H. Langweg, M. Meier, B.C. Witt, D. Reinhardt (Hrsg.): Sicherheit 2018,
Lecture Notes in Informatics (LNI), Gesellschaft für Informatik, Bonn 2018 285

Usability von Security-APIs für massiv-skalierbare vernetzte Service-orientierte Systeme

Peter Leo Gorski[1]

Abstract: Kontemporäre Service-orientierte Systeme sind hochgradig vernetzt und haben zudem die Eigenschaft massiv-skalierbar zu sein. Diese Charakteristiken stellen im besonderen Maße Anforderungen an die Datensicherheit der Anwender solcher Systeme und damit primär an alle Stakeholder der Softwareentwicklung, die in der Verantwortung sind, passgenaue Sicherheitsmechanismen effektiv in die Softwareprodukte zu bringen.

Die Effektivität von Sicherheitsarchitekturen in service-orientierten Systemen hängt maßgeblich von der richtigen Nutzung und Integration von Security-APIs durch eine heterogene Gruppe von Softwareentwicklern ab, bei der nicht per se ein fundiertes Hintergrundwissen über komplexe digitale Sicherheitsmechanismen vorausgesetzt werden kann. Die Diskrepanz zwischen komplexen und in der Anwendung fehleranfälligen APIs und einem fehlenden Verständnis für die zugrundeliegenden Sicherheitskonzepte auf Seiten der Nutzer begünstigt in der Praxis unsichere Softwaresysteme. Aus diesem Grund ist die Gebrauchstauglichkeit von Security-APIs besonders relevant, damit Programmierer den benötigten Funktionsumfang effektiv, effizient und zufriedenstellend verwenden können.

Abgeleitet von dieser Problemstellung, konzentriert sich das Dissertationsvorhaben auf die gebrauchstaugliche Ausgestaltung von Security-APIs und den Herausforderungen die sich aus den Methoden zur Evaluation der Usability in typischen Umgebungen der Softwareentwicklung ergeben.

Keywords: Security-APIs; Usability; Usable Security; Softwareentwicklung

1 Einleitung

Application Programming Interfaces (dt. Schnittstellen zur Anwendungsprogrammierung) oder kurz *APIs*, sind aufgrund der fortschreitenden digitalen Transformation in allen Lebensbereichen und des hohen Vernetzungsgrades heutiger Technik allgegenwärtig. Der Grund dafür liegt in dem Konzept, das mithilfe von APIs umgesetzt werden kann: die einfache Nutzung bzw. *Wiederverwendung* von bereits verfügbaren *Funktionalitäten* durch *Abstraktion*. Bei Funktionalitäten handelt es sich z.B. um bereits entwickelte Programmteile, Implementierungen von mathematischen Lösungsverfahren, GUI-Elementen, Sicherheitsmechanismen, Hardwarefunktionen oder komplexe Daten wie aktuelle Wettervorhersagen.

[1] Technische Hochschule Köln, Data & Application Security Group, Betzdorfer Straße 2, 50679 Köln / Technische Universität Berlin, Quality and Usability Lab, Ernst-Reuter-Platz 7, 10587 Berlin peter.gorski@th-koeln.de

doi:10.18420/sicherheit2018_26

Die Menge an Funktionalitäten umfasst potenziell alle Problemstellungen, die durch Software gelöst werden können. Durch die Abstraktion in Form einer API reduziert sich die Komplexität, welche der Problemlösung bzw. Implementierung anhaftet. Dabei werden Informationen reduziert, die für die Nutzung der eigentlichen Funktionalität irrelevant sind, wodurch „einfach“ wiederverwendbare Softwarekomponenten entstehen. Diese können als Teillösungen weiterer Problemstellungen dienen. Damit ist es möglich, ein System aufzubauen, das sich aus aufeinander aufbauenden Softwarebausteinen zusammensetzt und mit jedem hinzukommenden Bauteil an Funktionalität und Komplexität zunimmt. APIs bilden die elementaren Bindeglieder dieses modularen Aufbaus.

Diesem Prinzip folgend, basieren wiederverwendbare Komponenten von Sicherheitssoftware auf komplexen digitalen Sicherheitskonzepten. Security-APIs werden vor diesem Hintergrund von spezialisierten Entwicklern entworfen und implementiert. Eine Security-API kann als Programmierschnittstelle definiert werden, die dem Softwareentwickler eine Sicherheitsfunktionalität zur Verfügung stellt, welche eine oder mehrere Sicherheitsregeln bei der Interaktion zwischen mindestens zwei Entitäten durchsetzt [GLI16]. Wenn es darum geht Sicherheitsmechanismen in diverse Softwareprodukte zu integrieren, greifen Programmierer auf diese Security-APIs zurück, insbesondere auch solche Entwickler, die nicht über ein fundiertes Hintergrundwissen über die eingesetzten Sicherheitsfunktionen verfügen. Die Sicherheit heutiger Software hängt daher vor allem von der effektiven Nutzung dieser Security-APIs durch eine heterogene Gruppe von Softwareentwicklern ab [Ge12, Fa13, Fa12, Na16, Ac17, LIG17].

2 Problemstellung

Eine wichtige Erkenntnis aus dem Forschungsgebiet *Usable Security und Privacy* ist, dass nutzerzentrierte Ansätze einen wichtigen Faktor für die kontinuierliche Verbreitung, Akzeptanz, Weiterentwicklung und besonders für die effektive, effiziente und zufriedenstellende Anwendung von Sicherheitskomponenten bilden [FHLIM10]. Werden Security-APIs nicht entsprechend den Anforderungen von Softwareentwicklern ausgestaltet, kommt es zur unwillentlichen oder auch willentlichen Fehl- bzw. Nichtbenutzung, wodurch die Sicherheitsdienste nicht mehr oder nur sehr eingeschränkt erbracht werden.

Softwareentwickler nehmen dabei eine Schlüsselrolle ein, denn der Fehler eines einzelnen Programmierers bei der Implementierung einer Sicherheitsfunktion hat durch die starke Vernetzung heutiger Systeme einen unmittelbar negativen Lawinen-Effekt auf die Sicherheit von Endanwendern. Daraus resultierende Sicherheitsprobleme sind z.B. im Bezug auf das Transport-Layer-Security-Protokoll (TLS) [DR08] aufgedeckt worden. Dies konnte für Programmierer ohne Sicherheitsexpertise [Na16, Ac17] aber auch für Sicherheitsexperten [Du17] belegt werden.

In diesen Fällen ist die Zertifikatsvalidierung von Softwareentwicklern fehlerhaft implementiert worden. Dadurch führt die Prüfung der Authentizität von empfangenen Daten zu

falschen Ergebnissen. Authentische und nicht authentische Quellen können somit nicht mehr unterschieden werden. Das Fehlen dieses grundlegenden Sicherheitsdienstes kann von einem Angreifer durch einen sogenannten Man-in-the-Middle Angriff [OW15] ausgenutzt werden, um die Vertraulichkeit und Integrität der ausgetauschten Daten aufzuheben. Betroffen waren bzw. sind Android und iOS Applikationen [Fa12, Fa13] und eine Vielzahl von SSL (Secure Socket Layer) – aus dem später TLS hervorgegangen ist – Libraries und Frameworks [Ge12].

Massiv-skalierbare vernetzte Service-orientierten Systemen adressieren eine hohe Anzahl von Endgeräten, welche von Endanwendern in diversen Bereichen des alltäglichen Lebens eingesetzt und genutzt werden. Bei einem Angriff auf diese Art von Systemen ist die Anzahl potenzieller Angriffsziele dementsprechend hoch. Als Beispiel findet der Architekturstil REST (Representational State Transfer) [Fi00], als Entwurfskonzept für solche Systemen, ubiquitär Anwendung in Domänen wie z.B. dem klassischen Internet, dem Internet der Dinge, der Industrie 4.0 oder der Gesellschaft 5.0 [CE17]. Diese Begriffe charakterisieren Vernetzung einhergehend mit einem stetigen Anstieg der Digitalisierung. Die daraus resultierenden Sicherheitsanforderungen müssen von Softwareentwicklern in die diversen Systeme implementiert werden. Eine übergeordnete Forschungsfrage der Usable Security Disziplin lautet daher: Wie lässt sich das Risiko ineffektiver Sicherheitsfunktionen durch geeignete Unterstützung der Softwareentwickler minimieren?

3 Stand der Forschung

Einige grundlegende Ansätze zur Beantwortung dieser Frage hat die Forschung auf dem Gebiet der allgemeinen API-Usability vorgeschlagen [Bl06, St09, Cl10, SK15]. Dabei wurden vor allem drei wichtige Faktoren identifiziert und adressiert: Das API-Design, Dokumentationen und Werkzeuge für den Umgang mit APIs. Ein vorgeschlagenes Modell der API-Usability gibt einen strukturierten Überblick über die bisherigen Erkenntnisse [GLI16].

Diese reichen jedoch nicht alleine aus, um die Gebrauchstauglichkeit von Security-APIs beschreiben zu können, denn aus dem Sicherheitskontext der *Security-APIs* ergeben sich zusätzlich besonders zu berücksichtigende Usability Aspekte [GS16, GLI16], wie z.B. der *Endanwenderschutz*, die *Prinzipientreue*, die *Einschränkbarkeit* oder die *Konfigurierbarkeit*, deren gebrauchstaugliche Ausgestaltung erst noch erforscht werden muss.

Dies gilt auch für spezifische Anforderungen, die sich für die *Uniform Interface* Richtlinie des Architekturstils REST ergeben, um z.B. in Benutzerstudien Berücksichtigung zu finden. Aus den genannten Gründen gilt dies insbesondere für die Gebrauchstauglichkeit von Security-APIs als Teil eines noch zu entwickelnden REST-Security Framework [LING17], deren Funktionalität die Sicherheit von REST-basierten Softwaresystemen gewährleisten soll. Ebenso sind Fragestellungen im Bezug auf die gebrauchstaugliche Ausgestaltung von Sicherheitsmechanismen in Entwicklungsumgebungen sowie von Dokumentationen offen. Aktuelle Arbeiten zeigen hier insbesondere Schwächen bei der Deckung des speziellen

Informationsbedarfs von Softwareentwicklern, bei der Implementierung von Sicherheitsfunktionalitäten auf [Ac16, LIG17].

Um damit verbundene Probleme zu adressieren, müssen Erwartungshaltungen und Informationsbedarfe von Softwareentwicklern erhoben und effektive Ansätze erforscht werden, wie Informationsflüsse zwischen den Entwicklern von Security-APIs und den Anwendern mittels Verknüpfungen von API-intrinsischen Informationen, der Informationsaufbereitung in Dokumentationen und Werkzeugen innerhalb von Entwicklungsumgebungen ermöglicht werden können. Dazu muss ebenfalls analysiert werden, ob es eine typische Entwicklungsumgebung für Softwareentwickler gibt, und wenn ja, wie diese aussieht oder ob eine Gruppierung verschiedener Ausprägungen möglich ist, um diese in Designprozessen und Methoden zur Evaluation berücksichtigen zu können. Bisher vorgeschlagene Evaluationsmethoden der allgemeinen Gebrauchstauglichkeit von APIs wurden von dem Forschungsgebiet der HCI (Human Computer Interaction) adaptiert [St09, Cl10, PFM10, SK15]. Spezielle Anforderungen an diese Methoden im Kontext der Usability Evaluation mit Softwareentwicklern müssen noch erforscht werden.

4 Forschungsfragen

Das Dissertationsvorhaben untersucht die aufgezeigten Fragestellungen bezüglich der gebrauchstauglichen Ausgestaltung von ubiquitär eingesetzten REST-Security Programmierschnittstellen zur effektiven und effizienten Verwendung durch Softwareentwickler. Dazu werden Aspekte wie das API Design, die Dokumentation und Werkzeuge zur Softwareentwicklung im Hinblick auf deren Einfluss auf die Gebrauchstauglichkeit untersucht. Als Grundlage dienen Erkenntnisse und Methoden aus den Bereichen der Usability-Forschung in Bezug auf generelle Programmierinteraktionen, die im Kontext von Sicherheitsbibliotheken transportiert bzw. angepasst, erweitert und evaluiert werden.

Das geplante Dissertationsvorhaben widmet sich zusammenfassend den folgenden Fragestellungen:

- Wie müssen Security-APIs ausgestaltet werden, damit diese von unerfahrenen bzw. nicht-spezialisierten Softwareentwicklern effektiv, effizient und zufriedenstellend verwendet werden können?
- Durch welche Methoden kann die Gebrauchstauglichkeit von Security-APIs evaluiert werden?
- Inwiefern können Unterstützungswerkzeuge zur Verwendung von Security-APIs (integriert in gängige Entwicklungsumgebungen) die Gebrauchstauglichkeit beeinflussen?
- Was für einer Taxonomie folgt das Forschungsfeld der Gebrauchstauglichkeit von Security-APIs?

Danksagung

Das Projekt ULS3 (Ultra-Large Scale Systems Security) wird unter dem Förderkennzeichen 13FH016IX6 im Förderprogramm “Forschung an Fachhochschulen” vom Bundesministerium für Bildung und Forschung (BMBF) gefördert.

Literaturverzeichnis

[Ac16] Acar, Yasemin; Backes, Michael; Fahl, Sascha; Kim, Doowon; Mazurek, Michelle L.; Stransky, Christian: You Get Where You’re Looking for: The Impact of Information Sources on Code Security. In: IEEE Symposium on Security and Privacy. S&P ’16, 2016. doi: 10.1109/SP.2016.25.

[Ac17] Acar, Yasemin; Backes, Michael; Fahl, Sascha; Garfinkel, Simson; Kim, Doowon; Mazurek, Michelle; Stransky, Christian: Comparing the Usability of Cryptographic APIs. In: IEEE Symposium on Security and Privacy. S&P ’17, 2017. http://www.ieee-security.org/TC/SP2017/papers/161.pdf.

[Bl06] Bloch, Joshua: How to design a good API and why it matters. In: OOPSLA ’06, Companion to the 21st ACM SIGPLAN Symposium on Object-oriented Programming Systems, Languages, and Applications. 2006. doi: 10.1145/1176617.1176622.

[CE17] CEBIT: Society 5.0: Japan treibt auf der CeBIT 2017 die Digitalisierung voran. 2017. Online: https://www.cebit.de/de/news-trends/news/society-5-0-japan-treibt-auf-der-cebit-2017-die-digitalisierung-voran-722 (13.02.2018).

[Cl10] Clarke, Steven: How Usable Are Your APIs? In (Oram, Andy; Wilson, Greg, Hrsg.): Making software: What really works, and why we believe it, Theory in practice, S. 545 – 565. O’Reilly, Beijing, 1. Auflage, 2010.

[DR08] Dierks, T.; Rescorla, E.: RFC 5246 - The Transport Layer Security (TLS) Protocol Version 1.2. Proposed Standard,, Internet Engineering Task Force (IETF), 2008.

[Du17] Durumeric, Zakir; Ma, Zane; Springall, Drew; Barnes, Richard; Sullivan, Nick; Bursztein, Elie; Bailey, Michael; Halderman, J. Alex; Paxson, Vern: The Security Impact of HTTPS Interception. In: The Network and Distributed System Security Symposium 2017. NDSS ’17, 2017. doi: 10.14722/ndss.2017.23456.

[Fa12] Fahl, Sascha; Harbach, Marian; Muders, Thomas; Smith, Matthew; Baumgärtner, Lars; Freisleben, Bernd: Why Eve and Mallory Love Android: An Analysis of Android SSL (In)Security. In: 19th ACM Conference on Computer and Communications Security. CCS ’12, 2012. doi: 10.1145/2382196.2382205.

[Fa13] Fahl, Sascha; Harbach, Marian; Perl, Henning; Koetter, Markus; Smith, Matthew: Rethinking SSL Development in an Appified World. In: 20th ACM SIGSAC Conference on Computer and Communications Security. CCS 2013, 2013. doi: 10.1145/2508859.2516655.

[FHLIM10] Fischer-Hübner, Simone; Lo Iacono, Luigi; Möller, Sebastian: Usable Security und Privacy. Datenschutz und Datensicherheit - DuD, 34(11):773–782, 2010. doi: 10.1007/s11623-010-0210-4.

[Fi00] Fielding, Roy Thomas: Architectural Styles and the Design of Network-based Software Architectures. Dissertation, University of California, 2000.

[Ge12] Georgiev, Martin; Iyengar, Subodh; Jana, Suman; Anubhai, Rishita; Boneh, Dan; Shmatikov, Vitaly: The Most Dangerous Code in the World: Validating SSL Certificates in Non-browser Software. In: 19th ACM Conference on Computer and Communications Security. CCS '12, 2012. doi: 10.1145/2382196.2382204.

[GLI16] Gorski, Peter Leo; Lo Iacono, Luigi: Towards the Usability Evaluation of Security APIs. In: Tenth International Symposium on Human Aspects of Information Security & Assurance. HAISA 2016, 2016. http://www.cscan.org/openaccess/?paperid=287.

[GS16] Green, Matthew; Smith, Matthew: Developers are Not the Enemy!: The Need for Usable Security APIs. IEEE Security & Privacy, 14(5):40–46, 2016. doi: 10.1109/MSP.2016.111.

[LIG17] Lo Iacono, Luigi; Gorski, Peter Leo: I Do and I Understand. Not Yet True for Security APIs. So Sad. In: The 2nd European Workshop on Usable Security. EuroUSEC '17, 2017. doi: 10.14722/eurousec.2017.23015.

[LING17] Lo Iacono, Luigi; Nguyen, Hoai Viet; Gorski, Peter Leo: On the Need for a General REST-Security Framework. ACM Transactions on the Web (TWEB), 2017. In review process.

[Na16] Nadi, Sarah; Krüger, Stefan; Mezini, Mira; Bodden, Eric: Jumping Through Hoops: Why Do Java Developers Struggle with Cryptography APIs? In: The 38th International Conference on Software Engineering. ICSE '16, 2016. doi: 10.1145/2884781.2884790.

[OW15] OWASP: Man-in-the-middle attack. 2015. Online: https://www.owasp.org/index.php/Man-in-the-middle_attack (13.02.2018).

[PFM10] Piccioni, Marco; Furia, Carlo A.; Meyer, Bertrand: An Empirical Study of API Usability. In: ACM/IEEE International Symposium on Empirical Software Engineering and Measurement. ESEM '13, 2013-10. doi: 10.1109/ESEM.2013.14.

[SK15] Scheller, Thomas; Kühn, Eva: Automated Measurement of API Usability: The API Concepts Framework. Information and Software Technology, 61:145–162, 2015. doi: 10.1016/j.infsof.2015.01.009.

[St09] Stylos, Jeffrey: Making APIs More Usable with Improved API Designs, Documentation and Tools. Dissertation, Carnegie Mellon University, 2009. http://www.cs.cmu.edu/ natprog/papers/CMU-CS-09-130-Stylos-Dissertation.pdf.