engineering and Methodology (TOSEM), 2014,23(4):29. [doi: 10.1145/2531921]

[31] Chugh R, Meister JA, Jhala R, Lerner S. Staged information flow for JavaScript. ACM SIGPLAN Notices, 2009,44(6):50−62. [doi: 10.1145/1542476.1542483]

[32] Wei S, Ryder BG. Practical blended taint analysis for JavaScript. In: Proc. of the 2013 Int'l Symp. on Software Testing and Analysis. ACM Press, 2013. 336−346. [doi: 10.1145/2483760.2483788]

[33] Guarnieri S, Pistoia M, Tripp O, Dolby J, Teilhet S, Berg R. Saving the world wide Web from vulnerable JavaScript. In: Proc. of the 2011 Int'l Symp. on Software Testing and Analysis. ACM Press, 2011. 177−187. [doi: 10.1145/2001420.2001442]

[34] Lekies S, Stock B, Johns M. 25 million flows later: Large-Scale detection of DOM-based XSS. In: Proc. of the 2013 ACM SIGSAC Conf. on Computer & Communications Security. ACM Press, 2013. 1193−1204. [doi: 10.1145/2508859.2516703]

[35] Stock B, Lekies S, Mueller T, Spiegel P, Johns M. Precise client-side protection against dom-based cross-site scripting. In: Proc. of the 23rd USENIX Security Symp. (USENIX Security 2014). 2014. 655−670. https://www.usenix.org/node/184492

[36] Denning DE. A lattice model of secure information flow. Communications of the ACM, 1976,19(5):236−243. [doi: 10.1145/360051.360056]

[37] Denning DE, Denning PJ. Certification of programs for secure information flow. Communications of the ACM, 1977,20(7):504−13. [doi: 10.1145/359636.359712]

[38] Goguen JA, Meseguer J. Security policies and security models. In: Proc. of the '82 IEEE Symp. on Security and Privacy. IEEE, 1982. 11−11. [doi: 10.1109/SP.1982.10014]

[39] Myers AC. JFlow: Practical mostly-static information flow control. In: Proc. of the 26th ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages. ACM Press, 1999. 228−241. [doi: 10.1145/292540.292561]

[40] Heintze N, Riecke JG. The SLam calculus: Programming with secrecy and integrity. In: Proc. of the 25th ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages. ACM Press, 1998. 365−377. [doi: 10.1145/268946.268976]

[41] Myers AC, Liskov B. Protecting privacy using the decentralized label model. ACM Trans. on Software Engineering and Methodology (TOSEM), 2000,9(4):410−442. [doi: 10.1145/363516.363526]

[42] Sabelfeld A, Sands D. Probabilistic noninterference for multi-threaded programs. In: Proc. of the 13th IEEE Computer Security Foundations Workshop (CSFW-13). IEEE, 2000. 200−214. [doi: 10.1109/CSFW.2000.856937]

[43] Pottier F, Simonet V. Information flow inference for ML. ACM Trans. on Programming Languages and Systems (TOPLAS), 2003, 25(1):117−158. [doi: 10.1145/596980.596983]

[44] Abadi M, Banerjee A, Heintze N, Riecke JG. A core calculus of dependency. In: Proc. of the 26th ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages. ACM Press, 1999. 147−160. [doi: 10.1145/292540.292555]

[45] Ashcraft K, Engler D. Using programmer-written compiler extensions to catch security holes. In: Proc. of the 2002 IEEE Symp. on Security and Privacy. IEEE, 2002. 143−159. [doi: 10.1109/SECPRI.2002.1004368]

[46] Volpano D, Irvine C, Smith G. A sound type system for secure flow analysis. Journal of Computer Security, 1996,4(2-3):167−87. [doi: 10.3233/JCS-1996-42-304]

[47] Foster JS, Fähndrich M, Aiken A. A theory of type qualifiers. ACM SIGPLAN Notices, 1999,34(5):192−203. [doi: 10.1145/301631.301665]

[48] Shankar U, Talwar K, Foster JS, Wagner D. Detecting format string vulnerabilities with type qualifiers. In: Proc. of the USENIX Security Symp. 2001. 201−220. https://www.usenix.org/conference/10th-usenix-security-symposium/detecting-format-string-vulnerabilities-type-qualifiers

[49] King D, Hicks B, Hicks M, Jaeger T. Implicit flows: Can't live with 'em, can't live without 'em. In: Proc. of the Int'l Conf. on Information Systems Security. Berlin, Heidelberg: Springer-Verlah, 2008. 56−70. [doi: 10.1007/978-3-540-89862-7_4]

[50] Saxena P, Molnar D, Livshits B. SCRIPTGARD: Automatic context-sensitive sanitization for large-scale legacy Web applications. In: Proc. of the 18th ACM Conf. on Computer and Communications Security. ACM Press, 2011. 601−614. [doi: 10.1145/2046707.2046776]

[51] Samuel M, Saxena P, Song D. Context-Sensitive auto-sanitization in Web templating languages using type qualifiers. In: Proc. of the 18th ACM Conf. on Computer and Communications Security. ACM Press, 2011. 587−600. [doi: 10.1145/2046707.2046775]

[52] Bates D, Barth A, Jackson C. Regular expressions considered harmful in client-side XSS filters. In: Proc. of the 19th Int'l Conf. on World Wide Web. ACM Press, 2010. 91−100. [doi: 10.1145/1772690.1772701]

[53] Hooimeijer P, Livshits B, Molnar D, Saxena P, Veanes M. Fast and precise sanitizer analysis with BEK. In: Proc. of the 20th USENIX Conf. on Security. USENIX Association, 2011. 1−1. http://dl.acm.org/citation.cfm?id=2028068

[54] Aho AV, Sethi R, Ullman JD. Compilers, Principles, Techniques. Boston: Addison Wesley, 1986.

[55] Nielson F, Nielson HR, Hankin C. Principles of Program Analysis. New York: Springer-Verlag, 2015.

[56] Andersen LO. Program analysis and specialization for the C programming language. Addison-Wesley Series in Computer Science, 1994,2(1):37−77.

[57] Reps T, Horwitz S, Sagiv M. Precise interprocedural dataflow analysis via graph reachability. In: Proc. of the 22nd ACM SIGPLAN-SIGACT Symp. on Principles of Programming Languages. ACM Press, 1995. 49−61. [doi: 10.1145/199448.199462]

[58] Sagiv M, Reps T, Horwitz S. Precise interprocedural dataflow analysis with applications to constant propagation. Theoretical Computer Science, 1996,167(1):131−70. [doi: 10.1016/0304-3975(96)00072-2]

[59] Li Y, Tan T, Zhang Y, Xue J. Program tailoring: Slicing by sequential criteria. In: Proc. of the ECOOP. 2016. http://drops.dagstuhl.de/opus/volltexte/2016/6109/

[60] Ming CC, Huo W, Yu HT. A survey of optimization technology of inclusion-based pointer analysis. Chinese Journal of Computers, 2011,34(7):1224−1238 (in Chinese with English abstract). [doi: 10.3724/SP.J.1016.2011.01224]

[61] Tripp O, Pistoia M, Cousot P, Cousot R, Guarnieri S. Andromeda: Accurate and scalable security analysis of Web applications. In: Proc. of the Int'l Conf. on Fundamental Approaches to Software Engineering. Berlin, Heidelberg: Springer-Verlag, 2013. 210−225. [doi: 10.1007/978-3-642-37057-1_15]

[62] Crandall JR, Chong FT. Minos: Control data attack prevention orthogonal to memory model. In: Proc. of the 37th Int'l Symp. on Microarchitecture (MICRO-37). IEEE, 2004. 221−232. [doi: 10.1109/MICRO.2004.26]

[63] Dalton M, Kannan H, Kozyrakis C. Raksha: A flexible information flow architecture for software security. ACM SIGARCH Computer Architecture News, 2007,35(2):482−493. [doi: 10.1145/1273440.1250722]

[64] Zhu DY, Jung J, Song D, Kohno T, Wetherall D. Tainteraser: Protecting sensitive data leaks using applicationlevel taint tracking. ACM SIGOPS Operating Systems Review, 2011,45(1):142−154. [doi: 10.1145/1945023.1945039]

[65] Venkataramani G, Doudalis I, Solihin Y, Prvulovic M. Flexitaint: A programmable accelerator for dynamic taint propagation. In: Proc. of the 2008 IEEE 14th Int'l Symp. on High Performance Computer Architecture. IEEE, 2008. 173−184. [doi: 10.1109/HPCA.2008.4658637]

[66] Yoon MK, Salajegheh N, Chen Y, Christodorescu M. PIFT: Predictive information-flow tracking. In: Proc. of the 21st Int'l Conf. on Architectural Support for Programming Languages and Operating Systems. ACM Press, 2016. 713−725. [doi: 10.1145/2872362.2872403]

[67] Newsome J, Song D. Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software. In: Proc. of the Network and Distributed System Security Symp. 2005. 720−724.

[68] Nethercote N, Seward J. Valgrind: A program supervision framework. Electronic Notes in Theoretical Computer Science, 2003, 89(2):44−66. [doi: 10.1016/S1571-0661(04)81042-9]

[69] Zhu Y, Jung J, Song D, Kohno T, Wetherall D. Privacy scope: A precise information flow tracking system for finding application leaks. Technical Report, EECS-2009-145, Berkeley: University of California, 2009.

[70] Clause J, Li W, Orso A. DYTAN: A generic dynamic taint analysis framework. In: Proc. of the 2007 Int'l Symp. on Software Testing and Analysis. ACM Press, 2007. 196−206. [doi: 10.1145/1273463.1273490]

[71] Kemerlis VP, Portokalidis G, Jee K, Keromytis AD. libdft: Practical dynamic data flow tracking for commodity systems. ACM SIGPLAN Notices, 2012,47(7):121−132. [doi: 10.1145/2365864.2151042]

[72] Luk CK, Cohn R, Muth R, Patil H, Klauser A, Lowney G, Wallace S, Reddi VJ, Hazelwood K. Pin: Building customized program analysis tools with dynamic instrumentation. ACM SIGPLAN Notices, 2005,40(6):190−200. [doi: 10.1145/1064978.1065034]

[73] Dalvik executable format. 2016. https://source.android.com/devices/tech/dalvik/dex-format.html

[74] Qin F, Wang C, Li Z, Kim HS, Zhou Y, Wu Y. Lift: A low-overhead practical information flow tracking system for detecting security attacks. In: Proc. of the 39th Annual IEEE/ACM Int'l Symp. on Microarchitecture (MICRO 2006). IEEE, 2006. 135−148. [doi: 10.1109/MICRO.2006.29]

[75] Chen H, Wu X, Yuan L, Zang B, Yew PC, Chong FT. From speculation to security: Practical and efficient information flow tracking using speculative hardware. In: Proc. of the Int'l Symp. on Computer Architecture. IEEE Computer Society, 2008. 401−412. [doi: 10.1109/ISCA.2008.18]

[76] Bao T, Zheng Y, Lin Z, Zhang X, Xu D. Strict control dependence and its effect on dynamic information flow analyses. In: Proc. of the 19th Int'l Symp. on Software Testing and Analysis. ACM Press, 2010. 13−24. [doi: 10.1145/1831708.1831711]

[77] Kang MG, McCamant S, Poosankam P, Song D. DTA++: Dynamic taint analysis with targeted control-flow propagation. In: Proc. of the Network and Distributed System Security Symp. (NDSS 2011). San Diego, 2011.

[78] Cox LP, Gilbert P, Lawler G, Pistol V, Razeen A, Wu B, Cheemalapati S. Spandex: Secure password tracking for Android. In: Proc. of the 23rd USENIX Security Symp. (USENIX Security 2014). 2014. 481−494. https://www.usenix.org/node/184402

[79] Qing SH. Research progress on Android security. Ruan Jian Xue Bao/Journal of Software, 2016,27(1):45−71 (in Chinese with English abstract). http://www.jos.org.cn/1000-9825/4914.htm [doi: 10.13328/j.cnki.jos.004914]

[80] Zhang YQ, Wang K, Yang H, Fang ZJ, Wang ZQ, Cao C. Survey of Android OS security. Journal of Computer Research and Development, 2014,51(7):1385−1396 (in Chinese with English abstract). [doi: 10.7544/issnl1000-1239.2014.20140098]

[81] Felt AP, Chin E, Hanna S, Song D, Wagner D. Android permissions demystified. In: Proc. of the 18th ACM Conf. on Computer and Communications Security. ACM Press, 2011. 627−638. [doi: 10.1145/2046707.2046779]

[82] Corporation ID. Smartphone OS market share. 2015. http://www.idc.com/prodserv/smartphone-os-market-share.jsp

[83] The Android software stack. 2016. https://developer.android.com/guide/platform/index.html

[84] Christensen AS, Møller A, Schwartzbach MI. Precise analysis of string expressions. In: Proc. of the Int'l Static Analysis Symp. Berlin, Heidelberg: Springer-Verlag, 2003. 1−18. [doi: 10.1007/3-540-44898-5_1]

[85] Arzt S, Bodden E. StubDroid: Automatic inference of precise data-flow summaries for the Android framework. In: Proc. of the 38th Int'l Conf. on Software Engineering. ACM Press, 2016. 725−735. [doi: 10.1145/2884781.2884816]

[86] Wikipedia. Comparison of Web application frameworks. 2016. https://en.wikipedia.org/wiki/Comparison_of_web_frameworks

[87] Web framework. 2016. https://en.wikipedia.org/wiki/Web_framework

[88] ECMA Int'l. ECMAScript language specification. Version 5.1. 2011. http://www.ecma-international.org/ecma-262/5.1/Ecma-262.pdf

[89] Klein A. Dom based cross site scripting or XSS of the third kind. Web Application Security Consortium, 2005,4:365−372.

[90] Xia M, Gong L, Lyu Y, Qi Z, Liu X. Effective real-time Android application auditing. In: Proc. of the 2015 IEEE Symp. on Security and Privacy. IEEE, 2015. 899−914. [doi: 10.1109/SP.2015.60]

[91] Li L, Cifuentes C, Keynes N. Boosting the performance of flow-sensitive points-to analysis using value flow. In: Proc. of the 19th ACM SIGSOFT Symp. and the 13th European Conf. on Foundations of Software Engineering. ACM Press, 2011. 343−353. [doi: 10.1145/2025113.2025160]

[92] Li L, Cifuentes C, Keynes N. Precise and scalable context-sensitive pointer analysis via value flow graph. ACM SIGPLAN Notices, 2013,48(11):85−96. [doi: 10.1145/2555670.2466483]

[93] Yu H, Xue J, Huo W, Feng X, Zhang Z. Level by level: Making flow-and context-sensitive pointer analysis scalable for millions of lines of code. In: Proc. of the 8th Annual IEEE/ACM Int'l Symp. on Code Generation and Optimization. ACM Press, 2010. 218−229. [doi: 10.1145/1772954.1772985]

[94] Sui Y, Xue J. On-Demand strong update analysis via value-flow refinement. In: Proc. of the 24th ACM SIGSOFT Int'l Symp. on Foundations of Software Engineering. ACM Press, 2016. 460−473. [doi: 10.1145/2950290.2950296]

[95] Tan T, Li Y, Xue J. Making *k*-object-sensitive pointer analysis more precise with still *k*-limiting. In: Proc. of the Int'l Static Analysis Symp. Berlin, Heidelberg: Springer-Verlag, 2016. 489−510. [doi: 10.1007/978-3-662-53413-7_24]

[96] Sui Y, Ye S, Xue J, Zhang J. Making context-sensitive inclusion-based pointer analysis practical for compilers using parameterised summarisation. Software: Practice and Experience, 2014,44(12):1485−1510. [doi: 10.1002/spe.2214]

[97] Sarwar G, Mehani O, Boreli R, Kaafar MA. On the effectiveness of dynamic taint analysis for protecting against private information leaks on Android-based devices. In: Proc. of the 2013 Int'l Conf. on Security and Cryptography (SECRYPT). IEEE, 2013. 1−8. http://ieeexplore.ieee.org/document/7223198/

[98] Cui W, Peinado M, Cha SK, Fratantonio Y, Kemerlis VP. RETracer: Triaging crashes by reverse execution from partial memory dumps. In: Proc. of the 38th Int'l Conf. on Software Engineering. ACM Press, 2016. 820−831. [doi: 10.1145/2884781.2884844]

[99] Attariyan M, Flinn J. Automating configuration troubleshooting with dynamic information flow analysis. In: Proc. of the Usenix Conf. on Operating Systems Design and Implementation. USENIX Association, 2010. 1−11. https://www.usenix.org/conference/osdi10/automating-configuration-troubleshooting-dynamic-information-flow-analysis

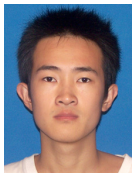
[100] Attariyan M, Chow M, Flinn J. *X*-Ray: Automating root-cause diagnosis of performance anomalies in production software. In: Proc. of the Presented as Part of the 10th USENIX Symp. on Operating Systems Design and Implementation (OSDI 2012). 2012. 307−320. https://www.usenix.org/node/170861

附中文参考文献:

[60] 陈聪明,霍玮,于洪涛,冯晓兵.基于包含的指针分析优化技术综述.计算机学报,2011,34(7):1224−1238. [doi: 10.3724/SP.J.1016.2011.01224]

[79] 卿斯汉.Android 安全研究进展.软件学报,2016,27(1):45−71. http://www.jos.org.cn/1000-9825/4914.htm [doi: 10.13328/j.cnki.jos.004914]

[80] 张玉清,王凯,杨欢,方喆君,王志强,曹琛.Android 安全综述.计算机研究与发展,2014,51(7):1385−1396. [doi: 10.7544/issnl1000-1239.2014.20140098]



王蕾(1989－),男,吉林白山人,博士生,主要研究领域为程序分析,软件安全.

李炼(1977－),男,博士,研究员,博士生导师,CCF 专业会员,主要研究领域为程序分析,编译,自动测试,软件安全.

李丰(1985－),女,博士,助理研究员,CCF 专业会员,主要研究领域为程序分析,故障定位.

冯晓兵(1969－),男,博士,研究员,博士生导师,CCF 杰出会员,主要研究领域为编程模型,编译优化.