Modellierung und formale Analyse von Betriebssystem-Sicherheitspolitiken

Abstract

Peter Amthor*

Heutige Anwendungsdomänen für Betriebssysteme bedingen zunehmend strengere Anforderungen an die Informationssicherheit. Beispiele hierfür finden sich etwa in der Servervirtualisierung (*Infrastructure as a Service*), in mobilen Systemen wie Smartphones und Smartwatches oder in der Automatisierung und Vernetzung im Straßenverkehr. Aus diesem Grund kommen in zunehmendem Maße informationstechnische Sicherheitspolitiken zum Einsatz, welche die Strategien zur Umsetzung solcher Sicherheitsanforderungen beschreiben und implementieren [24].

Die Maschinenrepräsentation einer Sicherheitspolitik nimmt somit eine zentrale Stellung als sicherheitskritischer Teil des Betriebssystems ein: Ihr Schutz gegenüber Fehlern und Angriffen, zugleich aber auch ihre Korrektheit sind entscheidend für die Wahrung von Sicherheitseigenschaften hinsichtlich etwa der Integrität des Systems oder der Vertraulichkeit von Anwendungsinformationen.

Diese Forderung nach Korrektheit hat sich in der Praxis als grundsätzlich problematisch erwiesen: Durch gravierende semantische Brüche, welche auf dem Weg von grundlegenden, in natürlicher Sprache spezifizierten Sicherheitsanforderungen hin zu einer Implementierung der Sicherheitspolitik zu überwinden sind, haben sich traditionelle Software-Engineering-Prozesse als zu fehleranfällig erwiesen; gleichzeitig schließt die Komplexität des resultierenden Systems dessen formale Verifikation auf Quellcode-Niveau weitgehend aus.

Als Konsequenz hieraus wird die Korrektheit von Betriebssystem-Sicherheitspolitiken auf einer abstrakteren Basis analysiert: mittels formaler Modelle und formaler Sicherheitseigenschaften (*modellbasiertes Security Engineering*). Hierbei sind noch immer praktische Hürden zu überwinden, auf die im Folgenden näher eingegangen werden soll.

Zugriffssteuerung in Betriebssystemen

Die bedeutendste Klasse von Sicherheitspolitiken, welche in Betriebssystemen zum Einsatz kommt, ist die der Zugriffssteuerungspolitiken [16, 23, 20, 7, 9, 11]. Ihre Maschinenrepräsentation bildet eine kompakte, zentralisierte logische Komponente der Betriebssystemfunktionalität, welche typischerweise im Kernel implementiert und somit von Nutzerprozessen isoliert ist. Diese umfasst als wesentliche Teile einen *Policy Decision Point* (PDP), welcher die Sicherheitspolitik auswertet und somit Zugriffsentscheidungen trifft, sowie mehrere *Policy Enforcement Points* (PEP), welche diese Entscheidungen durchsetzen.

Wir sprechen bei Systemen, welche dieses Architekturprinzip implementieren, im Folgenden von *politikgesteuerten Betriebssystemen*. Diese zeichnen sich insbesondere dadurch aus, dass ihre Implementierung einer Sicherheitspolitik die Einhaltung der Referenzmonitoreigenschaften Unumgehbarkeit (durch wohldefinierte PEPs), Manipulationssicherheit (durch Schutz des PDP) und praktische Validierbarkeit (durch funktional kompakten und zentralisierten PDP) verfolgt. Moderne Implementierungen dieses Konzepts finden sich etwa in den *Linux Security Modules* (LSM) [25], dem darauf aufbauenden SELinux [16] oder dem *Xen* Hypervisor [8].

Gerade unter der Annahme eines fehlerfreien, also nichtkorrumpierten politikgesteuerten Betriebssystems wird offensichtlich, welche entscheidende Bedeutung der Korrektheit einer Sicherheitspolitik (sprich der Entscheidungen des PDP) zukommt. Dies wiederum motiviert ihre formale Analyse auf Basis eines Zugriffssteuerungsmodells.

Politiksemantik

Zugriffssteuerungsmodelle formalisieren die Semantik einer Betriebssystem-Zugriffssteuerungspolitik. Diese kann sich, je nach Betriebssystem, unterscheiden hinsichtlich der Arten von Metainformationen, mit denen Prozesse und Systemobjekte attributiert werden, sowie der Logik von Politikregeln, welche diese auswerten. Diese Diversität hat in der Vergangenheit zu einer Vielzahl spezialisierter Zugriffssteuerungsmodelle geführt [26, 4, 19, 6].

Typische Klassen von Politiksemantiken sind beispielsweise hierarchische Vertraulichkeits- und Integritätsklassen

* Technische Universität Ilmenau, peter.amthor@tu-ilmenau.de

(*Multi-Level Security*, MLS), rollenbasierte (*Role-based Access Control*, RBAC) oder typenbasierte Zugriffsregeln (*Type Enforcement*, TE). Diese können, je nach Anwendungsdomäne, in Reinform oder untereinander kombiniert vorliegen; die SELinux-Zugriffssteuerungspolitik beinhaltet beispielsweise Regeln aus allen drei Klassen. Zugleich erfordern aussagekräftige Analysen eines Modells die präzise Abbildung der zugrunde liegenden Politiksemantik; entsprechend hoch ist der Komplexitätsgrad bei der Erstellung neuer Modelle für eine Zugriffssteuerungspolitik, die in einem (neu entwickelten oder existierenden) politikkontrollierten Betriebssystem durchgesetzt werden soll.

In der Praxis erfordert durchgängiges modellbasiertes Security Engineering für ein politikgesteuertes Betriebssystem daher die Definition eines spezialisierten Zugriffssteuerungsmodells.

Sicherheitseigenschaften

Neben der präzisen Formalisierung von Zugriffssteuerungssemantiken ist es erforderlich, auch den Korrektheitsbegriff, gegen den das entsprechende Modell verifiziert werden soll, zu formalisieren. Dieser ergibt sich aus den Sicherheitsanforderungen der zu analysierenden Politik.

Eine typische Sicherheitseigenschaft in der Domäne der Betriebssystem-Zugriffssteuerungspolitiken ist die potenzielle Rechteausbreitung [15, 18, 4, 5, 1]: In der Praxis ist diese für solche Bedrohungsszenarien von besonderem Interesse, bei denen eine politikkonforme (aber nicht erwünschte) Änderung von Zugriffsentscheidungen als Grundlage eines Angriffs herbeigeführt wird (*Privilege Escalation*). Beispiele hierfür sind Fragen wie „Kann ein Nutzerprozess jemals Schreibrechte auf eine Systemdatei erhalten?“ oder „Kann es jemals geschehen, dass ein Nutzerprozess Code mit Privilegien *x* ausführt?“. Solche Fragen müssen für eine konkrete Zugriffssteuerungspolitik formalisiert werden, beispielsweise in Form aussagenlogischer Invarianten, die sich maßgeschneidert auf das für die jeweilige Politik definierte Zugriffssteuerungsmodell beziehen.

Vor diesem Hintergrund ist es zudem erforderlich, dass ein Zugriffssteuerungsmodell das Laufzeitverhalten des Betriebssystems modelliert. Hierfür kommen typischerweise Beschreibungsmittel aus dem Bereich der Automatentheorie zum Einsatz. Eine wesentliche Grundlage hierfür liefert das HRU-Modell [12], dessen Automatenkalkül später zum Modellkern dynamischer Zugriffssteuerungsmodelle (*Model Core* [14, 17]) verallgemeinert wurde. Dieser muss für jedes konkrete Zugriffssteuerungsmodell instanziiert werden, was eine Redefinition der formalen Modellkomponenten mit sich bringt.

Für die tatsächliche Analyse muss schließlich eine hinsichtlich der Entscheidbarkeits- und Komplexitätseigenschaften des Problems adäquate Analysemethode gefunden werden, beispielsweise auf Basis von abstrakter Interpretation [22, 10, 13] oder Einschränkung der Modellsemantik [18, 21, 1]. Auch dieser Schritt ist in der Praxis politik- und somit systemspezifisch durchzuführen.

Modellierung mittels Entity Labeling

Aufgrund des Komplexitätsgrades der hier umrissenen formalen Schritte bedingt der Ansatz des modellbasierten Security Engineering einen beträchtlichen Aufwand, der für die Entwicklung von Betriebssystemen sowie die Erweiterung existierender Systeme infolge strengerer Sicherheitsanforderungen ein grundlegendes Hemmnis darstellt.

Die angesprochenen Hürden bei der Modellierung und Analyse von Zugriffssteuerungspolitiken gehen im Wesentlichen auf zwei Ziele zurück: (1.) die präzise Abbildung einer konkreten Politiksemantik und (2.) die Analyse einer konkreten Sicherheitseigenschaft. Gleichwohl lassen sich für beide Ziele Gemeinsamkeiten identifizieren, die unabhängig vom konkreten System und dessen Anforderungskatalog sind. Auf dieser Idee basiert das Prinzip des *Entity Labeling* [2, 3], welches den Prozess des modellbasierten Security Engineering auf die gemeinsamen Eigenschaften von Betriebssystem-Zugriffssteuerungspolitiken zuschneidet:

1. Politiksemantik: Aktiven und passiven Entitäten (*Entities*) werden durch Metainformationen sicherheitsrelevante *Labels* zugewiesen, welche wiederum Gegenstand logischer Ausdrücke (*Authorization Rules*) sind.
2. Sicherheitseigenschaft: Die Analyse der Sicherheitspolitik hat eine Ausprägung der Sicherheitseigenschaft „Rechteausbreitung“ zum Ziel, welche durch eine automatenbasierte Formalisierung der o. g. Politiksemantik beschrieben wird.

Ziel ist es, die Umsetzung kritischer Sicherheitseigenschaften in Betriebssystemen effizienter zu gestalten. Hierzu wird der hier vorgestellte Beitrag folgende Details zum *Entity-Labeling*-Prinzip und seiner Anwendung beinhalten:

- Die Abstraktion von Betriebssystem-Zugriffssteuerungspolitiken in Form von semantischen Oberklassen, die für konkrete Systeme spezialisiert werden,
- die Abstraktion von Rechteausbreitungseigenschaften, die basierend auf den Instanzen dieser Oberklassen in konkrete Definitionen überführt werden,
- die Abstraktion einer Analysemethode für diese Eigenschaften, die basierend auf deren konkreter Definition in einen Algorithmus überführt wird.

Diese werden prototypisch auf das politikgesteuerte Betriebssystem SELinux angewandt.

Literaturreferenzen

[1] Tahmina Ahmed and Ravi Sandhu. *Safety of ABAC$_\alpha$ Is Decidable*, pages 257–272. Springer International Publishing, Cham, 2017.

[2] Peter Amthor. *E-Business and Telecommunications: 12th International Joint Conference, ICETE 2015, Colmar, France, July 20–22, 2015, Revised Selected Papers*, chapter The Entity Labeling Pattern for Modeling Operating Systems Access Control, pages 270–292. Springer International Publishing, Cham, 2016.

[3] Peter Amthor. *An Aspect-oriented Approach to Model-based Security Engineering*. PhD thesis, Technische Universität Ilmenau, Ilmenau, Germany, March 2018.

[4] Peter Amthor, Winfried E. Kühnhauser, and Anja Pölck. Model-based Safety Analysis of SELinux Security Policies. In P. Samarati, S. Foresti, J. Hu, and G. Livraga, editors, *In Proc. of 5th Int. Conference on Network and System Security*, pages 208–215. IEEE, 2011.

[5] Peter Amthor, Winfried E. Kühnhauser, and Anja Pölck. Heuristic Safety Analysis of Access Control Models. In *Proceedings of the 18th ACM Symposium on Access Control Models and Technologies*, SACMAT '13, pages 137–148, New York, NY, USA, 2013. ACM.

[6] Guillaume Benats, Arosha Bandara, Yijun Yu, Jean-Noël Colin, and Bashar Nuseibeh. PrimAndroid: Privacy Policy Modelling and Analysis for Android Applications. In *2011 IEEE International Symposium on Policies for Distributed Systems and Networks (Policy 2011)*, pages 129–132. IEEE, 2011.

[7] Sven Bugiel, Stephan Heuser, and Ahmad-Reza Sadeghi. Flexible and Fine-Grained Mandatory Access Control on Android for Diverse Security and Privacy Policies. In *22nd USENIX Security Symposium (USENIX Security '13)*. USENIX, August 2013.

[8] George Coker. Xen Security Modules (XSM). Xen Summit Fall 2006, 2006. `http://www-archive.xenproject.org/files/xensummit_4/xsm-summit-041707_Coker.pdf` (accessed: 2018-07-27).

[9] Glenn Faden. Multilevel Filesystems in Solaris Trusted Extensions. In *Proceedings of the 12th ACM Symposium on Access Control Models and Technologies*, SACMAT '07, pages 121–126, New York, NY, USA, 2007. ACM.

[10] Anna Lisa Ferrara, P. Madhusudan, and Gennaro Parlato. Security Analysis of Role-Based Access Control Through Program Verification. In *Proceedings of the 2012 IEEE 25th Computer Security Foundations Symposium*, CSF '12, pages 113–125, Washington, DC, USA, 2012. IEEE Computer Society.

[11] Roger A. Grimes and Jesper M. Johansson. *Windows Vista Security: Securing Vista Against Malicious Attacks*. John Wiley & Sons, Inc., New York, NY, USA, 2007.

[12] Michael A. Harrison, Walter L. Ruzzo, and Jeffrey D. Ullman. Protection in Operating Systems. *Communications of the ACM*, 19(8):461–471, August 1976.

[13] Karthick Jayaraman, Vijay Ganesh, Mahesh Tripunitara, Martin Rinard, and Steve Chapin. Automatic Error Finding in Access-Control Policies. In *Proceedings of the 18th ACM conference on Computer and communications security*, CCS '11, pages 163–174, New York, NY, USA, 2011. ACM.

[14] Winfried E. Kühnhauser and Anja Pölck. Towards Access Control Model Engineering. In *Proc. 7th Int. Conf. on Information Systems Security*, ICISS'11, pages 379–382, Berlin, Heidelberg, 2011. Springer-Verlag.

[15] R. J. Lipton and L. Snyder. A Linear Time Algorithm for Deciding Subject Security. *Journal of the ACM*, 24(3):455–464, 1977.

[16] Peter A. Loscocco and Stephen D. Smalley. Integrating Flexible Support for Security Policies into the Linux Operating System. In Clem Cole, editor, *2001 USENIX Annual Technical Conference*, pages 29–42, 2001.

[17] Anja Pölck. *Small TCBs of Policy-controlled Operating Systems*. Universitätsverlag Ilmenau, May 2014.

[18] Ravi S. Sandhu. The Typed Access Matrix Model. In *Proceedings of the 1992 IEEE Symposium on Security and Privacy*, SP '92, pages 122–136, Washington, DC, USA, 1992. IEEE Computer Society.

[19] Wook Shin, Shinsaku Kiyomoto, Kazuhide Fukushima, and Toshiaki Tanaka. A Formal Model to Analyze the Permission Authorization and Enforcement in the Android Framework. In *2010 IEEE Second International Conference on Social Computing (SocialCom)*, pages 944–951, Aug 2010.

[20] Stephen Smalley and Robert Craig. Security Enhanced (SE) Android: Bringing Flexible MAC to Android. In *20th Annual Network & Distributed System Security Symposium (NDSS)*, February 2013.

[21] J. A. Solworth and R. H. Sloan. A layered design of discretionary access controls with decidable safety properties. In *IEEE Symposium on Security and Privacy, 2004. Proceedings. 2004*, pages 56–67, May 2004.

[22] Scott D. Stoller, Ping Yang, Mikhail Gofman, and C. R. Ramakrishnan. Symbolic Reachability Analysis for Parameterized Administrative Role Based Access Control. *Computers & Security*, 30(2-3):148–164, 2011.

[23] Robert Watson and Chris Vance. Security-Enhanced BSD. Technical report, Network Associates Laboratories, Rockville, MD, USA, July 2003.

[24] Robert N. M. Watson. A Decade of OS Access-control Extensibility. *ACM Queue*, 11(1):20:20–20:41, January 2013.

[25] Chris Wright, Crispin Cowan, Stephen Smalley, James Morris, and Greg Kroah-Hartman. Linux Security Modules: General Security Support for the Linux Kernel. In *Proceedings of the 11th USENIX Security Symposium*, pages 17–31, Berkeley, CA, USA, 2002. USENIX Association.

[26] Giorgio Zanin and Luigi Vincenzo Mancini. Towards a Formal Model for Security Policies Specification and Validation in the SELinux System. In *Proc. of the 9th ACM Symposium on Access Control Models and Technologies*, pages 136–145. ACM, 2004.