Integration von UX in den Security Engineering-Prozess

Katharina Joos[1,2], Tobias Straub[1]

Studiengang Wirtschaftsinformatik, Duale Hochschule Baden-Württemberg Stuttgart[1]
CI/ISI, Robert Bosch GmbH Stuttgart[2]

Zusammenfassung

Ein etablierter Security Engineering-Prozess gewährleistet, dass bei der Softwareentwicklung Sicherheitsaspekte systematisch berücksichtigt werden. Außerdem orientieren sich Unternehmen bei der Produktentwicklung zunehmend am Nutzererlebnis (UX) und machen auch hierfür verbindliche Vorgaben. Der vorliegende Beitrag beschreibt, wie getrennt entstandene Security Engineering- und UX-Prozesse in einem Großunternehmen zusammengeführt werden, um den besonderen Anforderungen benutzbarer Sicherheit gerecht zu werden. Auf Basis bekannter Usable Security-Prinzipien und -Patterns wurde ein Katalog für Entwickler erstellt. Anhand von drei für den Unternehmenseinsatz typischen Szenarien wurden Lösungen entwickelt und in Nutzertests und durch Experten evaluiert.

1 Einleitung

Bei der Entwicklung von Software für den sicheren Einsatz in Unternehmen ist eine Reihe von Anforderungen zu berücksichtigen. In der Regel gibt es neben zentralen Vorgaben zum Projektmanagement auch solche, die sich aus der Sicherheitsstrategie ableiten. Eine wichtige Rolle kommt dem Aspekt der Benutzbarkeit zu, da ansonsten die angestrebten Sicherheitsziele nicht wirksam erreicht werden können. Viele Firmen haben zudem UX als Kriterium in der Produktentwicklung erkannt, welches einen Wettbewerbsvorteil bieten kann.

Die Herausforderung für Architekten und Entwickler bei der Erstellung sicherer Software besteht darin, möglicherweise sehr umfangreiche Vorgaben, etwa aus dem BSI-Grundschutz oder unternehmensweiten Sicherheitsleitlinien, umsetzen zu müssen. Typischerweise wird im Software Enigeering nicht-funktionalen Anforderungen, insbesondere der Sicherheit, deutlich weniger Aufmerksamkeit gewidmet als funktionalen. Auch verfügen nicht alle Entwickler über ein entsprechendes Hintergrundwissen oder empfinden den zusätzlich durchzuführenden UX-Prozess als weitere Belastung. Dieser Beitrag stellt eine Vorgehensweise vor, mit deren Hilfe der Aufwand für den UX-Prozess, insbesondere die Nutzerforschung, reduziert werden soll.

Veröffentlicht durch die Gesellschaft für Informatik e. V. 2017 in
M. Burghardt, R. Wimmer, C. Wolff, C. Womser-Hacker (Hrsg.):
Mensch und Computer 2017 – Workshopband, 10.–13. September 2017, Regensburg.
 https://doi.org/10.18420/muc2017-ws05-0402

2 Entwurfswerkzeuge

Zur Unterstützung der Entwickler bietet sich an, das erforderliche Wissen in komprimierter und aufbereiteter Form verfügbar zu machen. Ein bekanntes Beispiel hierfür sind die Entwurfsmuster (engl. *design patterns*) der „Gang of Four" (Gamma et al., 2015), die im Software Engineering hinlänglich bekannt sind. In den Bereichen Usability und Sicherheit gibt es ähnliche Ansätze (s.u.).

Dieser Beitrag verwendet die gängigen Definitionen für die zentralen Entwurfswerkzeuge Prinzipien, Patterns und Richtlinien. Es erfolgt dennoch eine kurze Begriffsklärung, da deren Verwendung in der Literatur nicht immer ganz einheitlich und trennscharf[1] ist. Unter einem *Prinzip* (engl. principle) soll eine allgemeingültige Regel als Richtschnur für das Handeln verstanden werden, die knapp als Merksatz formuliert werden kann. Ein *Pattern*[2] ist eine konkrete Beschreibung der umzusetzenden Schritte, die für die Erreichung eines bestimmten Ziels benötigt werden unter Angabe der Anwendungsvoraussetzungen. Eine *Richtlinie* (engl. *guideline*) ist die Vorgabe einer genauen (eher kleinteiligen) Handlungsanweisung für eine bestimmte Situation.

Im Rahmen der Initiative „Einfach intuitiv – Usability für den Mittelstand" wurden im Projekt *USecureD* (Usable Security by Design) in der Literatur beschriebene Ansätze recherchiert und auf der Webseite usecured.de systematisch aufbereitet (Nehren et al., 2017). Die Liste der zugrunde gelegten Quellen kann dabei den Beschreibungsvorlagen für die Entwurfswerkzeuge entnommen werden.[3] Durch USecureD sind 23 Prinzipien, 33 Richtlinien und 47 Patterns beschrieben worden, wobei jeweils u. a. ihre Quellen aus der Literatur, Beispiele und Querverbindungen angegeben sind. Letztere sind in Abbildung 1 visualisiert.[4]

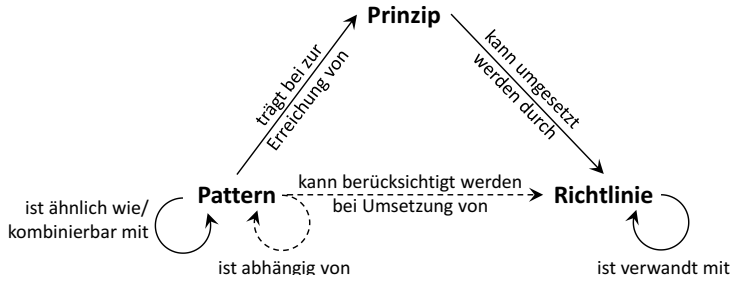


Abbildung 1: Zusammenspiel der Entwurfswerkzeuge (USecureD-Systematik).

[1] Es zeigte sich, dass es in der Praxis durchaus – je nach Blickwinkel – zu Übergängen zwischen den Typen der Entwurfswerkzeuge oder Grenzfällen, bei denen eine Einordnung nicht leicht möglich ist, kommen kann.

[2] Die Begriffe Pattern, Design Pattern und Entwurfsmuster werden hier synonym verwendet.

[3] Siehe Deliverables E2.1a, E2.1b, E2.1c auf der Webseite https://www.usecured.de/ergebnisse/.

[4] Die gestrichelten Pfeile deuten an, dass die zugehörigen Beziehungen in den Beschreibungsvorlagen zwar vorgesehen, aber von USecureD nicht benutzt werden.

2.1 Prinzipien

Bereits 1975 schlugen Saltzer und Schroeder für die Gestaltung sicherer Systeme acht Prinzipien wie etwa *Least Privilege* vor (Saltzer & Schroeder, 1975). Diese sind allgemein anerkannt, auch wenn sie nicht alle in gleichem Maße tatsächlich Eingang in die Entwicklung sicherer Systeme gefunden haben (Smith, 2012). Eines dieser Prinzipien (*Psychological Acceptability*) berührt dabei bereits Aspekte der Usability. Im eigentlichen Bereich der Usability definiert die Norm ISO 9241-110 sieben Grundsätze für die ergonomische Dialoggestaltung. Auch die Heuristiken von Nielsen und Molich (1990), die ursprünglich für den Zweck der Evaluation formuliert wurden, stellen allgemeine Prinzipien dar. Unter dem Begriff *User-Centered Security* (Zurko & Simon, 1996) wird der Ansatz verstanden, Sicherheit und Benutzbarkeit in Einklang zu bringen, indem die Anforderungen der Benutzer bei der Systementwicklung in den Mittelpunkt gestellt werden. Als erster hat Garfinkel (2005) hierfür ein umfassendes Instrumentarium beschrieben. Aufgrund des Interpretationsspielraums, den Prinzipien als möglichst allgemeingültig formulierte Grundsätze haben, ergänzt Garfinkel diese durch Anwendungsfall-bezogene *Patterns* mit einem hohen Konkretisierungsgrad, der die Umsetzung erleichtern soll.

2.2 Patterns

Patterns wurden erstmals in der Architektur beschrieben, bevor sie auf die Softwarearchitektur übertragen wurden. Die maßgeblichen Elemente eines Patterns sind ein *sprechender Name*, die *Problemstellung* (beschreibt die Situation, in der das Pattern anzuwenden ist, um ein spezifisches Problem zu lösen), weitere *Bedingungen* für die Anwendbarkeit sowie die Elemente zur *Lösung* des Problems (Gamma et al., 2015).

Patterns für den Bereich der IT-Sicherheit wurden etwa von Kienzle et al. (2002) beschrieben. Unterschieden wird dabei zwischen *strukturellen Patterns*, die im Endprodukt implementiert sein können (z.B. hinsichtlich des Umgangs mit Passwörtern), und *prozeduralen Patterns*, die helfen, den Entwicklungsprozess zu verbessern (etwa die Empfehlung, dafür zu sorgen, dass sich alle Entwickler für Sicherheitsaspekte verantwortlich fühlen und Auswirkungen ihrer Komponenten bedenken).

2.3 Richtlinien

Für den Einsatz der Entwurfswerkzeuge bei der Robert Bosch GmbH erfolgte – ganz ähnlich wie in (Garfinkel, 2005) – eine Beschränkung auf Prinzipien und Patterns. Es sollte möglichst nur einen Einstiegspunkt für Entwickler geben, um die Komplexität zu reduzieren. Hierfür erschienen Patterns am besten geeignet wegen ihrer Zuordnung zu Anwendungsfällen und auch ihrer Bekanntheit aus der Softwarearchitektur. Die Patterns nehmen Bezug auf zugehörige Prinzipien, so dass deren Beschreibung als Hintergrundwissen dienen kann.

Eine Ergänzung der Prinzipien und Patterns um Richtlinien ist zu einem späteren Zeitpunkt denkbar. Die Herangehensweise, *Checklisten* bestehend aus einzelnen Richtlinien zu bilden,

welche sich sowohl für die Entwicklung als auch die Überprüfung eignen, ist in der IT-Sicherheit etwa durch die Maßnahmenkataloge des BSI ohnehin schon etabliert.

3 Ausgangssituation

Im Folgenden werden die bei der Robert Bosch GmbH verbindlichen Prozesse für Produktentwicklung und Security Engineering sowie der UX-Referenzprozess kurz vorgestellt. Des Weiteren werden die organisatorischen Verantwortlichkeiten im Unternehmen beschrieben.

3.1 Unternehmensvorgaben

Bei der *Produktentwicklung* gibt es grundsätzlich zwei Methoden, die gewählt werden können: Auf dem Wasserfallmodell basiert der *phasenorientierte* Ansatz, der in Projektvorbereitung, Konzept, Realisierung, Produktionsvorbereitung und Stabilisierung gegliedert ist. Für den *agilen* Ansatz ist SCRUM die anzuwendende Methodik.

Die zentrale IT-Sicherheitsstrategie gibt den Rahmen vor für eine *Enterprise Information Security Architecture*. Diese definiert Maßnahmen basierend auf den Grundschutzkatalogen des BSI und der Norm NIST SP 800-53[5]. Sie ist außerdem die Grundlage für die Bedrohungsanalyse im *Security Engineering-Prozess*, der den phasenorientierten Ansatz erweitert. Aus den abgeleiteten Anforderungen an das zu entwickelnde Produkt wird ein Sicherheitskonzept formuliert, in welchem auch konkrete Maßnahmen festgelegt sind. Vor der Freigabe muss das Produkt ein Audit durchlaufen, um mögliche Sicherheitslücken finden und gegebenenfalls nachbessern zu können.

Im *UX-Referenzprozess* werden nach der Festlegung der Rahmenbedingungen und Ziele zu Projektstart vier Phasen iterativ durchlaufen: Phase 1 dient dazu, ein Verständnis für den Nutzer und seine aktuelle Situation zu schaffen. Hierbei sind Nutzerbefragungen, der Besuch des Nutzers am Arbeitsplatz, z. T. auch Nutzertests, üblich. In Phase 2 werden aus Nutzeranforderungen Ideen entwickelt. Teilweise entstehen in dieser Phase auch schon erste Lösungsansätze. Die zuvor entwickelten Ideen werden in Phase 3 konkretisiert und erste Konzepte in Mock-Ups umgesetzt. Phase 4 schließt den Zyklus mit Nutzertests. Die Erkenntnisse fließen in die Entwicklung neuer Lösungen ein.

3.2 Verantwortlichkeiten

Aktuell gibt es verschiedene Funktionen auf Geschäftsbereichsebene, die hinsichtlich der IT-Sicherheit unterstützen. Dazu gehören die entsprechende Zentralabteilung, der *Product Security Officer* und der *Data Security Officer,* der vom *Data Security Partner* im Projekt

[5] Aktuelle Version 4 verfügbar unter http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf.

vertreten wird. Im Projekt ist vor allem der *Projektleiter* für die Einhaltung des Security Engineering-Prozesses verantwortlich, der bei Bedarf von der Zentralabteilung für IT-Sicherheit unterstützt wird. Als zentraler Ansprechpartner für UX im Geschäftsbereich fungiert der *UX Owner*. Ihm melden die *UX Champions* relevante Projekte in ihrer Abteilung. Sie unterstützen zudem die *UX Advokaten*, die in den Projekten für die Einhaltung des UX-Referenzprozesses verantwortlich sind. Abbildung 2 gibt einen Überblick über die wichtigsten Funktionen, Verantwortlichkeiten und Prozesse.

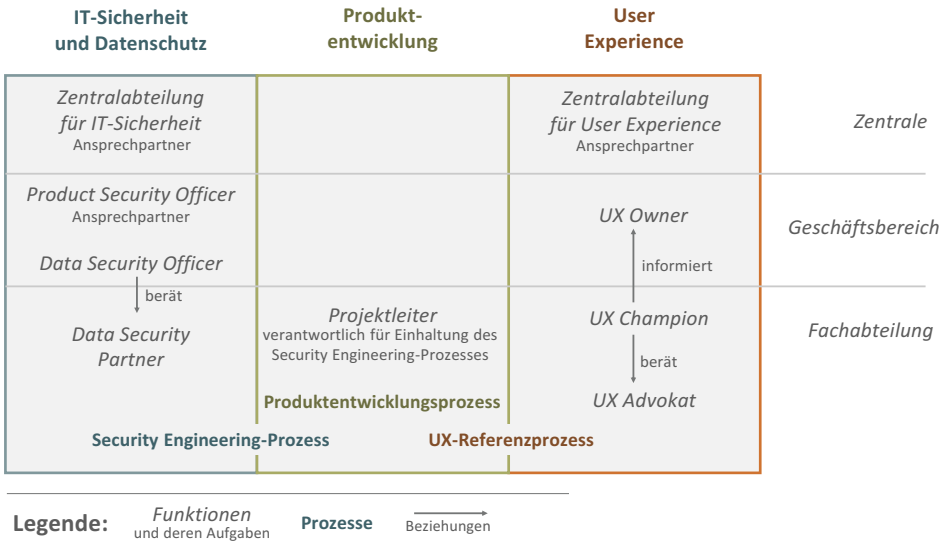


Abbildung 2: Auswahl der wichtigsten Funktionen im Projekt bezüglich IT-Sicherheit und UX.

4 Lösungsentwicklung

Durch die oben beschriebenen Prozesse werden sowohl UX als auch die Belange der IT-Sicherheit in der Produktentwicklung berücksichtigt. Allerdings wurden beide Aspekte bei der Robert Bosch GmbH bisher noch getrennt voneinander betrachtet. Schnittstellen gibt es wenige, insbesondere auch keine konkrete Funktion, die beide Bereiche verbindet. Eine engere Verzahnung ist aber aus Unternehmenssicht unumgänglich.

4.1 Vorgehensweise

Um sicherzustellen, dass IT-Sicherheitslösungen nutzerfreundlich gestaltet werden, wurde der Ansatz gewählt, UX-Aspekte in den Anforderungskatalog der Enterprise Information

Security Architecture zu integrieren. Hierfür wurden zunächst bekannte[6] Prinzipien geprüft und die für den Unternehmenskontext relevanten ausgewählt – wo nötig, in erweiterter oder zusammengefasster Form. Diese Prinzipien wurden bei der Entwicklung neuer Lösungen, die wiederum mit Nutzern getestet wurden, angewendet. Mit den Erkenntnissen der Nutzertests wurden schließlich die Patterns formuliert. Dieses Vorgehen ist an den UX-Referenzprozess angelehnt. Es besteht daher die Erwartung, dass Entwickler den späteren Aufwand für die Nutzerforschung in ihrem Projekt stark reduzieren können.

4.2 Verankerung von Usable Security in bestehende Prozesse

Die Zentralabteilung für IT-Sicherheit trägt die Verantwortung, die Kataloge der Prinzipien und Patterns zu pflegen und bei Bedarf zu erweitern. Diese Erweiterungen erfordern dasselbe Vorgehen, welches in den folgenden Abschnitten vorgestellt wird. Durch die Integration der Kataloge in den Security Engineering-Prozess soll die Anwendung der Patterns sichergestellt werden.

Die Übertragung der Aufgaben an die Zentralabteilung für IT-Sicherheit erfolgte vor dem Hintergrund, dass im Unternehmen die Verantwortung für UX generell in den Fachbereichen liegt und bewusst nicht bei einer zentralen Stelle. Ein Teil der Mitarbeiter der Zentralabteilung sind entsprechend für UX geschult. Insbesondere sind der dortige UX Champion und die dortigen UX Advokaten mit den Methoden vertraut.

Schulungen für Entwickler sind noch nicht in größerem Maßstab erfolgt, doch werden die entwickelten Kataloge allen Entwicklern im Intranet verfügbar gemacht. Ihre Verwendung geschieht dabei am einfachsten über die Suche nach einem passenden Anwendungsfall im Pattern-Katalog. Für einen Entwickler aus dem Bereich Identitätsmanagement (IdM) könnte dies etwa der Anwendungsfall „Authentifizierung“ sein. Er findet dort zu seinem Aufgabengebiet passende Patterns. Diese verweisen auf die zugrundeliegenden Prinzipien, so dass sich darüber vertiefende Informationen beschaffen lassen.

4.2.1 Festlegung von Anwendungsfällen

Für die Definition von Patterns wurden drei Anwendungsfälle (AF) nach der Einsatzhäufigkeit im Unternehmen priorisiert.[7] Die formulierten Patterns sprechen primär die Nutzerfreundlichkeit für den Endnutzer, also den Mitarbeiter des Unternehmens, an.

(AF 1) *Authentifizierung*: beinhaltet Anforderungen an die Authentifizierung nach Wissen, Besitz und basierend auf biometrischen Merkmalen
(AF 2) *Autorisierung und Verwaltung von Zugriffsrechten*: enthält Patterns für File Storage und File Sharing und beinhaltet insbesondere den Umgang mit Schutzklassen
(AF 3) *Social Engineering:* Bedrohungen durch Täuschung und Schadsoftware werden am Beispiel von E-Mail betrachtet

---

6 Unter Verwendung insbesondere der Ergebnisse von USecureD, der dort bereits genannten Quellen sowie von (Cooper et al., 2014).

7 Bei (AF 1) und (AF 2) handelt es sich um Querschnittsaufgaben zur Umsetzung grundlegender Sicherheitsziele, die sich an vielen Stellen wiederfinden. (AF 3) wurde ausgewählt, da E-Mail ein weit verbreiteter Übertragungsweg für Schadsoftware ist.

4.2.2 Auswahl relevanter Prinzipien

Der Katalog von Prinzipien wurde mit der Methode des Clustering entwickelt, welche auf der Qualitativen Inhaltsanalyse (Mayring, 2015) basiert. Möglicherweise relevante Prinzipien wurden recherchiert, auf Post-its notiert und gruppiert. Ähnliche Prinzipien wurden so derselben Kategorie zugeordnet oder gar zusammengefasst. Auf diese Weise entstanden teilweise auch neue Prinzipien. Wie in Abschnitt 2 beschrieben, sind Prinzipien auf einem hohen Abstraktionsniveau formuliert und daher verschiedentlich anwendbar. Sie haben für die Entwickler eine Begründungsfunktion, warum ein Pattern bestimmte Anforderungen enthält. Prinzipien können dabei auch genutzt werden, wenn noch keine Patterns für ein Themengebiet entwickelt wurden.

4.2.3 Übertragung der Prinzipien auf die Anwendungsfälle

Im nächsten Schritt wurden Lösungen in Form von Mock-Ups[8] für die oben aufgeführten Anwendungsfälle mit Hilfe der entwickelten Prinzipien entworfen. Dabei wurde geprüft, welche Prinzipien anwendbar sind, um sowohl die Nutzerfreundlichkeit als auch die Sicherheit zu erhöhen. Für (AF 1) wurden verschiedene Alternativen der Benutzerauthentifikation verglichen, für (AF 2) wurde das Konzept eines neuen Dokumentenverwaltungssystems entworfen. Für (AF 3) wurden unterstützende Funktionen im Mail-Client zur Erkennung von Phishing entwickelt.

4.3 Evaluation

Die Mock-Ups wurden mit Endnutzern mit Hilfe der Methode *Thinking Aloud* (Ericsson & Simon, 1980), eingerahmt durch eine Vor- und Nachbefragung, getestet. Für die Tests wurden fünf Probanden aus verschiedenen Unternehmensbereichen, mit und ohne IT-Hintergrund rekrutiert. Im UX-Umfeld ist es üblich, mit wenigen Nutzern zu testen, dafür aber mehrere Iterationen durchzuführen. Laut (Nielsen, 2000) können schon mit fünf Nutzern ca. 85% der Erkenntnisse gewonnen werden. Auf mehrere Iterationen wurde bisher noch verzichtet, da in diesen nur noch Details diskutiert werden (Nielsen, 2000). Eine allzu große Detailtiefe ist aber für die Entwicklung der Patterns ohnehin nicht erforderlich. An den jeweils ca. einstündigen semistrukturierten Experteninterviews nahmen drei Spezialisten mit mehrjähriger Erfahrung im Bereich IT-Sicherheit teil.

4.4 Ableitung von Patterns

Die Erkenntnisse der Evaluation mündeten in einen Katalog von insgesamt 39 Patterns, die auf 14 Prinzipien verweisen. Da eine ausführliche Darstellung aus Platzgründen hier nicht möglich ist, wird im Folgenden nur auf zwei der Patterns näher eingegangen und für ein längeres Beispiel auf den Anhang verwiesen.

---

[8] Um die Aufwände gering zu halten, erfolgte die Erstellung der Mock-Ups – je nach Komplexität des Anwendungsfalls – mit dem Werkzeug Balsamiq, per Bildbearbeitung von Screenshots sowie mit Hilfe von Powerpoint-Präsentationen mit Steuerelementen.

Das Pattern *Single Sign-On (SSO)* wendet sich dem Problem zu, dass Nutzer – gerade im betrieblichen Kontext – mit der Zahl ihrer Passwörter überfordert sind und sich daher mit möglicherweise unsicheren Mechanismen behelfen, um sie nicht zu vergessen. Das Grundprinzip, dass Nutzer nach einer initialen Authentifikation an einem zentralen System transparent für weitere Systeme autorisiert werden, wurde sowohl von Endnutzern als auch Experten durchgängig positiv bewertet. Letztere betonten aber das Schadenspotential eines kompromittierten Accounts und sprachen sich daher für die Vorgabe einer Zwei-Faktor-Authentifizierung aus. Dagegen hatten wiederum die Endnutzer nichts einzuwenden, sofern tatsächlich auch alle Systeme über SSO erreichbar sind.

Sehr uneinheitlich bewerteten die Nutzer dagegen das Pattern *Angabe der Schutzklassen*. Hintergrund ist, dass (zumindest in der Theorie) unternehmensweit alle Informationen nach ihrem Schutzbedarf klassifiziert werden sollten. Die entwickelte Lösung sah vor, den Nutzer bereits beim Anlegen von Dokumenten zur Angabe einer Vertraulichkeitsstufe zu zwingen. Ein Teil der Nutzer empfand die Pflichtangabe als störend, würde sich aber eine Möglichkeit der Klassifikation auf freiwilliger Basis wünschen. Nur eines der momentan im Unternehmen eingesetzten Systeme unterstützt die Klassifikation bereits, so dass das erstellte Pattern nun die Implementierung einer Möglichkeit zur Angabe der Schutzwürdigkeit empfiehlt. Wie eine solche Angabe verpflichtend gemacht werden kann, ohne die Usability zu sehr zu beeinträchtigen, muss aber noch weiter untersucht werden.

5 Bewertung und Ausblick

Um die Integration von User Experience in den Security Engineering-Prozess bestmöglich zu gewährleisten, wurde auf Basis bestehender Ansätze ein eigener Katalog von Design-Prinzipien und -Patterns für drei typische Anwendungsfälle entwickelt. Ausgangspunkt war die Überlegung, dass die Herangehensweise in Form von Katalogen Entwicklern eine leichtgewichtige, aber dennoch effektive Hilfestellung bietet, um Produkte nutzerfreundlich und sicher zu gestalten. Durch Patterns können Anforderungen benutzbarer Sicherheit detailliert ausgedrückt werden. Anders als Richtlinien, die möglicherweise als zu kleinteilig und einengend empfunden werden, lassen Patterns dem Entwickler noch genügend Flexibilität im Hinblick auf die konkrete Umsetzung im Projekt.

Wir erwarten, dass eine konsequente Verwendung der Patterns die Ausgangsbasis für den UX-Referenzprozess deutlich verbessert und somit die Zahl der nötigen Iterationen reduziert. Zudem soll der Ansatz für eine frühzeitige Berücksichtigung von Anforderungen sorgen, um das Problem zu vermeiden, dass sich Usable Security gegen Ende des Entwicklungsprozesses nicht mehr einfach erreichen lässt (Yee, 2004).

Die entwickelten Patterns und Prinzipien beruhen auf den Bedürfnissen der Nutzer. Da sich diese nur langsam ändern (Cooper et al., 2014), sind Kataloge auch ein nachhaltiger Ansatz. Eine ständige Pflege und Weiterentwicklung sollte nicht erforderlich sein, vielmehr dürfte eine periodische Überprüfung der Relevanz und Gültigkeit der Prinzipien und Patterns genügen. Eine Evaluation der Konzepte auf Basis von Mock-Ups erfolgte mit Endnutzern.

Denkbar ist aber auch eine vorherige Heuristische Evaluation, sofern entsprechend viele Usability-Experten verfügbar sind (Nielsen und Molich (1990) schlagen die Zahl von drei bis fünf Experten vor).

Kataloge sind nur eine Möglichkeit, die User Experience bei sicherheitsrelevanten Produkten zu optimieren. Natürlich sind sie aber keine automatische Garantie für Verbesserungen: Zum einen besteht die Gefahr, dass Patterns von den Entwicklern gar nicht angewandt werden – sei es aus purer Unkenntnis, mangelnder Erfahrung oder weil sie sich der Bedeutung von Usable Security nicht bewusst sind. Zum anderen ist es möglich, dass sich Entwickler zu sehr auf die Patterns verlassen und selbst keine oder nur unzureichende Nutzerforschung betreiben.

Erste praktische Erfahrungen werden aktuell in Zusammenarbeit mit einem UX Advokaten und den Entwicklern eines Projekts zur Migration des unternehmensweiten IdM-Systems gewonnen. In den nächsten Monaten sollen die entwickelten Patterns in die ohnehin in Überarbeitung befindliche Enterprise Information Security Architecture integriert werden. Dabei soll auch eine mögliche Ergänzung der Patterns mit Beispielcode bis hin zur Bereitstellung von Referenzimplementierungen für bestimmte Funktionen diskutiert werden.

Literaturverzeichnis

Cooper, A., Reimann, R., Cronin, D. & Noessl, C. (2014). *About face, The essentials of interaction design*, 4. Auflage, Indianapolis: Wiley.

Ericsson, K. A. & Simon, H. A. (1980). Verbal Reports as Data. *Psychological Review, 87*(3), 215-251.

Gamma, E., Richard, H., Johnson, R. & Vlissides, J. (2015). *Design Patterns – Entwurfsmuster als Elemente wiederverwendbarer objektorientierter Software.* Frechen: mitp.

Garfinkel, S. (2005): *Design Principles and Patterns for Computer Systems That Are Simultaneously Secure and Usable*. Dissertation, Massachusetts Institute of Technology.

Kienzle, D. M., Elder, M. C., Tyree, D. & Edwards-Hewitt, J. (2002). Security patterns repository version 1.0. *DARPA, Washington D.C*.

Mayring, P. (2015). *Qualitative Inhaltsanalyse – Grundlagen und Technik.* Weinheim: Beltz Verlag.

Nehren, P., Schmitt, H. & Lo Iacono, L. (2017). Usable Security – Werkzeuge für Entwickler. Mittelstand-Digital – Wissenschaft trifft Praxis, 6, 14-20,

Nielsen, J. (1993). *Usability Engineering.* San Diego: Academic Press.

Nielsen, J. (2000). *Why You Only Need to Test with 5 Users.* https://www.nngroup.com/articles/why-you-only-need-to-test-with-5-users/, (Abruf: 02.06.2017).

Nielsen, J. & Molich, R. (1990). Heuristic evaluation of user interfaces. *Proceedings of the SIGCHI conference on Human factors in computing systems, ACM,* 249-256.

Saltzer. J. H. & Schroeder, M. D. (1975). The protection of information in computer systems. *Proceedings of the IEEE, 63*(9), 1278-1308.

Smith, R. E. (2012). A Contemporary Look at Saltzer and Schroeder's 1975 Design Principles. *IEEE Security & Privacy, 10*(6), 20-25.

Yee, K. (2004). Aligning security and usability. *IEEE Security & Privacy, 2*(5), 48-55.

Zurko, M.E. & Simon, R. R. (1996). User-centered security. *Proceedings of the 1996 workshop on New security paradigms (NSPW '96), ACM,* 27-33.

Anhang: Beispiel-Pattern

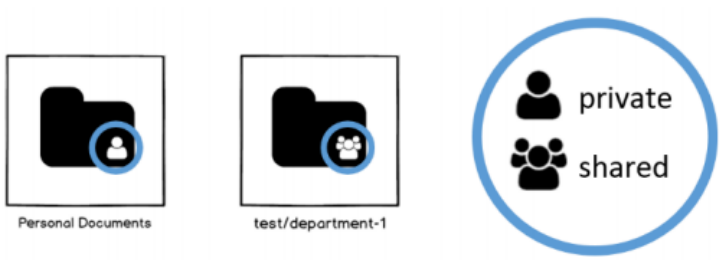
Angewandt auf den zweiten Anwendungsfall, führt das Prinzip "Sichtbarkeit", welches besagt, dass der aktuelle Systemstatus sowie laufende Vorgänge dem Nutzer transparent gemacht werden müssen, zum nachfolgend beschriebenen Pattern. Damit soll das Problem gelöst werden, dass bei Dateiablagen wie etwa Laufwerken oder webbasierten Speicherdiensten Zugriffsberechtigungen anderer Personen nicht immer leicht erkennbar sind und getroffene Einstellungen zu unberechtigter Kenntnisnahme oder Veränderung von Informationen führen können. Interessanterweise bieten selbst gängige Systeme wie der Windows Explorer, Microsoft Sharepoint oder Dropbox keine Möglichkeit, wirklich auf einen Blick zu erkennen, welche der Ordner und Dateien für andere Personen freigegeben wurden, obwohl dies, wie im Pattern beschrieben, leicht umsetzbar wäre.

Als Lösungsmöglichkeiten werden eine grafische Hervorhebung und eine Informationsleiste vorgeschlagen, welche mittels Mock-Up (siehe Abbildung 3) getestet wurden.

Abbildung 3: Mock-up eines Dokumentenverwaltungssystems mit Hinweisen auf geteilte Ordner und Dateien.

Die befragten Nutzer vermissten vergleichbare Funktionen in den von ihnen verwendeten Systemen. Über die Notwendigkeit einer Informationsleiste zusätzlich zu den Icons waren sie sich dagegen uneins, etwa die Hälfte empfand sie als überflüssig. Aus den Erkenntnissen wurde unter anderem das Pattern entwickelt, welches hier auszugsweise wiedergegeben wird. Mit seiner Hilfe können Entwickler die Sicherheit in ihrem System erhöhen, ohne die Nutzerfreundlichkeit zu beeinträchtigen. Für die Kennzeichnung können sie, wie vorgeschlagen, Icons einsetzen oder eigene Ideen entwickeln. Ein Beispiel wäre, an einer freigegebenen Ressource die Profilbilder der autorisierten Benutzer anzuzeigen (soweit im jeweiligen System vorhanden).

Pattern: Anzeige von Berechtigungen Dritter

Kontext/ Problem	Nutzern ist unklar, ob auf eigene Objekte (Ordner, Dateien, andere Ressourcen) von anderen Personen zugegriffen werden kann. Nutzer vergessen, nicht mehr benötigte Berechtigungen wieder zu entziehen.
Beispiel	
Beschreibung	Zeige dem Nutzer die Berechtigungen anderer Personen an.
Anforde-rungen	• *Zeige an, wenn eine Datei oder ein Ordner auch für weitere Personen freigegeben ist.* Der Nutzer soll auf einen Blick sehen können, ob auch andere Personen Zugriff auf ein Dokument oder einen Ordner haben. • *Zeige diese Information direkt am Objekt an.* Hierfür eignen sich Icons oder der Einsatz von Farben.
Abhängigk.	keine Abhängigkeiten (setzt keine anderen Patterns voraus)
Prinzipien	Sichtbarkeit, Rückmeldung an den Nutzer, Konsistenz

Kontakt

Katharina Joos <katharina.joos@de.bosch.com>
Tobias Straub <tobias.straub@dhbw-stuttgart.de>