Howard • Gantenbein • Curzi

Microsoft Azure Security

Bewährte Methoden, Prozesse und Grundprinzipien für das Entwerfen und Entwickeln sicherer Anwendungen in der Cloud

Microsoft

dpunkt.verlag

Michael Howard ist derzeit Principal Security Program Manager im Azure Data Platform Team und arbeitet an der Sicherheitstechnik. Er ist einer der ursprünglichen Architekten des Microsoft Security Development Lifecycle und unterstützt verschiedene Kunden aus den Bereichen öffentliche Verwaltung, Militär, Bildung, Finanzen und Gesundheitswesen bei der Sicherung ihrer Azure-Workloads. Er war der Leiter Application Security für die Olympischen Spiele in Rio 2016; die Anwendungen wurden auf Azure gehostet.

Heinrich Gantenbein ist Senior Principal Consultant für Cybersecurity in Microsofts Industry Solutions Delivery. Mit mehr als 30 Jahren Erfahrung in der Softwareentwicklung und in der Beratung bringt er eine Fülle von praktischem Know-how in seine Rolle ein. Heinrich Gantenbein ist spezialisiert auf Azure-Sicherheit, Bedrohungsmodellierung und DevSecOps.

Simone Curzi ist Principal Consultant von Microsofts Industry Solutions Delivery. Er verfügt über mehr als 20 Jahre Erfahrung in verschiedenen technischen Rollen bei Microsoft und hat sich seit mehr als 10 Jahren ganz dem Thema Sicherheit gewidmet. Als anerkannter Experte für Bedrohungsmodellierung und Microsoft Security Development Lifecycle-Experte, ist Simone Curzi ein regelmäßiger Redner auf internationalen Konferenzen wie Microsoft Ready, Microsoft Spark, (ISC)2 Security Congress, Carnegie Mellon's SEI DevOps Days und Security Kompass Equilibrium. Simone Curzi ist außerdem Autor eines Open-Source-Tools zur Bedrohungsmodellierung, Threats Manager Studio.

Microsoft Azure Security

Bewährte Methoden, Prozesse und Grundprinzipien für das Entwerfen und Entwickeln sicherer Anwendungen in der Cloud

Michael Howard
Heinrich Gantenbein
Simone Curzi

dpunkt.verlag

Microsoft

Michael Howard
Heinrich Gantenbein
Simone Curzi

Übersetzung: Rainer G. Haselier
Lektorat: Sandra Bollenbacher
Lektoratsassistenz: Friederike Demmig
Copy-Editing: Petra Heubach-Erdmann, Düsseldorf
Satz: Gerhard Alfes, mediaService, Siegen, *www.mediaservice.tv*
Herstellung: Stefanie Weidner
Umschlaggestaltung: Eva Hepper, Silke Braun
Druck und Bindung: mediaprint solutions GmbH, 33100 Paderborn

ISBN:
Print 978-3-86490-985-6
PDF 978-3-98890-088-3
ePub 978-3-98890-089-0
mobi 978-3-98890-090-6

1. Auflage 2024

Wieblinger Weg 17
69123 Heidelberg

ISBN of the English language edition: 978-0-13-790875-2
German language edition published by DPUNKT.VERLAG GMBH, Copyright © 2024

Hinweis:
Dieses Buch wurde mit mineralölfreien Farben auf PEFC-zertifiziertem Papier aus nachhaltiger Waldwirtschaft gedruckt. Der Umwelt zuliebe verzichten wir zusätzlich auf die Einschweißfolie. Hergestellt in Deutschland.

Schreiben Sie uns:
Falls Sie Anregungen, Wünsche und Kommentare haben, lassen Sie es uns wissen: *hallo@dpunkt.de*

5 4 3 2 1 0

Inhaltsverzeichnis

Kapitel 2

Kapitel 3

Kapitel 4

Kapitel 5

Kapitel 6

Kapitel 7

Kapitel 8

Teil 2: Sichere Implementierung

Kapitel 9

Kapitel 10

Kapitel 12

Kapitel 13

Für meine Familie, die Geduld mit mir hatte, als ich ein weiteres Buch schrieb.
– Michael

Mit Liebe zu meiner Frau Denyse, um ihr für ihre Unterstützung zu danken.
– Heinrich

Mit Dankbarkeit und Liebe zu meiner Familie, Silvia, Alice und Viola, die mich unermüdlich erdulden und unterstützen.
– Simone

Danksagungen

Dieses Buch behandelt eine Vielzahl komplexer sicherheitsrelevanter Themen. Beim Schreiben eines Buches müssen wir als Autoren sicherstellen, dass unsere Fakten stimmen und unsere Anleitungen korrekt sind. Dies können wir nur erreichen, indem wir Personen in den Azure-Produktgruppen befragen und indem wir Personen, die Experten auf ihrem jeweiligen Gebiet sind, um Unterstützung bitten. Wir möchten uns daher an dieser Stelle für die Hilfe und Unterstützung der folgenden Personen bei Microsoft bedanken:

Amar Gowda, Amaury Chamayou, Anthony Nevico, Antoine Delignat-Lavaud, Barry Dorrans, Ben Co, Ben Hanson, Ben Oberhaus, Bhuvaneshwari Krishnamurthi, Dan Simon, David Nunez Tejerina, Dhruv Iyer, Eric Beauchesne, Eustace Asanghanwa, Hannah Hayward, Jack Richins, Jakub Szymaszek, Jenny Hunter, Joachim Hammer, Jon Lange, John Lambert, Josh Brown-White, Ken St. Cyr, Kozeta Garrett, Luciano Raso, Mark Simos, Michael McReynolds, Michael Withrow, Mirek Sztajno, Nicholas Kondamudi, Niels Ferguson, Panagiotis Antonopoulos, Pieter Vanhove, Prasad Nelabhotla, Rafael Pazos Rodriguez, Robert Jarret, Rohit Nayak, Run Cai, Sameer Verkhedkar, Shubhra Sinha, Srđan Božović, Steven Gott, Sylvan Clebsch, Taylor Bianchi, Thomas Weiss, Vikas Bhatia und Yuri Diogenes.

Andere Microsoft-Kollegen waren in beratender Funktion tätig und halfen stundenlang bei einigen der komplexeren Teile des Buches. Es sind:

Kyle Marsh, Mark Morowczynski und Bailey Bercik (für Identität); Dave Thaler, Graham Berry und Vikas Bhatia (für Confidential Computing); und Hervey Wilson (für Schlüsseltresor)

Wir haben auch Feedback von Personen außerhalb von Microsoft erhalten, die über spezielles Fachwissen verfügen:

Arun Prabhakar (Boston Consulting Group), Avi Douglen (Bounce Security), Brook S. E. Schoenfield (True Positives, LLC), Dave Kaplan (AMD), David Litchfield (Apple), Donna McCally (HITRUST), Izar Tarandach (Squarespace), Lotfi Ben Othmane (Iowa State University), Mark Bode (AMD), Mark Cox (RedHat), Mark Curphey (Crash Override), Matthew Coles (Dell Technologies), Michael F. Angelo (Micro Focus), Mike Dietz und Robert Seacord (Woven Planet) sowie Shane Gashette und Steve Christey Coley (MITRE)

Wir möchten uns bei unseren technischen Reviewern bedanken, die jeden Aspekt unserer Entwürfe unter die Lupe genommen haben. Die technischen Reviewer waren:

Altaz Valani (Security Compass), Hasan Yasar (Software Engineering Institute, Carnegie Mellon), Jonathan Davis (Microsoft), Mike Becker (Microsoft) und Rick Alba (Microsoft)

Dieses Buch wäre ohne die Mitarbeiter von Pearson/Microsoft Press nicht möglich gewesen: Loretta Yates, die »ja« gesagt hat; Charvi Arora, die uns dazu gebracht hat, unsere Termine einzuhalten; und schließlich Kate Shoup, die unseren Text hervorragend redigiert hat und dabei unseren Ton und unsere inhaltlichen Intentionen beibehalten hat.

Schließlich möchten wir Scott Guthrie für das Schreiben des Vorworts und für die Leitung des großartigen Teams von Microsoft Azure danken.

Michael Howard
Austin, Texas

Heinrich Gantenbein
St. Paul, Minnesota

Simone Curzi
Perugia, Italien

Vorwort

Im vergangenen Jahrzehnt haben wir einen dramatischen Wandel in der Art und Weise erlebt, wie Unternehmen Technologie nutzen, um ihre Geschäftsabläufe völlig neu zu erfinden und zu verändern. Die jüngsten globalen Herausforderungen und unvorhersehbare, weitreichende Ereignisse haben diesen Wandel nur noch beschleunigt, und die Unternehmen mussten sich neu orientieren und anpassen, um die Bedürfnisse ihrer Kunden und Mitarbeiter zu erfüllen und die Widerstandsfähigkeit ihres Unternehmens sicherzustellen.

Diese digitale Transformation wurde zum Teil durch technologische Fortschritte und Hyperscale-Cloudanbieter wie Microsoft Azure ermöglicht, die Unternehmen die nötige Flexibilität bieten, um neue Effizienzen und Fähigkeiten zu realisieren. In dieser Ära der beispiellosen Transformation, einschließlich der Migration in die Cloud, gibt es jedoch auch neue Bedrohungen und Anforderungen an die Sicherheit und den Datenschutz.

Wenn Menschen an Sicherheit denken, denken sie oft an Endpunktschutz, Firewalls und Anti-Malware-Tools, die von entscheidender Bedeutung sind; aber Architekten und Entwickler können die Anwendungssicherheit während des Designs und der Entwicklung nicht ignorieren. Dieses Buch, »Microsoft Azure Security«, ist eine unverzichtbare Ressource für das Verständnis der wesentlichen Elemente eines durchgängig sicheren Softwaredesigns und der Entwicklung auf Azure. Es behandelt zwei Bereiche, die mir sehr am Herzen liegen – die Sicherheit von Azure und die Softwareentwicklung.

Die Microsoft Cloud bietet viele Zuverlässigkeits- und Sicherheitsvorteile im Vergleich zu On-Premises-Lösungen, aber Architekten und Entwickler können grundlegende Sicherheitspraktiken nicht ignorieren, wenn sie auf Azure bereitstellen. Cloudbasierte Lösungen verwenden ein Modell der geteilten Verantwortung, und ein Teil der Sicherheitsverantwortung liegt sowohl beim Mandanten als auch beim Cloudanbieter. »Microsoft Azure Security« bietet eine ganzheitliche und leicht zugängliche Ressource für jeden, der sichere Workloads auf Azure entwickelt. Leser, die an Azure-Lösungen arbeiten, erhalten ein zeitgemäßes Verständnis für sichere Entwicklung, Design und Implementierung.

Die Autoren Michael, Heinrich und Simone verfügen zusammen über jahrzehntelange Erfahrung im Bereich der Anwendungssicherheit. Sie haben mit öffentlichen Auftraggebern und Unternehmen – großen und kleinen – zusammengearbeitet, die alle in der Lage waren, sichere Lösungen auf Azure zu entwerfen, zu entwickeln, bereitzustellen und zu verwalten. Ich weiß, dass die Autoren sich dafür einsetzen, allen, die auf Azure entwerfen und entwickeln, dabei zu helfen, die Zuverlässigkeit, Skalierbarkeit und Sicherheit zu erreichen, die von ihren Organisationen und Endbenutzern gefordert werden.

Dieses Buch ist ein unverzichtbarer Leitfaden für jeden Architekten und Entwickler, der sichere, geschäftskritische Lösungen auf Azure bereitstellt.

Scott Guthrie
Executive Vice President
Cloud + AI-Gruppe, Microsoft

Einleitung

Mitte 2021, während einer Aufzeichnung des Azure-Security-Podcasts, wurde Michael Howard vom Azure-Sicherheitsexperten und Autor Yuri Diogenes gefragt, ob er plane, ein Update zu seinem Buch »The Security Development Lifecycle« zu schreiben. Ohne zu zögern, antwortete Michael: »Nein!«

Doch damit war die Angelegenheit noch nicht erledigt.

Die Frage, die Yuri Diogenes stellte, legte den Grundstein. In den nächsten Wochen schmiedeten wir drei – Michael, Heinrich und Simone – einen Plan, um dieses Buch zu schreiben. Gemeinsam haben wir mit Hunderten von Kunden zusammengearbeitet, um ihnen dabei zu helfen, geschäftskritische Lösungen auf Azure zuverlässig einzusetzen. Dieses Buch ist die Krönung dieser praktischen Erfahrung.

Wir haben dieses Buch nicht nur geschrieben, um Ihnen zu zeigen, wie Sie sichere Lösungen auf Azure entwerfen und entwickeln können, sondern auch, um Ihnen pragmatische Ratschläge zu geben. Das in Abbildung E-1 dargestellte Venn-Diagramm zeigt, wie wir dieses Buch sehen.

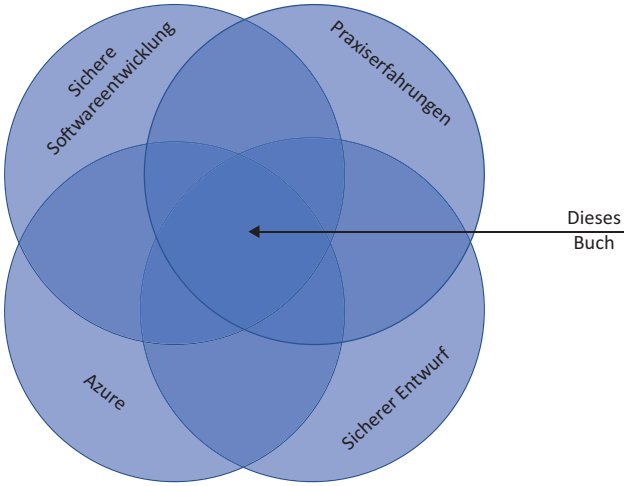

Abbildung E-1 Der Schnittpunkt der in diesem Buch behandelten Themenbereiche

Einige Bereiche von Azure werden nicht behandelt, da dieses Buch sonst zu einem ziemlichen Wälzer werden würde. Insbesondere behandeln wir keine Themen wie die folgenden:

- **Privileged Access Workstation (PAW)** Eine Arbeitsstation mit privilegiertem Zugriff ist eine Arbeitsstation, die nur für administrative Aufgaben vorgesehen ist. Sie hat keinen Zugriff auf E-Mail, allgemeines Web-Browsing und andere Produktivitätsaufgaben. PAWs

werden von Konten mit erhöhten Berechtigungen verwendet, um Aktionen in Umgebungen mit hohem Risiko durchzuführen, zum Beispiel in der Produktion, bei der Kontenverwaltung und in anderen Bereichen. Mehr über PAWs erfahren Sie hier: *https://learn.microsoft.com/azure-stack/ruggedized/customer-replaceable-unit/privileged-access-workstation*.

- **Bedingter Zugriff und Multi-Faktor-Authentifizierung (MFA)** Diese Aufgaben werden häufig von einem Identitätsteam übernommen, und die Infrastruktur sollte bereits vorhanden sein. Bedingter Zugriff und MFA sind jedoch entscheidend für die Sicherheit einer Azure-basierten Lösung. Hier erfahren Sie mehr zu diesem Thema: *https://learn.microsoft.com/azure/active-directory/conditional-access/overview*.
- **Datenschutz** Dies ist ein Buch über Sicherheit. Obwohl sich Sicherheit und Datenschutz überschneiden, geht es bei der Sicherheit hauptsächlich darum, ein System und seine Daten gegen unbefugte Nutzung zu schützen, während es beim Datenschutz um den Umgang mit personenbezogenen Daten geht. Man kann Sicherheit ohne Datenschutz haben, aber es gibt keinen Datenschutz ohne Sicherheit.

Wir haben uns relativ kurzgefasst, indem wir viele Links zu externen Informationen eingefügt haben, anstatt einige Themen in diesem Buch ausführlich zu behandeln.

Wie ist dieses Buch organisiert?

Dieses Buch ist nicht dazu gedacht, von vorne bis hinten gelesen zu werden. Das können Sie natürlich tun, aber wir haben versucht, die Kapitel so unabhängig wie möglich zu gestalten, damit sie auch einzeln gelesen werden können. Dennoch gibt es Querverweise zwischen den Kapiteln, und es kann sein, dass Sie manchmal einen Abschnitt eines anderen Kapitels lesen müssen, um das Gesamtbild zu verstehen.

Außerdem werden im Buch mehrere Möglichkeiten zur Erledigung einer Aufgabe beschrieben, wie die folgenden:

- Verwendung des Azure-Portals (obwohl es nicht üblich ist, das Azure-Portal in Produktionssystemen zu verwenden, da die Bereitstellung in der realen Welt in der Regel eine Pipeline nutzt, um Ressourcen zu pushen)
- Verwendung der Azure-Befehlszeilenschnittstelle (Command-line Interface, CLI)
- Verwendung von PowerShell-Code
- Verwendung vollständigerer Code-Beispiele in verschiedenen Sprachen wie C#, Python, JavaScript und anderen

Wir haben Codebeispiele und Schnipsel in unser GitHub-Repository unter *https://github.com/AzureDevSecurityBook* hochgeladen, besuchen Sie es also regelmäßig. Lokalisierte Versionen der Begleitdateien sind auf der Produktseite des Buches verfügbar: *https://dpunkt.de/produkt/microsoft-azure-security*

Wer sollte dieses Buch lesen?

Für wen ist dieses Buch gedacht? Es richtet sich an alle, die Lösungen auf Azure bereitstellen – seien es Architekten, Entwickler oder Tester –, die vielleicht nicht viel über Sicherheit wissen, aber möchten, dass ihr Design und ihr Code so sicher wie möglich sind. Wir decken in diesem Buch viel ab, aber wir gehen auch auf viele komplexe Themen ein.

Ein letzter Punkt: Wenn Sie das NIST Cybersecurity Framework (NIST CSF) verwenden, dann sind Sie mit dessen Kernkomponenten vertraut: Identifizieren, Schützen, Erkennen, Reagieren und Wiederherstellen (identify, protect, detect, respond, recover). Das Material in diesem Buch konzentriert sich hauptsächlich auf die Komponente Schützen und einige Aspekte der Komponente Erkennen. Wenn Sie für den produktiven Einsatz gedachte Lösungen auf Azure einführen möchten, muss Ihr Unternehmen die anderen vier Komponenten des NIST CSF abdecken. Weitere Informationen über das NIST CSF finden Sie in Kapitel 8, »Compliance und Risikomanagement«, und auf der NIST-Website unter *https://azsec.tech/81t*.

Vielen Dank fürs Lesen!

Konventionen und Features in diesem Buch

In diesem Buch werden die Informationen unter Verwendung von Konventionen dargestellt, die die Informationen lesbar und leicht nachvollziehbar machen sollen:

- Umrandete Elemente mit Beschriftungen wie *Hinweis* liefern zusätzliche Informationen.
- Text, den Sie eingeben (außer Codeblöcken), erscheint fett.
- Ein Pluszeichen (+) zwischen zwei Tastennamen bedeutet, dass Sie diese Tasten gleichzeitig drücken müssen. Zum Beispiel bedeutet: »Drücken Sie Alt + ⇆ «, dass Sie die Alt -Taste gedrückt halten, während Sie die ⇆ -Taste drücken.
- Ein vertikaler Balken zwischen zwei oder mehr Menüpunkten (z. B. *Datei* | *Schließen*) bedeutet, dass Sie das erste Menü oder den ersten Menüpunkt auswählen, dann das nächste und so weiter.

Systemvoraussetzungen

Die Beispiele und Szenarien in diesem Buch erfordern den Zugang zu einem Microsoft Azure-Abonnement und einen Computer, der eine Verbindung zu Azure herstellen kann. Weitere Informationen über ein Probeabonnement finden Sie auf dieser Website:

azure.microsoft.com/free

GitHub-Repo

Das GitHub-Repository des Buches enthält den englischsprachigen Beispielcode und Codeschnipsel; die Autoren werden es im Laufe der Zeit aktualisieren. Das Repository lautet *github.com/AzureDevSecurityBook*.

Lokalisierte Versionen der Begleitdateien sind auf der Produktseite des Buches verfügbar:
https://dpunkt.de/produkt/microsoft-azure-security

Errata und Support

Wir haben alle Anstrengungen unternommen, um die Richtigkeit dieses Buches und der dazugehörigen Inhalte zu gewährleisten. Sie können auf Aktualisierungen dieses Buches – in Form einer Liste der eingereichten Errata und der damit verbundenen Korrekturen – unter folgender Adresse zugreifen:

MicrosoftPressStore.com/SecureAzureSolutions/errata

Wenn Sie einen Fehler entdecken, der noch nicht aufgeführt ist, teilen Sie uns diesen bitte auf der gleichen Seite mit.

Weitere Unterstützung und Informationen zu Büchern finden Sie unter
MicrosoftPressStore.com/Support.

Mit Anmerkungen, Fragen oder Verbesserungsvorschlägen auf Deutsch zu diesem Buch können Sie sich auch an den dpunkt.verlag wenden:

hallo@dpunkt.de

Bitte beachten Sie, dass der Produktsupport für Microsoft-Software und -Hardware nicht über die oben genannten Adressen angeboten wird. Hilfe zu Microsoft-Software oder -Hardware finden Sie unter *support.microsoft.com*.

TEIL 1

Sicherheitsgrundsätze

KAPITEL 1

Prozesse für sichere Entwicklungszyklen

Am Ende dieses Kapitels

- verstehen Sie einige der Prozesse, die für die Entwicklung sicherer Software erforderlich sind.
- können Sie innerhalb Ihrer Organisation dazu beitragen, eine Sicherheitskultur zu entwickeln.
- sind Sie in der Lage, den Zweck der verschiedenen Arten von Umgebungen für die Entwicklung bis hin zur Produktion zu erläutern und welche unterschiedlichen Sicherheitsmaßnahmen sie erfordern.

Entwickler sind die Hauptursache für Kompromittierungen

Die Hauptursache für Gefährdungen sind nicht Hacker, Angreifer oder andere ruchlose Akteure. Vielmehr sind wir, die Softwareentwickler, die Hauptursache für Sicherheitslücken. Laut einer Analyse von Contrast Security aus dem Jahr 2020 sind fast 50 Prozent aller Kompromittierungen auf Schwachstellen in Anwendungen zurückzuführen – Schwachstellen, die letztlich von Softwareentwicklern geschaffen wurden. Eine Zusammenfassung des Berichts können Sie hier lesen: *https://azsec.tech/lvz*.

Als Softwareentwickler können wir nicht viel gegen Angriffe tun. Sie werden auf jeden Fall stattfinden. Was wir aber tun können, ist, die Sicherheit unseres Codes zu verbessern. Wir kommen nicht darum herum, dass das Design unseres Systems und die Qualität unseres Codes den Unterschied zwischen einem fehlgeschlagenen und einem erfolgreichen Angriff ausmachen können. Wir müssen die Art und Weise ändern, wie wir Software entwerfen und entwickeln, um die Sicherheit so nahtlos wie möglich und mit so wenig Reibungsverlusten wie möglich zu verbessern. Das entscheidende Wort hier ist *Reibung*. Sicherheit wird oft als eine Art Steuer angesehen, die Entwickler zahlen müssen und die die Entwicklung behindert. Sie steht einfach im Weg. Wir müssen Prozesse und Aufgaben einbeziehen, die die Sicherheit erhöhen, die so reibungslos wie möglich sind und einfach als ein weiterer wichtiger Aspekt bei der Erledigung der Aufgabe angesehen werden.

Natürlich sind auch Werkzeuge wichtig. Aber sie sollten nicht blindlings eingesetzt oder als einzige Quelle für die Sicherheit Ihrer Lösung betrachtet werden. Ganz gleich, wie viele Tools oder Automatisierungsfunktionen Sie bei Ihren Entwicklungsverfahren einsetzen, letztendlich sind es Menschen, die Software erstellen, und auch Ihre Sicherheitslage hängt von Menschen

ab. Wie das Sprichwort sagt: »Ein Dummkopf mit einem Werkzeug ist immer noch ein Dummkopf.« Wir müssen also nicht nur in die neuesten Sicherheitstools investieren, sondern auch in menschliches Sicherheitskapital und Prozesse.

Dieses Kapitel beschäftigt sich sowohl mit dem Prozess und den menschlichen Aspekten der Praktiken für die Softwareentwicklung; aber auch die technischen Aspekte kommen nicht zu kurz. Das Ziel ist, wie erwähnt, bei der Bereitstellung von sicheren Softwarelösungen so reibungslos zu sein wie möglich.

Einführung in den Microsoft Security Development Lifecycle

Der Microsoft Security Development Lifecycle (SDL) entstand Anfang der 2000er-Jahre und wurde im Laufe der Jahre angewandt und angepasst. Ein Sprichwort sagt: »Es gibt nichts Neues unter der Sonne«, und das trifft auf den SDL zu. Der SDL unterscheidet sich jedoch durch die Menge an unterstützender Dokumentation, Werkzeugen, Forschungsergebnissen und Vordenkern, die Microsoft öffentlich zugänglich gemacht hat.

Was also ist der SDL? Der SDL besteht aus einer Reihe von Praktiken zur Verbesserung der Softwaresicherheit. Er verfolgt zwei übergreifende Ziele:

- Verringerung der Anzahl von Sicherheitslücken in Ihrem Code
- Verringerung der Schwere der Schwachstellen, die Sie übersehen

Diese beiden Ziele führen zu einer gewissen Spannung in Ihrer Sicherheitsstrategie. Sie möchten die sicherste Software entwickeln, müssen aber gleichzeitig damit rechnen, dass Ihnen etwas entgeht und dass sich die Strategien der Angreifer mit der Zeit weiterentwickeln. Was heute sicher und korrekt ist, kann morgen angreifbar sein.

Um Ihr Wissen zu vervollständigen, empfehlen wir Ihnen, die vollständige Liste der SDL-Anforderungen zu lesen, die Sie hier finden:
***https://www.microsoft.com/securityengineering/sdl/practices*.**

Qualität ≠ Sicherheit

Wir hören oft, dass Leute sagen: »Wenn man die Qualität verbessert, dann verbessert sich auch die Sicherheit.« Diese Aussage klingt zwar plausibel, aber es gibt keine Beweise, die diese Aussage stützen. Keine. Softwarequalitätsprogramme finden nur selten Sicherheitsprobleme, denn Sicherheitsprobleme sind etwas anderes als Qualitätsprobleme. Außerdem wird Sicherheit, wie wir später in diesem Buch erörtern, oft als »zusätzliche« Funktionalität definiert.

Angenommen, Sie haben eine Anwendung, die lediglich die folgenden Datenbankoperationen durchführt:

- Hinzufügen eines neuen Benutzers (Erstellen/*Create* in CRUD)
- Lesen der Details eines Benutzers (Lesen/*Read* in CRUD)
- Bearbeitung der Benutzerdaten (Aktualisierung/*Update* in CRUD)
- Löschen eines Benutzers (Löschen/*Delete* in CRUD)
- Drucken der Benutzerdaten

Sie erstellen einige Tests, die erfolgreich oder (absichtlich!) fehlschlagen sollen, und überprüfen dann diese Erfolge und Fehlschläge. Wenn alle Tests, die den Erfolg überprüfen sollen, erfolgreich sind und alle Tests, die das Scheitern überprüfen sollen, pflichtgemäß scheitern, könnte man zu dem Schluss kommen, dass die Anwendung keine Fehler aufweist. Dies ist jedoch nicht der Fall. Wenn die Anwendung beispielsweise eine SQL-Injection-Schwachstelle aufweist, die es einem Tester (oder Angreifer) ermöglicht, alle Benutzer zu lesen oder eine Datenbanktabelle zu löschen, wird die Anwendung dennoch alle Erfolgstests bestehen und alle Fehlertests nicht bestehen. Die Moral von der Geschichte ist, dass Sie Ihren Softwareentwicklungsprozessen Sicherheit als eigenen Faktor hinzufügen *müssen*.

Sicherungsmerkmale vs. Sicherheitsmerkmale

Das Microsoft SDL konzentriert sich auf die Sicherung Ihrer Software, nicht nur auf das Hinzufügen weiterer Sicherheitsfunktionen. Sicherheitsfunktionen sind wichtig, aber Sie können nicht einfach jedes beliebige Sicherheitsprodukt in Ihre Lösung einbauen und sie als sicher bezeichnen. Die Funktionen, die Sie Ihren Lösungen hinzufügen, müssen ebenfalls sicher sein. Diese philosophische Perspektive stellt einen wichtigen Sinneswandel für viele dar, die glauben, sie könnten ein Produkt kaufen und es als erledigt betrachten – vor allem, wenn so viele Unternehmen ihre Produkte als Allheilmittel verkaufen.

Kurz, *Sie* müssen sich auf die Disziplin der Softwareentwicklungssicherheit konzentrieren. Sie können diese Verantwortung nicht auf ein Produkt abwälzen.

SDL-Komponenten

Die wichtigsten Aufgaben und Anforderungen von Microsoft SDL sind wie folgt:

- Sicherheitsschulung
- Definieren Ihrer Bug Bar
- Analyse der Angriffsfläche
- Modellierung von Bedrohungen
- Definieren Ihrer Toolchain
- Verbotene Funktionalität vermeiden

- Werkzeuge zur statischen Analyse verwenden
- Dynamische Analysetools verwenden
- Review des Sicherheitscodes
- Reaktionsplan bei Zwischenfällen zur Hand haben
- Penetrationstests durchführen

Schauen wir die einzelnen Punkte genauer an.

Agile SDL

Sie werden vielleicht denken, dass diese Liste mit Anforderungen vor allem für ein Wasserfallmodell gilt. Tatsächlich handelt es sich aber nur um eine Liste von Aufgaben, die erledigt werden müssen; sie gibt nicht an, wann Sie sie erledigen müssen. Es kann sein, dass Sie nur einen Sprint damit verbringen, an einigen Aufgaben zu arbeiten, und diese Arbeit wird dann für alle zukünftigen Sprints gelten. Wir werden erläutern, wie man jede dieser Aufgaben in einer agilen Umgebung am besten anwendet.

Sicherheitsschulung

Sicherheitsschulungen sind ein Muss. Aber mit Sicherheitsschulung meinen wir nicht die Schulung »Passwörter nicht wiederverwenden!«, sondern die Schulung »Sicherheit bei der Anwendungsentwicklung«.

Lange Zeit verlangte Microsoft von allen *technischen* Mitarbeitern, dass sie an allgemeinen Sicherheitsschulungen teilnehmen, und die Mitarbeiter können dies auch heute noch tun, wenn sie wollen. Heutzutage verwenden wir jedoch ein schlankeres Schulungsmodell, das eher bereichsspezifisch ist. So verlangen wir zum Beispiel von unseren technischen Mitarbeitern, dass sie sicherheitsrelevante Kurse besuchen, die sich auf ihre Rolle beziehen, anstatt eine allgemeine Schulung zu absolvieren.

Wenn ein Entwickler beispielsweise serverseitigen Node.js-JavaScript-Code für eine Webanwendung schreibt, muss er Cross-Site-Scripting-Probleme (XSS), die sichere Verwendung von Cookies und andere Web- und HTTP-bezogene Sicherheitsprobleme und Abwehrmaßnahmen verstehen. Wenn dieselbe Anwendung mit einer SQL-Datenbank kommuniziert, sind solide Kenntnisse über SQL Injection, Datenbankverbindungen mit geringsten Rechten und das sichere Speichern von Verbindungszeichenfolgen ebenfalls wichtig. Es ist jedoch sehr wahrscheinlich, dass dieser Entwickler die potenziellen Probleme der Speicherbeschädigung bei der Verwendung von `strcpy()` in C nicht verstehen muss. Er könnte also einfach lesen, ein Video ansehen oder eine Onlineschulung zu den für ihn wichtigen Themen absolvieren.

Es gibt mehrere Schulungsmodi, daher sollten Sie den Modus wählen, der für Sie als Entwickler, Architekt oder Tester am besten geeignet ist. Unterschätzen Sie jedoch nie den Wert eines kurzen Videos, das den Kern des Problems auf den Punkt bringt!