DSGVO – 11/2020 – Daten löschen in komplexen Systemlandschaften

DSGVO – 11/2020 – Daten löschen in komplexen Systemlandschaften

[expand title="mehr lesen…"]

Daten löschen in komplexen Systemlandschaften

Ballast abwerfen

Frank Heisel

Viele Unternehmen setzen das in der DSGVO geforderte Löschen obsoleter Daten entweder gar nicht oder nur unzureichend um. Da sich die Datenschutzaufsichtsbehörden in jüngster Zeit verstärkt für das Thema interessieren, müssen Firmen umdenken.

-tract

- Das datenschutzkonforme Löschen personenbezogener Informationen in verteilten Systemen ist eine anspruchsvolle Aufgabe, denn es gibt meist zahllose Abhängigkeiten zwischen den Daten.
- Dabei müssen die Verantwortlichen nicht nur rechtliche,

sondern auch viele technische Aspekte berücksichtigen.
- Bislang haben viele Unternehmen in dieser Hinsicht wenig getan. DSGVO, bissige Datenschutzbehörden und hohe Bußgelder zwingen sie nun zum Umdenken.

Personenbezogene Daten lassen sich in einem einzelnen System noch recht einfach löschen. In komplexen, heterogenen IT-Welten sieht die Sache anders aus. Hier gilt es, die Integrität, Verfügbarkeit und Konsistenz der Informationen nicht durch übereiltes Hantieren in den Datenbeständen zu gefährden.

Art. 17 Abs. 1 DSGVO räumt einem Betroffenen beim Vorliegen bestimmter Gründe das Recht ein, etwa die Betreiber eines Onlineshops dazu aufzufordern, seine persönlichen Daten unverzüglich aus den Systemen zu entfernen. Das anlassbezogene Löschen muss beispielsweise bei Widerruf einer erteilten Einwilligung erfolgen [Art. 17 Abs. 1 lit. b) DSGVO] oder bei - einem erfolgreichen Widerspruch gegen die Verarbeitung [Art. 17 Abs. 1 lit. c) DSGVO]. Der Artikel 17 regelt das sogenannte Recht auf Vergessenwerden.

In diesem Fall ergreift der Betroffene die Initiative. Um seinen Wunsch zu erfüllen, muss das Unternehmen einen Prozess implementiert haben, der das sofortige Prüfen der Anfrage einleitet. Dieser Prozess erfasst alle zugehörigen Datensätze, die über mehrere Systeme verteilt sein können. Nach der Überprüfung, ob und welche Daten gelöscht werden können, startet die zuständige Fachabteilung den Prozess manuell. Falls die Entwickler eine Löschfunktion eingerichtet haben, sollte man sie natürlich verwenden. Eine solche einfache Löschung funktioniert allerdings nur bei Systemen, die keine oder nur wenige Schnittstellen zu vor- und nachgelagerten Rechnern haben.

Das anlasslose Löschen verlangt das Entfernen von Daten, ohne dass jemand eine explizite Anfrage stellt. Diese Pflicht ergibt sich aus Art. 17 Abs. 1 lit. a) DSGVO. Danach müssen

Unternehmen personenbezogene Daten löschen, wenn der Zweck der Verarbeitung entfällt und keine Gründe gemäß Art. 17 Abs. 3 DSGVO dagegensprechen.

Die Aufgabe besteht nun darin, Löschfristen für einzelne Datensätze und -felder festzulegen, Löschregeln zu definieren und sie regelmäßig anzuwenden. Die Bereinigungsfrequenz dürfte sich in der Regel an den Aufbewahrungsfristen orientieren – je kürzer sie sind, desto höher die Rate der Wiederholungen. Bei wenigen und einfach strukturierten Daten lässt sich dieser Vorgang vergleichsweise einfach umsetzen, indem man einzelne Skripte zeitgesteuert ausführt oder manuell eingreift. Im Verbund mit anderen Systemen und verteilter Ablage der Daten kann das Löschen dagegen nur als Gesamtkonzept funktionieren, da ein ungeregelter Datenfluss zwischen den Systemen gelöschte Daten mit der nächsten Übertragung wieder einspielen kann.

Daten liegen überall verstreut

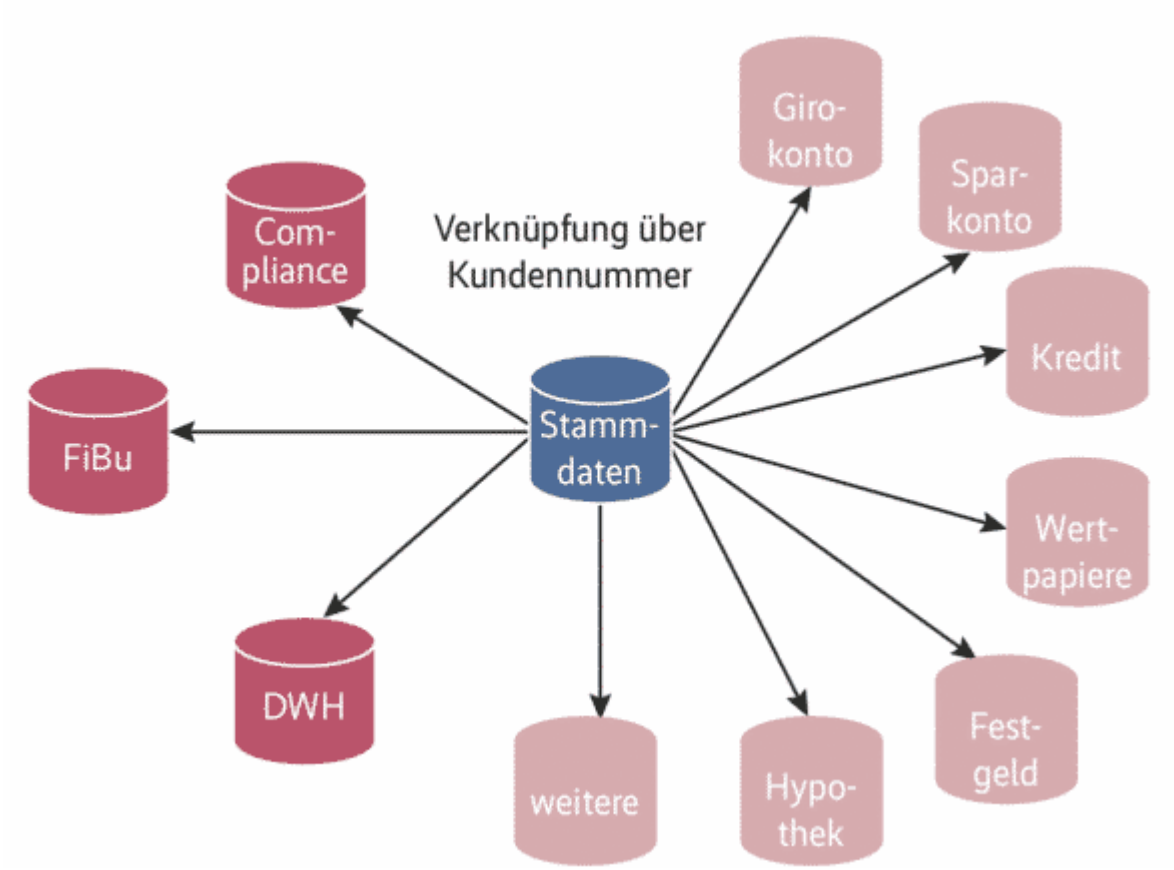
In größeren Organisationen besteht die IT-Landschaft gewöhnlich aus zahlreichen heterogen organisierten Anwendungen. Personenbezogene Daten werden zu verschiedenen Zwecken in vielen Datenbanken mehrfach oder verteilt gespeichert. Und nicht nur in den Produktivsystemen: Auch Entwicklungs- und Testumgebungen arbeiten oft damit.

In einem Webshop beispielsweise liefert der Kunde mindestens Name und Anschrift ab. Die Rechnungen werden als PDF gespeichert, per E-Mail versendet und an nachgelagerte Applikationen wie die Finanzbuchhaltung weitergereicht. Der Webshop will meist mehr wissen. Rechnungen müssen jedoch nur die in § 14 Abs. 4 UStG genannten Informationen enthalten. Alle erhobenen Daten landen normalerweise in einem CRM-System oder in einem Data Warehouse. Das Löschen einzelner Einträge muss daher in verschiedenen Systemen zu unterschiedlichen Zeitpunkten erfolgen.

Der Umfang der gespeicherten Daten verdient ebenfalls

Beachtung. Shop- oder CRM-Anwendungen erfassen außer dem Namen weitere Daten wie Kundennummer, Lieferanschrift, Rechnungsanschrift, Telefonnummer und E-Mail. Diese Daten sind notwendig, um die Bestellung zu bearbeiten, etwa um die Lieferung durch eine Spedition zu veranlassen und den Versand per E-Mail anzukündigen. Nach erfolgreicher Zustellung braucht man einiges davon zur Erfüllung des Vertrags jedoch nicht mehr. So sind für das Erstellen der Rechnung neben den Informationen zu den gelieferten Waren oder Leistungen lediglich der Name und die Anschrift des Empfängers, nicht jedoch die E-Mail-Adresse oder Telefonnummer erforderlich. Dafür entfällt dann der Verarbeitungszweck.

Komplexe Systeme verbinden die personenbezogenen Daten beispielsweise über ein Merkmal wie die Kundennummer (Abbildung 1). Die eigentlichen Daten liegen in der Stammdatenbank, alle anderen Systeme greifen lediglich über die Verknüpfung darauf zu. Zudem existieren viele Schnittstellen für den Datenaustausch. Bei einer Analyse des Datenflusses gehören alle Schnittstellen, über die personenbezogene Informationen laufen, in die Betrachtung. Wesentlichen Einfluss auf das Einsetzen einer datenschutzkonformen Löschung haben Faktoren wie der Umfang der personenbezogenen Daten, die Häufigkeit der Übertragung und die Art der Schnittstelle.



So etwa sieht die Systemlandschaft in einer Bank aus. Jedes Datenhaltungssystem speichert nur Teile eines kompletten Datensatzes. Löschen ist hier kompliziert (Abb. 1).

Akribische Untersuchungen sind notwendig

Die Fachabteilungen müssen für jede Schnittstelle zunächst erheben und dokumentieren, welche Daten sie wohin übermitteln. Entscheidend ist dabei, ob sie ausschließlich über ein systemweit eindeutiges Merkmal wie die Kundennummer verknüpft werden oder ob weitere Parameter im Spiel sind. Hier sollte man im Sinne einer guten „Privacy by Design“ darauf achten, dass nur notwendige Daten fließen. Bei Ausgangsrechnungen wäre es ausreichend, lediglich den Namen und die Anschrift des Rechnungsempfängers zu übertragen, nicht jedoch die weiteren Kontaktdaten.

Ein weiterer Aspekt ist die Häufigkeit der Übermittlung. Die meisten Schnittstellen dürften regelmäßig Daten erhalten und weitergeben. Falls das empfangende System eigene autonome Löschroutinen anbietet, müssen Übermittlungs- und

Löschfrequenz aufeinander abgestimmt sein, da es sonst vorkommen kann, dass Daten erneut übertragen werden und jemand sie im Zielsystem wieder löschen muss. Bei der Implementierung von Löschprozessen spielt daher die Art der Schnittstelle eine entscheidende Rolle.

Werden die Daten in Dateien verschickt, muss sichergestellt sein, dass auf dem abgebenden System oder einem eventuell zwischengeschalteten Fileserver keine Kopien verbleiben, da der Verarbeitungszweck nun erfüllt ist. Gegebenenfalls benötigt der Fileserver ein eigenes Löschkonzept, und er darf nicht als Backup oder Archiv missbraucht werden.

Eine Möglichkeit, Abhängigkeiten zu berücksichtigen, bieten zentrale Löschsysteme, die ihre Arbeit über alle angeschlossenen Systeme koordinieren und dafür sorgen, dass keine Daten verschwinden, die anderweitig noch benötigt werden. Man spricht in solchen Fällen von einer Orchestrierung der Löschung. Es gibt einige kommerzielle Produkte, etwa das Modul SAP ILM für die SAP-Welt, eine Alternative dazu namens - Cronos von der gleichnamigen Unternehmensberatung aus Münster und das universal einsetzbare Customer Data Deletion Management (CDMS) von Impetus.

Die zentrale Löschinstanz sammelt von allen Systemen die personenbezogenen Daten ein und hält sie in einer eigenen Datenbank. Die Löschregeln informieren bei Bedarf die Systemverantwortlichen über zu löschende Datensätze. Administratoren können dann das Löschen manuell oder automatisiert anstoßen und einen Nachweis an das zentrale Löschsystem schicken, das diesen archiviert. Hier angesiedelte Regeln sollten das unkontrollierte Löschen in den betroffenen Ablageorten verhindern. So lässt sich beispielsweise sicherstellen, dass der Stammdatensatz eines Kunden, der noch laufende Verträge hat, erst dann freigegeben wird, wenn alle Verträge beendet und die jeweiligen Aufbewahrungsfristen ausgelaufen sind.

Weiterer Vorteil einer zentralen Instanz: Unternehmen können sich bei einer Anfrage schnell einen Überblick darüber verschaffen, ob und welche Informationen zu dem Betroffenen vorliegen. Das Löschen lässt sich bei gesetzlichen Aufbewahrungsfristen mit einer entsprechenden Regel unterdrücken.

Manchmal müssen Daten verweilen

In bestimmten Situationen muss das Löschen grundsätzlich verhindert werden. Als Beispiel seien hier eine laufende Betriebsprüfung oder ein andauernder Rechtsstreit genannt. Bei einer Betriebsprüfung müssen die Daten bis zum Abschluss vorliegen. Hier bietet es sich an, die Frist zu verlängern. Relevante Daten zur Rechnungslegung liegen oft in Jahresarchiven, deren Löschung sich aussetzen lässt. Sie muss nach Abschluss der Prüfung manuell angestoßen werden. In jedem Fall sollte der verantwortliche Fachbereich vor dem Auslösen der regulären Löschung über den Vorgang mitbestimmen. Um nicht zu viele Daten auszunehmen, ist eine genaue Prüfung der Systeme und Daten notwendig.

Während eines Rechtsstreits müssen die benötigten Daten ebenfalls zur Verfügung stehen. In diesem Fall sollte die Rechtsabteilung die Fälle, deren Daten nicht gelöscht werden sollen, an den für das Löschen verantwortlichen Fachbereich übermitteln. Der nimmt die betroffenen Daten von der Löschung aus und entfernt sie nach Abschluss des Verfahrens einzeln. Alternativ bietet es sich an, alle benötigten Daten in einem separaten Archiv zu sichern, das sich in der Verantwortung der Rechtsabteilung befindet. Ist der Rechtsstreit beendet, stößt sie die datenschutzkonforme Löschung an.

Um die Rechenschaftspflicht gemäß Art. 5 Abs. 2 DSGVO zu erfüllen, ist das Unternehmen verpflichtet, die erfolgreiche Löschung nachweisen zu können. Dabei muss es darauf achten, keinen neuen Fundus personenbezogener Daten zu erzeugen. Das Ausführen eines SQL-Löschbefehls und das Speichern der

Ergebnismenge wäre beispielsweise ein neuer Datenbestand, dem allerdings die Rechtsgrundlage fehlt.

Bei einer anlassbezogenen Löschung sind die Verantwortlichen verpflichtet, den gesamten Vorgang der Anfrage zu dokumentieren. Hier kann auch das erfolgreiche Löschen von Daten einfließen. Mit Screenshots, den SQL-Befehlen oder dem negativen Ergebnis einer Suchabfrage ließe sich der technische Vorgang nachweisen.

Schwieriger ist dagegen das Dokumentieren des anlasslosen Löschens nach Zeitablauf. Das erledigen in der Regel automatisierte Skripte, die in den Datenbanken parametrisierte Routinen ausführen. Als Nachweise können hier die Systemeinstellungen gelten, die zeigen, dass die Löschroutinen regelmäßig arbeiten, sowie die Skripte selbst. Zusätzlich müssen die Zuständigen die Parametrisierung der einzelnen Aufrufe und das Ergebnis belegen. Dabei dürfen nur die Informationen im System verbleiben, die die erfolgreiche Ausführung dokumentieren, nicht jedoch die gelöschten Daten selbst.

Backup und Restore sind außen vor

Anders als in Archiven darf in Backups niemand etwas ändern, da deren Zweck in der Wiederherstellung von Daten liegt. Das Herausnehmen einzelner Daten aus einem Backup kann die Konsistenz der Daten verletzen und das Wiederherstellen verhindern. Bei einem Restore geraten jedoch auch die gewollt gelöschten Daten zurück ins System. Um diesen Vorgang datenschutzkonform abzuwickeln, müssen sie im zurückgespielten Datenbestand nochmals gelöscht werden. Das lässt sich beispielsweise anhand der Löschnachweise erledigen. Das Vorgehen beim Wiederherstellen sollte im entsprechenden Verfahren dokumentiert und getestet sein.

Genormtes Vorgehen

Eine gute Grundlage für Löschkonzepte findet sich in der Norm DIN 66398. Sie zeigt eine Vorgehensweise zum Erstellen solcher Konzepte und bietet Orientierungshilfe für Unternehmen aller Größen. Projekte, die das erfolgreich umsetzen wollen, kommen ohne die betroffenen Fachbereiche, die IT-Abteilung und den Datenschutzbeauftragten nicht aus.

Ein solches Projekt könnte nach folgenden Schritten vorgehen (Abbildung 2):

Schritt für Schritt: Vorgehensmodell zum regelkonformen Löschen personenbezogener Daten (Abb. 2)

System-Scoping: Zunächst müssen die Zuständigen alle Systeme ermitteln, die personenbezogene Daten verarbeiten. Hierzu sollte das Unternehmen in einem Regelwerk Datenschutzklassen vorgeben. Häufig erfolgt das Klassifizieren im Rahmen der Schutzbedarfsfeststellung, die neben den Kriterien Vertraulichkeit, Integrität und Verfügbarkeit auch erfasst, ob personenbezogene Daten verarbeitet werden.

Untersuchung der Daten: Weiterhin ist für alle erkannten Systeme detailliert aufzunehmen, um welche Daten es geht. Sie lassen sich dann in vorgegebene Kategorien einsortieren.

Fachliche Löschregeln: Aus den vorliegenden Informationen können die Administratoren nun fachliche Löschregeln ableiten. Dabei ermitteln sie für jede Datenkategorie, ob Aufbewahrungsfristen existieren und welche Löschfristen sich daraus ergeben.

Analyse der Schnittstellen und Abhängigkeiten: Zu den einzelnen Systemen müssen die Fachabteilungen weiterhin die Schnittstellen zu vor- und nachgelagerten Systemen finden und analysieren. Dabei erfassen sie die Quell- und Zielsysteme, die Datenkategorien, die Häufigkeit der Übertragung und die Art der Schnittstelle. Mit dieser Dokumentation können sie Abhängigkeiten untersuchen.

Fachliches Löschkonzept: Aus den gesammelten Informationen wird ein Löschkonzept erstellt.

Entscheidung für eine Löschlösung: Abhängig von der Analyse muss das Unternehmen eine Löschstrategie formulieren – etwa, ob es eine zentrale, eine dezentrale oder eine Mischlösung einführen will.

Technische Löschkonzepte: Sie beschreiben die konkrete Durchführung des Löschens. Auch Produktion und Ablage der Löschnachweise werden hier dokumentiert.

Test: Vor der Einführung der Löschroutinen müssen diese ausführlich geprüft werden. Dabei sind sowohl die vollständige Löschung innerhalb des Systems als auch die Abhängigkeiten zu anderen Systemen einzubeziehen.

Umsetzung: Nach erfolgreichem Test können die Zuständigen die Löschroutinen auf die einzelnen Systeme loslassen. In der Regel kann die Einführung in Stand-alone-Systemen, die keinerlei Schnittstellen nach außen haben, unabhängig vom Rest der IT-Welt erfolgen. Die meisten Systeme sind jedoch mit anderen verbunden und müssen gleichzeitig in Produktion genommen werden.

Ablage der Nachweise: Darüber muss ebenfalls eine Entscheidung fallen. Eine zentrale Löschlösung bietet oftmals eine Ablagemöglichkeit an. Wird dezentral gelöscht, müssen die Zuständigen über eine gemeinsame Ablagemöglichkeit und einen Prozess nachdenken, der die Nachweisführung sicherstellt.

Fazit

Das Ganze ist genauso kompliziert, wie es sich anhört. Die Vielzahl der Systeme, die Abhängigkeiten, der Umfang der Daten und ihre Redundanz erschweren die Umsetzung. Hinzu kommt, dass mit dem Löschen von Daten ein Kulturwandel einhergeht, da der jederzeitige Zugriff auf Daten in Vor-DSGVO-Zeiten wichtiger schien als das datenschutzkonforme Löschen.

Die mit der Einführung der Datenschutz-Grundverordnung drastisch gestiegenen und von den Aufsichtsbehörden inzwischen auch durchgesetzten Bußgelder sollten reichen, den Unternehmen das Umdenken zu erleichtern. Bei entsprechenden Projekten muss man sich auf möglicherweise lange Laufzeiten von mehreren Monaten bis zu einigen Jahren einstellen. (jd@ix.de) Frank Heisel

war bei einer Big-Four-Wirtschaftsprüfungsgesellschaft als verantwortlicher Prüfungsleiter für System-, Prozess- und IKS-Prüfungen tätig. Er ist als externer Datenschutzbeauftragter für mehrere Unternehmen benannt und berät Unternehmen zum Thema Datenschutz und internes Kontrollsystem.

[/expand]

DSGVO – 11/2020 – Löschung personenbezogener Daten

DSGVO – 11/2020 – Löschung personenbezogener Daten

[expand title="mehr lesen…"]

Löschung personenbezogener Daten

Seid sparsam

Tobias Haar

Das Datenschutzrecht verlangt die Löschung personenbezogener Daten, wenn sie nicht mehr rechtmäßig verarbeitet werden dürfen. Wann und wie das zu erfolgen hat, ist oft eine schwierige Einzelfallentscheidung.

-tract

- Das Datenschutzrecht verlangt die Löschung personenbezogener Daten, wenn in Unternehmen oder Organisationen der Verarbeitungszweck dieser Daten entfällt oder keine Einwilligung des Betroffenen vorliegt.
- Schwierigkeiten bei der Erfüllung der Vorgaben bereiten zu wenig spezifizierte Regelungen und Definitionen, einander widersprechende Speicherfristen, eine Datenverarbeitung durch Dritte und mehr.
- Was die DSGVO letztlich fordert, ist ein verantwortlicher, gesetzeskonformer Umgang mit personenbezogenen Daten im Rahmen eines auf die eigene Organisation zugeschnittenen Datenschutz- und -löschkonzepts.

Die Datenschutz-Grundverordnung hat sich mit ihren Pflichten in den letzten zwei Jahren zum Angstgegner vieler Unternehmen entwickelt. Die drohenden Bußgelder sind enorm, das Risiko von Abmahnungen durch Konkurrenten und Verbände ist real. Auf der anderen Seite müssen Unternehmen personenbezogene Daten mitunter ein Jahrzehnt oder noch länger speichern, um ihren vertraglichen und gesetzlichen Pflichten zu genügen. Eine nicht immer einfache Gratwanderung – im Detail entstehen unlösbar erscheinende Konflikte. Hier den Überblick zu behalten, ist selbst für Juristen und Datenschutzbehörden eine stetige Herausforderung, zumal sich die Rahmenbedingungen auch ändern.

Um sich dieser Herausforderung zu stellen, hilft es, die zugrunde liegenden Vorgaben des Datenschutzrechts zu kennen. In Zweifelsfällen muss man sich damit behelfen, herauszufinden, was die jeweilige Intention des Gesetzgebers für bestimmte gesetzliche Regelungen ist. Um Unternehmensentscheider hierbei zu unterstützen, ist die Bestellung eines betrieblichen Datenschutzbeauftragten für viele Unternehmen verpflichtend. Hilft all dies nicht, bleibt stets die Möglichkeit, sich ratsuchend an die Datenschutzbehörden zu wenden. Das ist im Einzelfall womöglich immer noch besser, als sich bei Mängeln in der Compliance erwischen zu lassen. Es gibt erste Bußgeldentscheidungen der Datenschutzaufsicht, die das belegen. Hier zeigen sich Parallelen zum Kartell- und Wettbewerbsrecht, das Kooperation (und mitunter auch das Auftreten als Kronzeuge) belohnt.

Das geschützte Gut

Was schützt das Datenschutzrecht? Es gilt ausschließlich für personenbezogene Daten. Diesen Begriff definiert Art. 4 Nr. 1 DSGVO ausführlich und auch anhand von Beispielen als „alle Informationen, die sich auf eine identifizierte oder identifizierbare natürliche Person (im Folgenden ‚betroffene Person') beziehen; als identifizierbar wird eine natürliche

Person angesehen, die direkt oder indirekt, insbesondere mittels Zuordnung zu einer Kennung wie einem Namen, zu einer Kennnummer, zu Standortdaten, zu einer Online-Kennung oder zu einem oder mehreren besonderen Merkmalen identifiziert werden kann, die Ausdruck der physischen, physiologischen, genetischen, psychischen, wirtschaftlichen, kulturellen oder sozialen Identität dieser natürlichen Person sind“.

Diese Definition reicht sehr weit. Es gilt – und das wurde bereits vom Europäischen Gerichtshof entschieden – ein objektiver Datenbegriff. Es kommt nicht darauf an, ob ein für die Datenverarbeitung verantwortliches Unternehmen mit vorhandenen Daten einen Bezug zu einer „identifizierte[n] oder identifizierbare[n] natürliche[n] Person“ herstellen kann. Es kommt darauf an, ob es irgendeine Stelle auf der Welt gibt, die etwa aus mehreren Datenpunkten einen Personenbezug herstellen könnte. Oftmals wird eingeschränkt, dass der Bezug eines Datums zu einer Person mit „verhältnismäßigen Mitteln“ herstellbar sein muss.

Als bekanntes Beispiel dienen IP-Adressen. Ein Webseitenbetreiber kann alleine aus der IP-Adresse eines Webseitenbesuchers nicht auf dessen Person Rückschlüsse ziehen. Weil es aber der Internetzugangsprovider kann, liegt auch für den Webseitenbetreiber ein personenbezogenes Datum vor und die Pflichten aus der DSGVO greifen. Hier zeigen sich Schnittstellen zur umstrittenen Vorratsdatenspeicherung. Auch sie führt rechtlich zu einer Pflicht des Internetanbieters, bestimmte Daten nicht zu löschen.

Es gibt in Art. 5 der DSGVO für die Datenverarbeitung übergreifende Grundsätze, die im Umgang mit personenbezogenen Daten stets berücksichtigt werden müssen: Diese müssen rechtmäßig und für den Betroffenen transparent verarbeitet werden. Sie unterliegen einer Zweckbindung und dürfen nicht ohne Weiteres für andere Zwecke verarbeitet werden. Ihre Verarbeitung muss auf das notwendige Maß reduziert werden. Und schließlich müssen sie richtig und „angemessen sicher“

verarbeitet werden.

Zeitlich begrenztes Speichern

Eine Herausforderung stellt für viele verarbeitende Stellen die Vorgabe der „Speicherbegrenzung“ dar. Danach dürfen personenbezogene Daten nur „in einer Form gespeichert werden, die die Identifizierung der betroffenen Personen nur so lange ermöglicht, wie es für die Zwecke, für die sie verarbeitet werden, erforderlich ist“. Wer diese Vorgaben nicht einhält, muss die Daten löschen.

Die DSGVO regelt dabei nicht, welcher in Tagen, Monaten oder Jahren bemessene Zeitraum im Einzelfall gilt. Das „nur so lange“ kann oftmals nur mittels des Zwecks der Datenverarbeitung oder gesetzlicher Vorgaben beantwortet werden. Ist Zweck der Datenverarbeitung die Erfüllung eines Vertrages, zum Beispiel die Lieferung von Versandartikeln, darf der Händler die Daten selbstverständlich für die Bearbeitung der Bestellung, den Versand und die Abrechnung der Lieferung verarbeiten und sie dabei auch speichern.

Im Versandhandel kommt hinzu, dass das Speichern bis zum Ablauf der Widerrufsfrist für Verbraucher gestattet ist. Danach müsste der Händler die Daten jedoch wieder löschen. Spitzfindige könnten bereits argumentieren, dass eine Speicherung bis zum Ablauf der regelmäßigen Verjährungsfrist für Gewährleistungsmängel nicht mehr gestattet ist. Denn im Gewährleistungsfall müsste der Kunde nachweisen, wann und von wem er ein Produkt erworben hat.

Spezialgesetz sticht Datenschutz

An dieser Stelle „hilft“ das Steuer- und das Handelsrecht. Danach müssen etwa Rechnungen bis zu zehn Jahre aufbewahrt werden. Diese Vorschriften haben als Spezialgesetz Vorrang vor dem Datenschutzrecht. Erst nach zehn Jahren dürfen die Rechnungen gelöscht werden, nach Datenschutzrecht müssen sie

das dann aber auch. Auch Buchungsbelege, „Bücher und Aufzeichnungen“ und andere Unterlagen unterliegen dieser Aufbewahrungsfrist.

Nur sechs Jahre aufzubewahren sind empfangene und abgesandte Handels- und Geschäftsbriefe sowie sonstige Unterlagen, soweit sie für die Besteuerung von Bedeutung sind. Auch E-Mails mit entsprechendem Inhalt können „Briefe“ nach diesen Vorgaben sein. Anstatt zu löschen, müssen Unternehmen insbesondere bei ausscheidenden Mitarbeitern prüfen, ob sie deren E-Mail-Accounts weiterhin verfügbar halten müssen, um bei Bedarf darauf zugreifen zu können.

Bei Verträgen kommt meist hinzu, dass die Aufbewahrungsfrist erst mit Ende der Vertragslaufzeit zu laufen beginnt. Enthalten sie etwa personenbezogene Daten eines Vermieters, müssen sie trotz der Vorgaben des Datenschutzrechts sechs volle Jahre nach Ende des Mietvertrags aufbewahrt werden. Die Fristen können sich beispielsweise bei laufenden Steuerverfahren zudem verlängern. Eine Verletzung der Aufbewahrungspflicht kann mit Bußgeldern belegt werden oder zu unangenehmen Steuerschätzungen führen. Können Beweise in Gerichtsverfahren wegen Löschung nicht mehr vorgelegt werden, drohen finanzielle Nachteile durch entsprechende Urteile.

Die Art und Weise der Aufbewahrung von Steuerunterlagen ergibt sich aus den „Grundsätzen zur ordnungsgemäßen Führung und Aufbewahrung von Büchern, Aufzeichnungen und Unterlagen in elektronischer Form sowie zum Datenzugriff (GoBD)“ [1]. Auch IP-Adressen mit Zeitstempel können steuerrelevante Informationen sein, die aufbewahrt werden müssen. Sie können beispielsweise gebraucht werden für den Umsatzsteuernachweis, ob eine „elektronisch erbrachte Dienstleistung“ von einem Verbraucher in Deutschland (16% bzw. 19% Umsatzsteuer) oder etwa in Ungarn (dann 27%) abgerufen wurde. Hierzu zählen etwa Streaming, Premium-Features in Computerspielen, E-Books, Software- oder App-Downloads und dergleichen.

Die Behörde – dein Freund und Helfer

Den meisten Steuerbehörden genügt es, wenn eine IPv4-Adresse um das letzte Oktett verkürzt gespeichert wird, um diesen Nachweis zu führen. Ähnliches gilt für teilgeschwärzte Kreditkartennummern oder eine IBAN. Im Einzelfall ist hier eine Abstimmung sowohl mit den Steuerbehörden als auch mit den Datenschutzbehörden erforderlich. Als Unternehmen kann man auch beide Behörden bitten, die Diskussion gemeinsam zu führen und eine gemeinsame Lösung zu erarbeiten.

Der Grundsatz der Datensparsamkeit und Zweckbindung verlangt, die personenbezogenen Datensätze nach Zweckerreichung aus allen Systemen zu löschen, die nicht für steuerliche Belange eingesetzt werden. Hierzu zählt etwa ein Customer-Relationship-Management-System (CRM). Die Pflicht zur Speicherung von Daten umfasst aber nicht auch das Recht, diese in sämtlichen Systemen stets verfügbar zu halten. Die Zweckbindung verlangt, dass nur insoweit gespeichert wird, wie die Daten auch künftig benötigt werden. Eine Ausnahme ist dann möglich, wenn ein Betroffener einer längeren Datenspeicherung zugestimmt hat. Dies kommt beispielsweise bei Kundenaccounts von Amazon und Co. in Betracht. Ob diese Einwilligung stets nach DSGVO-Grundsätzen wirksam ist, ist eine andere Frage.

Auch aus anderen Rechtsbereichen ergibt sich eine Pflicht zur Aufbewahrung personenbezogener Daten. Das gilt etwa für das Arbeitsrecht, das Sozialversicherungsrecht, das Produkthaftungsgesetz et cetera. Die IHK Pfalz stellt eine Übersicht zur Verfügung (siehe ix.de/zy59), die auch kurios anmutende Kategorien wie „Essensmarkenabrechnungen“ auflistet.

Nicht mehr dem Zweck entsprechend benötigte und keinen Aufbewahrungspflichten mehr unterliegende personenbezogene Daten sind zu löschen. Das ergibt sich bereits aus allgemeinen Datenschutzgrundsätzen, insbesondere aber aus Art. 17 der DSGVO. Diese Vorschrift regelt das „Recht auf Vergessenwerden“. Sie spiegelt die Pflichten nach Art. 5 der

DSGVO und verlangt vor allem die Löschung bei Zweckwegfall, Widerruf einer Einwilligung und unrechtmäßiger Verarbeitung. Es handelt sich dabei um ein Recht des Betroffenen gegenüber der Daten verarbeitenden Stelle. Das bedeutet aber nicht, dass eine Löschung nur dann stattfinden muss, wenn er dieses Recht explizit geltend macht. Wann immer keine gesetzliche Rechtfertigung oder wirksame Einwilligung des Betroffenen vorliegt, müssen personenbezogene Daten gelöscht werden.

Die DSGVO definiert nicht, was rechtlich unter Löschung zu verstehen ist. Das war bis zum Inkrafttreten der früheren Fassung des Bundesdatenschutzgesetzes noch anders, dort war das Löschen von Daten als das „Unkenntlichmachen von Daten“ festgelegt. So definieren es auch Wikipedia und Gerichtsurteile nach dem Strafgesetzbuch, etwa beim strafbaren „Ausspähen von Daten“.

Den Personenbezug entfernen

Löschung personenbezogener Daten bedeutet allerdings nicht, dass die Daten auch physisch vernichtet werden müssen. Es genügt, wenn ihnen der Personenbezug genommen wird. Das ist ein bedeutender Unterschied, wie die österreichische Datenschutzbehörde auf Anfrage eines Versicherungsunternehmens geklärt hat. Werden personenbezogene Daten zu anonymisierten Daten, dürfen sie nach der DSGVO auch weiterhin zeitlich unbefristet verarbeitet werden. Die DSGVO ist auf solche Daten schlicht nicht anwendbar.

Einen Grenzfall bildet die Pseudonymisierung personenbezogener Daten. Im Erwägungsgrund 26 zur DSGVO heißt es dazu: „Einer Pseudonymisierung unterzogene personenbezogene Daten, die durch Heranziehung zusätzlicher Informationen einer natürlichen Person zugeordnet werden könnten, sollten als Informationen über eine identifizierbare natürliche Person betrachtet werden.“ Fachleute deuten das Wort „sollten“ als Einschränkung dahingehend, dass es datenschutzrechtlich vertretbar ist, wenn die Verbindung zwischen einem

pseudonymisierten Datum und einer Person nur mit erheblichem Aufwand für die verarbeitende Stelle oder einen Dritten herzustellen ist. Was unter einem „erheblichen Aufwand“ zu verstehen ist, muss im Einzelfall entschieden werden.

Um sich der technischen Herausforderung der Löschung von Daten oder jedenfalls des Personenbezugs zu nähern, bieten Abschnitt CON. 6 „Löschen und Vernichten“ im IT-Grundschutz des Bundesamtes für Sicherheit in der Informationstechnik oder etwa DIN 66399 praktische Anleitungen (siehe ix.de/zy59). Der Schwerpunkt liegt dabei aber auf der Vernichtung im Sinne von sicherer Entsorgung von Datenträgern et cetera, die dann als datenschutzkonform gelöscht gelten.

Unternehmen müssen diesen Vorgaben und weiteren Abschnitten im IT-Grundschutz zufolge bei Bedarf ein Datenlöschkonzept erarbeiten und umsetzen, das auf die individuellen Gegebenheiten ausgerichtet ist. Dies fordert letztlich auch die DSGVO. Dabei folgt sie dem Ansatz, dass Daten verarbeitende Unternehmen Datenschutz eigenverantwortlich sowie gemäß den gesetzlichen Vorgaben umsetzen und dokumentieren müssen.

Hilfreich bei der Bestimmung der Löschfristen verschiedener Datenarten und beim Erstellen eines Löschkonzepts kann auch die DIN 66398 sein. Sie ist keine Norm, sondern eine „Leitlinie zur Entwicklung eines Löschkonzepts mit Ableitung von Löschfristen für personenbezogene Daten“ (siehe Abbildung).

Die Leitlinie DIN 66398 enthält Hilfestellungen für die kniffligen Fragen, die sich Unternehmen im Zusammenhang mit DSGVO-konformem Datenlöschen stellen. *Deutsches Institut für Normung*

Dauerproblem Cloud

Schwierig wird die Datenlöschung oft dann, wenn externe Anbieter im Auftrag Daten verarbeiten oder speichern. Hierzu zählen auch Cloud-Anwendungen. Bei deren Auswahl muss ein

Unternehmen darauf achten, dass die DSGVO-Vorgaben eingehalten werden und eben auch das Löschen personenbezogener Daten sicher und nachhaltig möglich ist. Das Datenschutzrecht unterscheidet bei der Verantwortlichkeit nicht, ob ein Unternehmer selbst oder ein (Cloud-)Dienstleister für ihn personenbezogene Daten verarbeitet.

Recherchen nach Anbietern im Bereich Datenvernichtung und Datenlöschung über Suchmaschinen führen zu einer großen Anzahl an Treffern. Die Auswahl eines solchen Anbieters ist schwierig. Sie muss aber sorgfältig getroffen werden, denn der Verantwortliche kann sich nicht durch Schlamperei oder Fehler eines externen Dienstleisters freizeichnen. Zudem muss ein Auftragsverarbeitungsvertrag nach den Vorgaben der DSGVO abgeschlossen werden, denn der Dienstleister kommt mit personenbezogenen Daten in Berührung. Befindet sich die Cloud außerhalb der EU – etwa in den USA – oder lässt sich dies nicht ausschließen, tauchen weitere datenschutzrechtliche Hürden auf. Jüngst wurde dies durch die EuGH-Entscheidung „Schrems II“ zur Unwirksamkeit des EU-US Privacy Shield deutlich (siehe *iX* 9/2020).

Das Datenschutzrecht ist grundsätzlich selbst bei unverhältnismäßig großem Aufwand für die Einhaltung der Vorgaben zu befolgen. In gewissem Umfang hilft Unternehmen hier, dass auch nach der Einführung der DSGVO anhand objektiver Kriterien geprüft werden kann, ob die Umsetzung von Löschpflichten im Einzelfall verhältnismäßig ist. Ist sie das nicht, besteht unter Umständen die Möglichkeit, personenbezogene Daten zu sperren, statt sie zu löschen.

Die DSGVO spricht hier auch von einer „Einschränkung der Verarbeitung“. Sie erfordert entsprechende technische und organisatorische Maßnahmen. Beispielsweise kann der Datenzugang innerhalb eines Unternehmens mittels Passwort auf wenige Personen beschränkt werden oder die Daten werden in andere Datenbanken übertragen, die gleichfalls nur beschränkt zugänglich sind. Wie stets müssen Lösungen erarbeitet werden,

die den jeweiligen Einzelfall DSGVO-konform abbilden.

Zur Reduzierung des Aufwands kann eventuell auch ein Data Lifecycle Management beitragen, also eine richtlinienbasierte Lösung, die bei Erreichen bestimmter Parameter zu einer automatischen Löschung von Daten führt. Dabei dürfen jedoch keine Fehler bei der Definition der Parameter auftreten – sie könnten zu hohen Folgeschäden führen.

Im Zweifel muss nach DSGVO eine personenbezogene Datenverarbeitung unterbleiben, wenn die Verarbeitung unzulässig oder auch die sichere Löschung unmöglich erscheint. So die Theorie. In der Praxis helfen Orientierungshilfen und andere Veröffentlichungen von Datenschutzbehörden in vielen Fällen dabei, die Vorgaben des Datenschutzrechts einzuhalten. Und oftmals müssen Unternehmer auch die Entscheidung treffen, ein verbleibendes Risiko einzugehen. Bis Einzelfragen gerichtlich geklärt sind, vergehen oft etliche Jahre. In Einzelfällen ist auch ein Spiel auf Zeit denkbar, wenn die Chancen die Risiken überwiegen. Angesichts der signifikant erhöhten Bußgelder kippt dieses Verhältnis aber zunehmend in Richtung eines inakzeptablen Risikos. Das ist vom Gesetzgeber auch durchaus gewollt.

Fazit

Die Frage, wie lange personenbezogene Daten gespeichert werden dürfen oder sogar müssen, muss jedes Unternehmen für sich entscheiden: Zahlreiche Spezialgesetze verlangen eine Speicherung auch über den eigentlichen Zweck hinaus. Hierbei hilft nur ein Ansatz über alle Unternehmensbereiche hinweg, der in ein Datenschutzkonzept mündet. Wann und wie personenbezogene Daten zu löschen oder zu vernichten sind, muss ebenfalls enthalten sein.

Dieses Thema ist für Unternehmen aber nur ein Ausschnitt aus der Gemengelage, wie sie mit Daten allgemein umzugehen haben. Neben personenbezogenen Daten muss auch für andere

„geschäftliche Informationen“ Ort, Dauer und Zweck einer Speicherung festgelegt werden. Auch hier gilt es, deren Löschung zu regeln. Das Gesetz zum Schutz von Geschäftsgeheimnissen verlangt die Erstellung eines Geheimnisschutzkonzepts. Es hilft alles nicht, Unternehmen müssen ihren Umgang mit Daten ganzheitlich angehen, um ihre eigenen Interessen zu vertreten und gesetzlichen Vorgaben zu entsprechen. Dienstleister können hier zwar helfen, die Verantwortung bleibt aber beim Unternehmen, um dessen Daten es geht. (ur@ix.de)

1. Quellen
2. Tobias Haar; Digitalbeleg; Neue Vorgaben zur elektronischen Buchführung; *iX* 3/2020, S. 92
3. Tobias Haar; Weckruf; EU-US Privacy Shield scheitert vor EuGH; *iX* 9/2020, S. 44
4. Die für das Datenlöschen relevanten DIN-Normen und das entsprechende BSI IT-Grundschutzkapitel sowie die IHK-Liste der verschiedenen Datenkategorien sind über ix.de/zy59 zu finden.

Tobias Haar, Rechtsanwalt, LL.M. (Rechtsinformatik), MBA,

ist Rechtsanwalt mit Schwerpunkt IT-Recht bei Vogel & Partner in Karlsruhe.

[/expand]

DSGVO – 11/2020

DSGVO – 11/2020

[expand title="mehr lesen…"]

Wo versteckte personenbezogene Daten lauern

Böse Überraschung

Martin Gerhard Loschwitz

Durch die Löschpflicht der DSGVO können vergessene Datensenken im Unternehmen zu einem echten Problem und verdammt teuer werden. Wo liegen heikle Daten, wie stöbert man sie auf und wie verhält es sich mit WORM-Archiven? iX hilft bei der Spurensuche.

-tract

- Die DSGVO sieht Löschpflichten für personenbezogene Daten vor, die nicht länger benötigt werden.
- Für Administratoren gerät der Versuch, die DSGVO-Vorschriften unter anderem mit Archivierungspflichten in Einklang zu bringen, oft zu einer Gratwanderung.
- Vielen Unternehmen ist gar nicht klar, wo sie überhaupt personenbezogene Daten speichern.
- Wichtig ist es, IT-Verantwortliche für versteckte Datensenken mit personenbezogenen Daten zu sensibilisieren, denn die technischen Lösungen für das Problem sind komplex und in vielen Fällen unbefriedigend.

Man stelle sich im eigenen Unternehmen die folgende Situation vor: Der Drucker druckt nicht. Alles Wackeln am Kabel vermag

das Problem nicht zu beseitigen, und letztlich stellt sich heraus, dass der Kollege Niemeyer sich auf seinem Rechner einen Virus eingefangen hat, der das gesamte System lahmlegt. Weil Herr Niemeyer sich den Virus eingefangen hat, indem er auf dubiosen Seiten unterwegs war, und weil seine Personalakte weitere unschöne Einträge enthält, setzt das Unternehmen ihn vor die Tür.

Ein halbes Jahr später – der Prozess vor dem Arbeitsgericht gegen Herrn Niemeyer ist noch in vollem Gange – flattert ein Brief der Landesdatenschutzbehörde ins Haus. Der Rechtsabteilung zieht es prompt die Schuhe aus: 500000 Euro, so heißt es dort, werden in Form einer Strafe wegen Verstoßes gegen die DSGVO fällig. Die Behörde habe stichhaltige Beweise dafür, dass das Unternehmen es seit Jahren versäume, Daten mit persönlichem Bezug im Sinne der DSGVO zu löschen, wenn diese nicht länger benötigt würden.

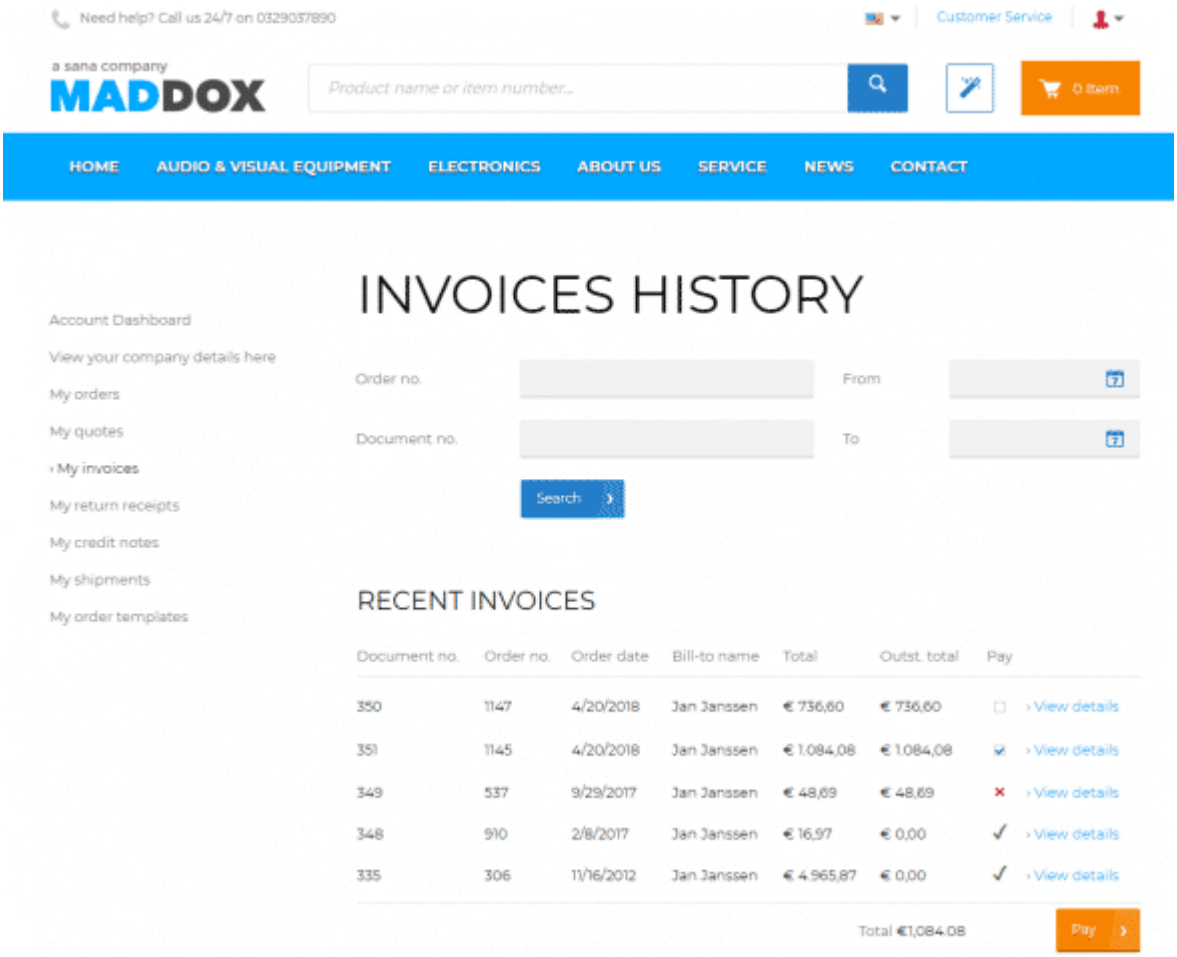
Klingt wie das Drehbuch eines schlechten Films, ist leider aber überhaupt nicht unwahrscheinlich. Die Datenschutzbehörden der Länder gehen mittlerweile alles andere als zimperlich mit Verstößen im DSGVO-Kontext um, wie das Beispiel Deutsche Wohnen eindrücklich bestätigt. Maja Smoltczyck, die Landesdatenschutzbeauftragte von Berlin, will von der Wohnungsgesellschaft 14,5 Millionen Euro Bußgeld eintreiben, und das mit eben diesem Vorwurf. Zwar wehrt sich die Deutsche Wohnen mit Händen und Füßen und wohl bald auch vor Gericht gegen den Bußgeldbescheid. Und in diesem Fall liegen die Dinge auch etwas anders, weil es hier nicht ein ehemaliger Mitarbeiter war, der den Stein ins Rollen brachte, sondern eine Kooperation mehrerer Datenschutzgruppen.

Das Beispiel zeigt jedoch eindrücklich: Die in der DSGVO festgeschriebene Löschpflicht kann für Unternehmen im Fiasko enden, ohne dass diese auch nur den leisesten Verdacht hätten, was da auf sie zurollt. Und Mitarbeiter mit Rachegelüsten, die über unzulässige Datensenken im Sinne der DSGVO im Unternehmen Bescheid wissen, können erheblichen Schaden anrichten.

Welche Optionen sich Firmen bieten

Was aber können Unternehmen tun? Eines sei gleich am Anfang dieses Artikels gesagt – das Ziel besteht an dieser Stelle nicht darin, über die Löschpflicht im Sinne der DSGVO zu debattieren oder im Detail auf sie einzugehen. Wer sich für die juristischen Spitzfindigkeiten interessiert, möge sich die anderen Artikel der Titelstrecke in dieser Ausgabe zu Gemüte führen. Hier geht es vielmehr um die Praxis – um die IT-Abteilung, die sich als Dienstleister für digitale Services im Unternehmen versteht und mit der DSGVO irgendwie umgehen muss. In dieser Position haben Systemverwalter das ausgesprochen unangenehme Problem, zwischen mehreren Stühlen zu sitzen. Denn bei ihrer alltäglichen Arbeit sind sie ja nicht nur den Regeln der DSGVO unterworfen, sondern auch anderer relevanter Gesetzgebung. Das Finanzamt etwa schreibt vor, dass bestimmte Arten von Belegen revisionssicher über Jahre und Jahrzehnte hinweg für spätere Überprüfungen aufzubewahren sind. Damit besteht im Sinne der DSGVO eine „andere Rechtsgrundlage“ für das Speichern von Daten.

Der Teufel steckt allerdings im Detail: Viele Softwarepakete etwa bieten erst gar keine Möglichkeit, nur bestimmte Teile eines Datensatzes zu löschen, andere jedoch zu erhalten. Verlangt ein Kunde etwa die Löschung seines Benutzerzugangs in einem Webshop, muss das Unternehmen dem Ansinnen grundsätzlich nachkommen, darf aber natürlich die Unterlagen behalten, die aufzubewahren es verpflichtet ist. Die meisten Webshops bieten dafür aber keine Funktion: Wer den Zugang löscht, löscht meist auch sämtliche damit verbundenen Dokumente.



Die DSGVO stellt Unternehmen vor allem bei komplexen Anwendungen wie Shopsystemen vor das Problem, Archivierungs- und Löschpflichten unter einen Hut zu bekommen (Abb. 1). *SANA Commerce*

Ähnliches gilt für Daten, die das Unternehmen nach dem WORM-Prinzip (Write Once, Read Many) zu archivieren verpflichtet ist. Hier kommen oft Hardware-Appliances zum Zug, die den gesetzlichen Vorgaben genügen. Vielfach haben sich Firmen bisher mit dem Thema DSGVO aber nicht ausführlich genug beschäftigt und archivieren ihre Daten erst gar nicht ausreichend fein abgestuft, um der DSGVO zu genügen.

Primär soll dieser Artikel Administratoren sensibilisieren und ihnen Anhaltspunkte dafür liefern, wo sich auf klassischen Serversystemen bedenkliche Daten befinden können. Der Text geht auch auf die Frage ein, welchen Aufwand im Sinne des DSGVO-konformen Speicherns von Daten sie betreiben sollen und müssen und wie sich das mit Tools angenehmer gestalten lässt. Vorrangig jedoch geht es darum, wie Administratoren ihre Sinne

schärfen, um potenziell gefährliche Datensenken im Unternehmen zu erkennen und zu eliminieren.

Datensenken

Der Begriff Datensenke stammt ursprünglich aus den Definitionen im Umfeld von Datenübertragungseinrichtungen (siehe ix.de/z9h2). Dort bezeichnet er eine empfangende Datenendeinrichtung.

In der IT gilt ein weiter gefasster Begriff: Einerseits zählt man nicht nur „empfangende“ Dienste wie Mailserver dazu, sondern generell alle Geräte und Dienste, die Daten vorhalten oder ablegen, vom Benutzerverzeichnis über Datenbanken, Geschäftsanwendungen, Shopsysteme bis hin zu smarten Endgeräten, Syslog-Diensten und Systemeinstellungen. In der Praxis bezeichnet man vor allem solche Orte als Datensenken, wo Daten hingesendet werden, um dort zu versacken – also genau das, was im DSGVO-Kontext zu vermeiden ist.

Viele Unternehmen haben zu wenig getan

Viele Unternehmen müssen sich den Vorwurf gefallen lassen, sich mit dem Thema der in der DSGVO verankerten Löschpflicht erst viel zu spät zu beschäftigen. Als die DSGVO in Kraft trat, gab es die heute aus Sicht von Administratoren unangenehmen Paragrafen ja bereits. Ganze vier Jahre hatten Firmen Zeit, sich technisch auf das vorzubereiten, was eine rigoros angewandte DSGVO für sie bedeuten würde. Passiert ist in dieser Richtung vielerorts wenig.

So ist vielen Verantwortlichen heute leider noch immer völlig unklar, wo in ihren Set-ups personenbezogene Daten anfallen. Viele Admins denken bei personenbezogenen Daten im Sinne der DSGVO zudem automatisch an die Daten der Endanwender. Das ist aber eine unvollständige Sicht auf die Dinge: Der bereits erwähnte Herr Niemeyer hat, wenn im Unternehmen gängige Compliance-Regeln zur Anwendung gekommen sind, einen eigenen

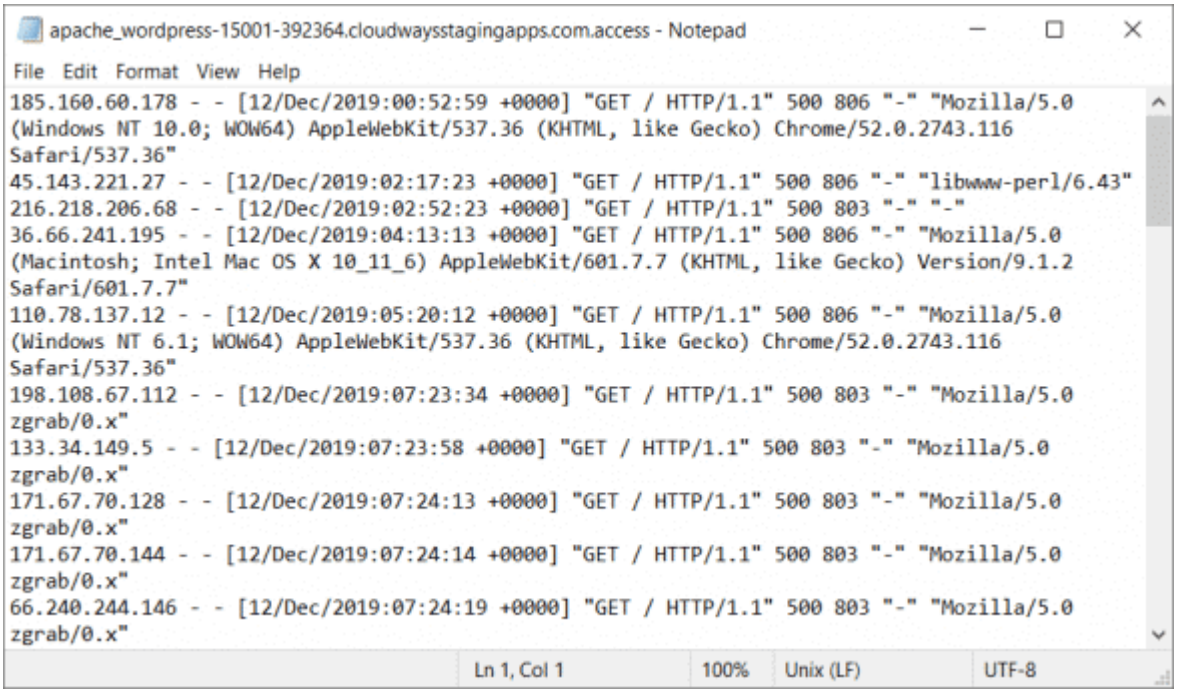
Account gehabt, mit dem er auf den Systemen hantierte.

Personenbezogene Daten sind daher beispielsweise auch die History der Shell, die der nun ehemalige Kollege im Rahmen seiner dienstlichen Tätigkeit genutzt hat. Verlangt er nach einer Einigung vor dem Arbeitsgericht, dass diese Daten entfernt werden, muss die Firma das ebenso tun, wie sie zum Löschen von Kundendaten verpflichtet wäre – freilich unter der einen zentralen Einschränkung, dass jene Daten behalten werden dürfen, zu deren Sicherung die Firma aus anderen Gründen verpflichtet ist.

Dreierlei Werkzeuge im Fokus

Wer vor der Aufgabe steht, Datensenken im eigenen Unternehmen zu finden und zu beseitigen, sieht sich eingangs einer Sisyphusarbeit gegenüber. Denn wenn Hunderte Systeme seit Jahren in Betrieb sind, ist die Wahrscheinlichkeit, dass sich auf diesen personenbezogene Daten finden, ausgesprochen hoch. Grob formuliert gibt es drei Arten von Werkzeugen, mit denen der Admin in diesem Kontext in Berührung kommt.

Da gibt es einerseits die Werkzeuge, die personenbezogene Daten produzieren (etwa die bereits erwähnte Shell in Form ihrer History) oder aufzeichnen (hierzu gehören auch Logdateien aller Art). Dann gibt es die Werkzeuge, deren Aufgabe ja gerade darin besteht, Daten langfristig zu archivieren. Ein solches Werkzeug ist der Firma Deutsche Wohnen zum Verhängnis geworden. Denn die hat Daten wie Gehaltsnachweise, SCHUFA-Auskünfte und ähnliche Unterlagen in ihrem zentralen Archivierungssystem gehabt, lange nachdem die dazugehörenden Mietverträge ausgelaufen waren – ein klarer Verstoß gegen die Löschpflicht, wie Berlins oberste Datenschützerin feststellte.



Ausufernde Webserver-Logs sind keine gute Idee, da auch IP-Adressen zu den personenbezogenen Daten gehören und daher einer Löschpflicht unterliegen (Abb. 2). *Cloudways*

Einfach verzichten können Unternehmen auf diese Geräte aber nicht, denn zum Speichern bestimmter Unterlagen über lange Zeiträume – und zwar revisionssicher – sind sie verpflichtet. Einfache Storage-Systeme reichen hier nicht aus, weil bei bestimmten Datenarten eine nachvollziehbare, nachträglich nicht mehr veränderbare Versionsgeschichte abrufbar sein muss. Dazu kommt spezielle Soft- wie Hardware zum Einsatz, die auf das fein granulierte Speichern von Daten aber nicht ausgelegt ist.

Die dritte Art von Werkzeug sind die Tools, die bei der Suche nach eventuell rechtswidrigen Datensenken unterstützen oder deren Entstehen von vornherein verhindern. Dabei handelt es sich einerseits um klassische Forensikwerkzeuge, mit denen sich versteckte Daten auffinden lassen – andererseits fallen in diese Kategorie aber auch ganz typische Werkzeuge im Kontext der Systemautomation. Die lassen sich möglicherweise auf bestehende Set-ups nur noch schwerlich anwenden, führen vor dem Hintergrund moderner Grundlagen der typischen Systemadministration jedoch dazu, dass Admins das Problem zukünftig umgehen.

Produzenten personenbezogener Daten erkennen

Die erste und oberste Frage bei der Untersuchung einer Umgebung im Hinblick auf personenbezogene Daten ist stets die nach den Datenproduzenten. Welche Programme legen – vor allem unbemerkt – personenbezogene Daten an und speichern sie an Orten, an denen man nicht damit rechnet? Im IT-Kontext der Gegenwart ergeben sich hier leider nahezu beliebig viele Möglichkeiten für Orte, an denen solche Daten anfallen.

Schon ein normales Linux-System bietet eine Vielzahl an Einfallstoren. Stellt man sich etwa einen Host mit Webserver vor, bestehen für die Administratoren vermutlich lokale Benutzeraccounts, über die die Administration erfolgt. Diese beinhalten aller Wahrscheinlichkeit nach zum größten Teil personenbezogene Daten. Teil des Offboardings von Kollegen muss es deshalb sein, auf sämtlichen Systemen etwaige Dateien zu löschen oder rechtskonform zu archivieren, auf die diese Kolleginnen und Kollegen Zugriff hatten. Doch ist es damit noch lange nicht getan, denn personenbezogene Daten fallen auch an anderen Stellen an.

Wer etwa HA-Proxy oder nginx als Load Balancer nutzt, lässt diese vermutlich Logdateien über die Zugriffe anlegen. Ähnliches gilt für Webserver, in deren Logdateien ebenfalls IP-Adressen landen. Der Europäische Gerichtshof hat IP-Adressen – und zwar sowohl dynamische als auch statische – mittlerweile als personenbezogene Daten eingestuft. Damit genießen sie eine besondere Schutzwürdigkeit. Was im Umkehrschluss auch bedeutet: Betreiber von Onlineangeboten dürfen diese Information grundsätzlich nur speichern, wenn damit ein „berechtigtes Interesse“ einhergeht. Die Sicherstellung der Servicequalität sowie die Möglichkeit, im Fehlerfalle Debugging zu betreiben, dürfen wohl als berechtigt wahrgenommen werden.

Wer auf seinen Webservern jedoch Logdateien der vergangenen sechs Monate hat, wird sich mit dem berechtigten Interesse schon schwerer tun: Warum etwa sollte es für den Betrieb eines Webshops von Relevanz sein, ob Ottilie Normalverbraucherin vor drei Monaten eine Bestellung aufgegeben hat, die zwischenzeitlich längst geliefert wurde und somit erledigt ist? Wer nicht schon aus Platzgründen Logrotation betreibt, tut auch im Sinne der DSGVO gut daran, diese zu aktivieren. Analoge Empfehlungen gelten auch für andere Geräte im Netzwerk, etwa Firewalls oder Systeme, die zur Intrusion Detection zum Einsatz kommen.

Datenbanken und Backups: Schatzkiste für Datenschützer

Nahezu jedes zeitgenössische IT-Set-up wird irgendwo eine Datenbank enthalten. Das ist völlig legitim, wenn die gespeicherten Daten nötig sind, um die Geschäftsbeziehung und die beiderseitigen Pflichten zu erfüllen. Das eingangs schon erwähnte Problem der Belege ist mittlerweile vielerorts dringend. Hier treffen zwei Rechtsmaterien aufeinander: die Pflicht der Shopbetreiber, Unterlagen für das Finanzamt aufzubewahren, und die Pflicht, benutzerbezogene Daten zu löschen, wenn sie diese nicht länger benötigen.

Aber: Auf die Datenbank selbst haben Systemverwalter in aller Regel keinen direkten Zugriff. Dieser geschieht oft durch eine Software hindurch, etwa eine Shopsoftware, die die Datenbank im Hintergrund nach eigenem Gutdünken verwaltet. Zwar könnte der Admin händisch an der Datenbank herumpulen – garantieren, dass der Shop danach noch funktioniert, wird dann aber keiner. Die Shopsoftware hingegen bietet die Funktion „Account löschen, aber Belege behalten“ oft einfach nicht. Zwar darf der Admin sich an dieser Stelle mit Fug und Recht aufregen, schließlich hatten die Shopbetreiber seit 2013 Zeit, entsprechende Funktionen einzubauen. Das nicht getan zu haben, grenzt hart an Vorsatz. Doch hat der Admin davon nichts, wenn

ihm die Datenschützer an den Fersen kleben.

Eine befriedigende Lösung gibt es an dieser Stelle leider nicht. Es empfiehlt sich, mit dem Betreiber von Software in Kontakt zu treten und sich über deren abgelegte Daten im Detail zu informieren. Die DSGVO sieht in engen Grenzen Ausnahmen für die Fälle vor, in denen das Löschen dem Anbieter nicht zumutbar ist – dazu später mehr. Möglicherweise lässt sich an der jeweiligen Stelle eine solche Ausnahme zur Anwendung bringen.

Vorsicht ist auch bei Backups geboten. Datenbanken und jede Art von Nutzdaten landen heute zuverlässig in Backups. Systeme mit niedrigem Automatisierungslevel legen zudem oft Logs und andere Nutzdaten in Backups ab. Für diese gilt dasselbe wie für die Daten auf den ursprünglichen Systemen auch: Wer etwa seine Datenbank zwar regelmäßig um nicht länger benötigte Daten erleichtert, diese aber weiterhin im Backup hat, gewinnt in Sachen Datenschutz gar nichts. Theoretisch müssten Backups regelmäßig ebenfalls auf entsprechende Daten untersucht werden, um diese zu löschen. Hier wird der Admin im Normalfall aber nicht mehr ohne juristische Hilfe auskommen, schon um zu erfassen, welche Daten zu löschen sind und welche erhalten bleiben dürfen.

Aufgepasst bei Managementtools

Auch bei Software, die im Büro organisatorischen Zwecken dient, ist erhöhte Vorsicht geboten. Nutzt die Assistenz der Geschäftsführung etwa spezielle Werkzeuge, um Termine der Chefs zu managen – beispielsweise Buchungssoftware für Restaurants –, werden diese schnell eine wahre Goldgrube im Hinblick auf personenbezogene Daten sein. Gerade das ist aber ein hervorragendes Beispiel für Daten mit sehr geringer Halbwertszeit.

Ob man mit der Repräsentantin eines anderen Unternehmens vor zwei Monaten im Café Einstein um 15 Uhr einen Kaffee getrunken

hat oder nicht, mag eine Information sein, die zu speichern legitim ist – etwa aus Gründen der Unternehmensstrategie. In der Mehrzahl der Fälle dürfte ein schneller Kaffee aber nicht von so großem strategischen Interesse sein, dass es legitim wäre, Aufenthaltsorte und -zeiten mehrerer Personen dauerhaft zu speichern. Blöd, wenn solche Daten dann über Jahre und Jahrzehnte erhalten bleiben, weil die Datenbank des Terminverwaltungstools nicht regelmäßig um alte Einträge erleichtert wird. Fällt ein solches Werkzeug bei einer DSGVO-Kontrolle den Prüfern auf, händigt man der Behörde am besten gleich die Zugangsdaten zum Onlinebanking für das Firmenkonto aus, das spart Zeit und Mühe.

Alles noch viel schlimmer: Hardware

Eine Misere, die Administratoren fast gar nicht auf dem Schirm haben, sind Clients und spezielle Geräte für die direkte Nutzung durch Anwender. Wer es etwa extrem praktisch findet, dass sich der Nobel-Kaffeevollautomat im Büro per App steuern lässt und für unterschiedliche Nutzer Geschmacksprofile für den Wunschkaffee anlegen kann, denkt besser noch mal nach. Denn ein Kaffeeprofil, das sich auf dem Gerät einer bestimmten ID oder einer MAC-Adresse zuweisen lässt, ist eindeutig zur jeweiligen Mitarbeiterin oder zum jeweiligen Mitarbeiter zurückverfolgbar. So praktisch das Feature also auch sein mag: Datenschutzrechtlich fährt ein Unternehmen besser, wenn es einen Bogen um derartige Funktionen macht. Denn ganz ehrlich: Wer denkt schon daran, beim Offboarding von Kolleginnen und Kollegen die Kaffeemaschine im Büro auf personenbezogene Daten hin zu untersuchen?

Das Internet of Things klingt verheißungsvoll und bringt Endanwendern nützliche Features, die für Firmen vor dem DSGVO-Hintergrund oft heikel sind (Abb. 3). *Melitta*
Ebenfalls beliebt in diesem Kontext sind Geräte, auf denen einmalig Benutzerzugänge für eine Verbindung zum Hersteller einzurichten sind. Die stellen gerade im Framework für Compliance kleinerer Unternehmen eine Herausforderung dar: Oft legen die Admins, die diese Geräte einrichten, nämlich einen generischen Zugang beim Hersteller an, der etwa ihren codierten Benutzernamen enthält („firmaxyz-maxmeier“). Und selbst wenn der Admin einen generischen Benutzernamen benutzt, kommt oft seine eigene E-Mail-Adresse zum Einsatz. Solche Details lassen sich praktisch nicht mehr auffinden, wenn der Admin das Unternehmen verlässt. Hier liegt es an der Firma, für alle Compliance-Regeln verbindlich vorzugeben, die generische Benutzernamen und die Verwendung von Mailboxen mit generischer Adresse ermöglichen.

Und die Liste lässt sich beliebig fortsetzen. Zeichnet das hauseigene Zugangssystem etwa auf, wann welche Tür geöffnet wird? Kein Problem, werden sich viele Admins denken, solange das Gerät nur aufzeichnet, dass die Tür aufgeht, aber nicht, wer sie öffnet. Anders sieht die Sache allerdings aus, wenn, um Lichter ein- und auszuschalten, im Gebäude auch smarte Bewegungsmelder zum Einsatz kommen, die ebenfalls

Aufzeichnungen führen. Aus der Information, dass um 08:05 Uhr die Türe geöffnet wurde und um 08:08 Uhr im Büro von Frau Müller das Licht anging, lässt sich aber korrelieren, dass diese vermutlich zuvor auch die Tür geöffnet hat – gerade in kleineren Unternehmen und gerade wenn das jeden Tag mit ähnlichen zeitlichen Abständen geschieht. Fans von Big Data geraten ins Schwärmen, Datenschützer hingegen ins Grübeln.

Corona lässt grüßen

Bietet das eigene Unternehmen Zugriff auf den internen Mailserver über das Internet? Gilt in der Firma gar eine BYOD-Philosophie? Wie praktisch! Gerade im Corona-Kontext und unter DSGVO-Aspekten aber alles andere als beruhigend. Denn einerseits gilt: Richtet sich ein Mitarbeiter auf einem privaten Gerät einen Zugang zu seinen Firmenmails ein, fällt dieses sofort unter dasselbe Compliance-Reglement wie alle anderen Geräte der Firma auch. Gerade weil es sich um ein privates Gerät handelt, hat die IT-Abteilung der Firma aber keine Handhabe mehr, zu bestimmen, was mit den Daten passiert. So erstrebenswert der Zugang zur Firmenmail von privaten Geräten aus auch sein mag – aus DSGVO-Sicht verbietet er sich eigentlich.

Dasselbe gilt im Corona-Kontext sogar für die Notebooks, die der Firma gehören und unter deren Compliance-Reglement stehen. Die DSGVO schreibt nämlich explizit vor, dass personenbezogene Daten so zu verarbeiten sind, dass der Zugang durch nicht berechtige Dritte unmöglich ist. Was im Büro noch umsetzbar ist, lässt sich im Homeoffice und innerhalb der eigenen vier Wände kaum rechtssicher implementieren – zumindest dann nicht, wenn kein eigener, abschließbarer Raum zur Verfügung steht. Mal eben die Mails auf dem heimischen Sofa im Wohnzimmer checken scheidet unter diesem Gesichtspunkt eigentlich aus. Dasselbe gilt analog auch für Co-Working-Spaces, in denen der Zugang zum eigenen Arbeitsplatz oft kaum einschränkbar ist.

Unternehmen sind in vielerlei Hinsicht dazu verpflichtet,

bestimmte Daten über einen langen Zeitraum hinweg vorzuhalten. Das ist im DSGVO-Kontext wie beschrieben zunächst kein Problem, weil in diesen Fällen eine andere Rechtsgrundlage besteht. Kommerzielle Lösungen, die für diese Aufgabe zum - Einsatz kommen, verfolgen jedoch das WORM-Prinzip. Sie sind hardware- wie softwareseitig um sämtliche Funktionen erleichtert, die das Modifizieren oder Löschen von Daten ermöglichen. Das wird für viele Unternehmen zum Bumerang, weil sie ihre Archive bisher nach dem Prinzip geführt haben, einfach alles zu archivieren, um sich die mühsame Arbeit des Herausfilterns der tatsächlich benötigten Elemente zu ersparen.

Archivsysteme sind ein Graus

Wer solche WORM-Konstrukte nutzt, wird ad hoc keine technische Lösung finden können, die mit vertretbarem Aufwand auch nur irgendwie realisierbar wäre. Die Deutsche Wohnen führt aus, dass das Gros der im Haus genutzten Software wie beschrieben das Löschen einzelner Dokumente aus Mieterakten aus den eben beschriebenen Gründen nicht beherrscht. Wollte man die Anbieter verpflichten, hier einzelne Teile von Datensätzen aus dem Archiv zu kratzen, müsste die Firma de facto zwei Systeme dieser Art beschäftigen: eines, in dem sie experimentell die Daten löscht, und ein zweites, in das sie dann den gültigen Datensatz überspielt, um wieder den Anforderungen an die eigene Löschpflicht zu genügen.

Archivsysteme wie dieses genügen gesetzlichen Anforderungen und geben WORM-Garantien ab, geraten damit oft aber in Widerspruch zur DSGVO (Abb. 4). *Fujitsu*

Immerhin sieht die DSGVO hier eine Hintertüre vor. Legen Unternehmen überzeugend dar, wieso der zu treibende Aufwand im Spannungsfeld von WORM-Pflichten einerseits und DSGVO-Löschpflicht andererseits zu hoch wäre, haben sie stattdessen die Option, Prozesse zu entwerfen, die verhindern, dass eigentlich zu löschende Daten wieder in Umlauf geraten. Begehren Kunden die Löschung von Datensätzen oder wäre die Löschung nötig, kann man sie dann mit einem entsprechenden Sperrvermerk versehen, der nicht zwingend im selben System hinterlegt sein muss.

Auf Verlangen muss die Firma die entsprechende Prozessdokumentation aber vorlegen können und nachweisen, dass sie in der Praxis tatsächlich zur Anwendung kommt. Und darauf verlassen, dass diese Regelung ewiglich gilt, sollten Unternehmen sich letztlich auch nicht – wer sich des Problems ernsthaft annehmen möchte, muss mit den Entwicklern etwaiger Programme zusammenarbeiten, um Datensätze sinnvoll aufteilbar zu machen. Das bedeutet am Beispiel der Deutsche Wohnen etwa ganz konkret: Die Software für die Verwaltung von Verträgen könnte die ursprünglich eingereichten Dokumente wie Gehaltsnachweise und SCHUFA-Auskünfte von sich aus sofort löschen, sobald ein Mietverhältnis in gegenseitigem Einvernehmen beendet ist. Dass es keine Option ist, das

Problem auszusitzen, zeigt das Beispiel der DW jedenfalls eindrücklich.

Ebbe bei hilfreichen Werkzeugen

Damit ist klar: Moderne IT-Umgebungen strotzen nur so vor Datensenken, die sich für Unternehmen als existenzbedrohend herausstellen können. Da ist es nur verständlich, dass Admins sich Werkzeuge wünschen, die ihnen beim Auffinden der Datentröge helfen und gegebenenfalls Alarm schlagen. Die schlechte Nachricht ist: Solche Tools sind kaum verfügbar. Werkzeuge zur forensischen Analyse beziehen sich eher auf Fälle, in denen ein Einbruch stattgefunden hat und Daten - bereits rausgetragen worden sind. Das ist im DSGVO-Kontext aber nur dann schädlich, wenn die Firma nicht nachweisen kann, dass sie sich an aktuelle technische Standards beim Absichern der eigenen Umgebung gehalten hat.

Dataloss Prevention Tools zäumen das Pferd von hinten auf und sind eher darauf ausgelegt, Prozesse zu installieren, die Datenverlust unwahrscheinlich machen. Das hilft aber nicht in bestehenden Umgebungen, in denen Grundsätze des Datenschutzes über Jahre hinweg nicht zur Anwendung gekommen sind.

Was können Unternehmen also tun? Wie bereits angedeutet, ist die richtige Reaktion auf die DSGVO-Löschpflicht in erster Linie die Einführung passender Prozesse in Kombination mit ausgiebiger Kommunikation mit den eigenen Lieferanten. Standardisierung, Automation und Orchestrierung nehmen viel Angriffsfläche. Überraschenderweise existiert mit der DIN 66398 sogar eine recht praxisorientierte Norm zu diesem Thema (siehe Kasten „Löschkonzepte gemäß DIN 66398“).

Löschkonzepte gemäß DIN 66398

Das DIN hat in der DIN 66398 Empfehlungen als „Leitlinie zur Entwicklung eines Löschkonzepts mit Ableitung von Löschfristen für personenbezogene Daten“ zusammengestellt. Darin finden

sich für das Normungsinstitut ungewöhnlich konkrete Vorgaben zur Vorgehensweise beim Erstellen eines Löschkonzepts.

In einem Blog des Forum-Verlags heißt es, dass demnach Unternehmen eine Reihe von Überlegungen anstellen und Schritte unternehmen sollten (siehe ix.de/z9h2):

- Datenarten, die es im Unternehmen gibt, kategorisieren und deren Aufbewahrungsfristen festlegen.
- Löschregeln für die jeweilige Datengruppe festlegen.
- Konkrete Umsetzungsregeln definieren.
- Die jeweils verantwortlichen Personen für die datenschutzkonforme Umsetzung benennen.
- Prozesse zur Dokumentation ausarbeiten und verbindlich festlegen.

Automation und Orchestrierung

Wer sein Set-up sinnvoll automatisiert und modernen Standards der Systemadministration folgt, minimiert das Risiko von Datensenken auf einzelnen Servern. Überhaupt sollte ein einzelnes System so wenige personenbezogene Daten enthalten wie möglich. Was im Klartext bedeutet: Sämtliche Benutzerzugänge der Systeme kommen aus dem LDAP. Der Prozess des Offboardings sieht Schritte vor, die Daten ehemaliger Kollegen auf Systemen zentralisiert und automatisiert zu entfernen, sofern diese für den Betrieb nicht nötig sind. Logdateien haben in modernen Systemen auf einzelnen Servern nichts mehr zu suchen. Zentralisiertes Logging ist stattdessen das Zauberwort – und ermöglicht es, zentral der DSGVO-Löschpflicht nachzukommen, zum Teil sogar vollständig automatisch.

Was spezielle Software wie Shopsoftware oder Vertragsverwaltungslösungen angeht, hilft nur die enge Kommunikation mit dem Hersteller. Falls der sein Produkt bisher nicht um eine Löschfunktion nach DSGVO-Standards erweitert hat, gilt es, Druck auszuüben, um das zu ändern.

Sich darauf zu verlassen, dass der „Machbarkeitsvorbehalt“ in der DSGVO dauerhaft erhalten bleibt, ist jedenfalls eine fragwürdige Strategie. Hier muss die Software a priori die Möglichkeit bieten, Daten eines Profils zu löschen, ohne dieses gleich vollständig aus dem Bestand zu entfernen.

Den Verheißungen des Internet of Things sollten Unternehmen vom Standpunkt der DSGVO her derzeit widerstehen. Smarte Lichtschalter und schlaue Kaffeemaschinen sind zwar bequem und angenehm. Doch sind sie zentral kaum administrierbar – und sie lassen sich nicht in automatisierte Compliance-Frameworks einbinden und werden so zum potenziellen Datentrog.

Fazit

Es ist eine Krux mit dem Datenschutz: Zu Recht formuliert die DSGVO ein hohes Recht der Menschen an den eigenen Daten, der besonderen Schutzbedürftigkeit dieser Daten und eine Verpflichtung für Daten verarbeitende Unternehmen, hohe Standards zu erfüllen. Für die Admins im Unternehmen geht das aber derzeit mit weitgehend unabwägbaren Risiken einher: In den wenigsten Firmen dürfte wirklich klar sein, wo sich möglicherweise noch alte Bestandsdaten verbergen, die längst den Weg in die ewigen Jagdgründe hätten antreten müssen. Wer jedoch diese Datensätze konsequent suchen und heben möchte, kommt um ein nahezu vollständiges Audit des eigenen Set-ups kaum herum. Und das behebt nur einen Teil der Schwierigkeiten.

Denn verschiedene Softwarelösungen und Archivimplementierungen, ganz gleich, ob in Hard- oder Software umgesetzt, verunmöglichen es, die Löschregeln der DSGVO in Gänze einzuhalten. Hier dürfte zumindest auf absehbare Zeit eine sehr enge Abstimmung mit dem eigenen Rechtsbeistand notwendig sein, um die Grenzen des Erlaubten zu kennen und sich an ihnen entlangzuhangeln. Aber langfristig muss die DSGVO zur Vereinheitlichung und Standardisierung von Set-ups im Rechenzentrum führen, weil sich damit viele Komplikationen im Hinblick auf Datenschutz von vornherein

vermeiden lassen.
(avr@ix.de)

1. Quellen
2. Tobias Haar; Seid sparsam; Löschung personenbezogener Daten; *iX* 11/2020, S. 58
3. Frank Heisel; Ballast abwerfen; Datenlöschen in komplexen Systemlandschaften; *iX* 11/2020, S. 64
4. Weitere Informationen zu Datenempfängern und zur DIN 66398: ix.de/z9h2

Martin Gerhard Loschwitz

ist Cloud Platform Architect bei Drei Austria und beackert dort Themen wie OpenStack, Kubernetes und Ceph.

[/expand]

OpenLDAP-Tutorial I: Konfiguration und Betrieb des freien Verzeichnisdienstes

OpenLDAP-Tutorial I: Konfiguration und Betrieb des freien Verzeichnisdienstes

[expand title="mehr lesen…"]

OpenLDAP-Tutorial I: Konfiguration und Betrieb des freien Verzeichnisdienstes

Schnell nachgeschlagen

Mark Pröhl

Ein sinnvolles Identity Management lässt sich in größeren Umgebungen nicht ohne Verzeichnisdienst realisieren. Neben Microsofts Active Directory ist das freie OpenLDAP einer der bekanntesten Vertreter dieser Gattung. Wie es sich einrichten lässt, zeigt dieser Auftakt eines dreiteiligen Tutorials.

iX-TRACT

OpenLDAP ist eine ernst zu nehmende Open-Source-Alternative zu kommerziellen Verzeichnisdiensten.

LDAP-Verzeichnisse speichern Daten in hierarchisch angeordneten Objekten und Attributen. Das Schema eines Verzeichnisdienstes legt fest, welche Klassen von Objekten und welche Attribute es gibt.

Mit einer einfachen Konfigurationsdatei ist der Server-Daemon *slapd* schnell betriebsbereit.

Zum Durchsuchen und Verwalten des Datenbestands nutzen die OpenLDAP-Kommandozeilenwerkzeuge das LDAP Data Interchange Format (LDIF).

Verzeichnisdienste auf Basis des Lightweight Directory Access Protocol (LDAP v3) spielen eine zentrale Rolle im Identity Management, wo man sie als Quelle für Benutzeridentitäten, Authentifizierungs- und Autorisierungsdaten einsetzt. Prominentes Beispiel für einen weitverbreiteten

Verzeichnisdienst mit diesem Fokus ist Active Directory. Daneben existieren etliche andere kommerzielle sowie freie LDAP-Implementierungen, die häufig universeller einsetzbar sind. Eine davon – OpenLDAP – ist das Thema dieses *iX*-Tutorials.

OpenLDAP ist eines der renommiertesten Open-Source-Projekte und bietet eine ausgereifte und standardkonforme LDAP-Referenzimplementierung. Der Verzeichnisdienst ist Bestandteil aller gängigen Linux-Distributionen und daher sehr weit verbreitet. OpenLDAP spielt durchaus in derselben Liga wie die zahlreichen kommerziellen Implementierungen diverser Hersteller: Die Software bietet gängige Funktionen wie Ausfallsicherheit oder Multimaster-Betrieb und übertrifft manche kommerzielle Konkurrenz in Sachen Performance und Skalierbarkeit (siehe „Onlinequellen“, [a]).

Im ersten Teil geht es um die Grundlagen von LDAP, konkret am Beispiel von OpenLDAP. Das Hauptaugenmerk liegt dabei auf dem Datenmodell, die Konfiguration des Dienstes selbst erfolgt zunächst nur rudimentär. Die folgenden Teile widmen sich dann im Detail der dynamischen Konfiguration, den Sicherheitsaspekten und den Themen Replikation und Synchronisation.

Derzeit liegt OpenLDAP in der Version 2.4.42 vor, Debian, Red Hat/CentOS und die meisten anderen Linux-Distributionen setzen noch ältere Releases ein. Aktuelle Pakete bekommt man distributionsübergreifend vom LDAP Tool Box Project (LTB, [b]). Das *iX*-Tutorial verwendet diese LTB-Pakete, wobei die Beispiele unter CentOS 7 mit *openldap-ltb-2.4.42-1* liefen.

Grundinstallation mit LTB

Listing 1: /etc/openldap/slapd.conf

```
include   /usr/local/openldap/etc/openldap/schema/core.schema
include
```

```
/usr/local/openldap/etc/openldap/schema/cosine.schema
include
/usr/local/openldap/etc/openldap/schema/inetorgperson.schema
pidfile   /usr/local/openldap/var/run/slapd.pid
argsfile  /usr/local/openldap/var/run/slapd.args
database  mdb
suffix    "o=tutorial"
directory /var/lib/ldap/tutorial
rootdn    "cn=root,o=tutorial"
rootpw    "P@ssw0rd"
access    to * by * read
```

Los geht es mit der Hauptkomponente des OpenLDAP-Verzeichnisdienstes, dem Daemon *slapd*. Dessen zeitgemäße Konfiguration und auch deren Sicherheitsaspekte sind Thema der nächsten Tutorialteile, an dieser Stelle belässt der Artikel es bei der sehr einfach gehaltenen *slapd.conf* aus Listing 1. Die verwendet als Namensraum für die LDAP-Daten *o=tutorial*, die *access*-Regel erlaubt anonymes Lesen für jedermann, die Definition eines *rootdn* und dessen Passwort gestattet diesem beliebige Schreibzugriffe. Gerade am Klartextpasswort erkennt man, dass diese Konfiguration so noch völlig unsicher ist und in den folgenden Tutorialteilen überarbeitet werden muss.

Installation und Konfiguration

Listing 3: /etc/yum.repos.d/ltb-project.repo

```
[ltb-project]
name=LTB project packages
baseurl=http://ltb-project.org/rpm/$releasever/$basearch
enabled=1
gpgcheck=1
gpgkey=file:///etc/pki/rpm-gpg/RPM-GPG-KEY-LTB-project
```

Mit folgenden Schritten lässt sich unter CentOS OpenLDAP aus dem LTB-Projekt einrichten. Zuerst erweitert man die Yum-Konfiguration um das LTB-Repository, indem man die in Listing 3 gezeigte Datei */etc/yum.repos.d/ltb-project.repo* anlegt und anschließend die folgenden Kommandos ausführt:

```
# yum update
# rpm --import http://ltb-project.org/wiki/
 lib/RPM-GPG-KEY-LTB-project
```

Dann installiert *yum install openldap-ltb* die benötigten Pakete. Danach ist */etc/default/slapd* wie in Listing 2 zu modifizieren und *slapd.conf* gemäß Listing 1 anzulegen. Jetzt legt man das Verzeichnis an, in dem die Daten für das Tutorial liegen sollen:

```
mkdir -p /var/lib/ldap/tutorial
chown ldap:ldap /var/lib/ldap/tutorial
chmod 0700 /var/lib/ldap/tutorial
```

Die beiden Aufrufe

```
systemctl start slapd.service
chkconfig slapd on
```

starten und aktivieren den Dienst.

Listing 2: /etc/default/slapd

```
IP="127.0.0.1"
PORT="389"
SLAPD_PATH="/usr/local/openldap"
SLAPD_PID_FILE="$SLAPD_PATH/var/run/slapd.pid"
SLAPD_CONF="/etc/openldap/slapd.conf"
SLAPD_CONF_DIR=""
SLAPD_SERVICES="ldap://$IP:$PORT"
SLAPD_PARAMS=""
SLAPD_BIN="$SLAPD_PATH/libexec/slapd"
SLAPD_USER="ldap"
SLAPD_GROUP="ldap"
```

Der Kasten „Installation und Konfiguration“ beschreibt die Schritte für das Einrichten des Dienstes mit LTB. Unter CentOS befindet sich *slapd.conf* im Verzeichnis */etc/openldap*. LTB konfiguriert diesen Pfad über den Parameter *SLAPD_CONF* in */etc/default/slapd* (Listing 2). Als Ergebnis läuft der *slapd* mit folgenden Parametern:

```
slapd -h ldap://127.0.0.1:389 -f
 /etc/openldap/slapd.conf -u ldap -g ldap
```

Der *slapd*-Parameter *–f* definiert den Pfad zur Konfigurationsdatei; *–u* und *–g* definieren den lokalen Benutzer-Account und dessen Gruppe, unter dem *slapd* laufen soll. Diese lauten beide *ldap*. *–h* legt fest, wie – also über welches Netzwerkinterface und welchen Port – sich der Daemon ansprechen lässt. Die korrespondierenden Parameter *IP="127.0.0.1"* sowie *PORT="389"* in Listing 2 führen dazu, dass *slapd* zunächst nur am Loopback-Interface auf den Standard-LDAP-Port 389 hört.

Erste Tests kann man mit dem Kommando *ldapsearch* anstoßen:

```
ldapsearch -H ldap://127.0.0.1
 -x -LLL -b "" -s base +
```

Listing 4: Erste LDAP-Suche gegen den RootDSE

```
# ldapsearch -H ldap://127.0.0.1 -x -LLL -b "" -s base +
dn:
structuralObjectClass: OpenLDAProotDSE
configContext: cn=config
namingContexts: o=tutorial
supportedControl: 1.3.6.1.4.1.4203.1.9.1.1
supportedControl: 2.16.840.1.113730.3.4.18
supportedControl: 2.16.840.1.113730.3.4.2
supportedControl: 1.3.6.1.4.1.4203.1.10.1
supportedControl: 1.3.6.1.1.22
supportedControl: 1.2.840.113556.1.4.319
supportedControl: 1.2.826.0.1.3344810.2.3
supportedControl: 1.3.6.1.1.13.2
supportedControl: 1.3.6.1.1.13.1
supportedControl: 1.3.6.1.1.12
supportedExtension: 1.3.6.1.4.1.4203.1.11.1
supportedExtension: 1.3.6.1.4.1.4203.1.11.3
supportedExtension: 1.3.6.1.1.8
supportedFeatures: 1.3.6.1.1.14
supportedFeatures: 1.3.6.1.4.1.4203.1.5.1
supportedFeatures: 1.3.6.1.4.1.4203.1.5.2
```

```
supportedFeatures: 1.3.6.1.4.1.4203.1.5.3
supportedFeatures: 1.3.6.1.4.1.4203.1.5.4
supportedFeatures: 1.3.6.1.4.1.4203.1.5.5
supportedLDAPVersion: 3
entryDN:
subschemaSubentry: cn=Subschema
```

liefert als Ergebnis den sogenannten RootDSE (Listing 4). Dieser Begriff steht für den DSA-specific Entry, einen Eintrag, in dem LDAP-Server (auch Directory System Agent oder kurz DSA genannt) Informationen für Clients vorhalten. Neben den LDAP-Features, -Controls und -Extensions, die jeweils mit einem Object Identifier (OID) aufgeführt werden, enthält der RootDSE auch die Information, für welche LDAP-Namensräume dieser Server zuständig ist: *namingContexts: o=tutorial*. Auch wo man das Schema abfragen kann, erfährt man hier: *subschemaSubentry: cn=Subschema*.

Eine weitere LDAP-Suche nach den Daten unter *o=tutorial* kann nun mit dem Kommando *ldapsearch –H ldap://127.0.0.1 –x –b o=tutorial* erfolgen. Das Ergebnis wird dabei zunächst *No such object* lauten – denn es existieren zu diesem Zeitpunkt noch keine Nutzdaten im Tutorialverzeichnis.

Daten im LDAP

Läuft der Dienst nach dem Abarbeiten der Installationsschritte gemäß Kasten „Installation und Konfiguration“, sollte man sich als Nächstes mit den dort zu speichernden Daten beschäftigen. Es geht also um die Frage, welche Daten man „im LDAP“ speichern kann und in welcher Form dies geschehen soll. Dazu zunächst etwas Theorie zum Datenmodell, das RFC 4512 [c] im Detail beschreibt.

LDAP-Verzeichnisse speichern ihre Daten in Form von Objekten oder auch Einträgen. Jedes Objekt besteht aus Attributen, denen man Werte zuordnen kann. Objekte gehören sogenannten Objektklassen an. Die legen fest, welche Attribute das Objekt haben kann (MAY-Attribute) und welche es haben muss (MUST-

Attribute). Beispielsweise muss nicht jedes Personenobjekt im LDAP eine Telefonnummer haben, aber Personen ohne Namen sind nicht erlaubt.

Welche Informationen sich in einem Verzeichnisdienst befinden, legt dessen Schema fest. Das definiert genau, welche Attribute und Objektklassen zur Verfügung stehen. Des Weiteren definiert es auch Syntaxregeln für die Attributwerte und Vergleichsoperationen, die bei Suchen im Verzeichnis Verwendung finden.

Grundsätzlich lassen sich in einem LDAP-Verzeichnis beliebige Daten speichern. Man kann also beliebige Dinge durch LDAP-Objekte darstellen und ihre Eigenschaften in passenden Attributen speichern. Ein weitverbreiteter Anwendungsfall von Verzeichnisdiensten besteht aber im Speichern von Daten zu Personen, Accounts und deren Berechtigungen; Letzteres in Form von Rollen oder Gruppen. Die Beispiele im Tutorial konzentrieren sich daher auf dieses Anwendungsszenario.

Namen und Hierarchien

Zum Referenzieren eines Objekts benötigt dieses einen Namen. Dazu ist zunächst der Begriff Relative Distinguished Name (RDN) von Bedeutung. LDAP sieht vor, den RDN eines Objekts aus dessen Attribut-Wert-Paaren zu konstruieren. Der RDN hat dann die Form *attribut=wert*. Üblicherweise bestimmt genau ein Attribut-Wert-Paar den RDN, es ist aber auch möglich, den RDN aus mehreren Paaren zusammenzusetzen. So können RDNs der folgenden Form entstehen: *attribut1=wert1+attribut2=wert2*.

Nun sind LDAP-Objekte hierarchisch in einer Baumstruktur organisiert. So hat jedes Objekt ein Elternobjekt (mit Ausnahme der Wurzel) und kann Kindobjekte haben. Dadurch entsteht der Directory Information Tree, kurz (DIT).

Der RDN alleine referenziert ein Objekt daher nur eindeutig unterhalb eines Elternobjekts. Für ein eindeutiges

Referenzieren eines Objekts im DIT fasst man seinen RDN und die Liste der RDNs seiner Eltern, Großeltern und aller weiteren Vorfahren bis zur Wurzel des DIT zusammen. So entsteht der Distinguished Name oder kurz DN. Der bezeichnet das Objekt eindeutig im gesamten DIT. Ein Objekt für das Tutorial könnte also den folgenden DN haben: *attribut=wert,attribut=wert,o=tutorial*.

Im Detail entsteht diese DIT-Struktur wie folgt: Der Name der Wurzel soll *o=tutorial* lauten. Dabei ist *o* (kurz für *organization*) der Name des Attributs, *tutorial* sein Wert und *o=tutorial* das namengebende Attribut-Wert-Paar, also der RDN der Wurzel.

In der Praxis kann man den DIT an die Hierarchie der betreffenden Organisation anlehnen und wie hier den Namen des Wurzelobjekts dem Organisationsnamen entnehmen. Dabei ist aber die Eindeutigkeit dieses Namens nicht garantiert. Ein anderer Weg, das Wurzelobjekt zu benennen, ist ebenfalls sehr üblich: Man lehnt es einfach an den DNS-Namen an. LDAP sieht dafür das Attribut *dc* (kurz für *domainComponent*) vor. Lautet der DNS-Name *tutorial.org*, wäre *dc=tutorial,dc=org* ebenfalls ein passender Name für die Wurzel des DIT dieses Tutorials.

Auch unterhalb der Wurzel lässt sich der LDAP-Baum nach verschiedenen Gesichtspunkten strukturieren: beispielsweise kann man organisatorische oder funktionale Aspekte heranziehen. Für die Verzweigungen verwendet man in der Praxis gerne Objekte der Klasse *organizationalUnit* und benennt diese mit dem *ou*-Attribut. Für organisatorische Einheiten einer Firma, wie Vertrieb oder Personalabteilung, wären also Zweige mit den RDNs *ou=sales* oder *ou=hr* durchaus üblich. Beim alternativen Strukturieren nach funktionalen Gesichtspunkten verwendet man ebenfalls *ou*. So ist es üblich, Mitarbeitern Benutzerobjekte unterhalb eines Zweigs mit dem Namen *ou=people* zu geben und Gruppen in *ou=groups* abzulegen.

Bei einer Mischung von organisatorischer und funktionaler

Strukturierung erhält man Teilbäume wie *ou=people,ou=marketing,o=tutorial*. So entsteht schnell eine tiefere Baumstruktur, die der Übersichtlichkeit durchaus dienen kann. In der Praxis hat es sich aber als sinnvoll erwiesen, nicht unnötig zu verzweigen, sondern den DIT möglichst flach zu gestalten. Gibt es also keinen zwingenden Grund für eine tiefere Verzweigung, reichen *ou=people,o=tutorial* und *ou=groups,o=tutorial* völlig aus.

DIT-Struktur für das Tutorial

Listing 5: Die Struktur des Tutorial-DIT

```
dn: o=tutorial
changetype: add
objectclass: top
objectclass: organization
o: tutorial

dn: ou=people,o=tutorial
changetype: add
objectclass: top
objectclass: organizationalUnit
ou: people

dn: ou=groups,o=tutorial
changetype: add
objectclass: top
objectclass: organizationalUnit
ou: groups

dn: ou=admins,o=tutorial
changetype: add
objectclass: top
objectclass: organizationalUnit
ou: admins
```

Neben *ou=people* für Personenobjekte wird das Tutorial auch noch Gruppen und LDAP-Administratoren unterhalb von *ou=groups* und *ou=admins* verwalten. Listing 5 beschreibt diese einfache

Struktur. Neben den drei *ou*-Objekten ist dort auch das Wurzelobjekt *o=tutorial* selbst enthalten.

Das Listing verwendet zum Beschreiben das sogenannte LDAP Data Interchange Format (LDIF, [d]), ein einfaches 7-Bit-ASCII-Format, mit dem sich LDAP-Objekte und Operationen an ihnen beschreiben lassen. LDIF-Dateien enthalten durch Leerzeilen getrennte Textblöcke, die jeweils ein Objekt oder Operationen daran beschreiben. Die erste Zeile pro Block enthält den DN, die zweite kann die Operation (*changetype*) enthalten. Erlaubte Werte sind hier beispielsweise *add*, *delete* oder *modify*.

Speichert man diese Beschreibung in einer Datei mit dem Namen *listing-5.ldif*, lässt sich der Inhalt mit folgendem Kommando dem DIT hinzufügen:

```
ldapadd -x -H ldap://127.0.0.1 -D cn=root,
 o=tutorial -w P@ssw0rd -f listing-5.ldif
```

Damit steht die Struktur des DIT fest.

Weitere Nutzdaten in den Verzeichnisbaum integrieren

Listing 6: Zwei Personenobjekte

```
dn: cn=Max Mustermann,ou=people,o=tutorial
changetype: add
objectclass: top
objectclass: person
cn: Max Mustermann
sn: Mustermann

dn: cn=Erika Musterfrau,ou=people,o=tutorial
changetype: add
objectclass: top
objectclass: person
cn: Erika Musterfrau
sn: Musterfrau
```

Traditionell dienen LDAP-Verzeichnisse zum Erfassen von Personendaten. Dazu gehören Attribute wie Vor- und Nachname, Geburtstag, aber auch Anschrift und Telefonnummer, E-Mail-Adresse et cetera. Listing 6 enthält zwei Personenobjekte. Die Beispiele sind bewusst einfach gehalten und sollen im Folgenden noch erweitert werden. Beide Personen gehören der Objektklasse *person* an. Für solche Objekte sieht LDAP hauptsächlich Namensattribute vor: Nachname (*sn*, kurz für *surname*) und *cn* (*commonName*). Darüber hinaus könnten sie über ein Passwort (*userPassword*) oder eine Telefonnummer (*telephoneNumber*) verfügen, aber nur *sn* und *cn* sind zwingend erforderlich. Das vorliegende Beispiel verwendet jeweils *cn* als namengebendes Attribut im RDN; *sn* wäre ebenfalls möglich, wenn auch unüblich.

Diese Beschreibung speichert man in einer Datei mit dem Namen *listing-6.ldif* und fügt sie wie zuvor per *ldapadd* hinzu. Damit läuft der Verzeichnisdienst und bietet bereits eine einfache Baumstruktur mit zwei Personenobjekten an. Das genügt schon, um sich ein wenig mehr mit den Suchmöglichkeiten zu beschäftigen, die LDAP bietet.

Anatomie einer LDAP-Suche

Listing 7: Eine rekursive LDAP-Suche

```
# ldapsearch -H ldap://127.0.0.1 -b o=tutorial -s sub -x -LLL
'(objectClass=*)' cn

 dn: o=tutorial

 dn: ou=people,o=tutorial

 dn: cn=Max Mustermann,ou=people,o=tutorial
 cn: Max Mustermann

 dn: cn=Erika Musterfrau,ou=people,o=tutorial
 cn: Erika Musterfrau
```

```
dn: ou=groups,o=tutorial

dn: ou=admins,o=tutorial
```

Listing 7 stellt eine typische LDAP-Suchoperation mit dem OpenLDAP-Werkzeug *ldapsearch* vor: *–H* gibt die LDAP-URI, also im Wesentlichen den Hostnamen oder die IP-Adresse des Servers an, *–b* die Suchbasis. Die Beispielsuche erfolgt also ab dem Wurzelobjekt *o=tutorial* – man könnte die Suche über *–b* auch auf einen Teilbaum einschränken. Der Parameter *–s* gibt den Scope der Suchoperation vor, mögliche Werte sind *sub* (durchsucht den DIT rekursiv unterhalb der angegebenen Suchbasis), *one* (findet nur direkte Kindobjekte der Suchbasis) und *base* (liefert nur das Basisobjekt selbst als Suchergebnis).

–x dient der Authentifizierung per sogenanntem Simple Bind, bei dem der LDAP-Client einen Namen (den Bind-DN) und das zugehörige Passwort benötigt. Mit *–D* könnte man den Bind-DN angeben, mit *–w* dessen Passwort. Anstelle von *–w* akzeptieren die OpenLDAP-Werkzeuge das Passwort auch interaktiv (über *–W*) oder aus einer Passwortdatei (mit *–y*). Ohne diese zusätzlichen Parameter führt *–x* zu einem anonymen Simple Bind. Der Parameter *–LLL* ist reine Kosmetik: Er reduziert die Menge der ausgegebenen LDIF-Daten etwas.

Des Weiteren enthält die Kommandozeile aus Listing 7 einen Suchfilter und eine Liste der gesuchten Attribute. *‚(objectClass=*)‘* ist der einfachste denkbare Suchfilter, der auf alle Objekte im DIT passt. Weil *cn* das einzige Suchattribut auf der Kommandozeile ist, ist auch nur dieses im Suchergebnis enthalten. Der Bind-DN *cn=root,o=tutorial* kommt in der Ausgabe nicht vor, denn den gibt es nicht im DIT, sondern lediglich als Konfigurationsparameter.

So liefert *ldapsearch* einen Überblick über die Objekte und auch ihre Anordnung im DIT. Für den Einstieg in LDAP empfiehlt sich die Kommandozeile. Sie soll hier im Tutorial

ausschließlich zum Einsatz kommen. Möchte man im täglichen Betrieb mehr Komfort, greift man zu einem der grafischen LDAP-Browser, wie Apache Directory Studio, Softeras LDAP Administrator, dem Userbooster von maxcrc oder dem webbasierten Web2LDAP.

Weitere LDAP-Operationen

Beide Personenobjekte besitzen bislang nur die nötigsten Attribute. Es wäre sicherlich nützlich, wenn man ihre Kontaktinformationen und Adressdaten ebenfalls speichern könnte. Sollen mit diesen Personenobjekten auch Betriebssystem-Accounts, beispielsweise für Linux, verbunden sein, kämen sogenannte POSIX-Attribute dazu. LDAP bietet nicht nur die Möglichkeit, zu suchen oder ganze Objekte mit der Add-Operation (über *ldapadd*) hinzuzufügen. Es lassen sich auch Objekte anpassen. Die Modify-Operation erlaubt das Hinzufügen, Ändern und Entfernen einzelner Attribute eines Objekts. Beschreibt man solche Modifikationen in einer LDIF-Datei, kann man diese unter OpenLDAP mit dem Werkzeug *ldapmodify* anwenden.

Listing 8: LDIF-Datei für LDAP-Modify-Operation

```
dn: cn=Max Mustermann,ou=people,o=tutorial
changetype: modify
add: telephoneNumber
telephoneNumber: +49 1234 5678
```

Herr Mustermann soll nun eine Telefonnummer erhalten. LDAP sieht hierfür das Attribut *telephoneNumber* vor. Der fiktive Wert soll *+49 1234 5678* lauten, Listing 8 zeigt die passende LDIF-Datei. Die MODIFY-Operation führt das folgende Kommando durch:

```
ldapmodify -x -H ldap://127.0.0.1 -D cn=root,
 o=tutorial -w P@ssw0rd -f listing-8.ldif
```

Hat das geklappt, sollen als Nächstes eine E-Mail-Adresse (Attribut *mail*) sowie eine Ortsangabe zu Herrn Mustermann im LDAP gespeichert werden. Dabei ist dem Administrator unklar,

welches Attribut sich für den Ort (englisch: „Locality“) eignet. Aus Listing 4 entnimmt man, dass das Schema des OpenLDAP-Servers unter der Suchbasis *cn=subschema* per LDAP abfragbar ist. Diese Funktion erweist sich bei der Suche nach geeigneten Attributen als äußerst praktisch. Ein *ldapsearch* mit den Parametern *–b cn=subschema –s base* + listet auch sämtliche Attributdefinitionen. Durchforscht man diese nach dem String „locality“, findet man die folgende Definition:

```
attributeTypes: ( 2.5.4.7
NAME ( 'l' 'localityName' )
DESC 'RFC2256: locality which this
 object resides in'
SUP name )
```

Das passende Attribut für den Ort lautet also einfach *l*. Der Versuch, eine zu Listing 8 analoge LDIF-Datei mit *l: Musterstadt* zu erzeugen und via *ldapmodify* anzuwenden, scheitert allerdings. Das Tool quittiert den Versuch mit den Meldungen „Object class violation“ und „attribute ‚l‘ not allowed“. Das liegt schlichtweg daran, dass die in Listing 6 für Herrn Mustermann verwendete Objektklasse *person* dieses Attribut nicht vorsieht.

Listing 9: Auslesen des Schemas des LDAP-Servers

```
# ldapsearch -H ldap://127.0.0.1 -x -LLL -b cn=subschemas base
+
[...]
objectClasses: ( 2.5.6.6
 NAME 'person'
 DESC 'RFC2256: a person'
 SUP top
 STRUCTURAL
 MUST ( sn $ cn )
 MAY ( userPassword $ telephoneNumber $ seeAlso $ description
) )

objectClasses: ( 2.5.6.7
 NAME 'organizationalPerson'
```

```
 DESC 'RFC2256: an organizational person'
 SUP person
 STRUCTURAL
  MAY ( title $ x121Address $ registeredAddress $
destinationIndicator $
    preferredDeliveryMethod $ telexNumber $
teletexTerminalIdentifier $
   telephoneNumber $ internationaliSDNNumber $
facsimileTelephoneNumber $
 street $ postOfficeBox $ postalCode $ postalAddress $
 physicalDeliveryOfficeName $ ou $ st $ l ) )

objectClasses: ( 2.16.840.1.113730.3.2.2
 NAME 'inetOrgPerson'
 DESC 'RFC2798: Internet Organizational Person'
 SUP organizationalPerson
 STRUCTURAL
  MAY ( audio $ businessCategory $ carLicense $
departmentNumber $ displayName $
  employeeNumber $ employeeType $ givenName $ homePhone $
homePostalAddress $
 initials $ jpegPhoto $ labeledURI $ mail $ manager $ mobile $
o $ pager $
 photo $ roomNumber $ secretary $ uid $ userCertificate $
x500uniqueIdentifier $
 preferredLanguage $ userSMIMECertificate $ userPKCS12 ) )
```

Durchsucht man das Schema nach weiteren Objektklassen, so findet man neben etlichen anderen auch die in Listing 9 dargestellten. Bei der Definition der Objektklasse *organizationalPerson* findet man in den unter dem Strichwort MAY gelisteten Attributen auch *l* – Objekte dieser Klasse können also einen Ort haben. *SUP person* bedeutet, dass *organizationalPerson* von der Klasse *person* abgeleitet ist. Daher erbt es deren Eigenschaften und kann diese um weitere (wie *l*) ergänzen.

Mit dem in der Definition enthaltenen Schlüsselwort *STRUCTURAL* hat es Folgendes auf sich: LDAP unterscheidet verschiedene Typen von Klassen: Die Klasse *top*, von der sich alle anderen

ableiten, ist vom Typ „abstract“, die meisten anderen Klassen sind entweder „structural“ oder „auxiliary“. Ein Objekt kann nur solchen strukturellen Objektklassen angehören, die auseinander oder aus der Klasse *top* abgeleitet wurden. *organizationalPerson* ist vom Typ „structural“, ebenso wie die Klasse *person* aus der sie abgeleitet wurde. Die strukturelle Klasse eines Objektes lässt sich per LDAP-Modify-Operation nachträglich nicht mehr ändern. Die bisherigen Benutzereinträge sind daher zu löschen und neu anzulegen. Vorher sollte man sich aber genau überlegen, welche strukturellen Objektklassen zum Einsatz kommen sollen. Spätestens wenn es um das Speichern der E-Mail-Adresse *m.mustermann@tutorial.org* geht, zeigt sich nämlich, dass *organizationalPerson* auch noch nicht der Weisheit letzter Schluss ist. Wie man Listing 9 entnehmen kann, fehlt in dessen Definition nämlich das Attribut *mail*. Dieses findet sich aber in der Liste der MAY-Attribute der Klasse *inetOrgPerson*.

Listing 10: Vollständiges Objekt für Herrn Mustermann

```
dn: cn=Max Mustermann,ou=people,o=tutorial
changetype: delete

dn: cn=Max Mustermann,ou=people,o=tutorial
changetype: add
objectclass: top
objectclass: person
objectclass: organizationalPerson
objectclass: inetOrgPerson
cn: Max Mustermann
sn: Mustermann
givenName: Max
initials: MM
displayName: Max Mustermann
description: Eine Beispielperson
jpegPhoto::
/9j/4AAQSkZJRgABAQEASABIAAD/2wBDABUOEBIQDRUSERIYFhUZHzQiHx0dH0
AuMCY
```

```
0TENQT0tDSUhUXnlmVFlyWkhJaY9qcnyAh4iHUWWUn5ODnXmEh4L/wgALCAAcA
BYBAREA/8QAFgABA
QEAAAAAAAAAAAAAAAABAUD/9oACAEBAAAAAZ9cRq2EmyQaMf/EAB8QAAICAQQ
DAAAAAAAAAAAAAAE
CAwQAERMhMiIxQf/aAAgBAQABBQKMAvtRAWEVcT1F5xWuv2t1sMC+mI5WMDn/x
AAcEAACAgIDAAAAA
AAAAAAAAAAAAQIQIUEREiL/2gAIAQEABj8C9GUJwGJ7RHNYkdVqnxuv/8QAHRA
BAAMAAwADAAAAAAAA
AAAAAAQARITFBUWGBkf/aAAgBAQABPyGhLL9nHX7NfA47HWp9kKsnQ1LhNm4Op
Zpxc3DCWHX5ibYGu
ra9w/U//9oACAEBAAAAECev/8QAGhABAQEBAQEBAAAAAAAAAAAAAAAREAIUExUf/
aAAgBAQABPxA4mvU
aAiH1WF6xCnV6J2lbzLC+uDWccigv5nftdVLVLvun7q7xHgJ5mJVM/d//Z
mail: m.mustermann@tutorial.org
telephoneNumber: +49 1234 5678
street: Musterstr. 21
st:: TXVzdGVybMOkbmRsZQo=m
l: Musterstadt
```

Ein entsprechend vervollständigtes Objekt für Herrn Mustermann mit einer Reihe weiterer praktischer Attribute zeigt das Listing 10. Das Objekt muss man wegen der Änderung der strukturellen Objektklasse löschen und neu anlegen. Es sollte sich nun problemlos mit *ldapmodify* anwenden lassen. Die Namen der Attribute sind selbsterklärend, interessanter ist hier die LDIF-Syntax: Im einfachsten Fall besteht eine Zeile aus einem Attributnamen, gefolgt von einem „*:*“ und dem zugehörigen Wert. „*::*“ bedeutet, dass der Wert Base64-kodiert ist, da LDIF nur 7-Bit-ASCII-Zeichen enthalten darf. Im Beispiel enthält der Wert des Attributes *st* (kurz für *stateOrProvinceName*) aber einen Umlaut; ebenso ist das binäre JPEG-Foto in Base64 einzutragen. Umbrüche sind in LDIF innerhalb eines Wertes ebenfalls erlaubt, dazu rückt man die Folgezeile mit einem Leerzeichen ein, wie das Beispiel zeigt.

Listing 11: LDIF-Datei zum Ändern mehrerer Attribute

```
dn: cn=Max Mustermann,ou=people,o=tutorial
```

```
changetype: modify
replace: street
street: Musterstr. 21
-
delete: jpegPhoto
-
add: userPassword
userPassword: SSHAs+o/41d+BW9fQUuQRzf9RtqpcMEDhB78
```

Mit LDIF lassen sich auch Attribute ändern: Es sei beispielsweise die Hausnummer 21 im Attribut *street* falsch und soll nach 12 geändert werden. Das Foto ist zu groß und soll gelöscht werden. Herr Mustermann soll LDAP auch für die Authentifizierung nutzen und benötigt daher ein Passwort. All diese Änderungen lassen sich wie in Listing 11 dargestellt anwenden. Den Hash für das *userPassword* erzeugt man mit dem Kommando *slappasswd*.

Fazit

Dieser erste Teil des Tutorials hat die Grundlagen von LDAP insbesondere im Hinblick auf das Datenmodell beleuchtet. Die DIT-Struktur für das Tutorial besteht aus einer Organisation (*o=tutorial*) und einer Reihe darunter angeordneter Organisationseinheiten. In der *ou=people* liegen bereits zwei Personenobjekte. Über Suchoperationen können Administratoren und Anwender nach Informationen in diesen Daten zu suchen. Zum Abfragen von Server-Eigenschaften wie dem Schema haben sich LDAP-Suchoperationen ebenfalls als nützlich erwiesen.

Mit LDIF-Dateien und den OpenLDAP-Werkzeugen lassen sich verschiedene Operationen am LDAP-Datenbestand durchführen. Bislang ist die Konfiguration des *slapd* noch sehr rudimentär und bedarf dringend einer Überarbeitung. Teil 2 des Tutorials vertieft daher die *slapd*-Konfiguration im Hinblick auf dynamische das Konfigurations-Backend. Dort werden die in Teil 1 erlernten LDAP-Operationen dann direkt auf Konfigurationsobjekte angewandt. (avr) Mark Pröhl ist Senior Solutions Architect bei der Tübinger Atos-Tochter science +

computing AG.

Schnell nachgeschlagen

- OpenLDAP
  1
- LDAP Tool Box Project
  2
- RFC-4512
  3
- RFC-2849
  4

[/expand]

Firewalls und Proxies von Anwendern ausgetrickst

Firewalls und Proxies von Anwendern ausgetrickst

[expand title="mehr lesen…"]

Firewalls und Proxies von Anwendern

ausgetrickst

Durchgegraben

Thomas Ronzon

Nicht nur Netzwerkprofis kennen Firewalls. Inzwischen wissen auch Anwender, wie sie die vermeintlich sichere Firmenmauer untergraben können.

Zu den Aufgaben von Firewalls gehört es, schädliche Dateien aus dem Netz fernzuhalten. Gerade hier kollidieren die Interessen von Administratoren und Anwendern am heftigsten. Die einen suchen nach Wegen, unwillkommene Exemplare zuverlässig herauszufiltern, die anderen danach, die daraus resultierenden Einschränkungen zu umgehen.

Da Schädlinge ihre Reise ins Firmennetz gern per E-Mail antreten, gehört das Prüfen von Mail-Anhängen zum Tagesgeschäft. Prüft der Mailserver jedoch nur die Dateiendungen, muss der Absender etwa ein gepacktes Archiv *datei.zip* nur mit dem Namen *datei.txt* versehen. Der Empfänger braucht die Datei lediglich wieder umzubenennen.

Gerade Administratoren mit Windows-Hintergrund scheinen für diesen Fehler anfällig zu sein. Deutlich zuverlässiger lässt sich erkennen, dass es sich etwa um ein mit dem ZIP-Algorithmus gepacktes Archiv handelt, wenn man den MIME-Type (Multipurpose Internet Mail Extensions) – quasi deren Fingerabdruck – bestimmt.

Ähnlich arbeiten Shell-Skripte auf unixoiden Betriebssystemen unter Zuhilfenahme des Befehls *file*, der den Dateityp anhand sogenannter Magic Patterns bestimmt. Auch für andere Sprachen stehen solche Werkzeuge bereit. Java-Entwickler etwa können auf das Java-Paket SimpleMagic zurückgreifen. In der Windows-Welt dagegen sind derartige Befehle unbekannt.

Doch auch die Prüfung des MIME-Typs lässt sich ganz einfach umgehen. Es genügt, die nicht druckbaren Zeichen einer binären Datei mit dem Befehl *uuencode datei1.zip datei1.zip > datei1.uue* oder einem grafischen Derivat für Windows in druckbare ASCII-Zeichen zu wandeln und mit *uudecode datei.uue* wiederherzustellen. Mit dieser Technik übertrug man in den Kindertagen des Internets binäre Dateien per E-Mail, da Mails nur eine 7-Bit-Kodierung besaßen. *uuencode* teilt drei Bytes der Binärdatei auf vier 6-Bit-Werte auf und ordnet ihnen druckbare ASCII-Zeichen zu.

Ähnlich wie Filter von E-Mail-Anhängen verhalten sich Proxies, wenn Anwender *.exe*-Dateien per Browser herunterladen wollen. Das Download-Ziel von *www.meinedomain.de/tool.exe* erkennen viele Proxies anhand der Endung als ausführbare Datei und sperren sie, nicht dagegen *www.meinedomain.de/tool.exe?*. Diese URL gaukelt Proxies, die nur die letzten drei Zeichen prüfen, die Endung *xe?* vor, während der Webserver das *?* als regulären Ausdruck erkennt, der für ein beliebiges, also auch kein Zeichen stehen kann. In der Regel verwirft er das Zeichen einfach und liefert die richtige Datei aus.

Umbenennen und teilen

Ein anderes Mittel vieler Administratoren besteht darin, dass sie nur Mails bis zu einer bestimmten Größe durchlassen. Damit wollen sie erreichen, dass große und damit in den Augen mancher Administratoren gefährliche Dateien nicht ins eigene Netz gelangen. In Zeiten teuren Speicherplatzes war dies sicherlich auch ein Mittel, um der Verschwendung von Ressourcen vorzubeugen.

Bei Anwendern kursieren deshalb althergebrachte Programme wie HJSplit, das bis in die 16-Bit-Zeit zurückreicht. Grafische Varianten sind für sämtliche Windows-Versionen, Linux, Mac OS und BSD verfügbar. Versierte Linux-Anwender zerlegen die Datei schlicht mit *split –b <size> daten.xls split-daten.xls*, wobei *<size>* die Größe der Teildateien in Byte angibt. Mit *cat*

*split-daten.xls.** > *daten.xls* setzen sie die Datei wieder zusammen.

So gut wie jeder Mitarbeiter hat neben seinem beruflichen Postfach einen privaten Account. Die Regeln dort sind meinst weniger restriktiv als bei Firmen-Accounts. Ein beliebtes Mittel, Dateien zu übertragen, ist das Senden an private Accounts. Der Kollege muss nun nur noch die E-Mail abholen und den Inhalt auf seinem Rechner speichern.

Theoretisch ließe sich der Zugang zum Webmailer sperren – aber: Webmailer gibt es wie Sand am Meer. Beliebt ist auch, die E-Mail mit dem Smartphone abzurufen. Dazu verbindet der Smartphone-Besitzer einen Rechner per Tethering, also über USB, WLAN oder Bluetooth, mit seinem Gerät und nutzt es als Hotspot.

Von Rechner zu Rechner

In vielen Firmen gehört das Sperren der USB-Ports zu den Standardmaßnahmen, vor allem, wenn sie nicht nur verhindern wollen, dass Schadsoftware auf den PC gelangt, sondern auch, dass Daten das Firmennetz verlassen. Auch das lässt sich leicht aushebeln.

Ein Anwender muss dazu nur auf einem anderen Rechner einen VNC-Server starten. Ist dort der File-Transfer aktiviert, kann er von seinem PC per VNC-Client darauf zugreifen. Noch brisanter ist die Lage bei Werkzeugen wie TeamViewer, da sie die Dateien nicht direkt an den zweiten Rechner übertragen, sondern auf einem Server zwischenspeichern. Das geschieht zwar verschlüsselt, bietet jedoch Angriffsflächen.

Noch einfacher ist der Weg über Filehoster. Auch hier stellt das Ablegen der Daten auf Servern des jeweiligen Dienstleisters wie Dropbox, Hightail oder Skype ein nicht zu unterschätzendes Sicherheitsrisiko dar. Das Sperren der Domains hilft wenig, da es mittlerweile so viele Filehoster

gibt, dass es einem ewigen Katz-und-Maus-Spiel gleichen würde.

Durch einen Tunnel zum SSH-Server und von dort weiter zum Webserver – auf diese Weise können versierte Anwender etwa die Sperren zu bestimmten Websites unterlaufen.
Weitere Optionen, firmenseitige Restriktionen zu umgehen, stehen fortgeschrittenen Anwendern offen. Eine davon ist das Tunneln, da für die Firewall nur das Tunnelprotokoll sichtbar ist. Ist etwa der Zugriff auf *sshserver* per SSH erlaubt, möchte der Anwender aber auf einen Webserver zugreifen, tunnelt er die HTTP-Anfrage per SSH. Windows-Benutzer greifen hier meist zu *putty*, Linux-Anwender brauchen auf ihrem Client nur die Datei *~/.ssh/config* anzulegen und mit dem folgenden Inhalt zu füllen:

```
Host <sshserver>
    HostName <sshserver.domain.de>
    User <user>
    LocalForward 20000
 <webserver>:80
```

Das verkürzt den Befehl *ssh –L 20000:<webserver>:80 <sshserver.domai.de>* zum Öffnen des Tunnels auf *ssh <sshserver>*. Eine Browseranfrage an *localhost* Port 20000 mit *http://localhost:20000* tunnelt die SSH einfach bis zum *sshserver* und leitet sie von dort an den Webserver Port 80 weiter (siehe Abbildung).

Alternativ kann man auf dem eigenen Arbeitsplatz einen Webserver starten. Windows-Benutzer haben etwa mit Abyss, HFS (HTTP File Server), jibble oder dem Mini-Webserver 2010 einige schlanke Apache-Alternativen zur Auswahl, Linux-Anwender können aus dem Vollen schöpfen. Eine Ad-hoc-Variante in Form eines Bash-Kommandos hat der Entwickler Răzvan Tudorică vor einigen Jahren ins Netz gestellt:

```
while true; do { echo -e
 'HTTP/1.1 200 OK\r\n';
 cat </dir/datei>; } |
```

```
nc -l 8888; done
```

Führt ein Anwender das Kommando auf seinem Linux-System <*linuxhost*> aus – so *netcat* vorhanden ist –, kann jeder andere, für den <*linuxhost*> erreichbar ist, mit seinem Browser über die URL *http://linuxhost:8888* auf die Datei zugreifen. Da *netcat* oder *nc* die Informationen zum Zugriff auf *stdout* ausgibt, kann der Anwender dort sehen, wann er die *while*-Schleife beenden kann.

Fazit

Ein Administrator verfügt über die Mittel, den gesamten Datenverkehr zu reglementieren und das Firmennetz abzuschotten. Ist der Betrieb von der Kommunikation mit anderen abhängig, etwa von der Zusammenarbeit mit anderen Unternehmen, ist das jedoch praxisfremd.

Besser wäre es, den Mitarbeitern geeignete Austauschwege und Zugänge zur Verfügung zu stellen. Wenn zudem die Mitarbeiter für die Vertraulichkeit der Firmendaten und die Gefahren sensibilisiert sind, sollten gefährliche Tricks wie das Ablegen von Firmendaten in privaten Dropbox-Accounts der Vergangenheit angehören. Gegen böswillige Spionage von innen ist allerdings kein Kraut gewachsen. (sun) Thomas Ronzon arbeitet als Projektleiter und Senior Softwareentwickler bei der w3logistics AG.

Literatur

- [1] Thomas Ronzon; Das Licht am Ende des Tunnels; JavaSpektrum 1/2013

Durchgegraben

- SimpleMagic – Simple Magic Content and Mime Type Detection Java Package
  1

- Mark’s Lab; UUDWin
  2
- HJ-Split Freeware multi-platform file splitters
  3
- Ultra VNC
  4
- putty
  5
- Blog von Razvan Tudorica
  6
- Heise Netze; Dusan Zivadinovic; Webserver als Shell-Einzeiler
  7
- Abyss Web Server
  8
- HFS – Http File Server
  9
- jibble
  10
- Mini-Webserver 2010
  11

[/expand]

IDEs: Alternativen zu Visual Studio und Eclipse

IDEs: Alternativen zu Visual Studio und Eclipse

[expand title="mehr lesen…"]

IDEs: Alternativen zu Visual Studio und Eclipse

Anders fahren

- Lazarus
  1
- Free Pascal
  2
- SharpDevelop
  3
- RAD Studio 10 Seattle
  4
- MonoDevelop
  5
- Xamarin
  6
- Für die Anwendungsentwicklung unterstützte Android-Geräte
  7
- The GTK Projekct
  8

[/expand]

SQL Server Backup und Recovery ohne teure Tools

SQL Server Backup und Recovery ohne teure Tools

[expand title="mehr lesen…"]

Selbstsicherung

Thorsten Kansy

Das Sichern von Datenbanken beherrscht Microsofts SQL Server mit Bordmitteln. Eine gute Kenntnis der Grundlagen und Fallstricke vorausgesetzt, stellt der Administrator im Ernstfall die Daten schnell und zuverlässig wieder her.

iX-TRACT

Microsofts Datenbank SQL Server bringt Konzepte und Werkzeuge fürs Sichern und Wiederherstellen einer Datenbank bereits mit.

SQL-Server-Backups lassen sich ins Unternehmens-Backup integrieren, der Kauf zusätzlicher Tools ist nicht unbedingt notwendig.

Mehrere Backup-Methoden, Kompression und Verschlüsselung stehen – je nach Version und Edition – zur Auswahl.

Microsofts relationales Datenbankmanagementsystem SQL Server

bietet von Haus aus genügend Tools und Fähigkeiten, die ohne zusätzliche kostenpflichtige Software Datenbanken sichern und wiederherstellen. Backups können bedarfsweise komprimiert und verschlüsselt werden – Automatisierung inklusive. Der Artikel gibt einen Überblick der Backup-Varianten und verrät Tipps für den Produktivbetrieb. Besonderheiten beim Sichern des SQL Server in virtuellen Maschinen sind jedoch nicht Thema.

Generell sollte man für ein Backup einen Zeitpunkt wählen, an dem der Server und die gesamte Infrastruktur (Netz, Storage) über möglichst viele Ressourcen verfügen. Dennoch kann ein Backup jederzeit stattfinden. SQL Server stellt sicher, dass der Inhalt des Backups exakt dem Zustand zum Start der Sicherung entspricht und nur freigegebene Transaktionen enthält. Dazu führt das System eine Log Sequence Number (LSN), mit der es jede Transaktion und jedes Backup versieht.

Zeitpunkt und Datenkonsistenz entscheiden

Während eines Backups durchläuft die Software die interne Struktur einer Datenbank, die auf 8 KByte großen Seiten basiert oder aus Stream-Daten besteht, und sichert nur belegte Bereiche. Zugleich überprüft sie, gewissermaßen als Nebeneffekt, die Konsistenz – ein weiterer Grund für ein Backup.

Der Backup-Dialog von SQL Server: Die physischen Daten liegen in mehreren Dateien, die in Filegroups organisiert sind (Abb. 1).

Auf die Frage, was gesichert werden soll, lässt sich nicht schlicht mit „meine Datenbank“ antworten. Viele Datenbanken bestehen aus Performancegründen oder wegen ihrer schieren Größe aus mehreren Dateien, die in File-Gruppen organisiert sind – Letztere sind der physische Speicherort der Daten. Dazu kommen Dateien für das Transaktionsprotokoll, das alle Änderungen an der Datenbank vor dem Umsetzen speichert.

Daher muss die korrekte Antwort auf die Frage lauten: „Die Inhalte meiner Datenbanken“ – ein wichtiger Unterschied. Das bedeutet im Detail, dass für ein komplettes Backup alle Datendateien inklusive der FileStream-Daten (beispielsweise aus dem FileTable-Feature) oder das Transaktionsprotokoll gesichert werden müssen. Letzteres lässt sich später jedoch nur wieder einspielen, wenn man auf eine davorliegende Sicherung von Daten zugreifen kann.

Außerdem ist das Sichern der eigenen Datenbank(en) meist zu kurz geplant. So enthält etwa die *master*-Datenbank alle Log-ins, was insbesondere beim Einsatz der SQL-Server-Authentifizierung die Kennwörter als Hash einschließt (das gilt allerdings nicht für Contained Databases). Zudem speichert das System in der *msdb*-Datenbank unter anderem alle Jobs des SQL-Server-Agenten und alle Wartungspläne. Weitere Datenbanken des sogenannten Systemkatalogs lassen sich entweder nicht sichern (*tempdb*) oder werden selten bis gar nicht genutzt (*model*). Deshalb sollte man die Datenbanken *master* und *msdb* neben den eigenen in der Planung der Backups berücksichtigen.

Recovery Model passend auswählen

Listing 1: Festlegen des Recovery Model für eine Datenbank

```
USE [master];

GO

-- Recovery Model Simple

ALTER DATABASE: [dotnetconsulting_Backups] SET RECOVERY SIMPLE
WITH NO_WAIT;

GO
```

```
-- Recovery Model: Bulk-logged

ALTER DATABASE [dotnetconsulting_Backups] SET RECOVERY BULK_LOGGED WITH NO_WAIT;

GO

-- Recovery Model: Full

ALTER DATABASE [dotnetconsulting_Backups] SET RECOVERY FULL WITH NO_WAIT;
```

Eine Option bestimmt maßgeblich die Art des Backups: Das Recovery Model (Wiederherstellungsmodell) steuert die Verwendung des Transaktionsprotokolls für eine Datenbank und ist für eine Sicherung so wichtig, dass sein aktueller Wert im Backup-Dialog (siehe Abbildung 1) zu Informationszwecken angezeigt wird. Drei Konfigurationswerte sind fürs Recovery Model erlaubt: Full, Bulk-logged und Simple (siehe Tabelle). Man ändert sie über die Eigenschaften der Datenbank (Karteireiter „Options“) oder per Transact-SQL (T-SQL) durch eine der drei Varianten, die Listing 1 zeigt.

Die in der Tabelle getroffene Aussage zur erwarteten Größe des Transaktionsprotokolls hängt von der Menge der Änderungen in der Datenbank (Daten und Struktur) und davon ab, ob Bulk-Operationen vorkommen. Außerdem enthält die Tabelle die Angabe, ob ein Just-in-Time (JIT) Recovery möglich ist, ob man also die Daten zu einem bestimmten Zeitpunkt wiederherstellen kann.

SQL Server kennt drei Arten von Sicherungen: vollständig, differenziell und Log-Backup. Ein vollständiges Backup umfasst alle Daten einer Datenbank, einer Filegroup oder einer Datei. Für ein Restore benötigt man lediglich diese Sicherung. Die Größe hängt von der Datenmenge ab.

Ein differenzielles Backup sichert alle Veränderungen seit dem letzten vollständigen Backup. Für ein Restore wird Letzteres

sowie das Differential Backup benötigt – es sind demnach immer zwei Backups beteiligt. In diesem Zusammenhang ist die Einstellung „Copy-Only“ wichtig (Abbildung 1). Hat man sie beim Erstellen des vollständigen Backups gesetzt, berücksichtigt ein Differential Backup dieses nicht, es ist gewissermaßen unsichtbar. Damit erstellt man aus der Reihe fallende Backups, ohne dass sie später beim Restore fehlen. Ein Differential Backup wird über die Zeit immer größer, je länger das letzte Backup zurückliegt.

Datenbanken und Logs sichern

Ein Log-Backup schließlich sichert das Transaktionsprotokoll. Dabei werden alle Informationen aus dem Protokoll gesichert, egal in welcher Datei sie sich befinden – welche Dateien zu sichern sind, lässt sich nicht angeben. Für ein Restore benötigt man ein vollständiges Backup (wahlweise mit anschließendem Differential Backup), an das sich ein oder mehrere Log-Backups in der richtigen Reihenfolge anschließen können. Auch hier ist die Copy-Only-Einstellung wichtig: Ist sie gesetzt, wird das Transaktionsprotokoll nicht abgeschnitten, sondern weitergeführt.

Der Zeitpunkt des letzten Backups ist im Eigenschaften-Dialog einer Datenbank ersichtlich (Abb. 2).
Ein Log-Backup kann starten, wenn zuvor mindestens ein vollständiges oder differenzielles Backup lief. Wann eine Datenbank zuletzt gesichert wurde, ist aus ihrem Eigenschaften-Dialog ersichtlich (siehe Abbildung 2). Ein Backup startet der Administrator per Dialog oder mit der T-SQL-Anweisung *BACKUP DATABASE* respektive *DATABASE LOG*.

Mehrere Ziele für Backups stehen zur Verfügung: Band, Datei und Cloud (vorgesehen ist dafür Windows Azure Storage). Ein Backup etwa auf DLT (Digital Linear Tape) ist zwar auch mit SQL-Server-eigenen Mitteln möglich, jedoch recht rudimentär und daher eher selten anzutreffen. Häufiger wird in eine Datei

gesichert, die das Backup-System des Unternehmens anschließend (zusammen mit anderen Dateien) auf ein anderes Medium überträgt. Somit lässt sich der SQL Server einfach in die Backup-Planung für die gesamten Firmendaten einbinden.

Damit Sicherungen nicht zu umfangreich geraten, komprimiert der SQL Server Backups optional. Dabei erreicht er eine Kompressionsrate von etwa 50 Prozent. Zwar ist das verglichen mit den bis zu 90 Prozent von nachträglich angewandten Werkzeugen wie WinZip oder 7Zip recht wenig, hat jedoch den Vorteil der Integration in den Backup-Prozess. Die Kompression spart Festplattenplatz und reduziert die Last fürs Speichersystem, beansprucht allerdings CPU-Leistung, die jedoch bei zeitgemäßer Hardware nicht übermäßig ins Gewicht fällt.

Listing 2: Verschlüsselung eines Backups

```
USE master;

GO

-- Database Master Key erzeugen

CREATE MASTER KEY ENCRYPTION BY PASSWORD = '123456';

GO

-- Zertifikat für Backup erstellen

CREATE CERTIFICATE BackupCertificate WITH SUBJECT = 'Zertifikat für Backupverschlüsselung';

GO

-- Backup mit AES256 durchführen

BACKUP DATABASE [dotnetconsulting_Backups]

TO DISK = N'{Pfad}'
```

```
WITH

  COMPRESSION,

   ENCRYPTION (ALGORITHM = AES_256, SERVER CERTIFICATE = BackupCertificate);
```

Außerdem bietet SQL Server das Verschlüsseln von Backups an, was bei schützenswerten Daten oder beim Sichern in die Cloud zur Gewährleistung der Datensicherheit beiträgt. Dazu setzt die Software ein Zertifikat mit einem symmetrischen Verschlüsselungsalgorithmus wie dem Advanced Encryption Standard (AES) ein. Der gewünschte Algorithmus und das Zertifikat müssen beim Erstellen des Backups angegeben werden. Wie das in T-SQL aussehen kann, zeigt Listing 2.

Wahlweise Kompression und Verschlüsselung

Die deutsche Version des SQL Server beweist beim Erstellen eines Backups (wohl unbeabsichtigt) Sinn für Humor mit folgender Meldung: „Das zum Verschlüsseln des Verschlüsselungsschlüssels für die Datenbank verwendete Zertifikat wurde nicht gesichert.“ Trotzdem verweist die Meldung auf den wichtigen Umstand, dass das verwendete Zertifikat (BackupCertificate) für ein Restore zur Verfügung stehen muss und daher seinerseits gesichert und sicher aufbewahrt werden sollte (mit der Anweisung *BACKUP CERTIFICATE*).

Listing 3: Restore einer Datenbank

```
USE [master];

RESTORE DATABASE [dotnetconsulting_Backups] FROM  DISK = {Pfad} WITH FILE = 1;
```

Will man eine Sicherung wiederherstellen, darf es zu diesem Zeitpunkt keine Client-Verbindungen zur betreffenden Datenbank geben, andernfalls scheitert das Restore mit einem Timeout. Ein schlichtes Restore mit T-SQL zeigt Listing 3. Mit dem

Zusatz *MOVE* bestimmt der Administrator, wie die Datenbankdateien wiederhergestellt werden sollen – etwa wenn der Zielserver eine unterschiedliche Verzeichnisstruktur enthält oder wenn die Wiederherstellung eine Kopie einer Datenbank erstellen soll. Übrigens akzeptiert die *RESTORE*-Anweisung bei Bedarf einen alternativen Name für die Datenbank.

Besteht ein Backup aus mehreren Teilen, etwa weil ein differenzielles Backup durchgeführt wurde, erlaubt der Zusatz *NORECOVERY* das Einspielen weiterer Dateien als Teil des Restore:

```
RESTORE DATABASE [dotnetconsulting_Backups]
 FROM  DISK = {Pfad} WITH FILE = 1, NORECOVERY;
```

Eine Datenbank im Object Explorer während des Restore (Abb. 3)

Bei einem Restore markiert das SQL Server Management Studio den Zustand der Datenbank mit einem entsprechenden Symbol und dem Wort „Restoring“ (Abbildung 3). Erst nach dem Überspielen der letzten Datei – ohne den Zusatz *NORECOVERY* beziehungsweise *RECOVERY* – lässt sich die Datenbank wieder regulär benutzen.

Grenzen der Bordmittel in großen Umgebungen

Die Tools fürs Backup des SQL Server stoßen jedoch an Grenzen. Mit zunehmender Zahl von SQL-Server-Installationen muss der Administrator immer mehr Datenbanken in unterschiedlichen Versionen verwalten, sichern und zudem erfolgreiche und fehlgeschlagene Sicherungsjobs überblicken. Zwar verteilt der SQL Server Aufgaben (die SQL Server Agent Jobs) dezentral, jedoch mangelt es dabei an Komfort und Transparenz. Zumal in einem Unternehmen das Sichern der SQL-Server-Datenbank nur ein Aspekt des Backups ist: Dateiserver, Exchange-Server und viele mehr kommen dazu. Außerdem verraten die Bordmittel nicht, wie lange die freien Ressourcen und Backup-Zeitfenster für einen

reibungslosen Betrieb reichen, denn Datenbanken wachsen und eine Prognose ist schwierig. Diese Aufgaben können nur umfassende Backup-Tools übernehmen.

Zwischen den Versionen und Editionen des SQL Server bestehen kleinere Unterschiede hinsichtlich der Sicherungstools. So hielt etwa die Kompression erst in SQL Server 2008 Einzug und stand damals nur der Enterprise Edition zur Verfügung. Ab SQL Server 2012 kommt jedoch auch die Standard Edition in den Genuss dieses Features. Generell bedeuten die Ressourcen einer SQL-Server-Installation auch für ein Backup eine Limitierung. Nutzt etwa eine Edition nur eine begrenzte Zahl an Prozessoren (Kernen) und eine begrenzte Menge Speicher, kann das Backup länger dauern. Andererseits sind in diesen Fällen die Datenbanken meist nicht allzu groß. Jedoch sei zur Beruhigung gesagt, dass jede Version oder Edition selbstverständlich in der Lage ist, ihre jeweilige maximale Datenbankgröße zu sichern (und wiederherzustellen).

Zum Schluss noch einige Tipps für einen einfacheren Alltag mit Backups und Restores:

– Backups sollten automatisiert laufen. Das erledigt entweder der SQL-Server-Agent, oder ein Backup ist Teil eines Wartungsplans.

– Bei der Planung sollte man Backups möglichst einfach halten. Wenn Zeit und Größe es erlauben, sind vollständige Backups zu bevorzugen.

– Der Administrator sollte regelmäßig Restores durchführen und die dafür benötigte Zeit messen. Das stellt sicher, dass eine Wiederherstellung auch im Ernstfall gelingt.

– Eine Sicherung ist nur so gut wie die dazugehörende Wiederherstellung. Wenn Letztere aus irgendwelchen Gründen nicht klappt, ist Erstere wertlos.

Fazit

SQL Server bietet alles Notwendige, um sowohl Backups als auch bei Bedarf einen Restore effizient und flexibel durchzuführen. Microsofts Bordmittel für seine Datenbankplattform decken also auch diese Phase des Betriebs ab. (tiw) Thorsten Kansy arbeitet als Berater, Softwarearchitekt und Coach mit den Schwerpunkten Datenbanken und Business Intelligence.

[/expand]

Progressive Web Apps: Service Worker und mehr

Progressive Web Apps: Service Worker und mehr

[expand title="mehr lesen…"]

Dienst am Kunden

- Progressive Web Apps
- Addy Osmani: Getting started with Progressive Web Apps
- Understanding Progressive Enhancement (Aaron Gustafson)
- Graceful degradation versus progressive enhancement
- What is a Polyfill?
- Nolan Lawsons Progressive web apps reading list
- Is ServiceWorker Ready?

- Can I use Service Workers
- Web App Manifest. Living Document
- Configuring Web Applications
- Compare stats
- About httparchive

Progressive Web Apps: Service Worker und mehr

Dienst am Kunden

Sebastian Springer

Designer kennen Progressive Enhancement seit gut zehn Jahren: Webseiten zunächst so zu planen, dass alle etwas damit anfangen können. Entwickler sogenannter Web-Apps hauchen mit diesem Prinzip ihren Webdokumenten die Eigenschaften von Apps ein.

iX-TRACT

Das Konzept der Progressive Web Apps bündelt eine Sammlung von Techniken und Best Practices, nach denen man eine Webapplikation aufbaut, die mehr wie eine App denn wie ein klassisches Webdokument wirkt.

Service Worker sind an der Schnittstelle zum Server angesiedelte Hintergrundprozesse, die Netzanfragen abfangen sowie beantworten können. Sie helfen, eine Web-App offlinefähig zu gestalten.

Weitere in Progressive Web Apps verwendete Techniken sind ein Manifest, das das Verhalten der Anwendung festlegt, sowie Push Notifications, über die Nachrichten an den Nutzer verschickt werden können.

Bei Progressive Web Apps (PWAs) verschwimmen durch den Einsatz fortgeschrittener Browsertechniken die Grenzen zwischen App und Webseite. Eine solche Anwendung kann auf einem Gerät installiert und selbst ohne bestehende Internetverbindung verwendet werden. Mit den den PWAs zugrunde liegenden Techniken wollen Browserhersteller, allen voran Google, den Betrieb von Applikationen auch über schwache und unzuverlässige Internetverbindungen ermöglichen, ohne auf die aktive Kommunikation mit dem Benutzer zu verzichten.

Progressive Web Apps sind weniger ein konkretes Muster, nach dem jemand eine Applikation erstellt, sondern mehr eine Sammlung von Techniken und Best Practices, nach denen man eine Webapplikation aufbauen sollte. Der Schlüssel zum Erfolg von PWAs ist die Tatsache, dass diese Anwendungen überall verfügbar sind und mit nahezu jedem Browser und auf einer Vielzahl von Systemen zum Einsatz kommen können. Mit sogenannten Service Workern kann man eine PWA vollständig offline betreiben oder nur Teile der Applikation nachladen, was den entstehenden Netz-Traffic erheblich verringert.

Einige Komponenten einer PWA erfordern es, dass die Kommunikation ausschließlich über gesicherte Verbindungen stattfindet, was letztlich den Benutzern zugutekommt. Wie erwähnt verbinden PWAs die Vorteile einer Webseite mit denen einer App. Ein Benutzer ist in der Lage, mit einem Klick die Anwendung auf seinem System zu installieren, verliert darüber aber nicht die Option, zur Navigation innerhalb der Applikation Links zu verwenden oder Bookmarks für bestimmte Status zu erstellen und diese mit anderen zu teilen. Durch den Einsatz von Standardtechniken wie HTML, CSS und JavaScript können auch Suchmaschinen PWAs auffinden. Gerade Google unterstützt diese Art von Anwendungen und stellt selbst zahlreiche Ressourcen wie eine umfangreiche Dokumentation und etliche Beispielapplikationen zur Verfügung. Ihre Stärke spielen PWAs vor allem bei mehrmaliger und regelmäßiger Verwendung aus, da hier die Caching-Strategien greifen und die

Performance der Applikation spürbar verbessern. Ziel einer PWA ist es, eine solche Art der Verwendung zu unterstützen und aktiv mit dem Benutzer zu kommunizieren.

Progressive Enhancement: Kern einer PWA

Wie der Name andeutet, basieren PWAs auf dem Konzept des Progressive Enhancement. Das Gegenstück dazu ist Graceful Degradation, was in der Vergangenheit vielen Webapplikationen als Grundlage diente. Hierbei entwickeln und testen die Webprogrammierer die Applikationen nur für die neuen Versionen der Mainstream-Browser. Falls noch Zeit und Budget übrig bleibt, finden noch kleinere Anpassungen statt, sodass auch ein älterer Browser die Applikation verwenden kann. Das führt dazu, dass sie sich nur noch eingeschränkt nutzen lässt, weil einige Features nur umständlich oder überhaupt nicht nutzbar sind.

Progressive Enhancement stellt dagegen den Inhalt einer Applikation in den Vordergrund. Es bedeutet, keinerlei Annahmen über die verwendeten Browser zu treffen. Die beste Beschreibung von Progressive Enhancement liefert Aaron Gustafson in einem Blogartikel bei A List Apart (Links wie dieser sind über den blauen Balken „Alle Links“ am Ende des Artikels zu finden), in dem er dieses Konzept mit einem Erdnuss-M&M vergleicht. Der Kern ist der Inhalt der Anwendung, der aus dem Markup und den Daten besteht. Alle Browser können ihn konsumieren, da es sich lediglich um Struktur und Information handelt.

Cascading Style Sheets (CSS) bilden den Schokoladenmantel von Progressive Enhancement und bereiten den Inhalt optisch ansprechend und gut nutzbar auf. Diese Schicht sorgt außerdem dafür, dass der Inhalt für das jeweilige Gerät und die verwendete Auflösung optimiert dargestellt wird. In dieser Schicht kommen zudem neuere CSS-Features zum Einsatz, die nicht mehr unbedingt jeder Browser unterstützt. Die Schwierigkeiten der Unterstützung lassen sich durch Polyfills

lösen. Selbst wenn der Browser einen Style nicht versteht, kann er den Inhalt anzeigen.

Schließlich bildet JavaScript die Zuckerschicht des Progressive Enhancement. Hier kommen die Service Worker, Push Notifications und viele andere Features zum Einsatz. Hinsichtlich der eingesetzten Frameworks und Bibliotheken treffen weder Progressive Enhancement noch PWAs eine Aussage. Eine Applikation kann in reinem JavaScript ohne ein Framework oder mit AngularJS beziehungsweise React erstellt werden. Das Konzept lässt sich auf nahezu jeder Plattform umsetzen. Im Kern geht es um eine alte Strategie: Separation of Concerns. Bei der Erstellung einer Anwendung soll der Entwickler die Struktur von der Darstellung und der Logik trennen.

Herzstück einer PWA: Service Worker

Schwankungen in der Qualität der Netzverbindung sind für Webentwickler allgegenwärtig. Deshalb ist es logisch, dass eine PWA solche Schwankungen in der Verbindung zum Server ausgleichen können muss. Die Spanne reicht von leicht verzögerten Antworten eines Servers bis hin zum vollständigen Nichtvorhandensein einer Verbindung.

Offlinefähige Applikationen sind in der Webentwicklung nichts Neues. Schon seit Langem unterstützen Browser den Application Cache, bei dem eine Manifest-Datei angibt, welche Dateien ein Browser herunterladen und welche er vom Server beziehen soll. Letztere lädt er einmal vom Server herunter, sie verbleiben danach im Cache des Browsers. Hat der Client eine Verbindung zum Server, prüft Ersterer, ob das Manifest verändert wurde. Wenn das der Fall ist, untersucht der Browser, ob die Dateien im Cache noch aktuell sind. Diese Methode für offlinefähige Webapplikationen lösen in Zukunft sogenannte Service Worker ab. Aus diesem Grund haben sich die Entwickler von Firefox entschlossen, seit Version 44 des Browsers eine Warnmeldung auf der Konsole anzuzeigen, wenn das AppCache-Feature verwendet wird.

Im Gegensatz zum AppCache, der auf Basis einer statischen Datei arbeitet, ist ein Service Worker ein Hintergrundprozess, der an der Schnittstelle zum Server angesiedelt ist und Netzanfragen abfangen sowie beantworten kann. Neben dieser Proxy-Funktion unterstützen Service Worker Features wie Push Notifications und Background Sync. Dazu später mehr.

Da ein Service Worker ein Hintergrundprozess ist, kann die Web-App nicht direkt mit dem DOM interagieren. Service Worker und die Webapplikation tauschen lediglich Informationen aus. Zu diesem Zweck kommt *postMessage* zum Einsatz. Einen ähnlichen Ansatz verfolgen Web Worker, die man als Hintergrundprozess nutzen kann, um rechenintensive Operationen aus dem Browser-Prozess auszulagern. Sie können ebenfalls mit dem Server kommunizieren. Ihr Schwerpunkt liegt jedoch auf der Ausführung von Applikationslogik.

Service Worker kümmern sich dagegen mehr darum, Infrastruktur für eine Applikation zur Verfügung zu stellen. Um die Abhängigkeit von einer permanent vorhandenen Netzverbindung zu lösen, müssen Webentwickler zunächst einen Service Worker erstellen. Für die lokale Entwicklung gelten keine weiteren Anforderungen. Soll die Applikation jedoch auf einem Server arbeiten, muss man sicherstellen, dass HTTPS verwendet wird. Der Grund dafür ist, dass ein Service Worker an einer zentralen Stelle in der Applikation installiert sein muss. Eine Man-in-the-Middle-Attacke hätte hier fatale Folgen. Eine verschlüsselte Verbindung erschwert das zumindest.

Dienst installieren, aktivieren und beenden

Der Lebenszyklus eines Service Workers besteht aus drei Phasen: Installieren, Aktivieren und nach der Arbeit Terminieren, um die Ressourcen des Systems zu schonen. Der Worker wird aktiviert, sobald er entweder ein *fetch*– oder ein *message*-Event erhält. Dieses ressourcenschonende Beenden des

Worker-Prozesses hat zur Folge, dass der Worker selbst nicht langfristig einen Status speichern kann. Sollte das erforderlich sein, kann man auf die IndexedDB des Browsers zurückgreifen.

Um den Worker-Prozess zu registrieren, greift man auf das *serviceWorker*-Objekt der globalen *navigator*-Eigenschaft des Browsers zurück. Die *register*-Methode des *serviceWorker*-Objekts akzeptiert einen Dateinamen als Argument. Die angegebene Datei enthält den tatsächlichen Quellcode des Service Workers. Die Registrierung sowie zahlreiche andere Funktionen eines Service Workers laufen entkoppelt vom Programmfluss ab (asynchron). Zur besseren Steuerung solcher asynchronen Operationen kommen Promises zum Einsatz. Der Rückgabewert der *register*-Methode ist ein solches Promise-Objekt, über dessen *then*-Methode man Callback-Funktionen für den Erfolgs- und Fehlerfall der Registrierung des Service Workers einbinden kann. Sie werden ausgeführt, wenn die Registrierung beendet ist. Die wiederum ist der einzige Schritt, der direkt in der Applikation stattfindet, alles Weitere setzt die Datei des Service Workers um.

Listing 1: Service Worker registrieren

```
if ('serviceWorker' in navigator) {
    navigator.serviceWorker.register('serviceWorker.js')
        .then(function success(reg) {
            console.log('Registration successful: ', reg);
        }, function failure(err) {
            console.log('Registration failed: ', err);
        });
}
```

Prüft man vor der Registrierung, ob der Browser dieses Feature unterstützt, und führt den Code nur aus, falls dies der Fall ist, erfüllt dies eine wichtige Anforderung: Die PWA funktioniert auch auf Systemen, die ein bestimmtes Feature noch nicht implementiert haben. Verzichtet man auf eine solche Überprüfung, wird eine Exception geworfen, was im

ungünstigsten Fall zur Beendigung der Applikation führt. Listing 1 enthält den Quellcode für die Registrierung eines Service-Worker-Prozesses. Er kann beliebig oft ausgeführt werden, da der Browser prüft, ob der Worker schon registriert ist, und den Vorgang in diesem Fall nicht wiederholt.

Der Worker-Prozess ist nur für das Verzeichnis verantwortlich, in dem die Datei liegt. Das bedeutet, dass ein Service Worker, dessen Datei im Document-Root-Verzeichnis des Webservers liegt, für sämtliche Anfragen auf diesen Webserver zuständig ist. Befindet sich die Datei in einem Unterverzeichnis, fängt er nur Anfragen in dieses Verzeichnis und darunterliegende ab.

Callback für statische Inhalte

Listing 2: Installieren eines Service Workers

```
self.addEventListener('install', function (e) {
    e.waitUntil(
        caches.open('app')
            .then(function (cache) {
                return cache.addAll(['/node/style/style.css'])
            })
    )
});
```

Auf das Registrieren folgt das Installieren. In dieser Phase geht es normalerweise darum, Caches für statische Inhalte wie Stylesheets, HTML- und JavaScript-Dateien sowie Bilder und andere Mediendateien anzulegen und zu befüllen. Das *install*-Event löst diesen Vorgang aus und setzt ihn in Form einer Callback-Funktion um. Die Behandlung des Cache ist ein zweistufiger asynchroner Prozess, der auf Promises basiert, die der JavaScript-Standard ECMAScript seit der Version 6 aus dem Jahre 2015 vorsieht. Zunächst erzeugt der Worker einen Cache mit einem frei wählbaren Namen und füllt ihn im zweiten Schritt mit einem Array von Dateinamen. Wie dieser Prozess genau funktioniert, zeigt Listing 2.

Listing 3: Service Worker aktualisieren

```
self.addEventListener('activate', function (e) {
    caches.keys().then(function (names) {
        return Promise.all(
            names.map(function (name) {
                if (name !== 'v1.0.2') {
                    return caches.delete(name);
                }
            })
        )
    })
});
```

Nachdem der Service Worker installiert ist, kann man im Aktivierungsschritt weitere Verwaltungstätigkeiten an den Caches durchführen. Typischerweise findet die Aktualisierung alter Cache-Inhalte in diesem Schritt statt. Der Quellcode aus Listing 3 löscht sämtliche Caches, die nicht den Namen v1.0.2 tragen. Das soll sicherstellen, dass nur die neueste Version der Dateien ausgeliefert wird.

Listing 4: Cache Handling

```
self.addEventListener('fetch', function (e) {
    e.respondWith(
        caches.match(e.request)
            .then(function (response) {
                if (response) {
                    return response;
                }

                return fetch(event.request);
            })
    )
});
```

Aktualisiert der Benutzer eine Seite, löst das *fetch*-Events aus, die ihrerseits Service Worker abfangen und beantworten können. Dazu registriert man eine Callback-Funktion für das *fetch*-Event. Die Repräsentation dieses Events verfügt über

eine Methode *respondWith*, mit der man steuern kann, wie die Antwort an den Browser aussehen soll. Die Methode akzeptiert eine Promise als Argument, die man beispielsweise durch ein *caches.match* erzeugen kann, wenn man im Cache nach einem passenden Treffer sucht. Ist kein Eintrag vorhanden, kann die Anfrage mit einem Aufruf der *fetch*-Funktion an den Server weitergeleitet werden. Listing 4 enthält den Quellcode für eine einfache Bedienung von Anfragen aus dem Cache.

Erzeugt man mit *caches.open* eine Referenz auf einen benannten Cache, kann ein Worker über die *put*-Methode diesem Cache neue Einträge hinzuzufügen, um zukünftige Anfragen aus dem Cache beantworten zu können.

Hat man sich an den Umgang mit den Promises gewöhnt, stellt der Einsatz von Service Workern in einer Applikation keine große Hürde mehr dar. Man muss sich nur stets vor Augen halten, dass Service Worker Teil von PWAs sind, die auf jedem System funktionieren sollen. Service Worker sollten daher lediglich die Funktionsvielfalt erweitern und die Benutzung verbessern und keinesfalls ein erforderlicher Bestandteil einer Anwendung sein.

Eine PWA sollte installierbar sein. Diese Anforderung gilt besonders für mobile Geräte. Da vor allem Google und Mozilla zu den Unterstützern von PWAs zählen, wundert es nicht, dass die Installierbarkeit einer Webapplikation auf Android-Geräten sich von selbst versteht. Apple geht auf seinen iOS-Geräten einen anderen Weg, der ein wenig Mehraufwand erfordert. Die Konfiguration der Applikation erfolgt für iPhones über verschiedene *link*– und *meta*-Tags. Genauere Informationen hierzu findet man auf Apples Developer-Seiten (siehe *iX*-Link unter „Configuring Web Applications“).

Zurück zur Installation auf einem Android-Gerät. Seit Chrome 39 lässt sich eine Webanwendung über das Browser-Menü auf dem Homescreen installieren.

Apples Safari bietet ein ähnliches Feature. In beiden Fällen handelt es sich nicht wirklich ums Installieren, sondern vielmehr um einen speziellen Link zum Browser, der allerdings dafür sorgt, dass sich eine Webanwendung wie eine native App anfühlt. Dieses Feature erreicht jedoch schnell seine Grenzen, wenn die Internetverbindung nicht zuverlässig ist, da der Browser standardmäßig keinerlei Dateien zwischenspeichert. Das bedeutet, dass man dieses Feature idealerweise mit dem Einsatz von Service Workern kombinieren sollte.

Mit Version 42 hat Chrome ein neues Feature mit dem Namen App Install Banner eingeführt. Diese Browserfunktion präsentiert dem Benutzer ein Banner, das ihm dieselben Optionen wie das bisherige „Add to Homescreen“ bietet, allerdings wesentlich prominenter sichtbar ist.

Wie die „App“ sich verhalten soll

Damit jetzt nicht jede Webseite ein solches Banner einführt und dem Benutzer bei jedem Besuch einer beliebigen Webseite zuerst ein Installationsbanner entgegenkommt, müssen einige Voraussetzungen erfüllt sein: Ein Service Worker muss für die Webapplikation registriert sein, sie muss per HTTPS ausgeliefert werden und über ein Web-App-Manifest verfügen (siehe unten). Damit das App Install Banner zur Anzeige kommt, muss der Benutzer eine Seite mindestens zweimal im Abstand von mindestens fünf Minuten besucht haben. Und für dieses Feature gilt wieder, dass es für Browser, die es unterstützen, einen Mehrwert für den Benutzer bietet, das Nutzungserlebnis für alle anderen allerdings nicht einschränkt.

Listing 5: Web-App-Manifest

```
{
  "short_name": "MyPWA",
  "name": "My first progressive web app",
  "icons": [
    {
      "src": "hello.png",
```

```
      "sizes": "96x96",
      "type": "image/png"
    }
  ],
  "start_url": "/index.html",
  "display": "standalone",
  "orientation": "landscape"
}
```

Eine der Anforderungen für das Funktionieren von App Install Banners ist das Vorhandensein eines Web-App-Manifests. Dahinter steckt nichts anderes als eine Beschreibungsdatei, die festlegt, wie sich die Applikation auf einem Gerät verhalten soll, wenn der Anwender sie als App auf dem Homescreen installiert hat. Diese Manifest-Datei ist nicht mit dem AppCache-Manifest zu verwechseln, das früher beim AppCache zum Einsatz kam. Beim Web-App-Manifest handelt es sich um eine JSON-Datei, die ein *link*-Element einbindet. Browser, die dieses Feature nicht unterstützen, ignorieren das Element schlicht. Von daher erfüllt das Web-App-Manifest die wichtigste Anforderung einer PWA: keine Einschränkung für ältere Browser (siehe Listing 5).

Die verschiedenen Eigenschaften bestimmen, wie sich die App verhält, wenn sie auf einem Gerät installiert ist. So kann man beispielsweise den Namen angeben, den das App Install Banner anzeigen soll. Unter der Eigenschaft *icons* lassen sich Icons in verschiedenen Auflösungen und Dateitypen aufführen – beispielsweise, welches auf dem Homescreen des Benutzers angezeigt wird. Die *start_url* gibt den Startpunkt der Applikation an. Die *display*-Eigenschaft definiert, wie der Browser in Erscheinung treten soll. Der Wert *browser* sorgt dafür, dass alle Browser-Controls wie der Zurück-Button verfügbar sind. *minimal-ui* schränkt die sichtbaren Steuerungselemente weiter ein, und *fullscreen* lädt die Applikation bildschirmfüllend und ohne Steuerungselemente. Mit dem Wert *standalone* übernimmt die Webapplikation das Look-and-Feel nativer Apps. Dieser Wert wird am häufigsten für

installierte PWAs verwendet. Die Eigenschaft *orientation* gibt an, ob die Applikation im Hochformat (*portrait*) oder Querformat (*landscape*) zu laden ist. Es gibt weitere Eigenschaften – Genaueres ist dem Working Draft des W3C (siehe den genannten blauen Balken „Alle Links“) zu entnehmen.

Progressive Web Apps sollen die Bindung zum Benutzer stärken und ihn über Änderungen auf dem Laufenden halten. Push Notifications (Listing 7) basieren auf den bisher vorgestellten Service Workern und dem Web-App-Manifest. Der Vorteil dieser Benachrichtigungen ist, dass der zugrunde liegende Service Worker zunächst terminiert und keine Ressourcen benötigt. Geht eine Nachricht ein, wird der Service Worker aktiv und kann mit der Nachricht umgehen. Das funktioniert sogar, wenn der Browser nicht aktiv ist.

Benachrichtigungen per *push* abonnieren

Listing 6: Push Notifications abonnieren

```
navigator.serviceWorker.register('sw.js').then(function(reg) {
    reg.pushManager.subscribe({
        userVisibleOnly: true
    }).then(function(sub) {
        console.log('endpoint:', sub.endpoint);
    });
});
```

Einer der Sicherheitsmechanismen von Push Notifications ist, dass Anwendungen nicht ohne Weiteres einem Browser eine Benachrichtigung schicken können. Für deren Versand kommen spezielle Dienste wie Googles Cloud Messaging (kurz GCM) zum Einsatz. Nachdem der Entwickler ein Projekt für den Notification Service der Applikation im GCM erstellt hat, muss er die Sender-ID in der Manifest-Datei der PWA eintragen. Danach wird die Applikation für Push Notifications registriert und ein Handler für das *push*-Event erstellt. Beide Schritte enthält der JavaScript-Code der Applikation. Das Abonnement der Push Notifications geschieht während der Registrierung des

Service Workers (siehe Listing 6).

Listing 7: Anzeige einer Push Notification

```
self.addEventListener('push', function(event) {
    event.waitUntil(
           self.registration.showNotification('New content
available', {
            body: '5 New datasets available',
            icon: 'images/icon.png'
        }));
});
```

Event-Handling wiederum findet im Service Worker selbst statt: über das Auslösen des *push*-Events, wenn eine Nachricht eingeht. Die *showNotification*-Methode stellt sicher, dass die Nachricht dem Benutzer angezeigt wird.

Außer *push*– gibt es mit *notificationclick* ein weiteres Event, das beim Umgang mit Push Notifications hilfreich ist. Ein Benutzer löst es durch einen Klick auf eine Benachrichtigung aus; es kann dazu dienen, Browserfenster in den Fokus zu bringen oder neu zu öffnen. Push Notifications sind ein mächtiges Werkzeug, da sie die Aufmerksamkeit des Benutzers auf sich ziehen – und das, obwohl die Applikation zu diesem Zeitpunkt geschlossen sein kann. Aus diesem Grund sollte man mit solchen Benachrichtigungen eher sparsam umgehen, da man damit schnell über das Ziel hinausschießen kann und den Benutzer eher verärgert, was im harmlosen Fall zu einer Deaktivierung der Push Notifications und im schlimmsten Fall zur Abkehr des Benutzers von der Applikation führt. Eine Push Notification sollte man nur im Falle eines für den Benutzer wichtigen Ereignisses versenden – keinesfalls zu Werbezwecken.

Ziel einer PWA ist eine optimale Performance. Um dies zu erreichen, muss die Applikation eine kurze Ladezeit haben. Und genau an diesem Punkt greift die App-Shell-Architektur ein. Die App Shell bezeichnet den Rahmen der Applikation, also nur das fürs Ausführen der App wirklich erforderliche HTML, CSS

und JavaScript. Alle weiteren Inhalte kann die Software zu einem späteren Zeitpunkt nachladen. Dieser Rahmen besteht in der Regel aus statischen Inhalten, die der Service Worker gut zwischenspeichern kann. Dynamische Daten lädt er erst, wenn die „Basis“ vorhanden ist.

So zeigt die App dem Benutzer zunächst das Grundgerüst der Applikation, wie die Navigation oder ähnliche sinnvolle Inhalte, die sich nicht allzu häufig ändern. Sobald weitere Inhalte verfügbar sind, werden sie ebenfalls angezeigt. Der Benutzer erhält auf diese Weise schnelles Feedback und kann die Applikation schon zu einem frühen Zeitpunkt nutzen. Eine solche Architektur eignet sich allerdings nicht für jede Applikationsart. Ihre Stärke kann die App Shell nur in einer dynamischen Applikation ausspielen. Besteht eine Anwendung hauptsächlich aus statischen Inhalten, ist diese Architektur eher hinderlich und wirkt sich durch die zusätzlichen Anfragen für das Nachladen der Daten negativ auf die Performance aus. Eine solche Architektur lässt sich durchaus auf älteren Browsern und auf verschiedenen Plattformen umsetzen. Ihre Vorzüge kommen jedoch vor allem im Zusammenspiel mit den Caching-Fähigkeiten der Service Worker zur Geltung.

Fazit

Obwohl längst nicht alle Browser Service Worker unterstützen (wie man bei caniuse.com nachschlagen kann, siehe „Alle Links“), sollte jeder Webentwickler die Ideen hinter Progressive Web Apps bei der Arbeit beherzigen. Das gilt für die Erweiterung bestehender Applikationen genauso wie für die Erstellung neuer.

Viele entwickeln Webapplikationen nur noch für die neuen Mainstream-Browser. Genau dagegen richten sich die Verfechter von Progressive Web Apps. Dabei verschwimmen die Grenzen zwischen Webdokument und App auf mobilen Geräten, indem eine PWA installierbar und ihr Verhalten über Manifest-Dateien beeinflussbar ist. Service Worker sorgen für einen

Performanceschub, indem Entwickler kontrollieren können, wann die PWA welche Anfragen aus dem Cache oder vom Server beantworten soll. Diese Caching-Möglichkeit geht bis zur vollständig offlinefähigen Anwendung, die sich bei einer bestehenden Verbindung mit dem Server synchronisieren kann. PWAs bieten mit diesen Funktionen erheblichen Mehrwert, außerdem stärken sie die Bindung zum Nutzer, indem sie ihn durch Push Notifications über aktuelle Änderungen auf dem Laufenden halten.

Google ist einer der größten Treiber hinter PWAs, aber Mozilla stellt ebenfalls immer mehr Ressourcen für diese Art von Applikationen zur Verfügung. Den Stand der Entwicklung sieht man vor allem bei den Service Workern. Firefox, Chrome und Opera kennen sie schon. Bei Microsoft ist die Unterstützung noch nicht so weit fortgeschritten, mit der Integration wurde bereits begonnen. Einzig Apple mit dem Safari-Browser hinkt im Rennen um die neuen Techniken hinterher. Aber selbst hier gibt es erste positive Anzeichen, dass dieses Feature früher oder später Einzug in den Browser halten wird. (hb) Sebastian Springer arbeitet als JavaScript-Entwickler bei der MaibornWolff GmbH in München. Als Dozent und Autor für JavaScript will er die Begeisterung für professionelle Entwicklung mit JavaScript wecken.

[/expand]

Fünf professionelle Profiling-Werkzeuge für PHP

Fünf professionelle Profiling-Werkzeuge für PHP

[expand title="mehr lesen…"]

Auf Touren

- Tideways
- New Relic APM
- Blackfire
- Xhprof
- Xdebug
- Profiling-Demo
- Faker
- ab – Apache HTTP server benchmarking tool

Fünf professionelle Profiling-Werkzeuge für PHP

Auf Touren

Jörn Wagner, Christian Ehmke

Ob Java, .NET, Ruby oder PHP – größere Websites sind längst mit normalen Softwareprojekten vergleichbar. Wie bei diesen helfen Testwerkzeuge und Profiler bei der Analyse. Ein Überblick über fünf solcher Profiling-Tools für PHP.

iX-TRACT

Profiling-Werkzeuge sollen bei der Analyse von Software in Bezug auf Laufzeit, Speicherverbrauch, Datenbankabfragen und weitere Parameter helfen.

Fünf für die PHP-Entwicklung gedachte Produkte hat *iX* eine Demo-Anwendung durchlaufen lassen, um ihre Effizienz zu prüfen.

Xdebug und Xhprof (Open Source) sowie Tideways, Blackfire und New Relic (kommerziell) eignen sich für unterschiedliche Szenarien und Geldbeutel.

iX-Wertung

Blackfire

⊕ einfache Installation und Bedienung

⊕ kostenlose Produktvariante, auch dauerhaft

⊕ Kollaboration im Team

⊕ Datenspeicherung in Europa

⊖ Team-Profile nicht öffentlich teilbar

⊖ kein Profiling auf Windows-Plattformen

New Relic

⊕ einfache Installation und Bedienung

⊕ kostenlose Produktvariante, auch dauerhaft

⊖ rudimentäres Profiling

⊖ keine Berücksichtigung nativer PHP-Funktionen

⊖ kein Profiling auf Windows-Plattformen

Tideways

⊕ einfache Installation und Bedienung

⊕ umfangreiche und übersichtliche Analysen

⊕ Berücksichtigung beliebter Frameworks

⊕ Datenspeicherung in Deutschland

⊖ kein Profiling auf Windows-Plattformen

⊖ nach Ablauf des Testzeitraums kostenpflichtig

Xdebug

⊕ kostenlos

⊕ einfache Installation auf eigenem System

⊖ hoher Overhead

⊖ schwierige Bedienung

⊖ Auswertung abhängig von Drittwerkzeugen

Xhprof

⊕ kostenlos

⊕ aussagekräftiges Auswertungswerkzeug mitgeliefert

⊕ Installation auf eigenem System

⊖ Installation mäßig aufwendig

⊖ unübersichtliche Auswertung

⊖ nicht zu vernachlässigender Overhead

Profiler sind wichtige Werkzeuge für jeden Entwickler, mit denen er das Laufzeitverhalten einer Anwendung oder eines Dienstes analysieren und Schwachstellen aufdecken kann. Unter dem Begriff Profiling versteht man die Erstellung eines Profils der Software in Bezug auf Laufzeit, Speicherverbrauch, Datenbankabfragen und weitere Parameter. Der häufigste Anwendungsfall bezieht sich auf das Zählen und Messen von Aufrufen der beteiligten Funktionen und Methoden. Ziel ist es, die Stellen im Code zu finden, an denen die meiste Zeit verbraucht wird, und diese zu optimieren. Daher ist es

wichtig, nach Änderungen einen guten Vergleich ziehen zu können, um Aussagen über die Wirksamkeit der vermeintlichen Verbesserungen treffen zu können.

Im Ökosystem der weitverbreiteten Webprogrammiersprache PHP waren bis vor wenigen Jahren Profiler nicht übermäßig etabliert. Eine routinemäßige Überprüfung des Laufzeitverhaltens einer Anwendung, die Performance-Probleme schon während der Entwicklung hätte aufdecken können, fand oft nicht statt. Meist erzeugte erst ein echter Performance-Engpass in der Produktivumgebung die Notwendigkeit zur Analyse. Dazu war es durchaus üblich, Konstrukte zur Erfassung genauer Zeitstempel wie manuelle Zeitmessungen im Code – gerne mit Stoppuhr-Klassen oder -Bibliotheken – zu nutzen, um den sogenannten Flaschenhals aufzuspüren. Dieses Trial and Error lag womöglich an den wenigen verfügbaren Lösungen zum Erstellen von Profilen mit PHP. Und von denen waren die meisten noch mit viel Overhead bei der Ausführung verbunden und erzeugten schwer verständliche Output-Dateien, sodass eine manuelle Analyse schnellere Erfolge zu verheißen schien, aber bei fortschreitender Komplexität immer mühsamer und ineffizienter wurde.

Mit Blackfire und Tideways sind nun kürzlich gleich zwei neue kommerzielle Profiler für PHP veröffentlicht worden, die sich zu den Open-Source-Lösungen Xdebug und Xhprof sowie der ebenfalls kommerziellen Lösung New Relic gesellen. Grund genug, diese Profiler unter die Lupe zu nehmen.

Profiling-Begriffe erklärt

Profiling kann man grundsätzlich auf zwei Arten durchführen: über statistische Auswertung (Sampling) oder Instrumentierung. Bei ersterer bleibt der Ablauf des ausgeführten Programms oder Skriptes unverändert, vielmehr hält ein Programm in periodischen Abständen über Betriebssystemaufrufe die Ausführung kurz an und zeichnet Messwerte zum aktuellen Codeblock und Ressourcenverbrauch auf. Diese Vorgehensweise

beeinträchtigt die Laufzeit des zu messenden Programms so gut wie gar nicht, kann aber nur stichprobenartige Mittelwerte liefern und insbesondere keinen Call Graph erzeugen (eine hierarchische Aufstellung des Codepfads mit genauen Ausführungszeiten). Daher verwenden viele Profiling-Tools, darunter alle hier vorgestellten, die Methode der Instrumentierung. Dabei werden in den Code Zählaufrufe injiziert, die genauere Messungen ermöglichen. Allerdings leidet darunter in der Folge die Ausführungsgeschwindigkeit, teilweise enorm. Ein guter Profiler sollte keinen zu großen Overhead produzieren, da dieser die Ergebnisse stark verzerrt und beispielsweise Nebenläufigkeitsprobleme gar nicht auffallen oder Timeouts auftreten, die es ohne Profiling nicht gegeben hätte, und er dadurch die Vergleichbarkeit der Messungen verhindert.

Von größtem Interesse ist zunächst die benötigte Zeit (Wall-Clock Time), die auf der Uhr an der Wand von Anfang bis Ende der Programmausführung verstreicht. Daneben kann noch die CPU-Zeit von Interesse sein – die Zeit, in der das Programm aktiv CPU-Ressourcen verbraucht hat, ohne die Sekunden, die zwischenzeitlich mit Warten auf I/O-Ressourcen oder Antwort von anderen Diensten verstrichen sind. Ebenfalls von Belang ist der benötigte Umfang an RAM, vor allem wenn es darum geht, Speicherlecks aufzuspüren.

Bei den Profiling-Ergebnissen wird oft zwischen interner und externer Zeit differenziert, verschiedene Profiler verwenden unterschiedliche Begriffe. Die interne oder exklusive Zeit besagt, wie lange die Funktion oder Methode für ihre Ausführung gebraucht hat. Externe oder inklusive Zeit bezeichnet den Ablauf von Anfang bis Ende der Methode, das heißt inklusive aller Aufrufe anderer Funktionen. Eine Funktion mit hoher externer, aber geringer interner Zeit ist daher gar nicht selbst der Verursacher des hohen Zeitverbrauchs, sondern von ihr aufgerufene Funktionen, die ihrerseits viel Zeit benötigen.

Demo-Anwendung für den Vergleich

Damit Interessierte die Werkzeuge unter gleichen Bedingungen testen und vergleichen können, haben die Autoren eine Demo-Anwendung mit dem PHP-Framework Symfony der Version 2.7 erstellt, die bei GitHub zur Verfügung steht (die URL ist über den blauen Balken „Alle Links“ zu finden). Mit dem ORM-Framework Doctrine wurden drei Entitäten erzeugt: *Person*, *Calendar* und *Entry* (Kalendereintrag). Faker (ebenfalls über den genannten blauen Balken zu erreichen) diente dazu, 50 Testdatensätze zu Personen mit insgesamt 200 Kalendern und 563 Kalendereinträgen zufällig zu erzeugen und in einer SQLite-Datenbank zu speichern. Dem MVC-Paradigma folgend lädt die Demo alle Personen zunächst in den Controller und übergibt die Daten an die View-Komponente. Dort erzeugt die Template-Engine Twig alle Einträge als HTML, liefert sie an den Controller wieder zurück, und der schickt sie an den Client (Webbrowser).

Zur Verdeutlichung sind bewusste, aber in der Praxis häufig beobachtbare Performance-Fehler eingebaut. Durch die Verwendung des Lazy Loading von Doctrine ORM erzeugt jeder Schleifenaufruf mit dem Zugriff auf die Eigenschaft einer Person mit einer Fremdschlüsselbeziehung eine erneute Datenbankabfrage, ebenso jeder Zugriff auf die Einträge eines Kalenders. Darüber hinaus erzeugt das Programm die Kalendereinträge als Sub-Template und bindet sie mit einer Twig-basierten *include*-Anweisung ein. Zudem ermittelt sie über einen Singleton-Ansatz die Anrede abhängig vom Geschlecht und gibt sie aus, aber durch einen Implementierungsfehler instanziiert sie die Singleton-Klasse bei jedem Aufruf erneut. Das sorgt für hundertfache SQL-Abfragen sowie ineffiziente Templating-Aufrufe und bietet damit eine brauchbare Grundlage, um auf die Suche nach Performance-Engpässen zu gehen.

Die Demo-Anwendung wird auf einem virtualisierten Server mit zwei CPU-Kernen und 2 GByte RAM betrieben. Als Betriebssystem ist Ubuntu 14.04.3 LTS installiert. Apaches HTTP-Server 2.4.7

dient als Webserver mit PHP 5.5.9 und aktiviertem Zend OPcache in der Version 7.0.3.

Xdebug

Historisch betrachtet das erste Werkzeug zum Aufspüren von Performance-Problemen in PHP-Code ist die Erweiterung Xdebug. Seit 2002 als Open-Source-Software verfügbar, liefert sie dem PHP-Entwickler viele wertvolle Informationen nebst aussagekräftigeren Fehlermeldungen und erlaubt das Debugging per Breakpoints. Zudem kann man den Profiler fallweise über den Request-Parameter *XDEBUG_PROFILE* aktivieren; das Programm erzeugt in diesem Fall Dateien, die mit dem C-Profiling-Werkzeug Cachegrind kompatibel sind. Installiert wird Xdebug als PHP-Extension über das PECL-Repository (PHP Extension Community Library).

QCacheGrind zeigt sich beim Auswerten der Demo-Anwendung auffallend bunt und ist eines der aussagekräftigeren Analysewerkzeuge (Abb. 1).
Die Auswertung erfolgt über eins der grafischen Werkzeuge zur Analyse von Cachegrind-Dateien, wie KCacheGrind für Linux, WinCacheGrind oder QCacheGrind für Windows. Je nach Werkzeug stehen mehr oder weniger intuitive und aussagekräftige Auswertungsmöglichkeiten zur Verfügung, so gibt es nicht für jedes Cachegrind-kompatible Programm eine Call-Graph-Darstellung. Die Analyse verlangt ein Einarbeiten in das jeweilige Visualisierungswerkzeug, aber mit einiger Erfahrung lassen sich die Gründe für die hohe Ausführungszeit der Demo-Anwendung finden.

Xdebug verwendet Instrumentierung zur Codeanalyse. Der größte Nachteil beim Einsatz ist der anfallende Overhead (siehe unten). Schon mit aktivierter Xdebug-Extension sinkt die Performanz rapide ab, beim Erstellen eines Profils beträgt sie nicht einmal mehr ein Viertel des ursprünglichen Wertes. Das erschwert nicht nur das Profiling selbst, an einen

Produktiveinsatz ist bei solch dramatischen Einbußen für den laufenden Betrieb der Anwendung nicht zu denken.

Xhprof

2009 veröffentlichte Facebook sein selbst entwickeltes Werkzeug zum Auffinden von Performanzschwachstellen als Open Source. Xhprof unterstützt Sampling sowie Instrumentierung und lässt so dem Anwender die Wahl zwischen geringem Overhead und genauer Messung. Die Möglichkeiten beim Sampling sind allerdings stark eingeschränkt, so ist das Sampling-Intervall fest auf 0,1 Sekunden eingestellt und liefert lediglich den Call-Stack zu den gemessenen Zeitpunkten. Außerdem sind die Flags zum Sammeln von CPU- und Speicherverbrauchsinformationen nicht verfügbar, sodass die einzigen Informationen darin bestehen, wie lange die gesamte Ausführung gedauert hat und in welcher Methode sich die Anwendung zu bestimmten Zeiten befand.

Wesentlich aussagekräftiger sind da schon die Informationen aus der Instrumentierung. Xhprof liefert zur Analyse gleich eine HTML-basierte Oberfläche mit, die die gewonnenen Informationen tabellarisch zeigt und Sortierung ermöglicht. Die Einbindung ist zunächst nicht besonders komfortabel, bis man die ersten Ergebnisse bekommt. Xhprof installieren Entwickler ebenfalls als PHP-Erweiterung (Extension), sie müssen sie aber im Code noch durch Aufrufe aktivieren. Ohne weitere Zusätze liefert Xhprof nur ein Array zurück, das Einträge zu Funktionen und je nach aktivierten Flags zwei bis fünf Messwerten zur Anzahl der Aufrufe, zu verstrichener Gesamtzeit, CPU-Zeit sowie Speicherverbrauch (Durchschnitts- und Spitzenwert) enthält.

Zur weiteren Verarbeitung müssen PHP-Verantwortliche Bibliotheken aus dem *utils*-Verzeichnis der Xhprof-Installation einbinden und das erstellte Profil wegschreiben. Beim Symfony-Framework besteht der Vorteil darin, dass alle Anfragen über einen zentralen Front-Controller laufen, in den man den

Xhprof-Code einfügen oder in dem man sogar einen eigenen Profiling-Front-Controller anlegen kann. Für die Demo haben die Autoren sich für Letzteres entschieden, um den Overhead einfach ermitteln zu können. Dieser ist ohne Profiling-Code vernachlässigbar, allerdings während des Profiling signifikant: Die Ausführungszeit verdoppelt sich.


Von Xhprof generierter Call Graph, deutlich erkennbar die rot eingefärbten Performance-Engpässe (Abb. 2)
Man kann außerdem eine tabellarische Darstellung ansehen, die alle Informationen übersichtlich aufbereitet und Details über jede Funktion per Klick auf ihren Namen liefert. Zunächst ist diese Liste auf die 100 meistaufgerufenen Funktionen bezogen, lässt sich aber beliebig erweitern. Bei installiertem Graphviz-Paket stellt Xhprof die Aufrufhierarchie als übersichtlichen Call Graph dar und färbt kritische Stellen gelb beziehungsweise rot ein. So sieht man auf den ersten Blick die beiden eingebauten Engpässe der Demo-Anwendung, an denen man mit Optimierungen beginnen würde. Xhprof bietet die Option, verschiedene Ausführungsläufe einander gegenüberzustellen und so die Verbesserungen (oder Verschlechterungen) übersichtlich anzusehen. Der sogenannte Diff-Report vergleicht zwei Läufe und listet die als Regressions und Improvements bezeichneten Verluste beziehungsweise Zuwächse an Performanz auf. Besonders interessant ist dabei der Diff-Call-Graph, der die Veränderungen grafisch darstellt.

Tideways

Tideways ist ein kommerzieller Realtime Profiler von der Qafoo GmbH für die PHP-Versionen 5.3 bis 5.6. Pünktlich zur Release von PHP 7 soll Tideways mit der neuen Version umgehen können. Zurzeit unterstützt sie nur Linux-Plattformen. Kontinuierlich sammelt Tideways Informationen zu HTTP- und I/O-Zugriffen, Datenbankabfragen und dem Cache. Diese Daten übermittelt das Werkzeug mit SSL-Verschlüsselung an die bei Qafoo in Berlin

stehenden Server. Persönliche Daten der Webseitenbenutzer überträgt die Software laut Tideways nicht. Bei Bedarf können Anwender diese Sicherheitseinstellungen ändern. Qafoo bereitet die Daten auf und stellt sie webbasiert zur weiteren Analyse zur Verfügung.

Nach dem kostenlosen Probezeitraum müssen Kunden einen volumenabhängigen Preis zahlen. Das minimale Paket kostet 40,00 Euro pro Monat. In diesem Paket hält Qafoo die Daten für 24 Stunden vor. Sollen sie länger zur Verfügung stehen, aggregiert Qafoo sie deutlich stärker.

Zu installieren bedeutet, die Tideways-Extension einzubinden und einen Daemon zu installieren. Die Software überwacht anschließend die gesamte Apache-PHP-Umgebung. Das Tideways Commandline Interface kann ein Entwickler nutzen, um einzelne Skripte im Dateisystem oder gezielt das Profiling eines HTTP-Aufrufs auszulösen. Optional kann er eine Chrome Extension nutzen, um die explizite Kontrolle über das Profiling und die damit verbundene Datengenerierung zu behalten. Schließlich landen die Daten auf dem Tideways-Server.

Tideways fasst alle gesammelten Informationen unter einer Transaktion zusammen, die einen spezifischen Programmablauf mit einem Namen identifiziert – mehrere HTTP-Requests an die gleiche URL unter derselben Transaktion. Dies ermöglicht unter anderem einen einfacheren Vergleich von Performance-Engpässen im Zeitablauf und eine Analyse der Gegenmaßnahmen. Den Transaktionsnamen legt eine automatisch ausgeführte Controller-Action-Kombination fest. Keiner Transaktion zugeordnete Daten fasst die Software im Sammelbecken *default* zusammen. Tideways unterstützt Zend Framework 1 und 2, Symfony2 Framework und Components, Shopware, Oxid, WordPress und Laravel. Weitere sollen folgen.

Standardmäßig führt das Programm für 10 % aller erfolgten HTTP-Requests ein Profiling durch. Diesen Wert kann man über einen Eintrag beispielsweise in *php.ini* individuell

konfigurieren. Die kontinuierliche Analyse des Systems ermöglicht es, im Bedarfsfall entsprechende Gegenmaßnahmen zeitnah einzuleiten. Tideways eignet sich folglich durchaus fürs Monitoring.

Die Webanwendung von Tideways (tideways.io) kann Organisationen, Anwendungen und Infrastrukturen logisch verwalten. Sie ist in der Lage, unterschiedliche Infrastrukturen mit derselben Anwendung zu analysieren und zu vergleichen. Beim Betrieb mehrerer PHP-Anwendungen in derselben Umgebung führt die Software alle Profiling-Ergebnisse unter derselben Rubrik auf. Alternativ kann man einen API-Key in der konkreten PHP-Anwendung konfigurieren. Dazu muss jemand den Quellcode der PHP-Anwendung um eine Zeile Quellcode erweitern. Auf der Einstiegsseite erhält der Benutzer eine Kurzübersicht über seine Anwendungen sowie aktuelle Informationen zur Reaktionszeit und aufgetretenen Fehlern. Exceptions und Fatal Errors erkennt das Tool ebenfalls und stellt sie mit Details in einer eigenen Rubrik dar.

Weitere Informationen zu einem Request können Anwender über eine Detailansicht abrufen. Grundsätzliche Indikatoren mit Angaben zu Gesamtdauer, Speicherverbrauch, Anzahl von Datenbankabfragen, Anzahl von HTTP-Requests und Wartezeit für I/O-Operationen zeigt Tideways an prominenter Stelle an. Es gruppiert Informationen nach *slow calls*, *controllers* und *views*, was bei dem verwendeten MVC-Framework Symfony2 sinnvoll ist.

Detaillierte Informationen über den Programmablauf einer Transaktion mit Angabe der einzelnen Funktionsaufrufe, ihre Dauer und Häufigkeit erhält man lediglich bei einem manuell ausgelösten Profiling. Dazu kann entweder die Chrome Extension, das Tideways Commandline Interface oder ein benutzerspezifischer GET-Parameter in der URL dienen.

Erst bei diesem Vorgehen stellt die Software den

Programmablauf durch einen Call Graph grafisch dar. Performance-Engpässe sind dort auf Anhieb erkennbar. Zusätzlich stehen für jeden Funktionsaufruf weitere Informationen zur Verfügung, etwa die Ausführungszeit (Wall Time) einer Funktion inklusive und exklusive der Zeit von Kinderfunktionen. Auch den Aufruf nativer PHP-Funktionen im Quellcode bietet die Software in der Übersicht. New Relic APM dagegen zeigt native PHP-Funktionen nicht an.

Vollständige SQL-Querys stellt Tideways zurzeit noch nicht dar, aber schon bald soll eine neue Version des Tools zur Verfügung stehen, die die vollständigen SQL-Querys anzeigt und PHP 7 unterstützt.

Blackfire

SensioLabs, die Macher des Symfony Frameworks, haben Ende Juli 2015 einen Profiler namens Blackfire in der Version 1.0 veröffentlicht. Das Werkzeug besteht aus mehreren Komponenten: Der Probe genannte Teil liefert als PHP-Erweiterung die Rohdaten der Analyse, ist aber nur aktiv, wenn ein Profil erstellt wird, und produziert im Übrigen keinen Overhead. Der Agent läuft als Daemon auf dem Server, aggregiert die Daten, bereitet sie auf und schickt sie an die Plattform blackfire.io zur Darstellung und Auswertung. Mit dem Companion, einer Erweiterung für den Chrome-Browser, können Anwender Auswertungen direkt vom Browser aus steuern.

Zum Einstieg bietet SensioLabs nicht nur einen begrenzten Testzeitraum für die Premium-Variante, sondern außerdem eine dauerhaft kostenlose Produktversion namens Hack an. Mit dieser lassen sich immerhin zwei Referenzprofile für die Dauer eines Tages speichern. Weitere Funktionen schlagen mit mindestens 82,50 Euro pro Monat für den Premium-Zugang zu Buche. Im Gegensatz zu Tideways misst Blackfire nicht automatisch, sondern nur bei expliziter Aktivierung. Die geschieht entweder über den Companion oder auf der Kommandozeile per *blackfire curl*. Der Companion nimmt Kontakt zur momentan besuchten

Website auf und versucht, eine Verbindung mit dem Blackfire-Konto des Benutzers herzustellen. Hierzu müssen Anwender aus Sicherheitsgründen zuvor die Server Credentials genannten Sicherheitstoken in der Konfiguration des Blackfire-Agenten hinterlegen. Der Companion initiiert danach die Datensammlung des Agenten, nimmt hierzu zehn Samples und ermittelt die Mittelwerte. Das ermöglicht Profiling auf Knopfdruck; nach dem Aktivieren des Profilers und einer kurzen Wartezeit bereitet Blackfire die Ergebnisse auf der blackfire.io-Plattform übersichtlich und interaktiv auf.

Über den Call Graph lassen sich sofort die kritischen Stellen erkennen und ein Klick auf die Funktionen öffnet Detailinformationen zu Zeit-, CPU- und Speicherverbrauch. Blackfire bietet zusätzlich Analysen zu Netzwerk- und SQL-Abfragen an, genau wie Tideways, aber im Gegensatz zu den Open-Source-Lösungen Xdebug und Xhprof. Diese Funktion steht allerdings nur in den kostenpflichtigen Premium- oder Enterprise-Versionen zur Verfügung. Ebenfalls aufpreispflichtig ist Team, eine Funktion, mit der sich Anwendungen von mehreren Benutzern profilen und die Ergebnisse im Team analysieren lassen. Allerdings können Anwender Team-Profile im Gegensatz zu Einzelprofilen nicht öffentlich über einen geheimen Link teilen, sodass man sich vorher über den Verwendungszweck im Klaren sein sollte.

Ein Vergleich zweier Profile in Blackfire nach Optimierung der ORM-Abfrage (Abb. 3)

Für den Vergleich zwischen Profiling-Aufrufen, insbesondere nach Änderungen am Code, gibt es die sogenannten Referenzprofile. Ein Profil kann Blackfire als Referenz speichern und erlaubt den Vergleich mit anderen aus der „Timeline“. Genauso wie bei Xhprof zeigt ein Call Graph die Änderungen visuell. Der Vorteil zum statischen PNG-Graphen von Xhprof ist hierbei die optionale Interaktion, da der Call Graph von Blackfire SVG nutzt. Man kann zoomen, den Ausschnitt verschieben sowie einzelne Teilbäume in den Fokus rücken und

beim Klick auf einen Knoten die Detailinformationen auf der linken Seite anzeigen lassen.

Als Vorteil von Blackfire stellt SensioLabs heraus, dass Kernfunktionen in PHP, die Entwickler ohnehin nicht optimieren können, im Vorhinein aggregiert sind und die Auswertung sich somit auf die interessanten Funktionsaufrufe beschränkt. Dies ist insbesondere im Vergleich zu Xdebug und Xhprof ein Vorteil, die alle Funktionen gleichberechtigt darstellen.

SensioLabs-Geschäftsführer Fabien Potencier sieht Profiling nach Unit-Tests als nächsten bedeutenden Schritt für Entwickler. Blackfire soll dabei unterstützen, indem es Profiling mit wenig Overhead produziert. Dieses Versprechen löst die Software ein, zumindest kann die Probe-Extension im Produktivbetrieb aktiviert bleiben. Beim Erstellen des Profils generiert Blackfire allerdings einen Overhead von circa 25 %, der die gemessenen Zeiten leicht verfälscht.

New Relic

New Relic gehört primär zur Klasse der Application-Performance-Management-Werkzeuge, bietet aber Funktionen für das Profiling von Anwendungen. Das Tool steht für die Plattformen PHP, Ruby, Java, .NET, Node.js und Python zur Verfügung und kann auf Linux- und Windows-Servern arbeiten. Um die Profiling-Funktion zu nutzen, muss der Kunde die Pro-Variante für 149 US-$ monatlich pro Host erwerben.

Zunächst muss man den sogenannten Agenten auf dem jeweiligen System installieren, der die Umgebung überwacht und alle relevanten Informationen sammelt. Jede Minute überträgt er die so gesammelten Daten an die Relic-Server. Erst die bereiten die Daten auf und stellen sie dem Benutzer webbasiert zur Verfügung. New Relic verarbeitet und speichert Kundendaten in den USA. Zusätzlich nimmt sich New Relic das Recht heraus, diese Daten auch außerhalb der USA zu verarbeiten. Dieses Recht ist in den AGB verankert, aber laut kürzlich erfolgter

Entscheidung des EuGH keine gesetzliche Ausnahme mehr.

Alle wichtigen Informationen, potenzielle Flaschenhälse, Fehler oder Störungen liefert das Dashboard. Die dargestellten Informationen können Anwender im Hinblick auf den betrachteten Zeitraum individuell auswählen. Ein Vergleich mit historischen Daten ermöglicht es, die Auswirkung von Änderungen am Quellcode zu bewerten.

New Relic APM fasst ebenfalls Informationen über einen HTTP-Request unter dem Begriff Transaktion zusammen. Eine Top-5-Anzeige der zeitaufwendigsten Transaktionen ermöglicht einen schnellen Einstieg in die Analyse. Auffällig ist, dass das Werkzeug keine Informationen zum Speicherverbrauch auf dieser Ebene anzeigt.

Die langsamsten Komponenten, üblicherweise Funktionsaufrufe, erscheinen in einer tabellarischen Übersicht – in der Demo der konkrete Klassenname, Funktionsname und deren Aufrufhäufigkeit. Mehr als ein Indiz dafür, dass der häufige Funktionsaufruf unnötig ist, gibt diese Information leider nicht her. Eine grafische Darstellung des Programmablaufs (Call Graph) sucht man vergebens. Alle Informationen zum Programmablauf stehen in einer baumartigen Tabelle. Verwendete PHP-Funktionen wie *sleep(5)*, die Ursache des Flaschenhalses sein könnten, stellt New Relic APM nicht dar. Die aufgeführten Informationen beziehen sich in der Demo-Anwendung immer auf Klassen und deren Funktionen.

Tabellarische Übersicht des Programmablaufs bei New Relic, deutlich erkennbar der rote Performance-Engpass (Abb. 4)
Aber nicht für jede Transaktion stehen Ablaufinformationen direkt zur Verfügung. Ob detailliertere Daten überhaupt gesammelt werden, hängt von dem zu konfigurierenden Schwellenwert Apdex T ab. Der beeinflusst den Wert, ab dem eine beobachtete Reaktionszeit als tolerierbar eingestuft

wird. Trifft das zu, löst New Relic APM keine Benachrichtigung beziehungsweise Warnung aus und sammelt keine weiteren Informationen zur Ablaufsteuerung.

Angaben zum Speicherverbrauch oder zu I/O-Zeiten gibt es nicht. Der Fokus von New Relic liegt hier bei der Darstellung von Antwortzeiten. Ob die Ursache einer schlechten Reaktionszeit eine komplexe Berechnung ist oder der wiederholte Zugriff auf das Dateisystem, kann nur der Blick auf den Quellcode zeigen.

Im Gegensatz zu den anderen Tools registriert New Relic nicht behandelte Exceptions und Warnings und stellt sie prominent im Dashboard dar – im günstigsten Fall außerdem inklusive der jeweiligen Stacktrace.

Die Verwendung von Profilern während der Entwicklung einer PHP-Anwendung ist sicherlich empfehlenswert, um frühzeitig eventuelle Flaschenhälse zu entdecken und geeignete Gegenmaßnahmen einzuleiten. Aber oft treten im produktiven Einsatz Situationen ein, die zuvor niemand in Betracht gezogen hat oder die aufgrund der wechselnden Betriebsumgebung außerhalb des Einflusses des Entwicklers liegen.

Profiling-Werkzeuge im produktiven Betrieb

Für eine grundsätzliche Aussage, ob der Einsatz eines der hier vorgestellten Profiling-Werkzeuge im produktiven Betrieb sinnvoll ist, haben die Autoren die Performance der Demo-Anwendung überprüft. Ausgehend von einer Konfiguration ohne Profiler haben sie den aktiven Einsatz jedes Profiling-Werkzeugs mit dem Apache HTTP Server Benchmarking Tool (*ab*, siehe wiederum den Balken „Alle Links“) verglichen.

Tideways und New Relic beeinflussen die Performance der PHP-Anwendung mit 11 und 13 % zwar erkennbar, aber dennoch so wenig, dass bei ausreichenden Systemressourcen ein Einsatz auf

dem produktiven System in Betracht gezogen werden kann. Vor allem wiegen die Vorteile des umfassenden Monitorings den geringen Overhead auf, aufkommende Schwierigkeiten lassen sich mit beiden Tools im Produktivsystem in Echtzeit erkennen und fördern so proaktives Handeln. Ein Einsatz von Blackfire im produktiven Betrieb ist trotz 27 % Overhead gerade noch denkbar. Zwar erstellt Blackfire nur ein Profil, sobald man es aktiv dazu aufruft, dennoch sollten Anwender in diesem Fall den Einfluss auf das System berücksichtigen. Ebenso verhält es sich mit Xhprof. Ein Einsatz von Xdebug auf einer produktiven Maschine ist angesichts des gewaltigen Overhead nicht empfehlenswert.

Fazit

Profiler sind wertvoll für die Analyse von Anwendungsperformanz. Mittlerweile steht eine Vielzahl an Profiling-Werkzeugen für PHP zur Verfügung. Welches am besten geeignet ist, kommt auf die jeweilige Anwendung und die zu beantwortenden Fragestellungen an. Jedenfalls braucht niemand mehr individuelle Zeitnahmen im Code selbstständig zu implementieren. Die Tools sind leicht zu installieren und Anwender können sie ohne oder mit wenigen Anpassungen am System sofort einsetzen. Selbst der Einsatz bestimmter Profiler auf Produktivumgebungen ist denkbar, um auftretende Störungen live analysieren zu können.

Die kostenlos verfügbaren Werkzeuge bieten schon einen ausreichenden Funktionsumfang und Entwickler können sie vollständig in der eigenen Umgebung einsetzen. Kommerzielle Produkte punkten mit Zusatzfunktionen wie SQL-Analyse, verbesserten Analysemethoden oder Teamfunktionen. Dafür verlangen die Dienste, dass die Anbieter die Daten nur auf ihren Plattformen verarbeiten und die Kunden nach Ablauf des Abonnements ihre Ergebnisse nicht mehr abrufen können. Wenn man einen solchen Service nutzt, sollte man sich zudem darüber im Klaren sein, dass die übermittelten Informationen Klassen-

und Methodennamen enthalten. Der Transport geschieht zwar verschlüsselt, und die Betreiber versprechen, dass sie diese Daten nicht weiterverarbeiten, aber bei manchem bleibt Skepsis. Wirkliche Wirtschaftsspionage ist mit diesen Informationen kaum realistisch, aber unter Umständen können Vertraulichkeitsvereinbarungen mit dem Kunden der Nutzung dieser Dienste einen Riegel vorschieben.

Blackfire in der Hack-Variante oder Xhprof eignen sich besonders, wenn erste Erkenntnisse gewonnen werden sollen – oder bei schmalem Budget. Tideways bietet fürs Geld übersichtliche Analysen und eine einfache Bedienung. APM-Werkzeuge wie New Relic gehen weit über das reine Profiling hinaus und eignen sich vor allem für Betreiber großer Webseiten. (hb) Jörn Wagner, Christian Ehmke arbeiten als Softwarearchitekten bei der Explicatis GmbH in den Bereichen Beratung, Konzeption und Realisierung von Web- und Anwendungssoftware.

[/expand]