Maiborn Wolff Memus IT
Sonderdruck aus OBJEKTspektrum

Ausgabe 03|2021
Deutschland € 12,90 Österreich € 13,90 Schweiz sfr 22,20

4 194164 112900 03

OBJEKTspektrum

Software-Architektur und IT für Profis

Security – Wie sich mit immer größerer Komplexität und Verantwortung umgehen lässt

Interview mit Markus Völter

Der Experte und Podcaster erklärt die Vorzüge von Domain-Specific Languages, die Technik und Fachlichkeit vereinen

- Softwarearchitektur – Mehr Wert und Innovation schaffen
- Schulungsumgebung in Docker und Kubernetes bereitstellen
- Flexible Anwendungsarchitektur – Rethinking Pizza Factory

www.OBJEKTspektrum.de

Security – Wie sich mit immer größerer Komplexität und Verantwortung umgehen lässt

G 6540F

Cloud-Security

Geteilte Verantwortung in der Cloud

Wenn ein Unternehmen ein eigenes Rechenzentrum betreibt, ist die Verantwortung klar geregelt: Sie liegt allein bei den eigenen Teams. Sie sind für die Sicherheit der Server und der darauf gespeicherten Daten verantwortlich. In der Cloud gehört das Rechenzentrum jedoch einer anderen Partei. Damit stellt sich die Frage der Verantwortung neu und anders.

Die Frage der Verantwortlichkeiten scheint oberflächlich einfach: Der Cloud-Provider verantwortet die Hardware. In der Praxis wird es schnell komplexer, denn durch Infrastructure as a Service, Platform as a Service und Software as a Service sind Soft- und Hardwaregrenzen unscharf. Dazu kommt, dass die Cloud-Provider ihre Zertifizierungen und organisatorischen wie technischen Security- und Datenschutz-Maßnahmen herausstellen. Hier könnte der Eindruck einstehen, dass eine Cloud automatisch viel sicherer ist, schließlich hat das eigene Rechenzentrum bei Weitem nicht so viele Prüfungen erfolgreich durchlaufen. Für das Rechenzentrum des Cloud-Providers selbst stimmt diese Annahme sicherlich auch. Doch wo beginnt die eigene Verantwortung als Application Owner?
Dies drückt sich in einer einfachen Formulierung (**siehe Abbildung 1**) aus:

- Geht es um die Security *der* Cloud oder
- um Security *in der* Cloud?

Die Zertifizierungen der Cloud-Provider zahlen in der Regel auf die Security *der* Cloud ein. Die Security *in der* Cloud ist dagegen eine geteilte Verantwortung. Und das wichtigste: Die beiden Verantwortungsbereiche müssen so zusammenpassen, dass keine Lücken entstehen. Wenn wir über Cloud und Security sprechen, müssen wir also sehr genau hinhören, welche der beiden Dimensionen gemeint ist.

Security *der* Cloud oder Security *in der* Cloud

Zuallererst ist die Frage der geteilten Verantwortung eine vertragliche. Welche Zusicherungen macht der Cloud-Provider und welche Pflichten hat der Kunde? Hier liegt durchaus eine Gefahr: Das Vertragliche liegt selten bei den Entwickler-Teams, sondern meistens bei Management und Einkauf. So kann es passieren, dass diejenigen, die die Anwendung implementieren und betreiben, nicht alle Details oder Auswirkungen von Regelungen im Vertrag kennen. Sie brauchen dieses Wissen aber, um ihre Verantwortlichkeit wahrzunehmen und die eigenen Maßnahmen lückenlos an die des Cloud-Providers anzuschließen.
Und dieser Punkt ist wesentlich: Aus Perspektive eines Hackers reicht eine Lücke im Gesamtsystem für einen erfolgreichen Angriff, unabhängig davon, ob die Zuständigkeit in die Verantwortung des Cloud-Providers oder des Kunden oder irgendwo dazwischen fällt.

"Shared Responsibility"-Modelle

Die Cloud-Anbieter verwenden sogenannte „Shared Responsibility"-Modelle, die definieren, wer was in Bezug auf die Cybersicherheit tut beziehungsweise tun muss. In der Regel kümmern sich die Anbieter um die Sicherheit der zugrunde liegenden Cloud-Infrastruktur, einschließ-

lich Hardware, Software, Netzwerke und physische Anlagen wie Gebäude und Stromversorgung sowie Anbindung.
Die Hoheit für die physikalischen Aspekte der Absicherung beim Cloud-Provider ist schon deswegen erforderlich, da der Kunde überhaupt keine Kontrolle oder Zugang zu diesem Bereich hat. Dies umfasst die Hardware, das Host-Betriebssystem und die physische Sicherheit der Gebäude und Infrastruktur. Durch die Verlagerung in die Cloud werden viele dieser logistischen Herausforderungen dem Kunden abgenommen. Diese Aufgabe dürften die meisten Cloud-Provider auch mit Bravour leisten, ist dies schließlich eines ihrer wichtigsten Assets; jeder Vorfall wäre ein immenser Image-Schaden.
Die Anwender müssen die Assets innerhalb der Cloud-Anwendung sichern und sich um Dinge wie Datenverschlüsselung, Identitäts- und Zugriffsmanagement und allgemeine Anwendungssicherheit kümmern. Ebenso müssen sie alle Dienste im Blick haben, die über APIs integriert werden.
Einfach ausgedrückt: Der Cloud-Anbieter sichert die „Commodity“ ab, die den Betrieb des Rechenzentrums sicherstellt, und bietet XaaS-Funktionen. Der Cloud-Kunde sichert das eigene System und ist verantwortlich, wie die Cloud-Dienste genutzt werden.
Was einfach klingt, wird in der Realität oft zur Herausforderung. Als wäre das Thema Cyber-Security nicht schon groß und komplex genug; hinzu kommt, dass die verschiedenen Cloud-Provider die Verantwortlichkeiten unterschiedlich regeln. Die Grenze zwischen der Verantwortung des Cloud-Providers und der des Kunden verläuft dabei nicht geradlinig, sondern verschiebt sich je nach Cloud-Provider. Dann braucht es einen Blick ins Kleingedruckte. Durch Nichtwissen oder Unklarheiten bei der Auslegung können Lücken entstehen, wo die Verantwortung des Anbieters endet und die des Benutzers beginnt – Lücken, die immer noch groß genug sind, damit Hacker folgenschwere Angriffe direkt in und mit der Cloud ausführen können (**siehe Abbildung 2**).
Und auch wenn die Maßnahmen zwischen Applikation und Cloud-Provider abgestimmt sind, können Einfallstore zum Beispiel durch Flüchtigkeitsfehler entstehen – etwa ein Datenbankport, der ungeschützt im Netz steht, oder ein Login mit admin/admin.
Die Frage nach dem Was beantwortet nicht die Frage nach der Priorität: Grundlage für die Entscheidung, wie die Security-Maßnahmen priorisiert werden, ist in jedem Einzelfall eine Schutzbedarfs- und Bedrohungsanalyse. Aus Angreifer-Perspektive sind Information und Algorithmen des Kunden, also zum Beispiel personenbezogene Daten oder proprietäre Verfahren, das wertvollste Gut.

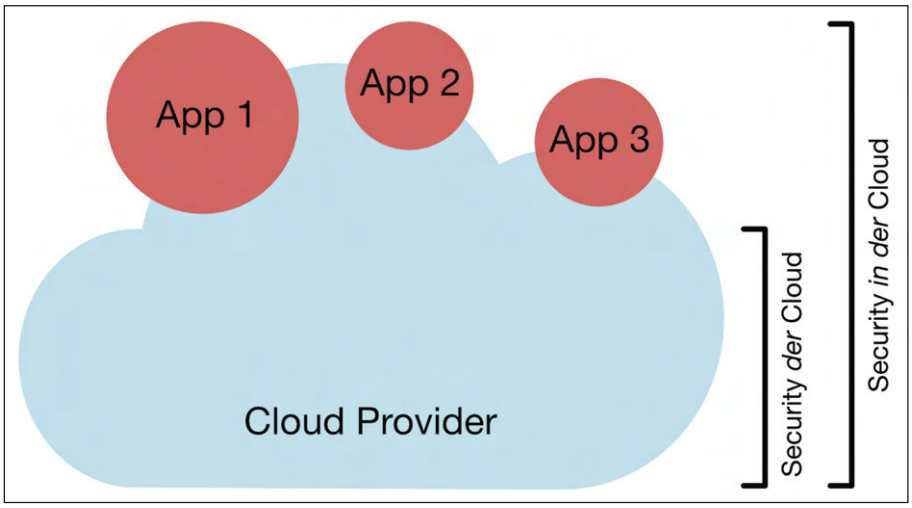

Abb. 1: Security der Cloud vs. Security in der Cloud

Verteilte Anwendungen, besondere Anforderungen an die Absicherung

Bei einem Sicherheitsvorfall denken die meisten wohl zuerst an einen *Datendiebstahl*. Natürlich gibt es je nach Setup sehr unterschiedliche Ursachen dafür, etwa veraltete Bibliotheken, übergreifende Zugriffe oder nicht abgesicherte Schnittstellen. Anders als bei der physikalischen Verkabelung zwischen Systemen im Rechenzentrum ist in der Cloud alles in Software definiert. So könnte ein Tippfehler in einem Infrastructure-as-Code-Quelltext dazu führen, dass plötzlich ein Backup-System frei im Internet zugänglich ist oder eine Verbindung zwischen einem ungesicherten Entwicklungsserver und der Produktivumgebung möglich wird. Leider liest man immer wieder, dass auf diese Weise Datenbanken und Dateiablagen ungesichert offen lagen, die ein wahrer Schatz für Hacker sind.
Auch harmlos aussehende *Workarounds* können zur Stolperfalle werden: Ich bin privat als Kunde eines Dienstes einmal Opfer geworden, als ein Entwicklungsserver automatisiert kurzfristig im Internet verfügbar gemacht wurde, damit ein Let's-Encrypt-Zertifikat ausgestellt werden konnte (für die HTTP-01 Challenge muss ein Token über http von Let's Encrypt validiert werden, bevor das Zertifikat generiert wird). Durch einen Fehler stoppte das Skript vorzeitig und der Ser-

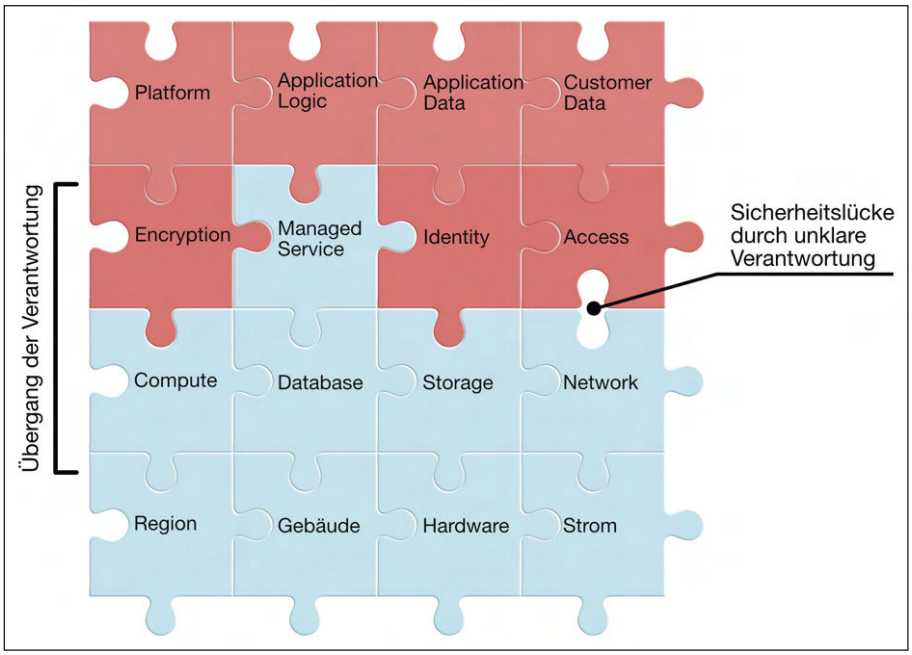

Abb. 2: Shared Responsibility: besonderer Fokus muss auf dem Übergang der Verantwortung liegen

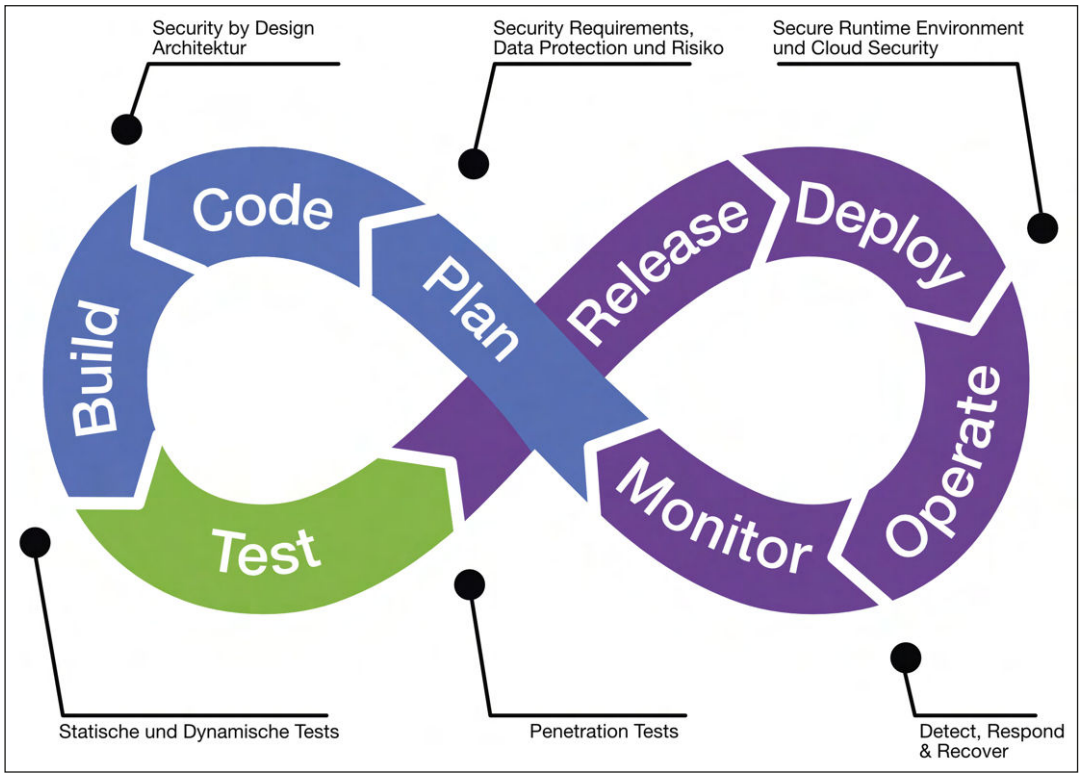

Abb. 3: Ganzheitliche Security wird in allen Teilen des Software Development Lifecycles berücksichtigt

ver blieb offen im Netz. Da in einem Zertifikat alle Domain-Namen der Virtual Hosts aufgelistet sind, konnte der Server mit dem richtigen Namen angesprochen werden und eine Lücke in einer ungepatchten Applikation ausgenutzt werden. Als der Hacker diese Hürde genommen hatte, hatte er freien Zugang ins interne Netz.

Eine weitere Gefahr ist der *Missbrauch von Ressourcen*, etwa um Spam zu versenden. Die Kunst bei Cloud-Anwendungen ist die Orchestrierung der Bausteine. Werden die Schnittstellen zwischen den Komponenten nicht sauber abgesichert, kann ein Angreifer sich zwischen zwei Komponenten – etwa Web-Client und API – einklinken und die Aufrufe für eigne Zwecke missbrauchen. So wird eine legitime Anwendung plötzlich zur Spam-Schleuder oder es werden aus Versehen gebührenpflichtige Services offengelegt, die massenhaft aufgerufen werden. Nicht selten nutzen Unternehmen zum Beispiel Dienste, die Adressen validieren, oder Machine-Learning-Funktionalität wie Bild- oder Spracherkennung abrufen. Auch diese Aufrufe müssen geschützt werden.

Gerade weil sich solche APIs automatisiert ansprechen lassen, können Hacker sie schnell durch entsprechende Skripte oder Scanner mit unzähligen Requests überfluten; entweder mit dem Ziel, den Service zu überlasten oder so viele Input-Parameter wie möglich auszuprobieren.

Good Practices schon bei den Grundlagen

Anwendungen in der Cloud sind so gut wie immer verteilte Anwendungen, denn nur so lassen sich die Vorteile der Cloud vollständig nutzen. Das bedeutet aber auch, dass es sehr viele Systemübergänge gibt, und nicht selten werden sowohl interne wie extern Ressourcen eingebunden. Die folgenden Grundlagen helfen, Security ganzheitlich zu betrachten. Damit werden die nachträglichen – und in der Regel besonders teuren – Fehlerbehebungen weniger.

Secure Software Development Lifecycle

Jede Umgebung kann nur so sicher sein, wie der implementierte Softwareentwicklungsprozess. Entwickler brauchen die Zuverlässigkeit, dass das Programmierte auch genau das ist, was deployt wird. Sonst ist es nachvollziehbarerweise kaum möglich, eine Anwendung abzusichern. Genau das leistet ein Secure Software Development Lifecycle (SSDLC). Statt sicherheitsrelevante Aktivitäten nur als Teil des Testens (und meist am Ende der Entwicklung) durchzuführen, integriert ein SSDLC Security in alle Phasen: In jeder Interaktion wird die Sicherheit sowohl bei Designüberlegungen als auch bei Architekturentscheidungen, beim Entwickeln und Test sowie beim Ausrollen und Betreiben berücksichtigt. Auf diese Weise wird Security ein integraler Bestandteil des Produkts und gemäß eines agilen Mindsets mit jeder Iteration verbessert. Ein Vorteil des SSDLC: Es entsteht ein hohes Maß an Transparenz und viel Dokumentation, welche Maßnahmen bereits zum Einsatz kommen und welche Aktivitäten zur Steigerung der Sicherheit durchgeführt werden. Dies erleichtert es Teams, auf die sich ändernden Anforderungen zu reagieren. Erkennt der Scanner in der Build-Pipeline eine Sicherheitslücke in einer benutzten Library, kann diese direkt mit dem nächsten Deployment aktualisiert werden.

Diese Flexibilität ist dringend notwendig, denn auch Hacker lernen dazu und nutzen immer bessere Methoden und Werkzeuge, um nach Schwachstellen zu scannen und diese mit heuristischen Herangehensweisen effizient auszunutzen. Selbst aus den einst simplen Brute-Force-Angriffen ist heute schon ein intelligentes Raten geworden. Ganz offen ausgesprochen: Wir haben es mit einem Gegner zu tun, der sich ständig anpasst und dazulernt. Also müssen wird das auch.

Ein SSDLC hat einen weiteren, ganz entscheidenden Vorteil: Er rückt das Thema Cyber-Security in den Mittelpunkt und macht es zur Aufgabe für das ganze Team. Wenn Sicherheit bereits im Requirements Engineering und im Design Eingang findet, in der Entwicklung und im Test implementiert wird und auch im Deployment und Betrieb einen hohen Stellenwert hat, entsteht ein ganzheitlicher Security-by-Design-Ansatz (**siehe Abbildung 3**). Dann werden technisch und organisatorisch Maßnahmen getroffen, die Sicherheit auf mehreren Ebenen sicherzustellen. Das kann bedeuten, Services so zu schneiden, dass sie mit minimalen Berechtigungen auskommen, oder dass sie im Fehlerfall kontrolliert neu starten und wieder in einen zuverlässigen Zustand wechseln.

Security by Design ist eine etablierte Praxis, die Sicherheit effizient und vor allem wirtschaftlich in die Anwendungsentwicklung zu integrieren. Somit wird Security nicht nur einfacher zu handhaben, sondern auch günstiger. Denn viele Schritte lassen sich automatisieren. So kann etwa mit einer statischen Code-Analyse

und mit Dependency- beziehungsweise Vulnerability-Scannern bei jedem Build oder zu einer voreingestellten Uhrzeit, etwa nachts, geprüft werden, ob im eigenen Code oder in eingesetzten Bibliotheken Schwachstellen vorhanden sind. Ein kontinuierliches Verbessern bringt mehr Sicherheit in die Softwareentwicklung als der Versuch, Sicherheit nachträglich mit viel Aufwand einzubauen.
Ein SSDLC entsteht jedoch nicht von selbst. In der Praxis hat sich gezeigt, dass es am besten funktioniert, wenn sich eine Person explizit dem Thema Security widmet. Das heißt natürlich nicht, dass diese Person alles alleine umsetzen muss. Vielmehr geht es darum, dass eine Person den Überblick hat und wichtige Aktivitäten anstößt und alle Stakeholder mit an Bord nimmt. Diese Rolle übernimmt der *Security Champion*, typischerweise ein Teammitglied, das diese Rolle zusätzlich zur bestehenden Aufgabe als Entwickler, Architekt oder Tester übernimmt. So sorgt der Security Champion aus dem Team heraus für die Entwicklung eines sicheren und robusten Produkts.

Cloud Security Tools
Die Absicherung der Anwendung ist der erste Schritt, der auch in einem traditionellen Rechenzentrum notwendig ist. Für eine Cloud-Anwendung ist es wichtig, dass die Software sich gut in die Landschaft und Sicherheitstools des Cloud-Anbieters integriert. In den meisten Fällen bedeutet das, entsprechende Log-Daten und Metriken zu nutzen. Auf diese Weise kann zum Beispiel ein vom Cloud-Provider angebotenes Security Information & Event Management (SIEM) komplexe Angriffsversuche erkennen.
Gängige Cloud Security Tools arbeiten mit einer Mischung aus maschinellem Lernen und dedizierter Abwehr von bekannten Angriffsvektoren. Je mehr anwendungsspezifische Parameter die Lösung bekommt, desto zuverlässiger kann sie eine Baseline des normalen Betriebs ermitteln. Der Nutzen liegt auf der Hand: Je zutreffender die Baseline ist, umso genauer sind Anomalien erkennbar. Angriffe zeigen sich oft in simplen Abweichungen vom Normalverhalten: So können etwa Traffic oder Ressourcenauslastung zunehmen, Fehlermeldungen gehäuft auftreten oder plötzlich ganz ausbleiben. Diese Abweichungen meldet ein SIEM umgehend.
Ein Vorteil eines SIEM-Tools: Anwender profitieren von Daten aus dem Verantwortungsbereich des Cloud-Providers. So kann ein Cloud-Provider beispielsweise erkennen, wenn bestimmte Muster verteilt über mehrere Kunden-Installationen auftreten, deren Anwendungen diese Ähnlichkeiten ansonsten nicht teilen. Für Spamfilter wird dieses Verhalten gerne genommen: Wenn nicht miteinander verbundene Personen plötzlich die gleichen Nachrichten bekommen, ist dies zumindest ungewöhnlich.
Ähnliche Schemata gibt es für APIs: Nach Bekanntwerden einer neuen Lücke in einer Library könnte der Cloud-Provider beispielsweise ein bestimmtes Angriffsmuster über alle Kunden beobachten, etwa Eingaben mit bestimmten Steuerzeichen. Letztlich kann der Provider schon reagieren, bevor dieser Angriff das Kundensystem erreicht. Auch DDoS-Attacken können solche Muster aufweisen. Diese wird ein Cloud-Provider ebenfalls eliminieren wollen, bevor sie die Kunden-Installationen erreichen.
Speist der Cloud-Provider diese Art der Information über mehrere Installationen ins eigene SIEM ein, können wir als Anwendungsverantwortliche handeln. Nur aus dem Wissen um die eigene Anwendung hätten wir dagegen keine Schlüsse ziehen können.
Zu guter Letzt gibt es in bestimmten Branchen und Bereichen regulatorische Anforderungen, die unter Umständen auch eine Zertifizierung erforderlich machen. Beispielsweise kann es sich lohnen, für die Gesundheitszertifizierung auf den Provider zu setzen.
Dazu kommt, dass der Kunde das ganz Netzwerk und das virtuelle Rechenzentrum in der Cloud über Software beziehungsweise Konfigurationen einrichtet. Auch hier gibt es Tools, die diese Konfiguration auf bekannte Schwachstellen prüfen. Es gibt kaum Situationen, in denen ein einzelner Service oder eine Datenbank direkt im Internet erreichbar sein muss. Fast immer ist hier ein API oder ein Gateway dazwischengeschaltet. Auch derartige Schwachstellen, die meist unbewusst und fälschlicherweise in der Konfiguration landen, lassen sich mit Analyse-Tools finden.

Mehrstufig absichern
Da Systeme in der Cloud in der Regel komplexe, verteilte Anwendungen sind, reicht es nicht, einzelne Teile abzusichern. Außerdem können, bekannt oder unbekannt, Bibliotheken und andere Software von Drittanbietern Lücken im System sein. Aus diesem Grund muss die Anwendung mehrstufig abgesichert sein.
Der äußere Schutz kann in vielen Fällen ein DDoS-Schutz und eine Web Application Firewall sein, die vom Cloud-Provider gestellt wird. Darauf allein sollte man sich aber nicht verlassen; auch wenn dadurch eine Vielzahl automatisierter Scans und Bots abgehalten wird, wird es auch Schaden verursachende Requests geben, die durchkommen. Diese müssen feingranularer abgefangen werden. Deswegen gehören zu einer mehrstufigen Sicherheitsarchitektur reduzierte Rechte auf den einzelnen Systemen, Antiviren- und Malware-Scanner, starke Verschlüsselung und Authentifizierung, genauso wie Firewalls und Intrusion-Detection-Systeme zwischen den Services.
Der mehrstufige Sicherheitskordon erschwert es einem Angreifer, sich von einem verwundbaren Server weiter vorzuarbeiten. So lässt sich auch im Falle eines erfolgreichen Einbruchs der Schaden immer noch lokal halten.

Absicherung prüfen
Das richtige Niveau an Absicherung ist erreicht, wenn man keine Hemmungen mehr davor hat, einen umfangreichen Penetrationstest gegen die Produktionsumgebung zu beauftragen. Diese Maßnahme prüft dann auch das Zusammenspiel zwischen eigenen Maßnahmen und den Schutzvorrichtungen des Cloud-Providers. Diesen Test empfehlen wir unbedingt, um die oben angesprochenen Lücken zwischen den Verantwortlichkeiten auszuschließen. Idealerweise wird auf der Umgebung getestet, bei der die Absicherung am wichtigsten ist.

Im Gespräch sein
Der Dialog über mögliche Lücken in der Absicherung zwischen Cloud-Provider und Kunde baut zudem vor für den Fall, der hoffentlich nicht eintritt: wenn die Anwendung doch gehackt wird. Wenn Einverständnis über Zuständigkeiten besteht, und beide Seiten ihren Teil getan haben, treten bei einem Sicherheitsvorfall weniger Konflikte auf. Denn seien wir ehrlich: In dieser Situation kann es schwer genug sein, einen kühlen Kopf zu bewahren. Schuldzuweisungen zwischen Parteien – egal ob berechtigt oder nicht – sind in diesem Moment fehl am Platz. Sie vergiften nur eine eh schon angespannte Situation.

Cloud Console
Zu guter Letzt darf man nicht vergessen, dass die virtuelle Umgebung für die Anwendung nicht physikalisch aufgebaut wird, sondern per Software und Konfiguration in der Cloud Console definiert wird. Dort lassen sich Regeln und Schutzmaßnahmen jederzeit neu definieren, somit ist sie eine äußerst kritische Komponente. Das bedeutet zum einen, dass das Berechtigungsschema geplant und ent-

sprechend umgesetzt werden muss. Nur ein kleiner Personenkreis braucht überhaupt erweiterte Rechte. Zusätzlich müssen diese aktuell gehalten und überwacht werden. Starke Passwörter und eine solide 2-Faktor-Authentifizierung sind hier Pflicht.

Ganzheitliche Security

Für den Produktivbetrieb muss es immer ganzheitliche Security sein: Moderne, komplexe IT-Systeme lassen sich nur absichern, wenn man einen ganzheitlichen Security-by-Design-Ansatz wählt; das setzt mindestens voraus, dass die Cloud, die Konfiguration von Infrastruktur und Services und auch die Anwendung in der Cloud sicher sind. Mehr noch, es erfordert, dass Build- und Deployment-Prozesse solide eingerichtet werden, die Netzwerke sauber segmentiert sind, Alerting und Monitoring aktiv genutzt werden und alle X-as-a-Service-Dienste sicher und zuverlässig integriert sind. Und das alles getestet und reproduzierbar. Dazu kommt, dass virtuelle Umgebungen Möglichkeiten bieten, die es in einem physikalischen Rechenzentrum nicht gab. Daher bieten Cloud-Provider oft erweiterte Security-Lösungen, bei denen es sich lohnt, genauer zu prüfen, welchen Mehrwert sie für die eigene Anwendung bringen.

Fazit

Dieser Grad an Absicherung hört sich umfangreich an – ist aber erreichbar, wenn man die Maßnahmen Schritt für Schritt einführt, oder gleich von Anfang an etabliert. Steht die Sicherheitsarchitektur, kann die Cloud sicherer sein als das eigene Rechenzentrum. Ein Blick in die vertraglichen Details ist wichtig, damit keine Lücken zwischen den Zuständigkeiten entstehen. Zusätzlich bietet ein Cloud-Provider viele Tools, die schon optimal auf die Umgebung abgestimmt sind. Außerdem sichert der Provider selbst vor bestimmten Angriffen, die mehrere Kunden-Systeme betreffen, und nutzt dafür eigene Informationen und Dienste. Damit der Schutz wirklich besteht, muss die eigene Anwendung auf die Umgebung beim Provider abgestimmt sein – diese Herausforderung gibt es im eigenen Rechenzentrum jedoch auch. Hier wirken sie nur bekannter.

Als letzten Baustein plädieren wir daher für regelmäßige Security-Weiterbildungen für alle Rollen im Entwicklungsteam. So entsteht ein gemeinsames Sicherheitsbewusstsein, das Security als Qualitätsdimension von Software denkt. ||

Der Autor

Philippe Schrettenbrunner (philippe.schrettenbrunner@maibornwolff.de) ist Principal Security Architect bei Maiborn-Wolff. Seine Leidenschaft für sicheres Coden kommt aus mehr als 15 Jahren Software-Engineering in Kundenprojekten. Neben dem Aufbau komplexer IT-Systeme gilt sein besonderes Interesse der On-the-job-Ausbildung von Kolleginnen und Kollegen im eigenen Team oder beim Kunden.