Hochschule Wismar
Fakultät für Ingenieurwissenschaften
Bereich Elektrotechnik und Informatik

Master-Thesis

Untersuchung von opensource Datenbanksystemen auf Härtungsmaßnahmen unter Betrachtung des BSI IT-Grundschutz-Kompendium

Eingereicht am: 20. Juli 2020
von: Henner Bendig
geboren am 21. Februar 1991
in Uelzen

Aufgabenstellung

In dieser Arbeit sind verschiedene Opensource Datenbanksysteme (DBS) nach den Anforderungen des IT-Grundschutzkompendiums des Bundesministeriums für Sicherheit in der Informationstechnik (BSI) zu untersuchen.

Hierbei ist vorrangig das Kapitel „APP.4.3: Relationale Datenbanken“ zu betrachten, wobei die anzuwendenden Maßnahmen ebenso auf die Umsetzbarkeit in Nicht-Relationalen Datenbanken untersucht werden sollen.

Es sind weiterhin zu analysieren, welche technischen und organisatorischen Maßnahmen zur Härtung der Datenbanksysteme eingesetzt werden können.

Anhand von Beispiel-Datenbanken sind die Erkenntnisse prototypisch zu illustrieren.

Kurzreferat

Die Relevanz von Opensource Software (OSS) steigt immer weiter an. Laut einer Studie der Bitkom setzen heutzutage 69% aller Unternehmen, bewusst OSS ein und bis zu 75% der Unternehmen planen OSS einzusetzen [1]. Gleichzeitig steigt die Gefahr von Angriffen auf moderne Technologien [2]. Datenbanksysteme (DBS) stehen heute schon auf Platz 4 der am häufigsten angegriffenen IT-Systeme [3]. Die Wichtigkeit von Daten im heutigen Zeitalter steigt noch immer weiter man, somit ist eine noch höhere Angriffswahrscheinlichkeit auf DBS wahrscheinlich.

In dieser Arbeit wurde die Umsetzbarkeit des Bausteins „APP.4.3 – Relationale Datenbanken“ des IT-Grundschutzkompendium der Bundesagentur für Sicherheit in der Informationstechnik (BSI) in den Opensource DBS Oracle MySQL, PostgreSQL und MongoDB untersucht. Die drei gewählten Systeme sind die derzeit am meist verbreiteten Opensource Datenbanksysteme und bedürfen daher erhöhte Aufmerksamkeit [4] [5].

Organisatorische Maßnahmen wurden anhand von Empfehlungen, Vorgaben und Beispielen erarbeitet. Technische Maßnahmen wurden nach Möglichkeit direkt in den DBS umgesetzt. Sofern keine interne Umsetzung notwendig war, wurden Opensource Erweiterungen von Drittanbietern gesucht und getestet.

Anhand der vom BSI bereitgestellten Auditorencheckliste wurden die Maßnahmen bewertet. Organisatorische Maßnahmen wurden aus der Wertung herausgenommen. Es wurden die Punktekategorien 0 für gar nicht umsetzbar, 1 für teilweise umsetzbar und 2 für komplett umsetzbar vergeben. MySQL erreichte 73% der möglichen Punkte, PostgreSQL 67% und MongoDB 47%. Ein Großteil der notwendigen Funktionen wird vom Originalhersteller zwar bereitgestellt aber ist nur in der kommerziell verfügbaren Variante der Systeme aktivierbar.

Abstract

The relevance of Open Source Software (OSS) is constantly increasing. According to a study by Bitkom, 69% of all companies today use OSS consciously and up to 75% of companies plan to use OSS [1]. At the same time the danger of attacks on modern technologies is increasing [2]. Database systems (DBS) are already in 4th place among the most frequently attacked IT systems [3]. The importance of data in today's age is still increasing, so an even higher probability of attack on DBS is likely.

In this thesis the feasibility of the module ‚APP.4.3 - Relational Databases‘ of the IT basic protection compendium of the German Federal Agency for Information Security (BSI) was examined in the open source DBS Oracle MySQL, PostgreSQL and MongoDB. The three chosen systems are currently the most widespread open source database systems and therefore require special attention [4] [5].

Organisational requirements were developed on the basis of recommendations, specifications and examples. Where possible, technical requirements were implemented directly in the DBS. As far as no internal implementation was possible, open source extensions of third-party providers were searched and tested.

The requirements were evaluated using the auditor checklist provided by the BSI. Organizational requirements were removed from the evaluation. The point categories 0 for not realizable at all, 1 for partially realizable and 2 for completely realizable were assigned. MySQL achieved 73% of the possible points, PostgreSQL 67% and MongoDB 47%. Most of the required functions are provided by the original manufacturer but can only be activated in the commercially available version of the systems.

Inhalt

.

1 Einleitung

Laut dem Open Source Monitor 2019 der Bitkom sind 75% aller Unternehmen daran interessiert, mehr Open Source Software (OSS) einzusetzen, 69% der befragten Unternehmen setzen bereits bewusst OSS ein [1].

In den letzten Jahren ist eine stetige Zunahme der Wahrscheinlichkeit, dass unsere Welt durch technische Gefahren angegriffen wird, zu verzeichnen. So sind im Jahr 2009 noch gar keine und 2019 bereits zwei der weltweiten Top 5 Risiken technologischer Natur. Dies ist durch die steigende Relevanz moderner Technologien im privaten und beruflichen Umfeld zu begründen [2].

Datenbanken sind mit ca. 15% auf Platz 4 der häufigsten Ziele von Cyber Attacken im Jahr 2019 [3].

Somit ist die Absicherung von Datenbanksystemen aus dem Opensource-Bereich von steigender Bedeutung. Ein frei-verfügbarer Leitfaden dieses zu realisieren bietet die Bundesanstalt für Sicherheit in der Informationstechnik (BSI) mit dem IT-Grundschutz Kompendium. Dieses behandelt in einem Kapitel konkret den Schutz von Datenbanksystemen [6].

Eine Härtung eines Datenbanksystems umfasst neben der Datenbank die weiteren beteiligten Komponenten wie Hostsysteme und Netzwerk. Diese Punkte werden in dieser Arbeit nicht betrachtet.

2 Vorbetrachtungen und Grundlagen

In diesem Kapitel werden notwendige Grundlagen beschrieben, die für das Verständnis dieser Arbeit essenziell sind.

2.1 Datenbanken

Zum Speichern und Organisieren großer Datenmengen werden IT-gestützte Datenbanksysteme (DBS) eingesetzt [7]. Ein DBS besteht hierbei aus zwei Bestandteilen [8]:

- Datenbasis: Die gespeicherten Daten werden als Datenbasis bezeichnet.
- Datenbankmanagementsystem (DBMS): Die Sammlung an Programmen, die zur Verwaltung, Steuerung und den Zugriff auf die Datenbasis benötigt werden.

Die Organisation der Datenbasis wird prinzipiell in zwei Bereiche geteilt, die im Folgenden beschrieben werden.

2.1.1 Relationales Datenmodell

Das relationale Datenmodell basiert auf Tabellen, welche untereinander in Beziehungen (Relationen) zueinanderstehen. Diese Tabellen bestehen aus Zeilen und Spalten. Eine Zeile entspricht einem Datensatz, auch Tupel genannt. Eine Spalte entspricht einem Attribut. Jeder Datensatz muss eindeutig referenzierbar sein, dazu werden sogenannte Schlüsselattribute genutzt [9].

Abfragen auf relationale Datenbanken werden mittels der Abfragesprache „Structured Query Language“ (SQL) erstellt. Daher werden relationale Datenbanken auch häufig „SQL-Datenbanken“ bezeichnet [8]. Heutzutage wird jedoch diskutiert, ob diese Bezeichnung nicht irreführend ist, da der heutige SQL-Standard mehr erlaubt, als in der relationalen Algebra abgebildet wird [10].

2.1.2 Nicht-relationale Datenmodelle

Das Feld der nicht-relationalen-Datenbanken ist dadurch gekennzeichnet, dass es alle Datenbanksysteme umfasst die nicht oder nicht nur auf dem relationalen Datenmodell basieren. Sie werden auch „NoSQL“-Datenbanken genannt, das Kürzel steht für „Not only SQL“. NoSQL-Datenbanken können nach Moniruzzaman und Hossain [11] in vier Kategorien aufgeteilt werden:

- Key-Value Stores: Key-Value Stores basieren auf einer Schlüssel-Wert--Zuordnung. Ein Schlüssel, oftmals eine Zeichenkette, verweist auf einen bestimmten Satz an Attributen. Die Anzahl und Ausprägungen der Attribute eines Wertes müssen nicht mit denen eines anderen übereinstimmen [11].
- Dokumentenorientierte Datenbanken: Ganz ähnlich wie bei Key-Value Stores werden bei dokumentenorientierten Datenbanken einem Schlüssel, je ein Wert zugeordnet. Der Wert ist ein Dokument, oftmals ein JSON („JavaScript Object Notation“) oder ein anderer semi-strukturierter Dokumententypen. Vorteil dieser Variante ist, dass dem System die zu speichernden Daten vorher nicht bekannt gemacht werden müssen, sondern die Dokumente jederzeit angepasst werden können [11].
- Wide-Column Stores: Das zugrundeliegende Datenmodell gleicht dem relationalen Datenmodell, der Unterschied ist, dass eine Spalte eine beliebige Anzahl von Attributen zugeordnet sein kann. Zusammenhängende Spalten werden in Spaltenfamilien zusammengefasst. Der Zugriff erfolgt über Zeilen-Schlüssel [11].
- Graph-Datenbanken: Die Graph-Datenbanken basieren auf der mathematischen Graphentheorie. Datensätze bilden die Knoten, die Verbindungen unter den Knoten sind als Kanten definiert. Dieses Modell orientiert sich an realen Objekten, ähnlich der Objektorientierten Programmierung. Dies ermöglicht deutlich komplexere Beziehungstypen, als im relationalen Datenmodell möglich sind [11] [9].

2.2 Auswahl der zu untersuchenden Datenbanksysteme

Kern dieser Arbeit ist die Untersuchung der Anwendbarkeit des IT-Grundschutz-Kompendiums auf Opensource Datenbanksysteme sein. Zu diesem Zweck werden die drei meistgenutzten Systeme betrachtet. Hierzu wird die Ranking-Webseite „db-engines.com“[1] genutzt. Der Betreiber der Webseite, „solidIT consulting & software development gmbh“[2] wertet hierzu folgende Kriterien aus:

- Anzahl der Erwähnungen des Systems auf Webseiten, mittels der Suchmaschinen Google[3], Microsoft Bing[4] und Yandex[5]
- Interesse in das System, mittels Google Trends[6]
- Häufigkeit von technischen Diskussionen über das System auf den Portalen „Stack Overflow“ [7]und „DBA Stack Exchange“[8]
- Anzahl an Job-Angeboten auf „Indeed“ [9]und „Simply Hired“[10], in denen das System erwähnt wird

[1] https://db-engines.com/; solidIT consulting & software development gmbh

[2] http://solid-it.at/; solidIT consulting & software development gmbh

[3] https://www.google.com/; Google Ireland Limited

[4] https://www.bing.com/; Microsoft Corporation

[5] https://yandex.com/; Yandex N.V.

[6] https://trends.google.com/; Google Ireland Limited

[7] https://stackoverflow.com/; Stack Exchang, Inc.

[8] https://dba.stackexchange.com/; Stack Exchange, Inc.

[9] https://indeed.com/; Indeed Ireland Operations Limited

[10] https://www.simplyhired.de/; SH, Inc.

- Anzahl an Personen-Profilen in den Job-Netzwerken „LinkedIn“[11] und „Upwork“[12], in denen das System erwähnt wird
- Relevanz in dem sozialen Netzwerk „Twitter“ [13]anhand der Erwähnungen des Systems

Die Auswertung ergibt für die Monate Februar und März 2020, sowie März 2019, sichtbar in Tabelle 1, das die Top 10 Datenbanken folgende sind, in absteigender Reihenfolge: Oracle, MySQL, Microsoft SQL Server, PostgreSQL, MongoDB, IBM DB2, Elasticsearch, Redis, Microsoft Access, SQLite. Hiervon sind entsprechend die Top 3 Opensource Datenbanken folgende: MySQL, PostgreSQL und MongoDB [4].

Tabelle 1 - Ranking über die meistgenutzten Datenbanksysteme aus einer ausgewerteten Menge von 250 unterschiedlichen Systemen. Stand März 2020. Quelle https://db-engines.com

Rang			Datenbanksystem
März 2020	Februar 2020	März 2019	
1.	1.	1.	Oracle
2.	2.	2.	MySQL
3.	3.	3.	Microsoft SQL-Server
4.	4.	4.	PostgreSQL
5.	5.	5.	MongoDB
6.	6.	6.	IBM DB2
7.	7.	8.	Elasticsearch
8.	8.	7.	Redis
9.	9.	9.	Microsoft Access
10.	10.	10.	SQLite

[11] https://linkedin.com/; LinkedIn Ireland Unilimited Company

[12] https://www.upwork.com/; Upwork Glocal Inc.

[13] https://twitter.com/; Twitter, Inc.

Das Vergleichsportal „TrustRadius“[14] erstellt durchgängig Vergleiche auf Grundlage von Softwarebewertungen, die durch Nutzer auf dem Portal veröffentlicht werden. Hierbei gibt es einen dedizierten Vergleich von Opensource Datenbanken. Zum Zeitpunkt dieser Arbeit im Mai 2020 ergibt diese Auswertung ebenso, dass die drei Systeme MySQL, PostgreSQL und MongoDB die am meist verbreiteten sind [5].

Auf diese drei Datenbanken wird in dieser Arbeit Bezug genommen.

Weitere mögliche Datenbank-Vergleiche, mit anderen Bewertungsgrundlagen, wären z.B. die Studien des Marktforschungsunternehmens „Forrester“ [12], welcher u.a. zu den Themen „NoSQL BigData Datenbanken“ [13] und „Databases-as-a-Service“ [14] Auswertungen vorgenommen haben.

2.2.1 MySQL Community

MySQL Community (im Folgenden „MySQL“) ist ein relationales Datenbanksystem, welches 1994 vom schwedischen Unternehmen „MySQL AB“ unter der Lizenz „GNU General Public License“ (GNU GPL) veröffentlicht wurde. Seit 2010 gehört MySQL zu Oracle Corporation [15].

2.2.2 PostgreSQL

PostgreSQL wird als objektrelationales Datenbanksystem bezeichnet, was einer Mischung aus dem objektorientierten und dem relationalen Datenmodell entspricht. 1986 wurden die ersten Teile des POSTGRES-Projektes an der University of California entwickelt. Das System steht unter der „PostgreSQL License“, welche nach eigenen Angaben der MIT- und der BSD-Lizenz entspricht [16].

[14] https://www.trustradius.com/, TrustRadius Inc.

2.2.3 MongoDB Community Server

Der MongoDB Community Server (im Folgenden „MongoDB“) gehört zu der Gruppe der Dokumentenbasierten Datenbanken, welche 2009 durch das Unternehmen MongoDB Inc. veröffentlicht wurde. Seit 2018 steht MongoDB unter der Lizenz „Server Side Public License“ (SSPL) [17].

2.3 Der IT-Grundschutz

Um einen einheitlichen Standard für die Informationssicherheit zu schaffen, hat das BSI im Jahr 1994 den IT-Grundschutz geschaffen. Dieser bietet ein systematisches Vorgehen für Unternehmen um „das Niveau der Informationssicherheit in Behörden und Unternehmen jeder Größenordnung zu erhöhen“ (vgl. [18]).

Im Jahr 2017 wurde der IT-Grundschutz grundlegend überarbeitet und aktualisiert. Das neue System umfasst zwei große Bereiche: „die BSI-Standards“ und „das IT-Grundschutz-Kompendium“, welche im Folgenden erläutert werden.

2.3.1 Die BSI-Standards zum IT-Grundschutz

Die BSI-Standards sind das Fundament des IT-Grundschutzes und beinhalten Methoden, Prozesse, Verfahren, Vorgehensweisen und Maßnehmen zu vielen Themen aus dem Bereich IT-Sicherheit. Die Standards sind wie folgt bezeichnet:

- BSI-Standard 200-1 – „Managementsysteme für Informationssicherheit“ [19]
- BSI-Standard 200-2 – IT-Grundschutz-Methodik [20]
- BSI-Standard 200-3 – Risikoanalyse auf Basis von IT-Grundschutz [21]
- BSI-Standard 100-4 – Notfallmanagement [22]

Die Nummern 200-X sind die bereits aktualisierten Versionen, der Standard 100-4 ist derzeit noch nicht aktualisiert veröffentlicht. Die Standards sind kompatibel zur Norm ISO/IEC 27001[15] und äquivalent zur Norm ISO/IEC 27002[16] und somit auch international anerkannt [23].

Zusätzlich gibt es ein weiteres Dokument, den „Leitfaden zur Basisabsicherung nach IT-Grundschutz“. Dies ist eine Einführung in die Umsetzung der BSI-Standards [24].

2.3.2 Das IT-Grundschutz-Kompendium

Das Kompendium bildet den konkreten Umsetzungsleitfaden und dient als Nachschlagewerk für den Umsetzungsverantwortlichen für IT-Sicherheit. Anhand der Bausteine werden unterschiedliche Maßnahmen für verschiedene Bereiche konkret beschrieben [6] [7]. Das IT-Kompendium wird seit der Umstrukturierung 2017 jährlich im Februar aktualisiert und bietet damit ein sich regelmäßig-aktualisierendes, umfassendes Handbuch.

Derzeit ist das Kompendium in drei Bereiche aufgeteilt:

- Allgemeine Gefährdungen: in diesem Abschnitt werden unterschiedliche Gefahren aus der Informationstechnik beschrieben
- Prozess-Bausteine: dieser Abschnitt beschreibt Maßnahmen für die Absicherung von Prozessen innerhalb einer Organisation
- System-Bausteine: dieser Abschnitt beschreibt Maßnahmen für konkrete Hardware- und Software-Systeme

[15] ISO/IEC 27001: IT-Sicherheitsverfahren – Informationssicherheits-Managementsysteme – Anforderungen

[16] ISO/IEC 27002. IT-Sicherheitsverfahren – Leitfaden für das Informationssicherheits-Management

Aktuell sind 10 Bausteine enthalten:

- ISMS: Sicherheitsmanagement
- ORP: Organisation und Personal
- CON: Konzeption und Vorgehensweise
- OPS: Betrieb
- DER: Detektion und Reaktion
- APP: Anwendungen
- SYS: IT-Systeme
- IND: Industrielle IT
- NET: Netze und Kommunikation
- INF: Infrastruktur

2.4 Das Testsystem

In diesem Kapitel wird das Testsystem beschrieben.

2.4.1 Das Hostsystem

Alle Untersuchungen wurden mit dem Betriebssystem CentOS Linux 8 (Core), Release 8.1.1911, durchgeführt.

2.4.2 Die Anwendungssysteme

Die unterschiedlichen Datenbanksysteme wurden mit folgenden Versionen auf dem Hostsystem installiert.

MySQL
Die eingesetzte Version ist 8.0.20 MySQL Community Server – GPL.

PostgreSQL
Die eingesetzte Version ist 12.2.

MongoDB
Die eingesetzte Version ist MongoDB 4.2.6 Community.

2.4.3 Die Beispieldatenbank

Um die Umsetzung besser zu erklären, ist für einige Anforderungen das Vorhandensein spezifischer Daten notwendig, Daher wurde Beispieldaten ausgewählt, um die Beschreibung zu vereinfachen. Die Konzepte sind nicht spezifisch für diese Daten, sondern lassen sich auf andere Daten übertragen.

Als Beispieldatenbank wird relationale Schulungsdatenbank von Prof. Dr. Kemper[17], eingesetzt. Das Entity Relationship Diagram (ERD) zu dieser Datenbank ist dem Anhang (9A.1 Beispieldatenbank) zu entnehmen. Die Datenbankenschemata sind für MySQL und PostgreSQL ebenfalls im Anhang hinterlegt.

Für MongoDB wurde die Datenbank in eine dokumentenorientierte Variante konvertiert. Eine Tabelle entspricht einer Collection, ein Datensatz ist ein Dokument. Ein Datenbank-Dump dieses Aufbaus ist im Anhang hinterlegt.

[17] Prof. Alfons Kemper, PhD; Technische Universität München

3 Umsetzung der vom BSI geforderten Maßnahmen

Zur Umsetzung der Maßnahmen wird im Folgenden zuerst die Maßnahme im Originalwortlaut beschrieben, anschließend für jedes der zu untersuchenden Systeme ein Umsetzungsvorschlag gemacht. Organisatorische Maßnahmen, die sich nicht auf eine technische Implementierung beziehen, werden allgemein gültig beschrieben und nicht auf ein konkretes System bezogen.

Die Maßnahmen sind unterteilt in „Basis-Anforderungen“, „Standard-Anforderungen“ und „Anforderungen für erhöhtem Schutzbedarf“. Zur eindeutigen Zuordnung wird die Nummerierung der Anforderungen aus dem Grundschutz-Kompendium mit aufgeführt.

3.1 Basis-Anforderungen

Laut BSI sind die Basis-Anforderungen wie folgt beschrieben: „das Minimum dessen […], was vernünftiger Weise an Sicherheitsvorkehrungen umzusetzen ist“ (vgl. [6]).

3.1.1 APP.4.3.A1 – Erstellung einer Sicherheitsrichtlinie für Datenbanksysteme

Anforderungsbeschreibung

„Ausgehend von der allgemeinen Sicherheitsrichtlinie der Institution MUSS eine spezifische Sicherheitsrichtlinie für Datenbanksysteme erstellt werden. Darin MÜSSEN nachvollziehbar Anforderungen und Vorgaben beschrieben sein, wie Datenbanksysteme sicher betrieben werden sollen. Die Richtlinie MUSS allen im Bereich Datenbanksysteme verantwortlichen Mitarbeitern bekannt sein. Sie MUSS grundlegend für ihre Arbeit sein. Wird die Richtlinie verändert oder wird von den Anforderungen abgewichen, MUSS dies mit dem ISB abgestimmt und dokumentiert werden. Es MUSS regelmäßig überprüft werden, ob die Richtlinie noch korrekt umgesetzt ist. Die Ergebnisse MÜSSEN sinnvoll dokumentiert werden.“

Umsetzungsvorschlag

Die meisten Unternehmen führen eine allgemeine IT-Sicherheitsrichtlinie ein. Im Rahmen dieser Arbeit wurde eine spezifische Richtlinie für ein fiktives Unternehmen formuliert, welche sich im speziellen mit Datenbanksystemen beschäftigt. Für die Umsetzung wurde sich an den Vorgaben und Beispielen des BSI [24] [25] [26] und exemplarisch an Veröffentlichungen anderer Unternehmen orientiert [27] [28].

Die ausgearbeitete Sicherheitsrichtlinie befasst sich mit dem Umgang mit Datenschutz, dem Stellenwert der Informationssicherheit für DBS, dem Einführungsprozess der Sicherheitsrichtlinie, den Verantwortlichkeiten und der Durchsetzungsdringlichkeit, sowie den einzuführenden Maßnahmen. Der komplette Wortlaut der Richtlinie ist der Anlage 9A.4.1 zu entnehmen.

3.1.2 APP.4.3.A2 – Installation des Datenbankmanagementsystems

Anforderungsbeschreibung

„Es MUSS sichergestellt sein, dass die Installationspakete des Datenbankmanagementsystems aus sicheren Quellen stammen. Bereits veröffentlichte Patches MÜSSEN eingespielt werden, bevor das DBMS betrieben wird.“

Umsetzungsvorschlag

Wenn neue Installations- oder Update-Pakete heruntergeladen werden, sollte unbedingt darauf geachtet werden, dass diese nur vertrauenswürdigen Quellen bezogen werden. Prädestiniert sind hier in jedem Fall die Hersteller selbst, die ein geeignetes Download-Portal bereitstellen. Die Authentizität der Webseite sollte mindestens über ein gültiges SSL-Zertifikat geprüft werden.

Einige Hersteller bieten zusätzlich zu dem Download eine Checksumme an (siehe Tabelle 2), anhand derer die Integrität der Downloaddatei geprüft werden kann. Sofern diese Möglichkeit geboten ist, sollte diese auch genutzt werden. Die Arten der Integritätsprüfungen können der Anlage 9A.4.2 entnommen werden. MySQL und MongoDB bieten eine qualifizierte Überprüfung der Setupdateien an, PostgreSQL hingegen nicht.

Ein automatisches Updaten aus den entsprechenden Quellen wird von keiner der untersuchten DBS angeboten.

Tabelle 2 - Übersicht der Download- und Integritätscheck-Möglichkeiten der DBS [29] [30]

	Download-portal	**Gültiges SSL-Zertifikat**	**Integritätscheck vorhanden**
MySQL	Download MySQL	JA	JA MD5, GPG, RPM (MD5+GPG)
PostgreSQL	Download PostgreSQL	JA	NEIN
MongoDB	Download MongoDB	JA	JA Linux und OSX: GP/GPG, SHA-256 Windows: SHA256

3.1.3 APP.4.3.A3 – Basishärtung des Datenbankmanagementsystems

Anforderungsbeschreibung

„Das Datenbankmanagementsystem MUSS gehärtet werden. Hierfür MUSS eine Checkliste mit den durchzuführenden Schritten zusammengestellt und abgearbeitet werden. Auch MÜSSEN alle Passwörter entsprechend den internen Anforderungen der Institution geändert werden. Alle Passwörter MÜSSEN verschlüsselt gespeichert werden. Die Basishärtung MUSS regelmäßig überprüft und, falls erforderlich, angepasst werden.“

Umsetzungsvorschlag

Der Konzern Deutsche Telekom AG (Telekom) betreibt seit 2010 das „Privacy and Security Assessment“-Verfahren (PSA-Verfahren) um alle Systeme innerhalb des Konzerns nach einem einheitlichen Sicherheitsstandard bewerten zu können. Ziel des PSA-Verfahrens ist es, ein adäquates Sicherheits- und Datenschutzniveau für alle Systeme zu gewährleisten und ist für alle Unternehmen des Konzerns verpflichtend. Mittlerweile hat das Verfahren das Zertifikat der ISO 27001. Um ein System nach dem PSA-Verfahren freizugeben, stellt die Telekom Anforderungskataloge („Statement of Compliance“, sogenannte „SoC-Listen“) für unterschiedlichste Systeme und Anwendungen bereit [31].

Das Center for Internet Security (CIS) ist ein US-Amerikanischer, Non-Profit – Verein, mit dem Ziel private und öffentliche Organisationen gegen Cyberangriffe zu sichern. Hierzu veröffentlicht der Verein u.a. regelmäßig kostenfreie Handlungsanweisungen, sogn. „CIS Benchmarks“, für unterschiedliche Systeme, sowie vorbereitete Images zu unterschiedlichen Systemen, auf denen diese Anweisungen bereits umgesetzt sind. Diese Anweisungen beinhalten „Best Practise“-Vorschläge verschiedener Unternehmen und Sicherheitsexperten zur Härtung der Systeme. Beteiligte Organisationen sind z.B. die „Information Systems Audit and Control Association“ (ISACA) und das „International Information Systems Security Certification Consortium“ (ISC2) [32] [33].

Auf Grundlage dieser zwei Beispiele von Telekom und CIS, wurde eine kombinierte Checkliste entworfen. Die Checkliste gliedert sich in die Bereiche „Allgemeine Anforderungen“, „Anforderungen an MySQL“, „Anforderungen an PostgreSQL“ und „Anforderungen an MongoDB“.

Die Passwörter müssen einer angemessenen Komplexität entsprechen. Aktuelle Forschungen und Empfehlungen [34] [35] [36] sehen wie folgt aus:

- Mindestens acht Zeichen lang
- Mindestens ein Zeichen aus jeder dieser Zeichengruppen muss enthalten sein:
    - Kleinschreibung
    - Großschreibung
    - Ziffern
    - Sonderzeichen
- Es dürfen keine bekannten Phrasen genutzt werden

Umsetzen ließen sich die Richtlinien mit internen Mitteln bei MySQL. Bei PostgreSQL mit einem zusätzlichen Plugin (Crackslib) oder einer Individualentwicklung. MongoDB bot keine Möglichkeit zur Umsetzung.

Die Verschlüsselung der Passwörter für Systemnutzer ist in allen drei Datenbanken als Funktion implementiert und standardmäßig aktiviert.

Die formulierten Checklisten, sowie Umsetzungsdetails zu der Passwortsicherheit sind Anlage 9A.4.3 zu entnehmen.

3.1.4 APP.4.3.A4 – Geregeltes Anlegen neuer Datenbanken

Anforderungsbeschreibung

„Neue Datenbanken MÜSSEN nach einem definierten Prozess angelegt werden. Wenn eine neue Datenbank angelegt wird, MÜSSEN Grundinformationen zur Datenbank nachvollziehbar dokumentiert werden.“

Umsetzungsvorschlag

Zur Umsetzung dieser Anforderung wurde ein Stammdatenblatt entworfen, welches die fachlichen, technischen und sicherheitstechnischen Punkte aufnimmt. Es wurde ein Prozess definiert, welcher die beteiligten Personen einbindet und ein geregeltes Vorgehen zur Definition der Daten im Stammblatt vorgibt. Beides wurde der Anlage 9A.4.4 angefügt.

Die beteiligten Personen ließen sich auf vier Profile herunterbrechen:

- Anfragender / Nutzer: Die fachliche Person, in der Regel ein Mitarbeiter einer Abteilung, hat den Bedarf an einer neuen Datenbank durch einen Geschäftsvorfall oder eine neue Aktion.
- Fachlich Verantwortlicher: Die dem Nutzer vorgesetzte Person, die den Geschäftsvorfall fachlich verantwortet und die Kosten rechtfertigen muss.
- Technisch Verantwortlicher: Ein Mitarbeiter aus der Abteilung „Informationstechnik“, der für die technische Umsetzung der Anforderungen aus dem Geschäftsbereich zuständig ist.
- Sicherheit Verantwortlicher: Der ISB oder eine von dem ISB beauftragte Person, die die Informationssicherheit für diesen Geschäftsbereich verantwortet.

Dieser Prozess sollte im Unternehmen mit technischen Mitteln umgesetzt werden, die das Weiterleiten und Benachrichtigen der beteiligten Personen automatisiert vornimmt. Auch sollte eine Stellvertreter-Regelung implementiert werden.

3.1.5 APP.4.3.A5 – Benutzer- und Berechtigungskonzept

Anforderungsbeschreibung

„Das Benutzer- und Berechtigungskonzept der Institution MUSS um die für Datenbankmanagementsysteme notwendigen Berechtigungen für Rollen, Profile und Benutzergruppen erweitert werden. Es MUSS ein Prozess etabliert werden, der regelt, wie Datenbankbenutzer und deren Berechtigungen angelegt, genehmigt, eingerichtet, modifiziert und wieder entzogen bzw. gelöscht werden. Dabei DÜRFEN immer NUR so viele Zugriffsrechte vergeben werden, wie für die jeweiligen Aufgaben erforderlich sind (Need-to-know-Prinzip).

Alle Änderungen SOLLTEN dokumentiert werden. Die eingerichteten Benutzer und die ihnen zugeordneten Berechtigungen MÜSSEN regelmäßig überprüft und, falls erforderlich, angepasst werden.“

Umsetzungsvorschlag

Es wurde ein rollenbasiertes Berechtigungskonzept (Role Based Access Control, RBAC) implementiert. RBAC ist seit 2004 ANSI/INCITS Standard [37] und kombiniert die zwei Sicherheitsprinzipien „Least Privileged“ und „Segregation of Duties“ (Funktionstrennung) [38]. RBAC zeichnet sich dadurch aus, dass einem Benutzer oder einer Gruppe nicht direkt Berechtigungen in einem System zugeordnet werden, sondern dass die Berechtigungen einer „Rolle“ zugewiesen werden. Diese Rolle kann dann wiederrum einer spezifischen Person A zugeordnet werden. Sollen die Berechtigungen an eine andere Person B weitergegeben werden, wird die Rolle A entzogen und B zugeordnet.

RBAC lässt sich nach Sandhu et al. [39] in vier Kategorien unterteilen, wobei sich die Kategorien oberflächlich wie folgt beschreiben lassen:

- RBAC 0 ist das Kernmodell von RBAC lässt die Definition von Nutzern, Rollen, Berechtigungen und die Zuordnung dieser drei Elemente untereinander zu.

- RBAC 1 inkludiert RBAC 0 und lässt zusätzlich die Möglichkeit von Hierarchien zu, d.h. eine Rolle kann von einer anderen Rolle erben und diese erweitern.
- RBAC 2 inkludiert RBAC 0 und lässt zusätzlich die Definition von Einschränkungen zu. Einschränkungen bedeuten, dass festgelegt werden kann, dass ein Nutzer, der die Rolle A hat, nicht zusätzlich die Rolle B haben darf.
- RBAC 3 ist die Kombination von RBAC 1 und 2 und lässt damit die Definition von Hierarchien und Einschränkungen zu.

Die Rollen beziehen sich auf fachspezifische Nutzerprofile, die der ISB zusammen mit den Fachverantwortlichen erarbeiten muss. Die Erstellung und Einführung dieses Richtlinienkonzeptes bringen zwar zunächst einen hohen Aufwand mit sich, dieser zahlt sich aber schnell in Sicherheit und Betrieb wieder aus. Beispielsweise können die Rollenbeschreibungen anhand der Stellenbeschreibung passieren. So kann schnell anhand personeller Veränderungen geprüft werden, ob ein Nutzer die Berechtigung weiterhin benötigt [40].

Zur Umsetzung der Anforderung wurden Prozesse und Anträge modelliert, welche die geforderten Abläufe darstellen:

- Neuen Benutzer anlegen
- Berechtigung hinzufügen / entziehen / ändern
- Benutzer entfernen

Ein einfaches RBAC-Konzept ist in folgenden Szenarien beschrieben:

„Annette Müller (Nutzername „amueller“) arbeitet im Prüfungsamt und soll auf die Tabellen „Professoren“ und „Assistenten“ lesend und auf die Tabelle „pruefen“ schreibend zugreifen.“

„Ernst Wagner (Nutzername „ewagner“) arbeitet in der Personalabteilung, Abteilung tarifliche Angestellte und soll auf die Tabelle „Professoren“ lesend, auf die Tabelle „Assistenten“ schreibend und auf die Tabelle „pruefen“ gar nicht zugreifen können.“

Es wurden unterschiedliche Rollen mit unterschiedlichen Rechten definiert:

- Rolle „PersonalEinsicht“:
  - Tabelle „Professoren“: Lesen
  - Tabelle „Assistenten“: Lesen
- Rolle „Pruefungsamt“
  - Tabelle „pruefen“: schreiben
- Rolle „PersonalTariflich“
  - Tabelle „Assistenten“: schreiben

Die Rollen werden folgendermaßen zugeordnet:

- amueller:
  - PersonalEinsicht
  - Prüfungsamt
- ewagner:
  - PersonalEinsicht
  - PersonalTariflich

Anschließend wurden weitere Anforderungen definiert, um die die RBAC 1 und 2 Kriterien zu testen.

„Alle Mitarbeiter aus der Abteilung „Personalabteilung Professoren“ sollen zusätzlich in der „Personalabteilung tarifliche Angestellte“ eingesetzt werden, daher sollen alle Inhaber der Rolle „PersonalTariflich“ die Berechtigungen der Rolle „PersonalProfessoren“ erhalten.“

„Mitarbeiter aus der Abteilung Personal Tariflich sollen niemals die Berechtigungen der Abteilung Prüfungsamt erhalten.“

Dazu soll eine weitere Rolle “PersonalProfessoren” angelegt werden und diese der Rolle „PersonalTariflich“ zugewiesen.

Außerdem soll die Einschränkung definiert werden, dass die Rolle „PersonalTariflich“ nicht mit der Rolle „Prüfungsamt“ vergeben werden darf.

Mit den Implementierungen kann gezeigt werden, dass die Kriterien für RBAC 0, 1, 2 und 3 eingehalten werden. In MySQL, PostgreSQL und MongoDB konnten die Anforderungen bis RBAC 1 erfüllt werden. RBAC 2 ist zum Zeitpunkt der Umsetzung in keinem der Systeme möglich gewesen.

Das regelmäßige Überprüfen der Datenbankberechtigungen sollte außerhalb des DBS selbst erfolgen. Daher ist der Eingriff von extern notwendig. Nichtsdestotrotz kann das System hierbei unterstützen. Beispielweise kann ein regelmäßiges Event dafür sorgen, das automatisch zu einem bestimmten Zeitpunkt eine Liste der Nutzer, samt den zugehörigen Rechten, ausgegeben wird.

Die modellierten Prozesse und die Umsetzung des Rechtesystems sind im Anhang, Abschnitt 9A.4.5, dargestellt.

3.1.6 APP.4.3.A6 – Passwortänderung

Anforderungsbeschreibung

„Alle Passwörter der Datenbankbenutzer MÜSSEN der Passwortrichtlinie der Institution entsprechen. Es MUSS gewährleistet sein, dass die Passwörter beim geringsten Verdacht eines Sicherheitsvorfalles geändert werden. Insbesondere bei privilegierten Datenbankaccounts und Dienstkonten SOLLTE ein Passwortwechsel sorgfältig geplant und gegebenenfalls mit den Anwendungsverantwortlichen abgestimmt werden.“

Umsetzungsvorschlag

Eine Vorgabe zur Komplexität von Passwörtern wurde bereits im Abschnitt „3.1.3 APP.4.3.A3 – Basishärtung des Datenbankmanagementsystems“ festgelegt. Aus dem Anforderungstext lassen sich weitere Punkte ableiten, die in den jeweiligen DBS umgesetzt werden sollen:

- Regelmäßiges Erneuern von Passwörtern
- Wiederbenutzung von alten Passwörtern
- Ungültig setzen aller Passwörter im System

In den MySQL konnten die Anforderungen ohne größeren Aufwand umgesetzt werden. PostgreSQL ermöglichte die Implementierung über Umwege zwei der drei Anforderungen. MongoDB hingegen bot keine dieser Funktionalitäten.

Die Umsetzungen dieser Anforderungen und Beschreibungen dazu sind im Anhang 9A.4.6 dargestellt.

3.1.7 APP.4.3.A7 – Zeitnahes Einspielen von Sicherheitsupdates

Anforderungsbeschreibung

„Vorhandene Sicherheitsupdates für das Datenbankmanagementsystem und das Betriebssystem MÜSSEN zeitnah installiert werden. Vorab MUSS auf einem Testsystem überprüft werden, ob die Sicherheitsupdates kompatibel sind und keine Fehler verursachen. Bevor ein Patch eingespielt wird, MUSS das Datenbanksystem gesichert werden. Zusätzlich MUSS eine verantwortliche Rolle definiert werden, die sich regelmäßig über bekannte Sicherheitslücken des Datenbankmanagementsystems sowie über verfügbare Sicherheitsupdates informiert. Des Weiteren MUSS geprüft werden, ob die Update-Intervalle des Datenbankmanagementsystems auf die Update-Zyklen des Herstellers abgestimmt werden können. Das Ergebnis SOLLTE nachvollziehbar dokumentiert werden.“

Umsetzungsvorschlag

Die Aufgabe zur Überprüfung auf neue Sicherheitsupdates sollte pro Datenbanksystem einer bestimmten Rolle zugewiesen werden, somit hat ein Mitarbeiter die Rolle „Securitymanager MySQL“, „Securitymanager PostgreSQL“ oder „Securitymanager MongoDB“. Eine Person kann auch mehrere dieser Rollen innehaben. Wichtig ist aber eine Stellvertreterregelung. Daher ist es empfehlenswert, die Updatemanager-Rolle auf unterschiedliche Personen zu übertragen, so dass untereinander eine Vertreterfunktion eingenommen werden kann.

Der Inhaber der Rolle muss die DBS regelmäßig auf wichtige Updates und Sicherheitslecks überprüfen. Sinnvoll ist hier ein direkter Kontakt zum Hersteller und eine aktive Beteiligung an vorherrschenden Communities. Einige Hersteller bieten auch direkte Benachrichtigungsdienste für sicherheitskritische Ereignisse an. Wichtig ist es außerdem, kein System zu betreiben welches den Status „End of Life“ (EoF) hat, da die Hersteller zu diesen Systemen keine Updates mehr bereitstellen [41].

Eine Übersicht möglicher Informationsquellen zu den jeweiligen DBS ist in Tabelle 3, Tabelle 4 und Tabelle 5 angegeben.

Das Anlegen von Sicherungen der Datenbanksysteme wird im Kapitel APP.4.3.A9 – Datensicherung eines Datenbanksystems detailliert beschrieben.

Tabelle 3 – MySQL: Möglichkeiten zum Informieren

MySQL		
Information	*Informationsquelle*	*Beschreibung*
Unterstützte Plattformen	Supported Platforms	Zeigt die aktuell unterstützen Betriebssysteme für die jeweiligen Versionen
EoF Ankündigungen	EoF Announcement	Listet Ankündigungen zum „End of Life“ für MySQL-Produkte auf.
Newsletter	MySQL Newsletter	Anmeldung zum monatlichen Newsletter von Oracle Corp. zum Produkt MySQL Datenbank
Webinare	MySQL Webinare	Oracle Corp. bietet regelmäßig Webinare zum Thema MySQL an.
Developer Forum	MySQL Forum	Das offizielle MySQL-Forum mit einem Update-Bereich, in den regelmäßig neue Ankündigungen veröffentlicht werden. Die Foren sind per RSS abonierbar
Blog	Planet-MySQL	„Planet MySQL“ ist ein Blog des Herstellers, um Nachrichten zum Thema MySQL zu veröffentlichen
Veranstaltungen	Events	Oracle Corp. bietet unterschiedliche Veranstaltungen in verschiedenen Ländern an
Update-Zyklen	Keine	Oracle Corp. veröffentlicht keine offizielle Aussage über die Update-Zyklen

Tabelle 4 - PostgreSQL: Möglichkeiten zum Informieren

PostgreSQL		
Information	*Informationsquelle*	*Beschreibung*
Mailing List	PostgreSQL Mailinglists	Postgres bietet thematisch-unterschiedliche Mailing-Listen zum Abonnieren an
Chatrooms und Foren	IRC SLACK PG-Forum Planet-Postgres	Zu Postgres gibt es eine große Gemeinschaft, die sich regelmäßig auf unterschiedlichen Kanälen austauscht.
Lokale Nutzergruppen	Usergroup	In 35 unterschiedlichen Ländern gibt es lokale Nutzergruppen, die sich zu Postgres austauschen.
Update-Zyklen	Versionierung	Postgres plant, pro Jahr ein Major-Update und pro Quartal ein Minor-Update herauszubringen. Der Support-Status (EoF) ist ebenfalls in der Liste aufgeführt.
Sicherheitskritische Ankündigungen	Security Announcements	Postgres veröffentlicht auf dieser Webseite regelmäßige sicherheitskritische Ankündigungen.

Tabelle 5 - MongoDB: Möglichkeiten zum Informieren

MongoDB		
Information	*Informationsquelle*	*Beschreibung*
Wichtige Ankündigungen	Alerts	Der Hersteller veröffentlicht auf dieser Seite kritische Ankündigungen zu den Themen Sicherheit, Datenintegrität und Betrieb.
Fehlerklassen und EoL	Support Policy	Für MongoDB gibt es unterschiedlichen Fehlerklassen, diese werden hier kategorisiert. Außerdem werden hier die EoF-Daten zu den jeweiligen Versionen veröffentlicht.
Forum	Forum	Das offizielle MongoDB Community Forum zum Austausch zwischen den Nutzern.
Developer Hub	Developer Hub	Ein offizieller Entwickler-Blog mit Ankündigungen, Neuigkeiten und Tutorials zum Thema MongoDB, zusätzlich werden hier spezielle Entwickler-Veranstaltungen angekündigt.
Veranstaltungen	Events	Der Hersteller kündigt hier unterschiedliche Veranstaltungen zu MongoDB an.
Ressourcen	Ressources	MongoDB veröffentlicht auf dieser Seite unterschiedliche Whitepaper, Webinare und Best-Practise Anweisungen.

3.1.8 APP.4.3.A8 – Datenbank-Protokollierung

Anforderungsbeschreibung

„Sicherheitsrelevante Ereignisse des Datenbanksystems MÜSSEN mit einem eindeutigen Zeitstempel protokolliert werden. Dabei MÜSSEN sich Art und Umfang der Protokollierung am Schutzbedarf der zu verarbeitenden Informationen orientieren. Zusätzlich MUSS geprüft werden, ob die Protokollierung der Fachanwendungen zusammen mit der Protokollierung der Datenbank alle erforderlichen Informationen abdeckt, um betriebs- und sicherheitsrelevante Veränderungen an der Datenbankinfrastruktur und den Anwendungen zu erkennen. Es SOLLTE so protokolliert werden, dass die Protokolldateien nicht nachträglich verändert werden können.“

Umsetzungsvorschlag

Die kommerziell-vertriebenen MySQL Enterprise und MongoDB Enterprise bieten abgestimmte Audit-Log[18]—Plugins an, die ein umfängliches Event-Logging zu erreichen [42] [43]. Diese Funktionen waren nicht in den Opensource-Varianten verfügbar. Daher wurde nach Alternativen zu den kommerziellen Plugins gesucht.

MySQL

MySQL bietet unterschiedliche Log-Funktionen an [44]:

- Error-Log: Zeichnet Probleme mit dem Starten, Ausführen und Stoppen des Dienstprogrammes auf.
- General Query Log: Zeichnet Verbindungen zum Server und erhaltene Statements vom Server auf
- Binary Log: Zeichnet Statements auf, die Änderungen an der Datenbasis vornehmen

[18] *Anmerkung des Autors: Bei IT-Systemen wird im Falle von Protokollierung, angelehnt an die englische Bezeichnung, in der Regel von „Logging“ gesprochen. Ein erstelltes Protokoll wird in diesem Sinne als „Log“ bezeichnet. Diese Begrifflichkeiten werden im Folgenden genutzt.*

- Relay Log: Zeichnet Änderungen auf, die von einem Replikationsserver gesendet werden
- Slow Query Log: Zeichnet Statements auf, die länger als eine definierte Zeit (*long_query_time*) zur Ausführung benötigen
- DDL Log: Zeichnet Metadaten auf, die die Datenstruktur verändern (DDL Statements)

In der Standardinstallation von MySQL ist keine dieser Log-Funktionalität aktiviert. Außerdem sind diese Log-Dateien nicht anpassbar und somit nur schwer auditfähig. Um Audit-fähiges Loging zu ermöglichen, wurden Alternativen herausgesucht.

Da es unterschiedliche Derivate gibt, die auf der Basis von MySQL Community entwickelt wurden und erweiterte Funktionalitäten bieten, wurden diese nach entsprechenden Lösungen untersucht. Außerdem gab es ein Plugin, welches von McAfee für MySQL entwickelt wurde, um diese Funktionalität zu bedienen. Daraus ergaben sich drei Alternativen:

Tabelle 6 - Übersicht von MySQL Audit-Plugins

Anbieter	Plugin	Quelle
MariaDB Foundation	Server_Audit	MariaDB Audit Plugin
McAfee LLC	Libaudit_plugin	McAfee Knowledge Center
Percona LLC	Audit_log	Percona Server Audit Log

Alle drei Varianten wurden mit einer Opensource-Lizenz angeboten, somit ist die volle Funktionalität einsehbar und das Risiko eine Schadsoftware einzuführen geringer. Es ließ sich zum Umsetzungszeitpunkt lediglich die Variante von Percona LLC in der aktuellen MySQL Server Version installieren. Die anderen waren mit den genutzten DBS-Versionen nicht kompatibel. Außerdem wurde offiziell nur das Betriebssystem Linux unterstützt. Details zur Umsetzung sind dem Anhang 9A.4.7.1zu entnehmen.

PostgreSQL

PostgreSQL Community stellte ein Error-Logging bereit, bei denen Fehlermeldungen standardmäßig in die den Fehlerdatenstrom des Systems gemeldet werden. Die Fehlerschwere konnte in der Konfiguration eingestellt

werden. Das Ziel konnte wahlweise auch eine „comma-seperated-value“ (csv)-Datei oder eine Textdatei sein [45]. Diese Art von Logging bot nicht ausreichend Aufzeichnung für ein Audit.

Die PostgreSQL Community bot mit pgAudit einen Defacto-Standard zur Generierung von Audit-Protokolldateien [46] für PostgreSQL-Server. Das Plugin ist Opensource und kann selbst kompiliert werden. Alternativ standen online auf unterschiedlichen Abruf-Servern vorkompilierte Versionen bereit. Ein weiteres Opensource Plugin, welches zu diesem Zweck von dem Unternehmen 2ndQuadrant Ltd. Entwickelt wurde, ist „Audit trigger 91plus“ [47]. Da pgAudit sich aber durchgesetzt hat und aktiver weiterentwickelt wird, wurde sich darauf fokussiert. Die Umsetzung dieser Anforderung wurde im Anhang A.4.7.2 beschrieben.

MongoDB

In der Community-Variante bot MongodB das Logging von Fehler, Warnungen und Informationen auf unterschiedlichen Komponenten. Diese Funktionalität ist aber nicht für den Regelbetrieb gedacht, sondern nur zur Fehlersuche. Der Hersteller weist explizit darauf hin, dass die Log-Funktionen das System verlangsamen [48].

Wie bereits im Einleitungssatz beschrieben wurde, bietet MongoDB Enterprise eine integrierte Audit Log-Lösung. Zum Zeitpunkt der Umsetzung gab es für MongoDB keine Möglichkeit, zusätzliche Funktionalitäten durch Plugins nachzuladen, daher existiert keine Lösung, um ein erweitertes Logging zu ermöglichen.

Alternativen wären hier das MongoDB-Derivat „Percona Server for MongoDB“, eine Entwicklung der Firma Percona LLC aufgebaut auf dem Opensource-Code von MongoDB. Percona hat in diese Variante ein Audit-Logging-System inkludiert [49]. Eine weitere Lösung, die die Einbindung externer Programme voraussetzt, wird in [50] diskutiert. Dies wird in dieser Arbeit nicht weiter betrachtet.

3.1.9 APP.4.3.A9 – Datensicherung eines Datenbanksystems

Anforderungsbeschreibung

„Es MÜSSEN regelmäßig Systemsicherungen des DBMS und der Daten durchgeführt werden. Auch bevor eine Datenbank neu erzeugt wird, MUSS das Datenbanksystem gesichert werden. Hierfür SOLLTEN die dafür zulässigen Dienstprogramme benutzt werden. Alle Transaktionen SOLLTEN so gesichert werden, dass sie jederzeit wiederherstellbar sind. Wenn die Datensicherung die verfügbaren Kapazitäten übersteigt, SOLLTE ein erweitertes Konzept erstellt werden, um die Datenbank zu sichern, z. B. eine inkrementelle Sicherung. Abhängig vom Schutzbedarf der Daten SOLLTEN die Wiederherstellungsparameter vorgegeben werden.“

Umsetzungsvorschlag

Gemäß der Anforderungsbeschreibung wurden die DBS auf unterschiedliche Sicherungsarten untersucht:

- Volle Datenbank-Sicherung
- Inkrementelle Datenbank-Sicherung
- Sicherung der Transaktionen

Zusätzlich wurden die Systeme dahingegen überprüft, ob aus dem DBS heraus eine zeitgesteuerte Sicherung möglich war.

MySQL

Die Oracle Corp. stellt zum Umsetzungszeitpunkt für MySQL unterschiedliche Backup-Methoden bereit. Die Enterprise-Variante bot nochmals zusätzliche Funktionalitäten. Zusätzlich beschreibt die Oracle Corp. in der Dokumentation unterschiedliche Backup-Strategien und wie diese mit dem DBS umgesetzt werden können [51]. Die drei betrachteten Sicherungsarten sind „mysqldump“ (volle Datensicherung), „binlog“ (inkrementelle Sicherung) und „General Logs“ (Transaktionssicherung). Im Anhang A.4.8.1 sind die Vorgehensweise für die drei Sicherungsarten im Detail beschrieben.

PostgreSQL

PostgreSQL bot ebenfalls unterschiedliche Backup-Routinen an, die unterschiedliche Funktionen aufwiesen. Einfache SQL-Backups der Datenbanken konnten via „pg_dump“ ausgeführt werden. Das Sichern von Binärdateien erfolgte via „pg_basebackup“. Die Umsetzung dieser Sicherungen ist im Anhang A.4.8.2 beschrieben. Keines dieser Dienstprogramme bot eine direkte Methode zur Ausführung von inkrementellen Backups. Eine Möglichkeit, wie eine solche Funktion implementiert werden kann, ist aber bereits in der Opensource-Community veröffentlicht [52].

MongoDB

Datensicherung in MongoDB konnten mittels „mongodump“ erfolgen, einem Dienstprogramm, welches „BSON“ (Binary JSON) -Dateien erzeugt. Hingegen gab es in der Community-Variante keine Möglichkeit inkrementelle Backups zu Erzeugen. Eine „Point-In-Time-Recovery“ (PITR) war nur in unterschiedlichen, kostenpflichten Zusatzdiensten möglich, siehe z.B. MongDB Atlas [53] oder MongoDB Cloud Manager [54]. Die Sicherung und Wiederherstellungen von Transaktionen wurden offiziell nicht unterstützt. Einen Weg wie Transaktionen über die Replikationsfunktion von MongoDB möglich ist, sowie die Sicherung via mongodump, ist im Anhang A.4.8.3 beschrieben.

Zeitgesteuerte Sicherung

Eine Möglichkeit, zeitgesteuert Sicherungen am DBS vorzunehmen, unterstützte keines der Systeme. Da die Standard-Sicherungstools aber Kommandozeilen-Ausführung unterstützten, wäre es möglich, über das jeweilige Betriebssystem eine Zeitsteuerung zu konfigurieren, wie z.B. Cronjobs auf einem Linux-basierten System.

3.2 Standard-Anforderungen

Die Standard-Anforderungen sind nach dem BSI als „angemessene Sicherheit“ beschrieben [6].

3.2.1 APP.4.3.A10 – Auswahl geeigneter Datenbankmanagementsysteme

Anforderungsbeschreibung

„Bevor Datenbankmanagementsysteme beschafft werden, SOLLTEN Anforderungen an das DBMS definiert und in einem Anforderungskatalog dokumentiert werden. Danach SOLLTEN alle infrage kommenden Datenbankmanagementsysteme anhand dieses Katalogs bewertet werden. Die Ergebnisse SOLLTEN dokumentiert werden.“

Umsetzungsvorschlag

Prinzipiell ist die Auswahl eines geeigneten DBS vom Anwendungsfall abhängig und kann nicht generalisiert werden, sondern muss vom IT-Architekten sorgfältig ausgesucht und begründet werden. Die Auswahl beruht nicht zuletzt auch auf der Erfahrung der Entwickler und deren Kenntnisstand mit den jeweiligen Systemen. Nichtsdestotrotz haben sich die Systeme oftmals in einer Fachdomäne bewährt und sich für diesen Bereich auch optimiert. Bei den ausgesuchten Systemen stellen sich die Anwendungsfälle wie folgt dar:

Tabelle 7 - Zuordnung typischer Anwendungsfälle zu den DBS MySQL, PostgreSQL und MongoDB

Datenbanksystem	**Anwendungsfall**
MySQL	Stark besuchte Webseiten [55]
	eCommerce Anwendungen [55]
	Content Management Systeme, wie z.B. Typo3, Wordpress oder Drupal [55]
	Hoch performante WebApps [55]
PostgreSQL	Fachanwendungen aus bspw. der Finanzindustrie [56]
	Verwaltung von Geodaten [56]
	Skalierbare Anwendungen für Supply Chain Prozesse und Industrieprozesse [56]
	Analyse und Verarbeitung großer Datenmengen für z.B. wissenschaftliche Forschung [56]
MongoDB	Betrieb von Echtzeit Anwendungen und Auswertung von Echtzeit-Daten [57]
	Verwaltung von Internet of Things (IoT)-Daten [57]
	Mobile Anwendungen für tragbare Geräte, z.B. Smartphones [57]
	Datenverwaltung unterschiedlichster Datentypen in einer Collection [57]
	Vernetzte Computerspiele [57]

Neben derartigen Anwendungsbeispielen gibt es verschiedene Untersuchungen, die vor allem die Abfrage- und Bearbeitungsgeschwindigkeiten von Datenbanken unter definierten Zuständen vergleichend untersuchen [58] [59] oder Entscheidungshilfen für unterschiedliche Systeme bzw. Systemtypen geben [60] [61].

3.2.2 APP.4.3.A11 – Ausreichende Dimensionierung der Hardware

Anforderungsbeschreibung

„Datenbankmanagementsysteme SOLLTEN auf ausreichend dimensionierter Hardware installiert werden. Die Hardware SOLLTE über genügend Reserven verfügen, um auch eventuell steigenden Anforderungen gerecht zu werden. Zeichnen sich trotzdem während des Betriebs Ressourcenengpässe ab, SOLLTEN diese frühzeitig behoben werden. Wenn die Hardware dimensioniert wird, SOLLTE das erwartete Wachstum für den geplanten Einsatzzeitraum berücksichtigt werden.“

Umsetzungsvorschlag

Die Dimensionierung von Hardware ist immer abhängig von dem Anwendungsfall, somit ist eine pauschale Aussage nicht möglich. PostgreSQL und MongoDB gaben offizielle Aussagen, an denen man sich orientieren kann. MySQL veröffentlichte von offizieller Seite keine Informationen zur empfohlenen Hardware.

In den folgenden Tabellen (Tabelle 8, Tabelle 9, Tabelle 10) wird zu den Systemen eine Mindestvoraussetzung genannt, die sich aus unterschiedlichen Quellen und Erfahrungsberichten zusammensetzt.

Beachtet werden muss, dass die Größe des gesamten Systems über die Laufzeit wächst, sofern es keine geeigneten Prozesse gibt die z.B. Log-Dateien archivieren und dadurch das Wachstum reduzieren.

MySQL

Tabelle 8 - Hardwareempfehlungen für den Betrieb des MySQL-Servers

Hardware	Spezifikation
CPU	2 Kerne, 2,4 GHz
Arbeitsspeicher (RAM)	4 GB
Festplattenspeicher	5 GB
Quellen	(1) (2)

PostgreSQL

Tabelle 9 - Hardwareempfehlungen für den Betrieb des PostgreSQL-Servers

Hardware	Spezifikation
CPU	4 Kerne, 1GHz, x64
Arbeitsspeicher (RAM)	2 GB
Festplattenspeicher	512 MB, 1.5GB
Quellen	(1) (2) (3) (4) (5)

MongoDB

Tabelle 10 - Hardwareempfehlungen für den Betrieb des MongoDB-Servers

Hardware	Spezifikation
CPU	2 Kerne, 1GHz, x86/x64
Arbeitsspeicher (RAM)	256 MB
Festplattenspeicher	512 MB, 1.5GB
Quellen	(1) (2) (3)

3.2.3 APP.4.3.A12 – Einheitlicher Konfigurationsstandard von Datenbankmanagementsystemen

Anforderungsbeschreibung

„Für alle eingesetzten Datenbankmanagementsysteme SOLLTE ein einheitlicher Konfigurationsstandard definiert werden. Alle Datenbankmanagementsysteme SOLLTEN nach diesem Standard konfiguriert und einheitlich betrieben werden. Falls es bei einer Installation notwendig ist, vom Konfigurationsstandard abzuweichen, SOLLTEN alle Schritte vom ISB freigegeben und nachvollziehbar dokumentiert werden. Der Konfigurationsstandard SOLLTE regelmäßig überprüft und, falls erforderlich, angepasst werden.“

Umsetzungsvorschlag

Im Abschnitt 3.1.4 „APP.4.3.A4 – Geregeltes Anlegen neuer Datenbanken“ wurde bereits ein Datenbank-Stammdatenblatt angelegt. In diesem Datenblatt wurden technische und organisatorische Daten eines Datenbanksystems festgehalten und verpflichtend für jede Installation eingeführt.

Alle drei untersuchten Systeme boten zudem Konfigurationsdateien an, die als Templates abgespeichert und genutzt werden konnten. Diese Templates sollten zentral abgelegt und für die Erstellung neuer Systeme genutzt werden. Die genutzte Template-Version wird im Stammdatenblatt aufgenommen. Notwendige Abweichungen wurden in einer zusätzlich eingeführten Spalte vermerkt. Das Angepasste Stammdatenblatt ist dem Anhang zu entnehmen.

Eine weitere Möglichkeit, um systemunabhängig einheitliche DBS zu erstellen, ist die Container-Technologie. Container sind eine Art der Visualisierung von Systemen. Im Gegensatz zur Visualisierung eines kompletten Hostsystems, werden bei einem Container nur eine Anwendung mit allen notwendigen Abhängigkeiten isoliert hochgefahren [62]. In diesen Containern können vorkonfigurierte DBS abgelegt werden, die bei Bedarf durch spezielle Container-Software instanziiert und hochgefahren werden können. Änderungen können zur Laufzeit am System erfolgen und als neuer Container separat abgelegt werden. Diese Vorgehensweise wird in dieser Arbeit nicht weiter thematisiert.

MySQL

Die Konfiguration eines MySQL DBS wird in der Datei „my.cnf“ abgelegt. Diese Datei wird beim Starten des DBS eingelesen und setzt die gültigen Einstellungen. In einem Linux-System werden die Konfigurationen global aus dem Pfad „/etc/my.cnf“ eingelesen [63].

PostgreSQL

Bei PostgreSQL wird die Konfiguration des Systems in der Datei „postgresql.conf“ gespeichert. Diese wird beim Hochfahren des DBS eingelesen und die Einstellungen festgelegt. Der Standardpfad ist der Ordner des entsprechenden Datenbankclusters. Zusätzlich gibt es noch zwei weitere Dateien, die die Nutzerauthentifizierung regeln: „pg_hba.conf“ und „pf_ident.conf“ [64].

MongoDB

Die Konfigurationsdatei bei MongoDB liegt bei einem Linuxsystem standardmäßig unter dem Pfad „/etc/mongod.conf“ und wird beim Starten des DBS eingelesen, um die festgelegten Einstellungen zu setzen [65].

3.2.4 APP.4.3.A13 – Restriktive Handhabung von Datenbank-Links

Anforderungsbeschreibung

„Es SOLLTE sichergestellt sein, dass nur Verantwortliche dazu berechtigt sind, Datenbank-Links (DB-Links) anzulegen. Werden solche Links angelegt, MÜSSEN so genannte Private DB-Links vor Public DB-Links bevorzugt angelegt werden. Alle von den Verantwortlichen angelegten DB-Links SOLLTEN dokumentiert und regelmäßig überprüft werden. Zudem SOLLTEN DB-Links mitberücksichtigt werden, wenn das Datenbanksystem gesichert wird.“

Umsetzungsvorschlag

Datenbank-Links sind ein Konzept zur unidirektionalen Verbindung von einem physikalischen Datenbankserver mit einem Zweiten. Diese Funktion ist vor allem bekannt aus dem DBS Oracle. Hierbei werden in einer Datenbank feste Pseudonyme für die externen Quellen angelegt, über das auf die Daten per Queries zugegriffen werden kann. Diese Links können verschiedene Restriktionslevel aufweisen: private (nur der anlegende User kann den Link sehen und nutzen), public (alle Nutzer der Datenbank können den Link sehen und nutzen) und global (der DB-Link kann über die Datenbank hinaus eingesehen und genutzt werden) [66].

In anderen Datenbanksystemen ist diese Funktionalität zumeist nicht implementiert. Von den untersuchten Datenbanken wieß nur PostgreSQL eine entsprechende Funktion auf, die sich über ein Plugin „dblink“ nachinstallieren ließ. Details hierzu sind dem Anhang 9A.4.10 zu entnehmen.

3.2.5 APP.4.3.A14 – Überprüfung der Datensicherung eines Datenbanksystems

Anforderungsbeschreibung

„Die vorgenommenen Datensicherungen SOLLTEN regelmäßig daraufhin überprüft werden, ob die Integrität der Sicherungsdateien noch gewährleistet ist. Die verantwortlichen Mitarbeiter SOLLTEN zudem regelmäßig üben, wie sich Datenbanken im Notfall schnell wiederherstellen lassen.“

Umsetzungsvorschlag

In Abschnitt 3.1.9 „APP.4.3.A9 – Datensicherung eines Datenbanksystems“ wurden für die unterschiedlichen Systeme unterschiedliche Sicherungsmaßnahmen beschrieben. Auf diese Maßnahmen wird im folgenden Abschnitt Bezug genommen.

Neben den integrierten Maßnahmen ist es möglich, die Sicherungsdateien in ein übliches Paketformat wie „tar.gz“ zu packen. Im Dateiheader von „tar.gz“-Dateien ist eine Prüfsumme enthalten [67]. Es gibt unterschiedliche Programme, die das Prüfen dieser Prüfsumme erlauben, wie beispielsweise „gzip“ [68].

MySQL

Für MySQL-Server wurden die Sicherungsarten „mysqldump“, „binlog“ und „General Log“ identifiziert. Für „mysqldump“ existiert ein impliziter Test, indem das Backup in ein MySQL-System eingespielt wird, „binlog“ bietet eine Checksummenüberprüfung mit der zyklischen Redundanzprüfung CRC32 (englisch cyclic redundancy check, daher meist CRC). Die „General Logs“ bieten keine Validierungsoption.

PostgreSQL

Die Sicherungsprogramme „pg_dump“ bzw. „pg_dumpall“ bietet für PostgreSQL umfangreiche Sicherungsmaßnahmen. Hierzu werden WAL-Dateien genutzt. Für „pg_dump“ existiert ein impliziter Test, indem das Backup in ein PostgreSQL-System eingespielt wird. WAL-Dateien enthalten zwar eine CRC32-Prüfsumme, jedoch existiert derzeit kein Weg, diese Dateien manuell gegen diese Prüfsumme zu testen [69].

MongoDB

Für MongoDB wurden die zwei Funktionen „mongodump“ und „oplog“ zur Datensicherung ausgewählt. Derzeit ist keine Prüfung der Integrität der Ergebnisse aus diesen Sicherungen implementiert. Zur Generellen Funktionsprüfung wird auch hier das Einspielen der Sicherungsdateien auf einem Testsystem empfohlen.

Notfallübungen

Das regelmäßige Üben von IT-Notfällen gehört heutzutage in alle Notfallmanagementverfahren. Das BSI hat einen eigenen Standard zu diesem Thema definiert, den BSI-Standard 100-4 „Notfallmanagement“ [22] und im IT Grundschutz-Kompendium einen Systembaustein „DER.4: Notfallmanagement“ [6] eingeführt. Der BSI-Standard 100-4 beschreibt im Kapitel 8 „Tests und Übungen“ detailliert, welche unterschiedlichen Arten von Übungen es gibt, welchen Aufwand diese haben, wie sie aufzubauen sind und unter welchen Umständen welche Übungsarten angemessen sind. Auch andere Institutionen und Verbände haben verpflichtend Übungen in ihr IT-Notfallkonzept aufgenommen, wie z.B. in den „Mindestanforderungen an das Risikomanagement“ (MaRisk) festgeschrieben in den „Bankaufsichtlichen Anforderungen an die IT“ (BAIT) der „Bundesanstalt für Finanzdienstleistungen“ (BaFin) [70] oder auch als Bestandteil der Information Technology Infrastructure Library (ITIL) [71].

Zusätzlich gibt es unterschiedliche kommerzielle Schulungszentren, die IT-Notfallübungen anbieten [72] [73].

3.2.6 APP.4.3.A15 – Schulung der Datenbankadministratoren

Anforderungsbeschreibung

„Es SOLLTE gewährleistet sein, dass nur ausreichend geschulte Mitarbeiter das Datenbankmanagementsystem administrieren. Es SOLLTE ein Schulungsplan erstellt werden, mit dem sichergestellt wird, dass Datenbankverantwortliche rechtzeitig zu Themen der Informationssicherheit und Performance sowie zu den Funktionen neuer Versionen des Datenbankmanagementsystems geschult werden."

Umsetzungsvorschlag

Die Oracle Corp. nennt in einem Bericht von 2017 als ersten Punkt zur Erhöhung der Sicherheit einer Datenbank, die Schulung von Mitarbeitern und empfiehlt explizit hierfür Budget und Zeit zur Verfügung zu stellen [74]. In [75] wird außerdem empfohlen, alle Mitarbeiter eines Unternehmens entsprechend ihrer Tätigkeit zu schulen. Daher wurde hier ein mehrstufiges, aufeinander aufbauendes Konzept vorgeschlagen. In Stufe 1 ist vorgesehen, dass alle Mitarbeiter, unabhängig Ihrer Tätigkeit, eine Grundausbildung im Bereich IT-Sicherheit erhalten sollen. Stufe 2 betrifft alle Mitarbeiter, die direkt als IT-Mitarbeiter angestellt sind oder viele ihrer Tätigkeiten in diesem Bereich haben. Diese Stufe umfasst eine Fachausbildung für IT-Sicherheit, die tiefergehendes Wissen aufbaut. Als Stufe 3 ist die Expertenausbildung vorgesehen. Hierbei geht es darum, ausgewählte Mitarbeiter, die zuständig für ein bestimmtes System sind, spezialisiert darauf zu schulen. Für die zu untersuchenden Datenbanken sind in Tabelle 11 geeignete Schulungsprogramme exemplarisch aufgeführt. Diese Schulungen umfassen IT-Sicherheitsthemen und technische Spezifika zum jeweiligen DBS, somit sind die Anforderung diesbezüglich erfüllt.

Das BSI stellt im Umsetzungsrahmenwerk (UmRa) eine Vorlage zur Dokumentation von Schulungsplänen bereit [76].

Tabelle 11 - Vorschläge für DBA Schulungen für die DBS MySQL, PostgreSQL und MongoDB

Datenbank-system	Schulungsname	Anbieter	Quelle
MySQL	MySQL for Database Administrators	Oracle Corporation	LINK
PostgreSQL	Free Postgres Training	EnterpriseDB Corporation	LINK
MongoDB	Free training courses designed for Database Administrators	MongoDB, Inc.	LINK

3.2.7 APP.4.3.A16 – Verschlüsselung der Datenbankanbindung

Anforderungsbeschreibung

„Das Datenbankmanagementsystem SOLLTE so konfiguriert werden, dass Datenbankverbindungen immer verschlüsselt werden. Die dazu eingesetzten kryptografischen Verfahren und Protokolle SOLLTEN den internen Vorgaben der Institution entsprechen."

Umsetzungsvorschlag

Gesicherte Verbindungen werden heutzutage über das Protokoll „Transport Layer Security" (TLS) ermöglicht, welcher auf dem veralteten Protokoll „Secure Socket Layer" (SSL) basiert, und oftmals noch unter der Bezeichnung „SSL" bekannt ist. Dieses Protokoll sorgt dafür, dass vor der Kommunikation zwischen zwei Partnern ein kryptografisches Verfahren vereinbart und ein gemeinsamer Schlüssel ausgetauscht werden, so dass die eigentliche Kommunikation nur verschlüsselt stattfindet [77].

Die Protokolle SSL und TSL existieren in verschiedenen Versionen. Die aktuelle Version ist TSLv1.3, welche 2018 veröffentlicht wurde. Das BSI empfiehlt für eine sichere Verbindung mindestens den Zertifikatsstandard TLSv1.2, bei Neuanlage von Projekten bereits TLSv1.3 [78].

Die drei DBS bauen in der Linux-Variante alle auf der Bibliothek OpenSSL in der Version 1.1.1 auf und können somit als gleichwertig betrachtet werden.

Die Implementierungsbeschreibungen der Verschlüsselungen sind dem Anhang 9A.4.12 zu entnehmen.

MySQL

MySQL implementiert das TSL-Protokoll auf Basis der Bibliothek OpenSSL [79] in Version 1.1.1. Von MySQL werden folgende TSL-Versionen unterstützt: TLSv1, TLSv1.1, TLSv1.2 und TLSv1.3. Die jeweiligen Versionen können vom Datenbankadministrator in der Konfiguration festgelegt werden.

PostgreSQL

Der PostgreSQL-Server nutzt OpenSSL und greift hierbei auf die installierte Version des Hostsystems zurück. Die unterstützen SSL/TLS-Versionen sind daher abhängig von der installierten OpenSSL-Version [80] [81]. Vorausgesetzt, dass die derzeit aktuelle Version 1.1.1 installiert ist, unterstützt PostgreSQL die Versionen TLSv1, TLSv1.1, TLSv1.2 und TLSv1.3 [82].

MongoDB

Der MongoDB-Server greift auf installierte, native SSL/TLS-Bibliotheken des Hostsystems zu, welche vom Betriebssystem abhängig sind. Bei Microsoft Windows wird „Secure Channel“ (Schannel) genutzt, bei Linux/BSD-Systemen „OpenSSL“ und bei macOS „Secure Transport“ [83]. Entsprechend eines Linux Systems mit installierter OpenSSL-Version 1.1.1, werden die Versionen TLSv1, TLSv1.1, TLSv1.2 und TLSv1.3 unterstützt [82]. MongoDB deaktiviert TLSv1, sobald eine neuere Version verfügbar ist, damit diese veraltete Version nicht mehr genutzt wird [83].

3.2.8 APP.4.3.A17 – Datenübernahme oder Migration

Anforderungsbeschreibung

„Es SOLLTE vorab definiert werden, wie initial oder regelmäßig Daten in eine Datenbank übernommen werden sollen. Nachdem Daten übernommen wurden, SOLLTE geprüft werden, ob sie vollständig und unverändert sind.“

Umsetzungsvorschlag

Zur einheitlichen Übernahme von neuen Daten oder Migration von Altsystemen wurde ein Prozess entwickelt, welcher sich an den Empfehlungen der Datenbankhersteller Oracle Corp. [84] und Amazon Web Services (AWS) [85] orientiert. Vorgeschlagene Phasen wurden in den Prozess integriert. Die Phase „Betrieb“ bzw. „Optimierung“ wurde in dieser Arbeit nicht behandelt, da diese für die Umsetzung der Anforderung nicht erforderlich sind. Zusätzlich wurde ein Antragsformular entworfen, welches die wichtigsten Daten für die Verarbeitung aufnimmt. Der Prozess als Ablaufplan und das Formular sind dem Anhang 9A.4.13 zu entnehmen.

3.2.9 APP.4.3.A18 – Überwachung des Datenbankmanagementsystems

Anforderungsbeschreibung

„Es SOLLTEN Parameter, Ereignisse und Betriebszustände des Datenbankmanagementsystems definiert werden, die für den sicheren Betrieb kritisch sind. Diese SOLLTEN mithilfe eines Monitoring-Systems überwacht werden. Für alle kritischen Parameter und Ereignisse SOLLTEN Schwellwerte festgelegt werden. Wenn diese Werte überschritten werden, MUSS geeignet reagiert werden. Es MÜSSEN z. B. die zuständigen Mitarbeiter alarmiert werden. Anwendungsspezifische Parameter, Ereignisse und deren Schwellwerte SOLLTEN mit den Verantwortlichen für die Fachanwendungen abgestimmt werden.“

Umsetzungsvorschlag

Zur Umsetzung dieser Anforderung wurde zuerst geprüft, ob es integrierte Mittel gab, um das Monitoring zu ermöglichen. Sollte es keine direkte, freiverfügbare Lösung geben, wurde nach einer alternativen freien Variante, vorzugsweise ebenso als Opensource-Version, gesucht.

Folgende Funktionen werden untersucht:

- Allgemeines Monitoring des Datenbanksystems, wie z.B. Hardwareperformance und Abfragedauer
- Festlegen von Schwellwerten für Alarmierungen
- Alarmierung außerhalb der Monitoring-Software, wie z.B. per E-Mail, SMS, oder weitere

Bei der Suche nach Opensource-Lösungen sind vor allem die Produkte des Unternehmens Percona LLC. positiv aufgefallen. Diese bieten insbesondere für Opensource-Datenbanksysteme zusätzliche Funktionen, die unter der gleichen Lizenz, wie das DBS ist, gestellt werden.

Beschreibungen zu den Untersuchungen der einzelnen DBS sind dem Anhang 9A.4.14 mit den jeweiligen Unterordnern zu entnehmen.

MySQL

In der MySQL Community Edition war kein Monitoring-Tool integriert. In der kostenpflichtigen Enterprise-Variante wäre das Tool „MySQL Enterprise Monitor“ inkludiert, welches laut Funktionsliste alle oben genannten Kriterien erfüllt [86]. Es wurde stattdessen das freie Opensource-Tool „Percona Monitoring and Management“ mit dem Plugin „PMM2 MySQL“ der Firma „Percona LLC“ untersucht. Die Untersuchung ergab, dass hiermit alle Funktionalitäten abgedeckt werden.

PostgreSQL

In der PostgreSQL Community Edition war kein Monitoring-Tool integriert. In der kostenpflichtigen Enterprise-Variante „EnterpriseDB Postgres“[19] wäre das Tool „EDB Postgres Enterprise Manager“ inkludiert, welches laut Funktionsliste alle oben genannten Kriterien erfüllt. Daher wurde stattdessen das freie Opensource-Tool „Percona Monitoring and Management“ mit dem Plugin „PMM2 PostgreSQL“ der Firma „Percona LLC“ untersucht. Die Untersuchung ergab, dass hiermit alle Funktionalitäten abgedeckt werden.

MongoDB

Für alle MongoDB Instanzen, einschließlich der Community-Variante, bot MongoDB eine freie Monitoring-Lösung an. Diese kann mit den entsprechenden Rechten aktiviert werden und wird auf einem Cloudserver des Herstellers gehostet. Jeder mit Zugriff auf die URL kann die Daten des Servers für die vergangenen 24 Stunden einsehen. Darüber hinaus war kein Monitoring möglich. Es wurden einige Hardwaredaten und einige Datenbank-Daten aufgenommen, wie z.B. CPU-Auslastung, Netzwerktraffic, aktive Operationen (lesend und schreibend) und die Anzahl gescannter Dokumente [87]. Es wurden keine Funktionen für Alarmierungen bereitgestellt. Alternativ konnten vom Hersteller MongoDB Inc. kostenpflichtige Dienste gekauft werden, z.B. MongoDB Cloud Manager oder MongoDB Ops Manager. Diese Dienste böten weitere

[19] https://www.enterprisedb.com/, EnterpriseDB Corporation

Funktionalitäten, als die Opensource-Variante ermöglicht, wodurch die Anforderungen abgedeckt werden [88].

Da nicht alle Funktionen in der freien Monitor-Lösung vorhanden waren, wurde nach Alternativen hierzu gesucht. Daher wurde das freie Tool „Percona Monitoring and Management MongoDB“ zusätzlich untersucht. Alle Funktionen wurden abgedeckt.

3.2.10 APP.4.3.A19 – Schutz vor schädlichen Datenbank-Skripten

Anforderungsbeschreibung

„Werden Datenbank-Skripte entwickelt, SOLLTEN dafür verpflichtende Qualitätskriterien definiert werden. Datenbank-Skripte SOLLTEN ausführlichen Funktionstests auf gesonderten Testsystemen unterzogen werden, bevor sie produktiv eingesetzt werden. Die Ergebnisse SOLLTEN dokumentiert werden.“

Umsetzungsvorschlag

Merkmale für eine gute Softwarequalität kann verschiedene Ausprägungen annehmen, hierzu zählen z.B. „Verständlichkeit“, „Korrektheit“ und „Robustheit“ [89]. Zur Einhaltung dieser Kriterien wurden Vorschläge gemacht, die von einer Programmiersprache auf die Abfragesprache „SQL“ übertragen werden können. Ausgewählt wurden „Statische Codequalität“ und „Komponententest“.

Statische Codequalität

Zur Steigerung der Qualität, ist es sinnvoll die statische Codequalität durch das Einführen von verpflichtenden Styleguides zu erhöhen. Es vereinfacht die Wartung des Codes für andere Programmierer und deckt übliche Fehlerquellen schnell auf [90] [91] [92]. Während Programmiersprachen wie C# [93] oder JavaScript [94] oftmals allgemeingültige Styleguides haben, existierte zum Umsetzungszeitpunkt kein einheitlicher Standard für die Datenbankabfragesprache SQL. Unterschiedliche Experten veröffentlichten bereits Vorschläge für einen SQL-Styleguide [95] [96] [97], beispielhaft ist die von der Mozilla Corporation publizierte Fassung im Anhang hinterlegt. Zusätzlich gab es Empfehlungen zum Umgang von öffentlichen SQL-Schnittstellen, z.B. von der Open Web Application Security Project (OWASP) Foundation [98].

Für MongoDB-Abfragen wurden zum Zeitpunkt noch keine Styleguides von größeren Herstellern veröffentlicht. Vereinzelt lassen sich Vorschläge von unabhängigen Entwicklern finden, z.B. [99].

Komponententests

Komponententests, oder aus dem Englischen oft „Unittests“, prüfen die Funktionalität einzelner Komponenten, isoliert von Abhängigkeiten. Oftmals werden jedoch die kleinsten Einheiten bei Datenbanken, die Abfragen, nicht explizit getestet [100] [101] [102].

Es wurden unterschiedliche Programme recherchiert, die bei der Erstellung von Komponententests für SQL-Abfragen unterstützen. Für die untersuchen Datenbankensysteme ist eine Reihe von Vorschlägen der Tabelle 12 zu entnehmen.

Tabelle 12 - Auflistung von Unit-Test-Frameworks für die DBS MySQL, PostgreSQL und MongoDB

Bezeichnung	Hersteller	Kompatibel mit	Quelle
STK/Unit	STK/Community	MySQL V5.x	LINK
MyTAP	Privat/Community	MySQL V5.x	LINK
pgTAP	Privat/Community	PostgreSQL 9.x	LINK
utMySQL	Privat/Community	MySQL V5.x	LINK
DbFit	Privat/Community	MySQL V5.x, PostgreSQL V9.x	LINK
NDBUnit	Privat/Community	MySQL V5.x, PostgreSQL V9.x	LINK
JEST	Facebook Inc.	MongoDB	LINK
mongoUnit	Privat/Community	MongoDB	LINK

Aufgefallen ist, dass für die Systeme MySQL und PostgreSQL nur Projekte aus der Opensource-Community vorhanden waren. Diese haben oftmals das Problem, dass sie nicht durchgängig gepflegt werden. So auch in diesen Fällen. Es existierten eine Reihe von Tools, doch wurde zum Testzeitpunkt keine aktuelle Version der Datenbanksysteme unterstützt, daher konnte diese Anforderung nicht erfüllt werden.

Unterstützung für MongoDB ist generell seltener zu finden, dafür waren die Projekte noch aktiv und unterstützten auch aktuelle Systemveröffentlichungen.

3.2.11 APP.4.3.A20 – Regelmäßige Audits

Anforderungsbeschreibung

„Bei allen Komponenten des Datenbanksystems SOLLTE regelmäßig überprüft werden, ob alle festgelegten Sicherheitsmaßnahmen umgesetzt und diese korrekt konfiguriert sind. Dabei SOLLTE geprüft werden, ob der dokumentierte Stand dem Ist-Zustand entspricht und ob die Konfiguration des Datenbankmanagementsystems der dokumentierten Standardkonfiguration entspricht. Zudem SOLLTE geprüft werden, ob alle Datenbank-Skripte benötigt werden und ob sie dem Qualitätsstandard der Institution genügen. Zusätzlich SOLLTEN die Protokolldateien des Datenbanksystems und des Betriebssystems nach Auffälligkeiten untersucht werden. Die Auditergebnisse SOLLTEN nachvollziehbar dokumentiert und mit dem Soll- Zustand abgeglichen werden. Abweichungen SOLLTE nachgegangen werden.“

Umsetzungsvorschlag

Es sollten unterschiedlich große Audits eingeführt werden, die regelmäßig stattfinden. Folgende Unterscheidungen mit unterschiedlichen Inhalten werden vorgeschlagen:

Große Audits umfassen das Prüfen der festgelegten Schutzmaßnahmen. Zum einen stellt das BSI zu jedem Baustein eine Checkliste bereit, anhand derer die Maßnahmen Punkt für Punkt abgehakt werden können. Die Checkliste für den Baustein APP.4.4 „Relationale Datenbanken“ kann dem Anhang 9A.4.16 entnommen werden. Zum anderen wurden in Kapitel 3.1.3 Checklisten für die untersuchten Datenbanksysteme angelegt, die die Basishärtung beschreiben. Zusätzlich müssen individuell aufgestellte Sicherheitsregeln dokumentiert und überprüft werden. Im Rahmen dieses Audits sollten auch die eingesetzten Datenbankskripte auf Sinnhaftigkeit geprüft werden.

Kleine Audits sollten periodisch häufiger durchgeführt werden. Sie umfassen das Prüfen der generierten Log-Dateien der Systeme auf Unregelmäßigkeiten und besondere Vorkommnisse. Diese Überprüfung sollte zwingend mit Tool-Unterstützung erfolgen, die automatisiert Logdateien analysieren und Vorkommnisse hervorheben. Beispiele sind die Opensource-Anwendungen Graylog[20], Elastic Stack[21] oder Logalyze[22]. Zusätzlich sollten in den kleinen Audits aktuelle „Key Performance Indicators“ (KPIs) vorgestellt werden. Beispiele für passende KPIs sind „Durchgeführte sicherheitsrelevante Mitarbeiterschulungen“ oder auch „Anzahl erkannter Sicherheitsvorfälle“ [103].

Zur regelmäßigen Kontrolle der Einhaltung der Sicherheitsmaßnahmen, sollten feste Termine definiert werden, an denen eine Überprüfung der Systeme vorgenommen wird. In der Zertifizierung nach ISO 27001, und respektiv auch dem BSI, ist festgelegt, dass einmal im Jahr eine Überprüfung der Sicherheitsstandards durch einen externen Auditor erfolgen muss [104]. Daran angelehnt sollten die großen Audits einmal im Jahr durchgeführt werden, die kleinen Audits einmal pro Monat.

Außerdem sollten die Sicherheitsvorkehrungen bei besonderen Ereignissen lageabhängig überprüft werden. Das bedeutet, wenn neue, relevante Sicherheitslücken bekannt werden, sollten die Beteiligten darüber beraten, ob und wann geeignete Gegenmaßnahmen eingeleitet werden können und müssen. Dies ist je nach Kritikalität des Falles im Einzelfall zu priorisieren. Auch das Aufdecken von konkreten Sicherheitsvorfällen in den Systemen stellt ein besonderes Ereignis dar, was die Beteiligten in einer Sondersitzung besprechen sollten, um die Schwachstelle abzustellen und ggf. weitere Maßnahmen, wie das Informieren von Datenschutzbeauftragten, einzuleiten.

---

20 https://www.graylog.org/products/open-source, Graylog

21 https://www.elastic.co/de/log-monitoring, Elastic NV

22 http://www.logalyze.com/, ZURIEL Ltd.

3.3 Anforderungen für erhöhten Schutzbedarf

Die Anforderungen für erhöhten Schutzbedarf haben sich, nach dem BSI, „in der Praxis bewährt“ und „zeigen auf, wie eine Institution sich […] zusätzlich absichern kann“ [6].

3.3.1 APP.4.3.A21 – Einsatz von Datenbank Security Tools

Anforderungsbeschreibung

„Es SOLLTEN Informationssicherheitsprodukte für Datenbanken eingesetzt werden. Die eingesetzten Produkte SOLLTEN folgende Funktionen bereitstellen:

- Erstellung einer Übersicht über alle Datenbanksysteme,
- erweiterte Konfigurationsmöglichkeiten und Rechtemanagement der Datenbanken,
- Erkennung und Unterbindung von möglichen Angriffen (z. B. Brute Force Angriffe auf ein Benutzerkonto, SQLInjection) und
- Auditfunktionen (z. B. Überprüfung von Konfigurationsvorgaben).“

Umsetzungsvorschlag

Zum Zeitpunkt der Arbeit existierten keine Informationssicherheitsprodukte speziell für Datenbanken, die Opensource oder frei verfügbar waren. Es gab unterschiedliche kommerzielle Hersteller, die geeignete Produkte vertrieben, wie z.B. Trustwave DbProtect[23] oder Imperva Data Protection and Database Audit[24]. Diese Anwendungen unterstützten die drei Systeme MySQL, PostgreSQL und MongoDB und boten einen ähnlichen Funktionsumfang, wie in der Umsetzung gefordert wird. Einzelne Funktionen können über andere Systeme auf Opensource-Basis erfüllt werden. Im Folgenden werden Umsetzungsvorschläge für die Einzelfunktionen aufgezeigt. Details sind im Anhang 9A.4.17 zu finden.

---

[23] https://www.trustwave.com/de-de/services/security-testing/dbprotect/, Truswave Holdings, Inc.

[24] https://www.imperva.com/products/data-protection/, Imperva, LLC

Erstellung einer Übersicht über alle Datenbanksysteme

Mit dem in Kapitel 3.2.9 vorgestellten Tool „Percona Monitoring and Management“, wurde eine Übersicht über alle konfigurierten Datenbanksysteme erstellt. Außerdem konnte hier der aktuelle Status des Systems abgefragt und individuelle Dashboards konfiguriert werden.

Erweiterte Konfigurationsmöglichkeiten und Rechtemanagement

MySQL Community und PostgreSQL boten beide je ein zusätzliches Administratortool, welches die Konfiguration des DBS und das Rechtemanagement ermöglicht. Für MongoDB existierte in der Opensource-Variante derzeit kein passendes Programm.

Erkennung und Unterbindung von möglichen Angriffen

Zum Umsetzungszeitpunkt existiert kein spezielles „Intrusion Detection System“ (IDS, Einbruch Erkennungs System) für die untersuchten Datenbanksysteme, das als freie oder Opensource-Variante verfügbar war. Es gab aber Opensource-Software, die generell eine Einbruchserkennung lieferten, welche auf dem Hostsystem des DBS installiert werden kann. Es werden zwar keine direkten Einbrüche ins DBS erkannt, aber zumindest Angriffe auf das Hostsystem. Ein Kandidat hierfür ist beispielsweise die Software „Snort“[25].

Auditfunktionen

Die mitgelieferten Tools zur Administration boten keine passende Auditfunktionalität. In Kapitel 3.1.8 wurden für die drei untersuchten Systeme schon erweiterte Audit-Log-Plugins vorgestellt. In Kombination mit einem Log-Management-Programm können bei bestimmen Log-Einträgen, wie z.B. das Ändern einer Konfiguration, Warnungen erzeugt werden. Ein bekannter Opensource Kandidat für so ein Logmanagement ist Graylog[26].

---

[25] https://www.snort.org/, Cisco Systems, Inc.

[26] https://www.graylog.org/

3.3.2 APP.4.3.A22 – Notfallvorsorge

Anforderungsbeschreibung

„Für das Datenbankmanagementsystem SOLLTE ein Notfallplan erstellt werden, der festlegt, wie ein Notbetrieb realisiert werden kann und welche Ressourcen dafür nötig sind. Zusätzlich SOLLTE der Notfallplan definieren, wie aus dem Notbetrieb der Regelbetrieb wiederhergestellt werden kann. Der Notfallplan SOLLTE die nötigen Meldewege, Reaktionswege, Ressourcen und Reaktionszeiten der Fachverantwortlichen festlegen. Auf Basis eines Koordinationsplans zum Wiederanlauf SOLLTEN alle von der Datenbank abhängigen IT-Systeme vorab ermittelt und berücksichtigt werden.“

Umsetzungsvorschlag

Die Kritikalität von IT-Systemen in Geschäftsbereichen ist bekannt und wird regelmäßig betont [105] [19]. Daher sollte jedes Unternehmen einen generellen Notfallvorsorgeplan haben, um die eigenen Systeme betriebsbereit zu halten, bzw. möglichst schnell Betriebsbereitschaft wiederherzustellen. Hierzu gibt es unterschiedliche Vorgehensweisen und Vorlagen. Im Standard 100-4: „Notfallmanagement“ [22] hat das BSI die zu betrachtenden Aspekte beschrieben, hierzu zählen z.B. die Business Impact Analyse, das Notfallvorsorgekonzept und das Krisenmanagement. Außerdem stellt das BSI eine beispielhafte Gliederung für ein Notfallvorsorgekonzept bereit. Zusätzlich bietet das BSI mit dem „Umsetzungsrahmenwerk“ (UmRa) unterschiedliche Vorlagen und Beschreibungen zur Umsetzung eines umfänglichen Notfallmanagements an [76].

Das „National Institute of Standards and Technology“ (NIST) hat ebenfalls ein Dokument publiziert, welches sich mit dem Thema befasst, genannt „Contingency Planning Guide for Federal Information Systems“ [106]. Die Inhalte sind zu großen Teilen überschneidend mit der Publikation des BSI.

In dieser Arbeit wurden explizit nur Datenbankensysteme behandelt, daher wird kein komplettes Notfallvorsorgekonzept entwickelt.

Eine Vorbereitung für die Umsetzung dieser Anforderung, sind die vorangegangenen Anforderungen, da diese einige Teile bereits abdecken:

- Die notwendigen Ressourcen können mit dem Umsetzungsvorschlag der Anforderung „APP.4.3.A11 – Ausreichende Dimensionierung der Hardware“ beantwortet werden.
- Es wurden bei der Erstellung der einzuhaltenden Prozesse die verschiedenen notwendigen Verantwortlichen identifiziert und benannt. Hierzu zählen z.B. die Fach-Administratoren des konkreten Systems, der ISB, der Securitymanager für das DBS, als auch die fachlich verantwortlichen Personen.

Auf Grundlage der Arbeiten des BSI und des NIST können die weiteren Anforderungen beantwortet werden.

3.3.3 APP.4.3.A23 – Archivierung

Anforderungsbeschreibung

„Ist es erforderlich, Daten eines Datenbanksystems zu archivieren, SOLLTE ein entsprechendes Archivierungskonzept erstellt werden. Es SOLLTE sichergestellt sein, dass die Datenbestände zu einem späteren Zeitpunkt wieder vollständig und konsistent verfügbar sind. Im Archivierungskonzept SOLLTEN sowohl die Intervalle der Archivierung als auch die Vorhaltefristen der archivierten Daten festgelegt werden. Zusätzlich SOLLTE dokumentiert werden, mit welcher Technik die Datenbanken archiviert wurden. Mit den archivierten Daten SOLLTEN regelmäßig Wiederherstellungstests durchgeführt werden. Die Ergebnisse SOLLTEN dokumentiert werden.“

Umsetzungsvorschlag

Zur Umsetzung eines umfänglichen Archivierungskonzeptes muss eine Liste aller archivierbaren Datenbanken und Datenbanksysteme vorhanden sein. Aus dieser Liste muss je nach Kritikalität entschieden werden, welche Daten in welchem Umfang gesichert werden. Dazu sollte eine Bewertung aller Datenbanken vorgenommen werden. Nach welchen Kriterien diese Bewertung vorgenommen wird, muss mit den Fachverantwortlichen abgestimmt werden. Je nach Unternehmen können diese Kriterien unterschiedlich sein, beispielsweise ob die Datenbank geschäftskritische Kundendaten für das Tagesgeschäft oder weniger sensitive Daten enthält. Am Beispiel der Uni-Datenbank könnte abgeleitet werden, dass die Daten mit den Prüfungsergebnissen der Studenten eine höhere Kritikalität haben, als die Raumnummer der Büros der Professoren. Die Business Impact Analyse, die in Abschnitt 3.3.2 vorgenommen wurde, kann hierbei als Entscheidungsgrundlage dienen.

Im nächsten Schritt muss betrachtet werden, wie oft die Daten sich verändern oder erneuert werden. Es kann höchstkritische Daten geben, die nur einmal im Jahr erneuert werden und hingegen unkritische Daten, die sich täglich verändern. Der Archivierungsintervall sollte sich an dieser Frequenz anlehnen, um unnötige Last durch die Archivierung zu vermeiden. Ebenso müssen sich Vorhaltefristen

an diesen Intervallen orientieren. Wenn die Daten nur einmal im Jahr erneuert werden und die Vorjahresdaten damit ihre Gültigkeit verlieren, ist es womöglich sinnvoll, nur die Vorjahresdaten zu Archivieren und alle Daten, die älter ein Jahr sind, zu löschen. Damit ergibt sich eine Vorhaltefrist von einem Jahr. Bestimmte Vertragsdaten mit Kunden, wie zum Beispiel Mahnbescheide, müssen aus rechtlichen Gründen bis zu 30 Jahre aufbewahrt werden [107].

Auf Grundlage der Kritikalität und der bestimmten Intervalle kann nun das technische Archivierungskonzept abgeleitet werden. Hierzu werden die möglichen technischen Sicherungsmöglichkeiten betrachtet, die in Abschnitt 3.1.9 erarbeitet wurden. Große Datenmengen, die sich regelmäßig ändern, sollten z.B. mit einer inkrementellen Datensicherung versehen werden, um Speicherplatz zu sparen und den Sicherungsvorgang zeitlich zu minimieren. Bei großen Datenmengen, die sich nicht so häufig ändern, ist hingegen jedes Mal eine volle Datensicherung empfehlenswert.

Die regelmäßigen Wiederherstellungstest können mit den in Abschnitt 3.2.5 vorgestellten Techniken vorgenommen werden. Es ist sinnvoll, die dort eingeleiteten Notfallübungen mit den Wiederherstellungstests zu kombinieren. Die Ergebnisse dieser Übungen sollten schriftlich fixiert werden. Sollten technische oder menschliche Diskrepanzen festgestellt werden, müssen diese im Nachgang der Übung besprochen werden und ein Vorgehen zur Abstellung dieser Probleme festgelegt werden.

Beispiele für ausgearbeitete Archivierungskonzepte sind das „Archivierungskonzept der Staats- und Universitätsbibliothek Hamburg“ und das „Archivierungs-Rahmenkonzept“ des Innenministeriums des Bundeslandes Nordrhein-Westfalen. Beide Konzepte sind im Anhang 9A.4.18 hinterlegt.

Weitere Vorgaben, wie z.B. das Aufführen verantwortlicher Personen und einzuhaltender Regularien werden im Systembaustein „OPS“ im Kapitel OBS.1.2.2 (Betrieb – Eigener Betrieb – Weiterführende Aufgaben – Archivierung) des IT-Grundschutzkompendium des BSI aufgeführt.

3.3.4 APP.4.3.A24 – Datenverschlüsselung in der Datenbank

Anforderungsbeschreibung

„Die Daten in den Datenbanken SOLLTEN verschlüsselt werden. Dabei SOLLTEN vorher unter anderem folgende Faktoren betrachtet werden:

- Einfluss auf die Performance,
- Schlüsselverwaltungsprozesse und -verfahren, einschließlich separater Schlüsselaufbewahrung und -sicherung,
- Einfluss auf Backup-Recovery-Konzepte,
- funktionale Auswirkungen auf die Datenbank, beispielsweise Sortiermöglichkeiten.“

Umsetzungsvorschlag

MySQL

Die Oracle Corp. bot in der Enterprise-Variante von MySQL eine integrierte Datenverschlüsselung, inklusive Schlüsselmanagement, an. Nach eigenen Angaben erfüllt der Hersteller hiermit die Richtlinien nach „Health Insurance Portability and Accountability Act“ (HIPAA), „Sarbanes-Oxley“ (SOx“) und „Payment Card Industry-Data Security Standard (PCI-DSS)“ [108].

In der Community-Variante bot MySQL die Funktion „`AES_ENCRYPT()`“, die es ermöglicht, Daten mit einer „Advanced Encryption Standard“ (AES)-Verschlüsselung mit einer Schlüssellänge bis zu 256 Bits zu verschlüsseln. Die Oracle Corp. gab selbst an, dass längere Schlüssel weniger performant sind. Standardmäßig wurde ein 128 Bit langer Schlüssel verwendet. Mit der Funktion „`AES_DECRYPT()`“ können die Daten wieder entschlüsselt werden [109].

In MySQL Community wurde kein Schlüsselmanagement angeboten, dass für den Austausch von Schlüsseln zu diesem Zweck verwendet werden kann. Zwar bietet der Hersteller mit „MySQL Keyring“ ein Schlüsselmanagement an, doch konnte dieses nur für interne Services und Plugins genutzt werden [110].

Wie zuvor erwähnt, hat die Verschlüsselung großen Einfluss auf die Performance. Bei jedem Lesen und Schreiben müsste die Berechnung der Verschlüsselung durchgeführt werden. Funktionen wie z.B. Sortieren können nicht mehr über die verschlüsselten Daten erfolgen, da hier dann nur die gehashten Werte sortiert werden würden. Daher eignet sich diese Verschlüsselung nicht für eine dauerhafte und vollständig umgesetzt Datenbankverschlüsselung.

PostgreSQL

Mit dem Plugin „pgcrypto“ unterstützte PostgreSQL unterschiedliche kryptografische Funktionen, die zur Datenverschlüsselung eingesetzt werden können. Hierzu zählen unter anderem „Secure Hash Algorithm“ (SHA) in den Varianten 1, 224, 256, 384 und 512, Blowfish und AES. Zusätzlich kann das Plugin mit OpenSSL kompiliert werden, dann werden alle Funktionen, die OpenSSL in der jeweiligen Version bietet, unterstützt.

„pgcrypto“ baut auf dem OpenPPG-Standard auf und unterstützt symmetrische und asymmetrische Verschlüsselungen. Mit den Funktionen „`pgp_sym_encrypt(data, pw, options)`“ bzw. „`pgp_sym_decrypt(data, pw, options)`“ können die Daten symmetrisch ver- bzw. entschlüsselt werden. In den Optionen kann dann der kryptografische Algorithmus festgelegt werden [111].

Auch bei PostgreSQL müsste jeder Zellwert einzeln verschlüsselt werden. Die Verschlüsselung hätte daher großen Einfluss auf die Performance. Bei jedem Lesen und Schreiben müsste die Berechnung durchgeführt werden. Funktionen wie z.B. Sortieren können nicht mehr über die verschlüsselten Daten erfolgen, da hier dann nur die gehashten Werte sortiert werden würden. Daher eignet sich diese Verschlüsselung nicht für eine dauerhafte und vollständig umgesetzt Datenbankverschlüsselung.

MongoDB

MongoDB Enterprise unterstützte die sogenannte „data at rest“-Verschlüsselung, d.h. Verschlüsselung von Daten, die abgelegt sind (im Gegensatz zu „data at transit“, Daten, die in Bewegung sind). Der Hersteller schrieb in seiner Dokumentation, dass eine zusätzliche Keymanagement-Software dringend empfohlen ist. Standards wie z.B. HIPAA, PCI-DSS und FERPA („Family Education Rights and Privacy Act“) werden mit der Verschlüsselung eingehalten. Standardmäßig wird der Algorithmus AES256-CBC („Cipher Block Chaining“), in der Implementierung von OpenSSL, genutzt. In der Version für Linux wurde zusätzlich der Algorithmus AES256-GCM (Galois/Counter Mode) angeboten [112].

Für die Community-Variante wurde zu diesem Zeitpunkt keine Verschlüsselungsmöglichkeit angeboten.

Allgemein

Die Opensource-Software „MyDiamo“ wurde vom Unternehmen Penta Security Systems angeboten und bot Datenverschlüsselung für die Datenbanken MySQL, PostgreSQL, MariaDB und Percona. Sie könnte als Alternative zu nativer bzw. kostenpflichtiger Dienste genutzt werden. [113]. Dieses Tool wurde in dieser Arbeit nicht weiter betrachtet.

3.3.5 APP.4.3.A25 – Sicherheitsüberprüfungen von Datenbanksystemen

Anforderungsbeschreibung

„Datenbanksysteme SOLLTEN regelmäßig mithilfe von Sicherheitsüberprüfungen kontrolliert werden. Bei den Sicherheitsüberprüfungen SOLLTEN die systemischen und herstellerspezifischen Aspekte der eingesetzten Datenbank-Infrastruktur (z. B. Verzeichnisdienste) sowie des eingesetzten Datenbankmanagementsystems betrachtet werden.“

Umsetzungsvorschlag

Sicherheitsüberprüfungen von IT-Systemen sollten nach definierten Standards und von unbeteiligten Dritten durchgeführt werden. Es gibt unterschiedliche kommerzielle Anbieter, die eine unabhängige Sicherheitsüberprüfung anbieten und eine entsprechende Zertifizierung durchführen können.

Das BSI bietet die „ISO 27001 Zertifizierung auf der Basis von IT-Grundschutz“ an und stellt hierzu online eine Liste aller derzeit berechtigten Auditoren bereit [114].

Andere Anbieter, wie z.B. der TÜV Süd, bieten Zertifizierungen nach weiteren Standards, wie z.B. nach IEC6244327, C2M228 oder COblT5[29] an [115].

Die Anforderung des BSI sieht vor, dass spezifische Aspekte der Datenbank-Infrastruktur betrachtet werden sollen. Daher ist es essenziell, einen Auditor zu beauftragen, welcher Erfahrung mit dem konkret zu untersuchenden DB-System hat.

---

[27] internationale Normenreihe über „Industrielle Kommunikationsnetze - IT-Sicherheit für Netze und Systeme“

[28] Cybersecurity Capability Maturity Model

[29] Control Objectives for Information and Related Technology

MySQL

Einige kommerzielle Anbieter boten Sicherheitsüberprüfungen für MySQL-Datenbanksysteme an. Dazu zählt z.B. Glock IT-Consulting GmbH [116].

Zusätzlich gab es unterschiedliche Opensource Initiativen, die eine automatisierte Überprüfung sicherheitskritischer Einstellungen ermöglicht. Diese Veröffentlichungen waren jedoch zumeist mehrere Jahre alt und daher nicht mehr auf aktuellem Stand. Beispiele hierfür sind DbDat (2018) [117] und MySAT (2018) [118]. Der Testdurchlauf mit MySAT hat fehlerfrei funktioniert und einen entsprechenden Bericht ausgegeben. Die Skriptdateien sind dem Anhang 9A.4.19.1 angefügt.

PostgreSQL

Für PostgreSQL gab es kommerzielle Anbieter, welche explizit eine Sicherheitsüberprüfung von PostgreSQL-basierten Datenbanksystemen anbieten. Beispiele hierfür sind das Unternehmen 2nd Quadrant Ltd. [119], welches maßgeblich an der Weiterentwicklung von PostgreSQL beteiligt waren und EnterpriseDB Corporation [120], welches mit EnterpriseDB eine eigene, kommerzielle PostgresSQL-Datenbank vertrieb.

Zusätzlich gab es unterschiedliche Opensource Initiativen, die eine automatisierte Überprüfung sicherheitskritischer Einstellungen ermöglichte. Diese Veröffentlichungen waren jedoch zumeist mehrere Jahre alt und daher nicht mehr auf aktuellem Stand. Ein Beispiel hierfür ist DbDat (2018) [117]. Der Testdurchlauf hat mit dem Testsystem nicht funktioniert. Die ausgegebenen Fehlermeldungen legten nahe, dass der Quellcode veraltet ist und nicht auf einem aktuellen System läuft. Der Quellcode ist dem Anhang 9A.4.19.2 angefügt.

MongoDB

Auf MongoDB-spezialisierte Beratungsunternehmen bieten oftmals auch Security-Beratung an und unterstützen bei der Erstellung von Sicherheitskonzepten. Beispielhaft seien hier Pythian [121] und DSP [122] genannt.

Zusätzlich gab es unterschiedliche Opensource Initiativen, die eine automatisierte Überprüfung sicherheitskritischer Einstellungen ermöglicht. Ein Beispiel hierfür ist MongoAudit [123]. Auf dem Testsystem verursachte dieses Programm bei einem Testdurchlauf einen Laufzeitfehler und lieferte daher kein Ergebnis. Die genutzte Quellcode-Version ist dem Anhang 9A.4.19.3 hinzugefügt.

4 Auswertung und Fazit

Die fachliche Bewertung der Umsetzung der Anforderungen aus dem IT-Grundschutzkompendium des BSI wurde anhand der einzelnen Anforderung vorgenommen. Anschließend wurde ein Fazit aus den Ergebnissen gezogen und ggf. weitere Maßnahmen vorgeschlagen.

4.1 Bewertung der Systeme nach Anforderungen

Zur Auswertung wurde die vom BSI bereitgestellte Checkliste herangezogen, die dem System-Auditor zur Überprüfung der Umsetzungen dient und in Kapitel 3.2.11 vorgestellt wurde. Die Checkliste ließ eine Bewertung in vier Kategorien zu:

„ja“	Für die volle Umsetzung der Anforderung
„teilw.“	Für eine nur zum Teil vollständige Umsetzung der Anforderung
„nein“	Für eine nicht erfolgte Umsetzung
„entbehrl.“	Für eine nicht notwendige Umsetzung

Anhand dieser Kategorien wurde auch in dieser Arbeit die Bewertung durchgeführt. Dabei wurden die Kategorien wie folgt gedeutet:

„ja“	Für die volle Umsetzung der Anforderung
„teilw.“	Für eine nur zum Teil mögliche oder schwer realisierbare Umsetzung der Anforderung
„nein“	Für eine nicht mögliche Umsetzung
„entbehrl.“	Für organisatorische Maßnahmen, die keiner vorrangig technischen Umsetzung im DBS bedürfen

Zur messbaren Bewertung wurden den Kategorien Zahlenwerte zugeordnet: „ja“ = 2; „teilw.“ = 1; „nein“ = 0; „entbehrl.“ = 0. Da die organisatorischen unabhängig vom eingesetzten DBS eingeführt werden mussten, wurden diese aus der Bewertung herausgenommen.

Die weiteren Spalten („Umsetzung bis“, „Verantwortlich“, „Bemerkung“ und „Kostenschätzung“) aus der bereitgestellten Checkliste wurden zur Auswertung nicht benötigt und daher hier weggelassen.

Abzüglich der zehn organisatorischen Maßnahmen, blieben noch 15 technische Anforderungen, die es umzusetzen galt. Somit war eine maximale Bewertung von (2 Punkte ×15 Anforderungen =) 30 Punkten möglich.

In der Auswertung erhielt MySQL 22 Punkte, PostgreSQL 20 Punkte und MongoDB 14 Punkte. Somit konnten in MySQL 73,33%, PostgreSQL 66,67% und MongoDB 46,67% der Anforderungen umgesetzt werden. Die konkrete Vergabe der Punkte kann in Tabelle 13 nachgelesen werden.

Tabelle 13 - Bewertung der DBS MySQL, PostgreSQL und MongoDB nach Anforderungen des BSI

Anforderung	Titel	MySQL	PostgreSQL	MongoDB	Anmerkung
APP.4.3.A1	Erstellung einer Sicherheitsrichtlinie für Datenbanksysteme	0	0	0	organisatorisch
APP.4.3.A2	Installation des Datenbankmanagementsystems	1	0	1	
APP.4.3.A3	Basishärtung des Datenbankmanagementsystems	2	2	1	
APP.4.3.A4	Geregeltes Anlegen neuer Datenbanken	0	0	0	organisatorisch
APP.4.3.A5	Benutzer- und Berechtigungskonzept	2	2	2	
APP.4.3.A6	Passwortänderung	2	1	0	
APP.4.3.A7	Zeitnahes Einspielen von Sicherheitsupdates	2	2	2	
APP.4.3.A8	Datenbank-Protokollierung	2	2	0	
APP.4.3.A9	Datensicherung eines Datenbanksystems	2	1	1	
APP.4.3.A10	Auswahl geeigneter Datenbankmanagementsysteme	0	0	0	organisatorisch
APP.4.3.A11	Ausreichende Dimensionierung der Hardware	0	0	0	organisatorisch
APP.4.3.A12	Einheitlicher Konfigurationsstandard von Datenbankmanagementsystemen	2	2	2	
APP.4.3.A13	Restriktive Handhabung von Datenbank-Links	0	1	0	
APP.4.3.A14	Überprüfung der Datensicherung eines Datenbanksystems	0	0	0	
APP.4.3.A15	Schulung der Datenbankadministratoren	0	0	0	organisatorisch
APP.4.3.A16	Verschlüsselung der Datenbankanbindung	2	2	2	
APP.4.3.A17	Datenübernahme oder Migration	0	0	0	organisatorisch
APP.4.3.A18	Überwachung des Datenbankmanagementsystems	2	2	2	
APP.4.3.A19	Schutz vor schädlichen Datenbank-Skripten	1	1	1	
APP.4.3.A20	Regelmäßige Audits	0	0	0	organisatorisch
APP.4.3.A21	Einsatz von Datenbank Security Tools	1	1	0	
APP.4.3.A22	Notfallvorsorge	0	0	0	organisatorisch
APP.4.3.A23	Archivierung	0	0	0	organisatorisch
APP.4.3.A24	Datenverschlüsselung in der Datenbank	1	1	0	
APP.4.3.A25	Sicherheitsüberprüfungen von Datenbanksystemen	0	0	0	organisatorisch
Summe		**22**	**20**	**14**	

4.2 Fazit und Ausblick

Die Umsetzung der organisatorischen Maßnahmen bedarf einigen personellen Aufwand und sollte von höchster, organisatorischer Stelle unterstützt werden, damit die notwendigen Ressourcen zur Umsetzung freigegeben werden. Es ist dennoch genauso wichtig, alle Mitarbeiter in die Umsetzung miteinzubeziehen und von der Wichtigkeit und Notwendigkeit der Aufgabe zu überzeugen, da diese Prozesse schlussendlich von diesen Personen eingehalten werden müssen.

Die Prozesse und Datenblätter sind ohne konkrete fachliche Anforderungen entworfen worden. In einer realen Umsetzung müssen diese an die Fachbereiche explizit angepasst werden. Das erfordert ein fundiertes Wissen in den technischen Sicherheitsanforderungen und den Fachanforderungen und sollte von Personen durchgeführt werden, die hier bereits Erfahrung haben. Im Realfall wird so ein Prozess in einer oder mehreren Testphasen evaluiert und iterativ angepasst. Das ist in den in dieser Arbeit vorgestellten Prozessen nicht passiert.

Die technischen Maßnahmen konnten in den untersuchten Opensource-Systemen nicht mal zu 75% umgesetzt werden. Zum Teil mussten weitere Softwareprodukte hinzugezogen werden, um die Umsetzung zu ermöglichen. Die Systeme MySQL und PostgreSQL wurden von vielen Entwickler-Communities unterstützt, so dass es hier zumeist adäquate Ergänzungen gab. Mit 73,33% konnten in MySQL die meisten Anforderungen umgesetzt werden.

In der nicht-relationalen Datenbank MongoDB konnten mit 46,67% die wenigsten Anforderungen umgesetzt werden. Das kann zum einem daran liegen, dass das System das jüngste der drei zu untersuchen ist (MySQL wird seit 1995 entwickelt, PostgreSQL seit 1996, MongoDB erst seit 2009). Die fehlende Unterstützung zur Einbindung externer Plugins erschwert die Erweiterung durch unabhängige Entwickler zusätzlich. Zum anderen muss bedacht werden, dass die Anforderungsliste explizit für relationale DBS entworfen wurde. Eine Version für nicht-relationale DBS ist vom BSI bereits in Planung, hierzu wurden aber bisher noch keine Details veröffentlicht [124].

Alle drei untersuchten Systeme werden von einem kommerziellen Anbieter bereitgestellt, der das gleiche Datenbanksystem mit erweiterten Funktionalitäten zum Kauf bzw. zur Miete anbietet. Diese erweiterten Funktionen dienen zum Großteil Sicherheits- und Überwachungsfunktionen, die für einen Regelbetrieb unbedingt erforderlich sind und die Anforderungen des BSI erfüllen würden. Die Anbieter nutzen die Verbreitung der Opensource Systeme bewusst, um mittels Lizenzvereinbarungen und erweiterten Funktionalitäten ein profitables Geschäft aufzubauen und die Weiterentwicklung zu finanzieren [125].

Ein Großteil der Funktionen, die zur Umsetzung der Anforderungen notwendig waren, mussten explizit aktiviert werden. Die Standardinstallation wies hiermit deutliche Sicherheitslücken auf und ein Nutzer ohne Kenntnis der Sicherheitsanforderungen wird diese nicht aktivieren. Im Fall der MongoDB mussten selbst grundlegende Einstellungen, wie die Authentifizierung von Benutzern, explizit aktiviert werden, da sie standardmäßig nicht aktiv waren. In einer Studie der Universität Bonn wurde aufgezeigt, dass selbst erfahrene Entwickler nur auf Nachfrage explizit Sicherheitsanforderungen aktivieren, sofern diese nicht direkt vorgegeben werden [126]. Daher sollte bei Sicherheitseinstellungen die „Opt-In“-Vorgehensweise zu einer „Opt-Out“-Strategie geändert werden. Somit müsste der Datenbankadministrator die Funktionen explizit deaktivieren, was wahrscheinlich in weniger Fällen vorkommt.

Das Bundesamt für Sicherheit in der Informationstechnik bietet mit dem IT-Grundschutzkompendium einen umfassenden Maßnahmenkatalog, an dem sich Sicherheitsbeauftragte orientieren können und eine offizielle Verifizierung der eigenen IT-Sicherheit erlangen können. Das Kompendium selbst enthält technische und organisatorische Maßnahmen, die als Vorgaben zu sehen sind, aber keine konkrete Implementierung vorschreiben. Das bedeutet, die Art und Weise wie die Anforderungen umgesetzt werden, ist bewusst offengehalten, damit die Anforderungen auf unterschiedlichste Systeme angewandt werden können und bedürfen deshalb explizites Fachwissen in den jeweiligen Systemen. Dennoch lassen sich nicht alle Anforderungen auf jedem System umsetzen, ein Beispiel hierfür ist die Anforderung APP.4.3.A13 „Restriktive Handhabung von Datenbanklinks“. Im Falle der nicht-relationalen Datenbanken hat das BSI bis

heute noch keinen Maßnahmenkatalog veröffentlicht, dabei sind nicht-relationale Systeme im eingangs beschriebenen Ranking auf vier Plätzen in den zehn meist-verbreiteten Datenbanksystemen vertreten [4]. Auch andere aktuelle Technologien, wie zum Beispiel Containerisierung, sind im Kompendium noch nicht vertreten. An dieser Stelle muss das BSI noch nacharbeiten.

In dieser Arbeit lag der Fokus auf den Datenbanksystemen. Die Anforderungen und auch die Recherchen zeigen aber, dass für eine vollständige Absicherung die Betrachtung der Netzwerke und Hostsysteme zwingend notwendig ist. Zusätzlich bieten die Datenbanksysteme auf den unterschiedlichen Hostsystemen teilweise unterschiedliche Funktionalitäten. Im nächsten Schritt wäre eine vergleichende Untersuchung auf unterschiedlichen Systemen sinnvoll.

5 Literaturverzeichnis

[1] Bitkom e.V., „Open Source Monitor 2019," Bitkom e.V., Berlin, 2020.

[2] World Economic Forum, „The Global Risks Report 2019, 14th Edition," World Economic Forum, Genf, 2019.

[3] Verizon Communications Inc., „2019 Data Breach Investigation Report," Verizon Communications Inc., New York City, 2019.

[4] solidIT consulting & software development gmbh, „DB Engines," 2020. [Online]. Available: https://db-engines.com/de/ranking. [Zugriff am 02 2020].

[5] TrustRadius Inc., „List of Top Open-Source Database Software 2020," 2020. [Online]. Available: https://www.trustradius.com/open-source-database. [Zugriff am 12 05 2020].

[6] Bundesamt für Sicherheit in der Informationstechnik, IT-Grundschutzkompendium, Köln: Reguvis Fachmedien GmbH, 2020.

[7] Bundesamt für Sicherheit in der Informationstechnik, „IT-Grundschutz-Kompendium," 2020. [Online]. Available: https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzKompendium/itgrundschutzKompendium_node.html. [Zugriff am 02 202].

[8] A. Kemper und A. Eickler, Datenbanksysteme - Eine Einführung, Berlin/Boston: Walter De Gruyter GmbH, 2015.

[9] A. Meier und M. Kaufmann, SQL & NoSQL Databases, Wiesbaden: Springer Vieweg, 2019.

[10] C. Date, „Relational Database Management: What First Normal Form Really Means,“ in *Data on Database: Writings 2000-2006*, New York, Apress (Springer-Verlag), 2006, pp. 103-138.

[11] A. B. M. Moniruzzaman und S. A. Hossain, „NoSQL Database: New Era of Databases for Big data Analytics - Classification, Characteristics and Comparison,“ *International Journal of Database Theory and Application,* Bd. 4, Nr. 6, 2013.

[12] Forrester Research, Inc., „Home · Forrester,“ 2020. [Online]. Available: https://go.forrester.com/. [Zugriff am 12 05 2020].

[13] N. Yuhanna, „The Forrester Wave™: Big Data NoSQL, Q1 2019,“ Forrester Research, Cambridge, United States, 2019.

[14] N. Yuhanna, „The Forrester Wave™: Database-As-A-Service, Q2 2019,“ Forrester Research Inc., Cambridge, United States, 2019.

[15] Oracle Corporation, „MySQL Developer Zone,“ 2020. [Online]. Available: https://dev.mysql.com/. [Zugriff am 02 2020].

[16] The PostgreSQL Global Development Group, „PostgreSQL,“ 2020. [Online]. Available: https://www.postgresql.org/about/. [Zugriff am 02 2020].

[17] MongoDB Inc., „MongoDB,“ 2020. [Online]. Available: https://www.mongodb.com/. [Zugriff am 02 2020].

[18] Bundesamt für Sicherheit in der Informationstechnik, „Der IT-Grundschutz,“ 2020. [Online]. Available: https://www.bsi.bund.de/DE/Themen/ITGrundschutz/itgrundschutz_node.html. [Zugriff am 02 2020].

[19] Bundesamt für Sicherheit in der Informationstechnik, *BSI-Standard 200-1: Managementsysteme für Informationssicherheit,* Bonn, 2017.

[20] Bundesamt für Sicherheit in der Informationstechnik, *BSI-Standard 200-2: IT-Grundschutz-Methodik,* Bonn, 2017.

[21] Bundesamt für Sicherheit in der Informationstechnik, *BSI-Standard 200-3: Risikoanalyse auf der Basis von IT-Grundschutz,* Bonn, 2017.

[22] Bundesamt für Sicherheit in der Informationstechnik, *BSI-Standard 100-4: Notfallmanagement,* Bonn, 2008.

[23] Bundesamt für Sicherheit in der Informationstechnik, *Zuordnungstabelle: Zuordnung ISO/IEC 27001 sowie ISO/IEC 27002 zum modernisierten IT-Grundschutz,* Bonn, 2018.

[24] Bundesamt für Sicherheit in der Informationstechnik, *Leitfaden zur Basis-Absicherung nach IT-Grundschutz,* Bonn, 2017.

[25] Bundesamt für Sicherheit in der Informationstechnik, „Webkurs IT-Grundschutz: Beschreibung des Beispielunternehmens RECPLAST GmbH," Bundesamt für Sicherheit in der Informationstechnik, Bonn, 2006.

[26] Bundesamt für Sicherheit in der Informationstechnik, „BSI - IT-Grundschutz - Profile," 2020. [Online]. Available: https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzProfile/itgrundschutzProfile_node.html. [Zugriff am 2020 04 11].

[27] Max-Planck-Gesellschaft, „IT-Sicherheitsrichtlinie der Max-Planck-Gesellschaft," Berlin, 2017.

[28] HiSolutions AG, Berlin, „Muster-IT-Sicherheitskonzepte der EKD," Evangelische Kirche in Deutschland (EKD), Hannover, 2014.

[29] MongoDB Inc., „MongoDB Manual," 2020. [Online]. Available: https://docs.mongodb.com/manual/. [Zugriff am 03 2020].

[30] Oracle Corporation, „MySQL :: MySQL Documentation," [Online]. Available: https://dev.mysql.com/doc/.

[31] Deutsche Telekom AG, „Privacy and Security Assessment," Deutsche Telekom AG, Bonn, 2017.

[32] Center for Internet Security, „CIS Fact Sheet," Center for Internet Security, East Greenbush, NY, 2019.

[33] Center for Internet Security, „CIS Center for Internet Security," 2020. [Online]. Available: https://www.cisecurity.org/. [Zugriff am 03 2020].

[34] Bundesamt für Sicherheit in der Informationstechnik, „BSI für Bürger - Passwörter," 2019. [Online]. Available: https://www.bsi-fuer-buerger.de/BSIFB/DE/Empfehlungen/Passwoerter/passwoerter_node.html. [Zugriff am 03 2020].

[35] Deutsche Telekom Gruppe, Technischer Basisschutz von IT-/NT-Systemen, Bonn: Deutsche Telkom AG, 2016.

[36] K.-P. L. Vu, R. W. Proctor, A. Bhargav-Spantzel, B.-L. (. Tai, J. Cook und E. E. Schultz, „Improving password security and memorability to protect personal and organizational information," *International Journal of Human-Computer Studies,* Bd. 65, Nr. 8, pp. 744-757, 2007.

[37] ANSI und INCITIS, „Role Based Access Control". America Patent ANSI INCITS 359-2004, 03 02 2004.

[38] M. Uddin, S. Islam und A. Al-Nemrat, „A Dynamic Access Control Model Using Authorising Workflow and Task-Role-Based Access Control," *IEEE Access,* Bd. 7, pp. 166676-166689, 2019.

[39] R. Sandhu, D. F. Ferraiolo und D. R. Kuhn, „The NIST Model for Role-Based Access Control: Towards a Unified Standard," in *Fifth ACM Workshop on Role-Based Access Control (RBAC '00)*, Berlin, 2000.

[40] M. Rouse, „What is role-based access control (RBAC)? - Definition from WhatIs.com," TechTarget, Inc., 09 2018. [Online]. Available: https://searchsecurity.techtarget.com/definition/role-based-access-control-RBAC. [Zugriff am 12 04 2020].

[41] N. Schmidt, A. Lüder, K. Hell, H. Röpke und J. Zawisza, „A generic model for the End-of-Life phase of production systems," in *IECON 2016 - 42nd Annual Conference of the IEEE Industrial Electronics Society*, Florence, 2016, pp. 5693-5698.

[42] Oracle Corporation, „MySQL :: MySQL 8.0 Reference Manual :: 6.4.5 MySQL Enterprise Audit," 2020. [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/audit-log.html. [Zugriff am 05 2020].

[43] MongoDB Inc., „Auditing — MongoDB Manual," 2020. [Online]. Available: https://docs.mongodb.com/manual/core/auditing/. [Zugriff am 05 2020].

[44] Oracle Corporation, „MySQL :: MySQL 5.7 Reference Manual :: 5.4 MySQL Server Logs," 2020. [Online]. Available: https://dev.mysql.com/doc/refman/5.7/en/server-logs.html. [Zugriff am 07 07 2020].

[45] The PostgreSQL Global Development Group, „PostgreSQL: Documentation: 12: 19.8. Error Reporting and Logging," 2020. [Online]. Available: https://www.postgresql.org/docs/12/runtime-config-logging.html. [Zugriff am 07 07 2020].

[46] PostgreSQL Development Community, „PostgreSQL Auditing Extension | PGAudit," [Online]. Available: https://www.pgaudit.org/. [Zugriff am 05 2020].

[47] 2ndQuadrant Ltd., „Audit trigger 91plus - PostgreSQL wiki," 2017. [Online]. Available: https://wiki.postgresql.org/wiki/Audit_trigger_91plus. [Zugriff am 05 2020].

[48] MongoDB, Inc., „Log Messages — MongoDB Manual," 2020. [Online]. Available: https://docs.mongodb.com/manual/reference/log-messages/. [Zugriff am 07 07 2020].

[49] Percona LLC, „MongoDB Audit Log: Why and How - Percona Database Performance Blog," 2020. [Online]. Available: https://www.percona.com/blog/2017/03/03/mongodb-audit-log-why-and-how/. [Zugriff am 05 2020].

[50] P. Murugesan und I. Ray, „Audit Log Management in MongoDB," in *IEEE 10th World Congress on Services*, Anchorage, AK, USA, 20214.

[51] Oracle Corporation, „MySQL :: MySQL Backup and Recovery :: 1 Backup and Recovery," 2020. [Online]. Available: https://dev.mysql.com/doc/mysql-backup-excerpt/5.7/en/backup-and-recovery.html. [Zugriff am 05 2020].

[52] S. Riggs, M. Nenciarini und G. Bartolini, „Incremental backup - PostgreSQL wiki," 07 03 2015. [Online]. Available: https://wiki.postgresql.org/wiki/Incremental_backup. [Zugriff am 05 2020].

[53] MongoDB Inc., „Cloud Provider Snapshots — MongoDB Atlas," 2020. [Online]. Available: https://docs.atlas.mongodb.com/backup/cloud-provider-snapshots/. [Zugriff am 05 2020].

[54] MongoDB Inc., „Restore from a Specific Point-in-Time — MongoDB Ops Manager 4.2," 2020. [Online]. Available: https://docs.opsmanager.mongodb.com/current/tutorial/restore-pit-snapshot-http/. [Zugriff am 05 2020].

[55] Amazon Web Services, Inc., „MySQL | Most Popular Open Source Relational Database | AWS," 2020. [Online]. Available: https://aws.amazon.com/de/rds/mysql/what-is-mysql/. [Zugriff am 12 05 2020].

[56] Cybertec Schönig & Schönig GmbH, „Lösungen: Wer verwendet PostgreSQL - Cybertec," 2020. [Online]. Available: https://www.cybertec-postgresql.com/de/postgresql-uebersicht/loesungen-wer-verwendet-postgresql/. [Zugriff am 12 05 2020].

[57] MongoDB, Inc., „Use Cases | MongoDB," 2020. [Online]. Available: https://www.mongodb.com/use-cases. [Zugriff am 12 05 2020].

[58] W. Puangsaijai und S. Puntheeranurak, „A Comparative Study of Relational Database and Key-Value Database for Big Data Applications," in *5th International Electrical Engineering Congress*, Pattaya, Thailand, 2017.

[59] M. M. Patil, A. Hanni, C. Tejeshwar und P. Patil, „A qualitative analysis of the performance of MongoDB vs MySQL Database based on insertion and retriewal operations using a web/android application to explore Load Balancing – Sharding in MongoDB and its advantages," in *International conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)*, Palladam, Indien, 2018.

[60] A. Monjaras, E. Bendezú und C. Raymundo, „Decision Tree Model to Support the Successful Selection of a Database Engine for Novice Database Administrators," in *2019 8th International Conference on Industrial Technology and Management*, Cambridge, UK, 2019.

[61] H. Farias, „How to choose the right type of database for your enterprise," 19 04 2019. [Online]. Available: https://www.infoworld.com/article/3268871/how-to-choose-the-right-type-of-database-for-your-enterprise.html. [Zugriff am 12 05 2020].

[62] P. Mishra, S. Bhatnagar und A. Katal, „Cloud Container Placement Policies: A Study and Comparison," in *International Conference on Computer Networks and Inventive Communication Technologies*, Coimbatore, India, Springer Nature Switzerland AG, 2019, pp. 513-524.

[63] Oracle Corporation, „MySQL :: MySQL 8.0 Reference Manual :: 4.2.2.2 Using Option Files," 2020. [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/option-files.html. [Zugriff am 16 05 2020].

[64] The PostgreSQL Global Development Group, „PostgreSQL: Documentation: 12: Chapter 19. Server Configuration," 2020. [Online]. Available: https://www.postgresql.org/docs/12/runtime-config.html. [Zugriff am 16 05 2020].

[65] MongoDB, Inc., „Configuration File Options — MongoDB Manual," 2020. [Online]. Available: https://docs.mongodb.com/manual/reference/configuration-options/. [Zugriff am 16 05 2020].

[66] Oracle Corporation, „Database Links," 2020. [Online]. Available: https://docs.oracle.com/cd/B28359_01/server.111/b28310/ds_concepts002.htm. [Zugriff am 16 05 2020].

[67] Free Software Foundation, Inc., „GNU tar 1.32: Basic Tar Format," 2019. [Online]. Available: https://www.gnu.org/software/tar/manual/html_node/Standard.html. [Zugriff am 17 05 2020].

[68] Free Software Foundation, Inc., „GNU Gzip," 2018. [Online]. Available: https://www.gnu.org/software/gzip/manual/gzip.html. [Zugriff am 27 06 2020].

[69] M. Banck, „Wie PostgreSQL Ihre Daten Sicher Hält,“ in *PGConf.DE*, Leipzig, Deutschland, 2019.

[70] Bundestanstalt für Finanzdienstleistungsaufsicht, „Bankaufsichtliche Anforderungen an die IT (BAIT),“ *Rundschreiben 10/2017 (BA) in der Fassung vom 14.09.2018,* 2018.

[71] ITIL®-Prozesslandkarte & ITIL®-Wiki, „Checkliste Protokoll Katastrophenübung | IT Process Wiki,“ 29 03 2019. [Online]. Available: https://wiki.de.it-processmaps.com/index.php/Checkliste_Protokoll_Katastrophen%C3%BCbung. [Zugriff am 17 05 2020].

[72] Heise Medien GmbH & Co. KG, „IT-Sicherheit: Notfallplanung und Notfallübungen,“ 2020. [Online]. Available: https://www.heise-events.de/workshops/notfallplanung. [Zugriff am 17 05 2020].

[73] Cyber Akademie GmbH, „IT-Notfallübungen – Krisenstabsübungen,“ 2020. [Online]. Available: https://www.cyber-akademie.de/it_notfall.jsp. [Zugriff am 17 05 2020].

[74] Oracle Corporation, „Datenbank-Sicherheit Grundüberlegungen,“ Kalifornien, USA, 2017.

[75] M. Alotaibi und W. Alfehaid, „Information Security Awareness: A Review of Methods, Challenges and Solutions,“ in *Internet Technology and Secured Transactions (ICITST-2018)*, University of Cambridge, Churchill College, 2019.

[76] Bundesamt für Sicherheit in der Informationstechnik, „BSI - IT-Grundschutz - BSI-Standards - Umsetzungsrahmenwerk zum Notfallmanagement nach BSI-Standard 100-4," 2020. [Online]. Available: https://www.bsi.bund.de/DE/Themen/ITGrundschutz/ITGrundschutzStandards/Umsetzungsrahmenwerk/umra.html. [Zugriff am 2020 06 12].

[77] Bundesamt für Sicherheit in der Informationstechnik, „Technische Richtlinie TR-02102-2 Kryptographische Verfahren: Empfehlungen und Schlüssellängen Teil 2 – Verwendung von Transport Layer Security (TLS) Version 2020-01," Bonn, 2020.

[78] Bundesamt für Sicherheit in der Informationstechnik, „Mindeststandard des BSI zur Verwendung von Transport Layer Security (TLS) nach § 8 Absatz 1 Satz 1 BSIG Ɛ Version 2.1 vom 09.04.2020," Bonn, 2020.

[79] OpenSSL Software Foundation, „OpenSSQL Cryptography and SSL/TLS Toolkit," 2018. [Online]. Available: https://www.openssl.org/. [Zugriff am 21 05 2020].

[80] The PostgreSQL Global Development Group, „PostgreSQL: Documentation: 12: 18.9. Secure TCP/IP Connections with SSL," 2020. [Online]. Available: https://www.postgresql.org/docs/12/ssl-tcp.html. [Zugriff am 21 05 2020].

[81] P. Eisentraut und 2ndQuadrant, „Setting SSL/TLS protocol versions with PostgreSQL 12 - 2ndQuadrant | PostgreSQL," 27 11 2019. [Online]. Available: https://www.2ndquadrant.com/en/blog/setting-ssl-tls-protocol-versions-with-postgresql-12/. [Zugriff am 21 05 2020].

[82] OpenSSL Software Foundation, „/docs/man1.1.1/man1/ciphers.html," 2018. [Online]. Available: https://www.openssl.org/docs/man1.1.1/man1/ciphers.html. [Zugriff am 21 05 2020].

[83] MongoDB, Inc., „Configure mongod and mongos for TLS/SSL — MongoDB Manual," 2020. [Online]. Available: https://docs.mongodb.com/manual/tutorial/configure-ssl/. [Zugriff am 21 05 2020].

[84] Oracle Corporation, „Successful Data Migration," Redwood Shores, CA, USA, 2011.

[85] Amazon Web Services, Inc., „AWS Prescriptive Guidance: Database migration strategy," Seattle, Washington, Vereinigte Staaten, 2020.

[86] Oracle Corporation, „MySQL :: MySQL Enterprise Monitor," 2020. [Online]. Available: https://www.mysql.com/products/enterprise/monitor.html. [Zugriff am 24 05 2020].

[87] MongoDB, Inc., „Free Monitoring — MongoDB Manual," 2020. [Online]. Available: https://docs.mongodb.com/manual/administration/free-monitoring/. [Zugriff am 29 05 2020].

[88] MongoDB, Inc., „Monitoring for MongoDB — MongoDB Manual," [Online]. Available: https://docs.mongodb.com/manual/administration/monitoring/. [Zugriff am 30 05 2020].

[89] R. Prof. Dr. Lackes und M. Dr. Siepermann, „Softwarequalität • Definition | Gabler Wirtschaftslexikon," 19 02 2018. [Online]. Available: https://wirtschaftslexikon.gabler.de/definition/softwarequalitaet-45483/version-268775. [Zugriff am 01 06 2020].

[90] C. Rylan, „Why Enforcing Code Style is Important," 30 08 2015. [Online]. Available: https://coryrylan.com/blog/why-enforcing-code-style-is-important. [Zugriff am 31 05 31].

[91] N. C. Zakas, „Why Coding Style Matters — Smashing Magazine,“ 25 10 2012. [Online]. Available: https://www.smashingmagazine.com/2012/10/why-coding-style-matters/. [Zugriff am 31 05 2020].

[92] B. Nice, „What is a Programming Style Guide and why should you care,“ 25 07 2019. [Online]. Available: https://medium.com/level-up-web/what-is-a-programming-style-guide-and-why-should-you-care-9019e51bb7ad. [Zugriff am 31 05 2020].

[93] Microsoft Corporation, „Codekonventionen für C# – C#-Programmierhandbuch | Microsoft Docs,“ 20 07 2015. [Online]. Available: https://docs.microsoft.com/de-de/dotnet/csharp/programming-guide/inside-a-program/coding-conventions. [Zugriff am 31 05 2020].

[94] Google LLC, „Google JavaScript Style Guide,“ 27 05 2020. [Online]. Available: https://google.github.io/styleguide/jsguide.html. [Zugriff am 31 05 2020].

[95] S. Holywell, „SQL style guide by Simon Holywell,“ 25 05 2020. [Online]. Available: https://www.sqlstyle.guide/. [Zugriff am 31 05 2020].

[96] Mozilla Corporation, „SQL Style Guide - Firefox Data Documentation,“ 28 05 2020. [Online]. Available: https://docs.telemetry.mozilla.org/concepts/sql_style.html. [Zugriff am 31 05 2020].

[97] GitLab Inc., „SQL Style Guide | GitLab,“ 31 03 2020. [Online]. Available: https://about.gitlab.com/handbook/business-ops/data-team/sql-style-guide/. [Zugriff am 31 05 2020].

[98] OWASP Foundation, Inc., „SQL Injection | OWASP,“ 2020. [Online]. Available: https://owasp.org/www-community/attacks/SQL_Injection. [Zugriff am 31 05 2020].

[99] J. Söndermann, „MongoStyleGuide: An opinionated guide to data modeling with MongoDB.,“ 10 08 2017. [Online]. Available: https://github.com/jsoendermann/MongoStyleGuide. [Zugriff am 01 06 2020].

[100] K. Nest, „Can You Unit Test Your Database? You Bet | TestRail Quality HubTestRail Quality Hub,“ 13 03 2018. [Online]. Available: https://blog.gurock.com/unit-test-database/. [Zugriff am 01 06 2020].

[101] A. Hafner, „Unit Testing, Databases, and You - Simple Talk,“ 30 05 2019. [Online]. Available: https://www.red-gate.com/simple-talk/sql/database-devops-sql/unit-testing-databases-and-you/. [Zugriff am 01 06 2020].

[102] D. Green, „Not unit testing your databases? You should be | Pluralsight,“ 18 02 2014. [Online]. Available: https://www.pluralsight.com/blog/software-development/unit-testing-databases. [Zugriff am 01 06 2020].

[103] M. Chowanetz , U. Dr. Laude und K. Klinner, Autoren, *Ein Kennzahlensystem für die Informationssicherheit.* [Performance]. Julius-Maximilians-Universität Würzburg , 2013.

[104] Bundesamt für Sicherheit in der Informationstechnik, „Zertifizierung nach ISO 27001 auf der Basis von IT-Grundschutz,“ Bonn, 2019.

[105] S. Kasulke und J. Bensch, Zero Outage: Kompromisslose Qualität in der IT im Zeitalter der Digitalisierung, Wiesbaden: Springer Gabler, 2017.

[106] M. Swanson, P. Bowen , A. W. Phillips , D. Gallup und D. Lynes , „Contingency Planning Guide for Federal Information Systems," *NIST Special Publication 800-34,* 05 2010.

[107] firma.de , „Aufbewahrungsfristen 2020 – Alle Fristen im Überblick – firma.de," 2020. [Online]. Available: https://www.firma.de/unternehmensfuehrung/aufbewahrungsfristen-fuer-2020-das-muessen-sie-wissen/. [Zugriff am 28 06 2020].

[108] Oracle Corporation, „MySQL :: MySQL Enterprise Encryption," 2020. [Online]. Available: https://www.mysql.com/de/products/enterprise/encryption.html. [Zugriff am 14 06 2020].

[109] Oracle Corporation, „MySQL :: MySQL 8.0 Reference Manual :: 12.13 Encryption and Compression Functions," 2020. [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/encryption-functions.html#function_aes-encrypt. [Zugriff am 14 06 2020].

[110] Oracle Corporation, „MySQL :: MySQL 8.0 Reference Manual :: 6.4.4 The MySQL Keyring," 2020. [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/keyring.html. [Zugriff am 14 06 2020].

[111] The PostgreSQL Global Development Group, „PostgreSQL: Documentation: 12: F.25. pgcrypto," 2020. [Online]. Available: https://www.postgresql.org/docs/current/pgcrypto.html. [Zugriff am 16 06 2020].

[112] MongoDB, Inc., „Encryption at Rest — MongoDB Manual," 2020. [Online]. Available: https://docs.mongodb.com/manual/core/security-encryption-at-rest/. [Zugriff am 18 06 2020].

[113] Penta Security Systems Co, „Ultimate Encryption Solution for Open Source DBMS," 2020. [Online]. Available: https://mydiamo.com/. [Zugriff am 21 06 2020].

[114] Bundesamt für Sicheheit in der Informationstechnik, „BSI - Zertifizierte 27001-Auditoren auf der Basis von IT-Grundschutz," 2020. [Online]. Available: https://www.bsi.bund.de/DE/Themen/ZertifizierungundAnerkennung/Managementsystemzertifizierung/Zertifizierung27001/Auditoren/iso27001auditoren_node.html. [Zugriff am 20 06 2020].

[115] TÜV SÜD AG, „Cyber Security Risk Assessment für Unternehmen | TÜV SÜD," 2020. [Online]. Available: https://www.tuvsud.com/de-de/dienstleistungen/cyber-security/cyber-security-assessments/cyber-security-risk-assessment. [Zugriff am 20 06 2020].

[116] Glock IT-Consulting GmbH, „SQL-Datenbank Audit und Checkliste Datenbanksicherheit," 2020. [Online]. Available: https://www.database-security.de/datenbank-audit-check/. [Zugriff am 20 06 2020].

[117] P. Maddux, „Db Database Assessment Tool," 26 06 2018. [Online]. Available: https://github.com/foospidy/DbDat. [Zugriff am 20 06 2020].

[118] XeniaLAB, „MySQL Database Security Assessment Tool," 30 09 2018. [Online]. Available: https://github.com/meob/MySAT. [Zugriff am 20 06 2020].

[119] 2ndQuadrant Ltd, „PostgreSQL Database Security Audit - 2ndQuadrant | PostgreSQL," 2020. [Online]. Available: https://www.2ndquadrant.com/de/services-2/postgresql-security-audit/. [Zugriff am 20 06 2020].

[120] EnterpriseDB Corporation, „PostgreSQL Security Compliance Assessment & Audit | Defend Against Data Breaches & Protect Database From Threats | EDB," 2020. [Online]. Available: https://www.enterprisedb.com/postgresql-security-assesment-audit-hardening-database. [Zugriff am 20 06 2020].

[121] Pythian Services Inc., „MongoDB Services | Consulting, Support & Management | Pythian®," 2020. [Online]. Available: https://pythian.com/mongodb-consulting/. [Zugriff am 20 06 2020].

[122] Database Service Provider Global Limited, „MongoDB Consultancy," 2020. [Online]. Available: https://www.dsp.co.uk/mongodb-consultancy. [Zugriff am 20 06 2020].

[123] Stampery, Inc., „Mongoaudit - MongoDB auditing and pentesting tool," 2020. [Online]. Available: https://github.com/stampery/mongoaudit. [Zugriff am 20 06 2020].

[124] Bundesamt für Sicherheit in der Informationstechnik, „BSI - Bausteine (Drafts) - Struktur der Modernisierung," 16 11 2017. [Online]. Available: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/IT-Grundschutz-Modernisierung/Struktur_Modernisierung.pdf. [Zugriff am 28 06 2020].

[125] K. M. Popp, „Commercial licensing for open source," in *Best Practices for commercial use of open source software*, Norderstedt, Germany, BoD-Books on Demand, 2020, p. 22ff.

[126] A. Naiakshina, A. Danilova, C. Tiefenau, M. Herzog, S. Dechand und M. Smith, „Why Do Developers Get Password Storage Wrong? A Qualitative Usability Study," in *ACM Conference on Computer and Communications Security*, Dallas, TX, USA, 2017.

[127] Oracle Corporation, „Datenbank-Sicherheit - Grundüberlegung," Oracle Corporation, Redwood Shores, CA, 2017.

[128] Oracle Corporation, „MySQL: Caching_sha2_password information," 22 01 2020. [Online]. Available: https://dev.mysql.com/doc/dev/mysql-server/latest/page_caching_sha2_authentication_exchanges.html. [Zugriff am 2020 04 05].

[129] The PostgreSQL Global Development Group, „PostgreSQL: Documentation: 12: PostgreSQL 12.2 Documentation," 13 02 2020. [Online]. Available: https://www.postgresql.org/docs/current/. [Zugriff am 05 04 2020].

[130] Internet Engineering Task Force (IETF), „RFC 5802 - Salted Challenge Response Authentication Mechanism (SCRAM) SASL and GSS-API Mechanisms," 07 2010. [Online]. Available: https://tools.ietf.org/html/rfc5802. [Zugriff am 05 04 2020].

[131] The PostgreSQL Global Development Group, „PostgreSQL: Documentation: 10: dblink," 2020. [Online]. Available: https://www.postgresql.org/docs/10/contrib-dblink-function.html. [Zugriff am 16 05 2020].

[132] Docker Inc., „Official Images on Docker Hub | Docker Documentation," 2019. [Online]. Available: https://docs.docker.com/docker-hub/official_images/. [Zugriff am 03 2020].

[133] MariaDB Corporation, MariaDB Foundation, „MariaDB Audit Plugin - MariaDB Knowledge Base," 2020. [Online]. Available: https://mariadb.com/kb/en/mariadb-audit-plugin/. [Zugriff am 05 2020].

[134] MongoDB Inc., „Auditing — MongoDB Manual," 2020. [Online]. Available: https://docs.mongodb.com/manual/core/auditing/. [Zugriff am 05 2020].

[135] Oracle Corporation, „MySQL :: MySQL 8.0 Reference Manual :: 6.3.2 Encrypted Connection TLS Protocols and Ciphers," 2020. [Online]. Available: https://dev.mysql.com/doc/refman/8.0/en/encrypted-connection-protocols-ciphers.html. [Zugriff am 21 05 2020].

[136] Bundesamt für Sicherheit in der Informationstechnik, „Zertifizierung nach ISO 27001 auf der Basis vonIT-Grundschutz," Bonn, 2019.

6 Tabellenverzeichnis

7 Verzeichnis der Abkürzungen

ANSI	American National Standards Institute
BaFin	Bundesanstalt für Finanzdienstleistungen
BSI	Bundesamt für Sicherheit in der Informationstechnik
BSON	Binary JSON
CRC	Zyklische Redundanzprüfung, englisch „cyclic redundancy check“, daher meist CRC
DB	Datenbank
DBMS	Datenbankmanagementsystem
DBS	Datenbanksystem
EoF	End of Life
GNU GPL	GNU General Public License
GPG	GNU Privacy Guard
IEC	International Electrotechnical Commission
INCITS	InterNational Committee for Information Technology Standards
ISACA	Information Systems Audit and Control Association
ISB	Informationssicherheitsbeauftragte/r
ISC2	International Information Systems Security Certification Consortium
ISMS	Information Security Management System

ISO	Internationale Organisation für Normung
ITIL	IT Infrastructure Library
JSON	JavaScript Object Notation
MaRisk	Mindestanforderungen an das Risikomanagement
MD5	Message-Digest Algorithm 5
NIST	National Institute of Standards and Technology
NoSQL	Not only SQL
PGP	Pretty Good Privacy
PIT(R)	Point-In-Time(-Recovery)
RBAC	Role Based Access Control
SQL	Structured Query Language
SSPL	Server Side Public License
TAR	Tape ARchiver
UmRa	Umsetzungsrahmenwerk der BSI

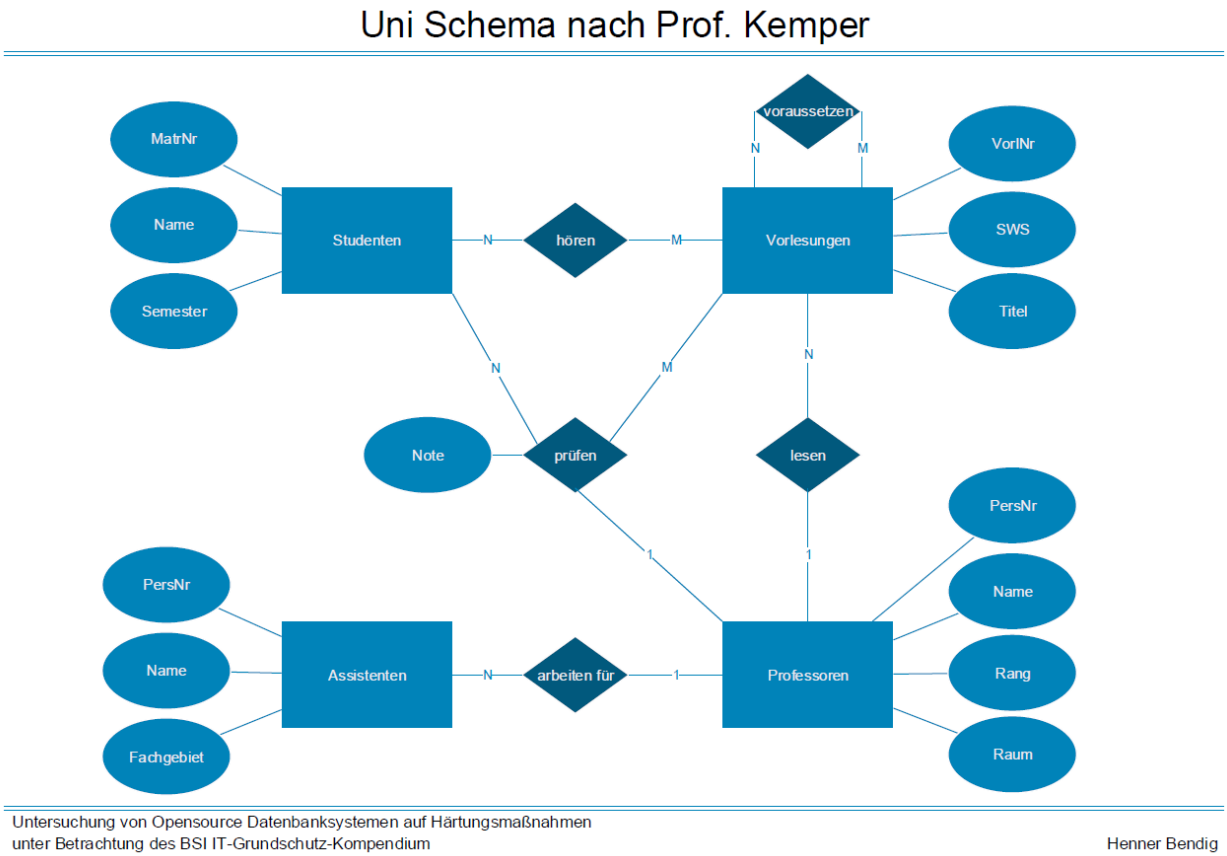
8 Thesen

1. Viele Sicherheitsmaßnahmen sind organisatorischer Natur und bedürfen die Mitarbeit aller Beteiligten, daher sollten alle in die Sicherheitsprozesse eingewiesen werden. Die Unterstützung muss von höchster Stelle kommen.
2. Die Umsetzung der organisatorischen Maßnahmen benötigt Erfahrung im IT-Sicherheitsbereich und in Management. Die Prozesse sollten in enger Abstimmung mit den Fachverantwortlichen und den Mitarbeitern erarbeitet werden, da diese Maßgeblich an der Durchführung beteiligt sind.
3. MySQL konnte ca. 49% und PostgreSQL ca. 44% der technischen Maßnahmen umsetzen. Hierzu mussten aber Drittanbieter Plugins und Software eingesetzt werden.
4. MongoDB konnte lediglich ca. 31% der Maßnahmen umsetzen. Das System ermöglicht keine Einbindung von Drittanbieter-Plugins.
5. Opensource Software wird stark von den zumeist dahinterstehenden Unternehmen getrieben. Kritische Funktionen, wie z.B. Sicherheitsmechanismen, bleiben oftmals einer kommerziellen Variante der Opensource Software vorbehalten.
6. Es sind viele Sicherheitsfunktionen vorhanden, die bspw. vom BSI gefordert werden. Diese müssen aber explizit aktiviert werden (Opt-In-Verfahren) und sind in Standardinstallationen daher nicht aktiv.
7. Das Bundesamt für Sicherheit in der Informationstechnik hat mit dem IT-Grundschutzkompendium einen umfassenden Maßnahmenkatalog zur Verbesserung der IT-Sicherheit veröffentlicht. Bei neueren Technologien müssen aber noch Bausteine nachgearbeitet werden.

9 Anlagen

A.1 Beispieldatenbank

A.1.1 Entity Relationship Diagram Uni Schema Kemper



Angefügte Dateien:

- UniSchema_ERD.pdf
- UniSchema_ERD.vsdx
- uni_mongodb.agz
- uni_mysql.sql
- uni_postgresql.sql

A.2 Inbetriebnahme

A.2.1 MySQL-Serverdaten

Port:	3306
Rootuser:	root
Passwort:	Dim1MT4D
Config-File:	/etc/my.cnf
Database:	uni

```
-- Terminalbefehl:
mysql -u root -p
Dim1MT4D
```

A.2.2 PostgreSQL-Serverdaten

Port:	5432
Rootuser:	postgres
Passwort:	Dim1MT4D
Config-File:	/var/lib/pgsql/12/data/postgresql.conf
Database:	uni

```
-- wechseln in den Nutzer "postgres"
su postgres
-- in die Postgres-Konsole wechseln
psql
```

A.2.3 MongoDB-Serverdaten

Port:	27017
Passwort:	Dim1MT4D
Rootuser:	admin
Config-File:	/etc/mongod.conf
Database:	uni

```
-- Server starten
mongod --config /etc/mongod.conf
-- Einloggen
mongo --tls --host 192.168.178.37 --tlsCertificateKeyFile /etc/ssl/
mongodb/client.pem --tlsCAFile /etc/ssl/mongodb/rootCA.pem -u admin
```

Angefügte Dateien:

- CentOS_Start.txt
- MySQL_Start.txt
- PostgreSQL_Start.txt
- MongoDB_Start.txt
- Keys / publicKey
- Keys / privateKey

A.3 Auswertung

Angefügte Datei:

- Vergleich.xlsx

A.4 Umsetzungsbeschreibungen

A.4.1 APP.4.3.A1

A.4.1.1 Vorbetrachtung

Dieses Dokument stellt eine beispielhafte Ergänzung einer Datenbank Sicherheitsrichtlinie zu einer allgemeinen IT-Sicherheitsrichtlinie eines Unternehmens (im folgendem „das Unternehmen“ genannt) dar und dient der Umsetzung eines geregelten Betriebes von Datenbanksystemen in diesem Unternehmen.

Es werden bewusst einige Punkte mit eingebracht, die in einer allgemeinen Richtlinie enthalten sein sollten, die aber von besonderer Bedeutung in einem Datenbanksystem sind.

Dieses Dokument ist im Rahmen der Master-Thesis „Untersuchung von Opensource Datenbanksystemen auf Härtungsmaßnahmen unter Betrachtung des IT-Grundschutz-Kompendium“ an der Hochschule Wismar von Henner Bendig entstanden.

A.4.1.2 Datenschutz Sicherheitsrichtlinie

Dieses Dokument beschreibt die Einführung einer speziellen Sicherheitsrichtlinie für Datenbanksysteme. Es wird eingeführt, wieso Datenbanken im Unternehmen eine besondere Schutzanforderung genießen und von welcher Bedeutung die für den Geschäftserfolgt sind. [127]

Des Weiteren werden konkrete Maßnahmen beschrieben, die von dem Betriebsteam der Systeme einzuhalten sind und die alle Nutzer der Systeme betreffen.

Diese Richtlinie wird verpflichtend für alle Mitarbeiter des Unternehmens eingeführt.

A.4.1.3 Stellenwert der Informationssicherheit für Datenbanksysteme

Eine funktionsfähige Informationstechnik und ein sicherheitsbewusster Umgang mit ihr sind wesentliche Voraussetzungen für die Einhaltung der IT-Sicherheitsziele Verfügbarkeit, Integrität und Vertraulichkeit von Informationen. Datenbanksysteme sind in der heutigen Zeit ein zentraler Bestandteil aller IT-Systeme und Anwendungen. In Ihnen lagern alle Informationen, die für ein erfolgreiches Unternehmertum notwendig sind. Die Größe und Komplexität von Datenbanken steigen kontinuierlich. Dementgegen sind Datenbanken sind mit ca. 15% auf Platz 4 der häufigsten Ziele von Cyber Attacken im Jahr 2019. [3] Daher tangiert der Schutz von Datenbanksystemen jeden Geschäftsbereich des Unternehmens.

Um einen standardisierten Schutz für alle unternehmensweiten Datenbankensysteme zu gewährleisten, hat sich die Geschäftsführung des Unternehmens dazu entschlossen, diese Sicherheitsrichtlinie speziell für Datenbanken zu erlassen. Ziel dieser Richtlinie ist es, den Sicherheitsstandard sukzessiv anzuheben. Hierzu orientiert sich das Unternehmen an dem IT-Grundschutz des Bundesamtes für Informationssicherheit (BSI).

In Abwägung der Gefährdungen, der Werte der zu schützenden Güter sowie des vertretbaren Aufwands an Personal und Finanzmitteln für IT-Sicherheit, hat die Unternehmensleitung bestimmt, dass ein **hohes IT-Sicherheitsniveau** angestrebt werden soll.

A.4.1.4 Einführung der Sicherheitsrichtlinie

Die Geschäftsführung des Unternehmens ist sich darüber im Klaren, dass ein solcher Sicherheitsstandard nicht sofort umzusetzen ist. So wurde in Abstimmung mit allen Bereichsleiter ein Stufenkonzept eingeführt, welches unter Betrachtung der einzusetzenden Mittel, die finanzielle Belastung und den Einfluss auf den Geschäftsalltag im angemessenen Verhältnis steht.

Orientierend am IT Grundschutz [6] sieht das Stufenkonzept wie folgt aus:

Stufe 1 Basis Absicherung: „*Die Basis-Absicherung ermöglicht es, als Einstieg in den IT-Grundschutz zunächst eine breite, grundlegende Erst-Absicherung über alle Geschäftsprozesse bzw. Fachverfahren einer Institution vorzunehmen*.“

Stufe 2 Standard Absicherung: „*Mit der Standard-Absicherung kann ein kompletter Sicherheitsprozess implementiert werden. Diese Absicherung entspricht weiterhin dem BSI-Standard 100-2 und ist kompatibel zur ISO 27001-Zertifizierung.*“

Stufe 3 Erhöhte Absicherung: *„Die erhöhte Absicherung beschreibt Maßnahmen, die über das dem Stand der Technik entsprechende Schutzniveau hinausgehen und bei erhöhtem Schutzbedarf in Betracht gezogen werden sollten.“*

Jede Stufe wird mittels eines Jahrs umgesetzt. Somit ergibt sich eine Laufzeit von drei Jahren für die Einführung.

Der ISB erarbeitet ein geeigneten Schulungsplan, zugeschnittenen auf den entsprechenden Tätigkeitsbereich der Mitarbeiter.

Diese Richtlinie wird regelmäßig, mindestens einmal jährlich, überprüft.

A.4.1.5 Verantwortlichkeiten und Durchsetzung

Zur Einführung, Umsetzung und Einbehaltung dieser Richtlinie wird ein Informationssicherheitsbeauftragter (ISB) ernannt, welche die volle Unterstützung der Geschäftsführung genießt. Der ISB steht im regelmäßigen Austausch mit der Geschäftsführung und allen Bereichsleitern und informiert diese über den Fortschritt und Hindernisse in der Einführung und Durchsetzung.

Beabsichtigte oder grob fahrlässige Handlungen, die die Sicherheit von Daten, Informationen, Anwendungen, IT-Systemen oder des Netzes gefährden, werden als Verstöße verfolgt. Dazu gehören beispielsweise:

- der Missbrauch von Daten, der finanziellen Verlust verursachen kann,
- der unberechtigte Zugriff auf Informationen bzw. ihre Änderung und unbefugte Übermittlung,
- die illegale Nutzung von Informationen aus dem Unternehmen,
- die Gefährdung der IT-Sicherheit der Mitarbeiter, Geschäftspartner und des Unternehmens und
- die Schädigung des Rufes des Unternehmens.

Bewusste Zuwiderhandlungen gegen diese Sicherheitsleitlinie werden bestraft – gegebenenfalls disziplinarisch, arbeitsrechtlich oder mit zivil- und strafrechtlichen Verfahren, in denen auch Haftungsansprüche und Regressforderungen erhoben werden können.

A.4.1.6 Maßnahmen

Die Maßnahmen orientieren sich an den Vorgaben des BSI IT-Grundschutzes. Prinzipiell sind die Maßnahmen des jeweilige gleichnamige Schutzlevels des IT-Grundschutz Kompendium für Datenbanksysteme, Kapitel APP.4.3, einzuhalten. Das Kompendium wird dieser Richtlinie angehängt und gilt als Bestandteil.

Maßnahmen können nach Bedarf vom ISB hinzugefügt, verschärft oder gemildert werden. Dies Bedarf die Abstimmung mit der Geschäftsführung und den betroffenen Bereichsleitern.

A.4.1.7 Anlage

IT-Grundschutz Kompendium, Bundesamt für Sicherheit in der Informationstechnik

Angefügte Datei:

- Sicherheitsrichtlinie.docx

A.4.2 APP.4.3.A2

A.4.2.1 MySQL:

A.4.2.1.1 Integritätsprüfung der MySQL-Setupdateien

Folgend der Anleitung[30], die von Oracle zur Überprüfung bereitgestellt wird, auf den Systemen unterschiedliche Möglichkeiten, die Integrität der Installationsdateien zu prüfen:

[30] https://dev.mysql.com/doc/mysql-installation-excerpt/5.7/en/verifying-package-integrity.html, 25.03.2020

A.4.2.1.1.1 Linux

Öffentliche Signatur downloaden und importieren

```
gpg --recv-keys 5072E1F5
```

Installationsdateien downloaden

```
curl -OL https://dev.mysql.com/downloads/gpg/?file=mysql-8.0.20-1.el8.x86_64.rpm-bundle.tar
```

Signatur downloaden

https://dev.mysql.com/downloads/gpg/?file=mysql-8.0.20-1.el8.x86_64.rpm-bundle.tar&p=23

Dateien verifizieren

```
gpg --verify mysql-8.0.20-1.el8.x86_64.rpm-bundle.tar.asc
```

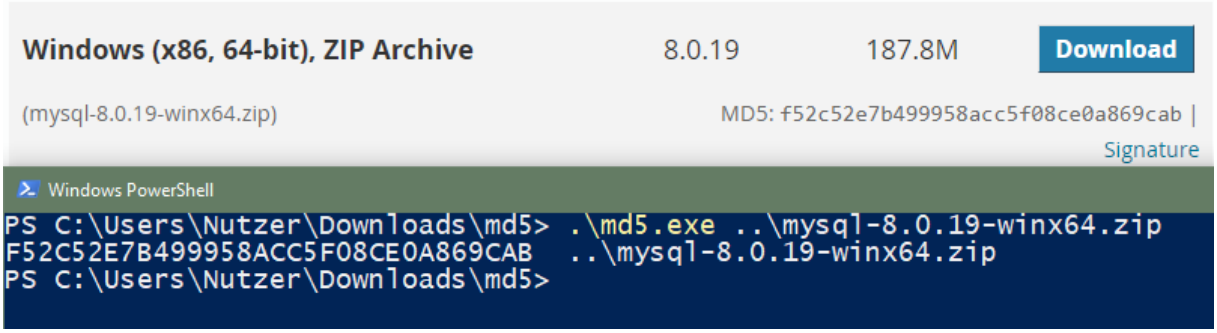
Die Ausgabe auf dem Testsystem lautet, die Signatur ist korrekt, aber der Absender nicht bekannt. Das Liegt daran, dass die Signatur nicht in der lokalen TrustDB aufgenommen wurde und daher noch nicht bekannt ist.

```
[root@centos mysql]# gpg --recv-keys 5072E1F5
gpg: key 8C718D3B5072E1F5: 3 duplicate signatures removed
gpg: key 8C718D3B5072E1F5: 102 Beglaubigungen wegen fehlender Schlüssel nicht geprüft
gpg: Schlüssel 8C718D3B5072E1F5: "MySQL Release Engineering <mysql-build@oss.oracle.c
om>" 29 neue Signaturen
gpg: keine ultimativ vertrauenswürdigen Schlüssel gefunden
gpg: Anzahl insgesamt bearbeiteter Schlüssel: 1
gpg:                       neue Signaturen: 29
[root@centos mysql]# rpm --import mysql_pubkey.asc
Fehler: mysql_pubkey.asc: Schlüssel 1 ist kein gesicherter öffentlicher Schlüssel.
[root@centos mysql]# curl -OL https://dev.mysql.com/get/Downloads/MySQL-8.0/mysql-8.0
.20-1.el8.x86_64.rpm-bundle.tar
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100  650M  100  650M    0     0  3986k      0  0:02:47  0:02:47 --:--:-- 4078k^[[B
[root@centos mysql]# curl -OL https://dev.mysql.com/downloads/gpg/?file=mysql-8.0.20-
1.el8.x86_64.rpm-bundle.tar&p=23
[1] 23396
[root@centos mysql]#   % Total    % Received % Xferd  Average Speed   Time    Time
  Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100 15150    0 15150    0     0  11845      0 --:--:--  0:00:01 --:--:-- 11845
^C
[1]+  Fertig                  curl -OL https://dev.mysql.com/downloads/gpg/?file=mysq
l-8.0.20-1.el8.x86_64.rpm-bundle.tar
[root@centos mysql]# gpg --verify mysql-8.0.20-1.el8.x86_64.rpm-bundle.tar.asc
gpg: die unterzeichneten Daten sind wohl in 'mysql-8.0.20-1.el8.x86_64.rpm-bundl
e.tar'
gpg: Signatur vom Di 28 Apr 2020 10:36:14 CEST
gpg:                mittels DSA-Schlüssel 8C718D3B5072E1F5
gpg: Korrekte Signatur von "MySQL Release Engineering <mysql-build@oss.oracle.co
m>" [unbekannt]
gpg: WARNUNG: Dieser Schlüssel trägt keine vertrauenswürdige Signatur!
gpg:          Es gibt keinen Hinweis, daß die Signatur wirklich dem vorgeblichen
 Besitzer gehört.
Haupt-Fingerabdruck  = A4A9 4068 76FC BD3C 4567  70C8 8C71 8D3B 5072 E1F5
[root@centos mysql]#
```

A.4.2.1.1.2 Windows

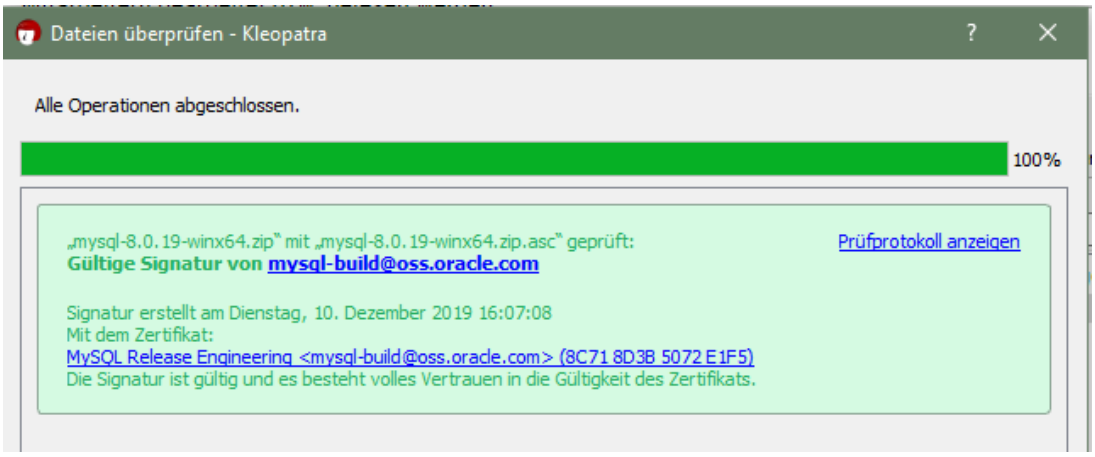
MD5

Auf einem Windows System ist standardmäßig kein MD5-Checksummen-Tool vorhanden. MySQL empfiehlt die Implementierung „MD5: Command Line Message Digest Utility“, welche samt Sourcecode von auf der Webseite Formilab.ch bereitsteht. [31]



GnuPG Signatur

Das von Oracle empfohlene Tool zur Prüfung der Signatur nennt sich „Gpg4Win“ und kommt mit einer GUI-Implementierung „Kleopatra“. Erhältlich ist dieses Opensource-Tool auf der Hersteller-Webseite [32]



Angefügte Dateien:

- MySQL_Setupintegritaet.docx
- Windows

---

[31] http://www.fourmilab.ch/md5/, 25.03.2020

[32] https://www.gpg4win.org/, 25.03.2020

    - mysql-8.0.19-winx64.zip
    - mysql-8.0.19-winx64.zip.asc
    - md5.zip
    - gpg4win-3.1.11.exe

- Linux
    - mysql_pubkey.asc
    - mysql-8.0.20-1.el8.x86_64.rpm-bundle.tar.asc
    - mysql-8.0.20-1.el8.x86_64.rpm-bundle.tar

A.4.2.2 MongoDB

A.4.2.2.1 Integritätsprüfung der MongoDB-Setupdateien

Folgend der Anleitung[33], die von MongoDB Inc. zur Überprüfung bereitgestellt wird, gibt es auf den untersuchten Systemen unterschiedliche Möglichkeiten, die Integrität der Installationsdateien zu prüfen.

A.4.2.2.1.1 Linux

PGP/GPG-Prüfung

Herunterladen der Installationspakete:

```
curl -LO https://fastdl.mongodb.org/osx/mongodb-macos-x86_64-4.2.8.
tgz
```

Herunterladen der öffentlichen Signatur:

```
curl -LO https://fastdl.mongodb.org/osx/mongodb-macos-x86_64-4.2.8.
tgz.sig
```

Downloaden und Importieren des öffentlichen Schlüssels:

```
curl -LO https://www.mongodb.org/static/pgp/server-4.2.asc
gpg --import server-4.2.asc
```

Verifikation der Signatur

```
gpg --verify mongodb-macos-x86_64-4.2.8.tgz.sig mongodb-macos-x86_6
4-4.2.8.tgz
```

[33] https://docs.mongodb.com/manual/tutorial/verify-mongodb-packages/ 28.03.2020

Die Ausgabe auf dem Testsystem lautet, die Signatur ist korrekt, aber der Absender nicht bekannt. Das Liegt daran, dass die Signatur nicht in der lokalen TrustDB aufgenommen wurde und daher noch nicht bekannt ist.

```
[root@centos mongodb]# curl -LO https://fastdl.mongodb.org/osx/mongodb-macos-x86_64-4.2.8.t
gz
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  113M  100  113M    0     0  3778k      0  0:00:30  0:00:30 --:--:-- 4032k
[root@centos mongodb]# curl -LO https://fastdl.mongodb.org/osx/mongodb-macos-x86_64-4.2.8.t
gz.sig
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   801  100   801    0     0   1592      0 --:--:-- --:--:-- --:--:--  1592
[root@centos mongodb]# curl -LO https://www.mongodb.org/static/pgp/server-4.2.asc
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1691  100  1691    0     0   3277      0 --:--:-- --:--:-- --:--:--  3270
[root@centos mongodb]# gpg --import server-4.2.asc
gpg: Verzeichnis `/root/.gnupg' erzeugt
gpg: Die "Keybox" `/root/.gnupg/pubring.kbx' wurde erstellt
gpg: /root/.gnupg/trustdb.gpg: trust-db erzeugt
gpg: Schlüssel 4B7C549A058F8B6B: Öffentlicher Schlüssel "MongoDB 4.2 Release Signing Key <p
ackaging@mongodb.com>" importiert
gpg: Anzahl insgesamt bearbeiteter Schlüssel: 1
gpg:                   importiert: 1
[root@centos mongodb]# gpg --verify mongodb-macos-x86_64-4.2.8.tgz.sig mongodb-macos-x86_64
-4.2.8.tgz
gpg: Signatur vom Do 11 Jun 2020 18:52:56 CEST
gpg:                mittels RSA-Schlüssel 4B7C549A058F8B6B
gpg: Korrekte Signatur von "MongoDB 4.2 Release Signing Key <packaging@mongodb.com>" [unbek
annt]
gpg: WARNUNG: Dieser Schlüssel trägt keine vertrauenswürdige Signatur!
gpg:          Es gibt keinen Hinweis, daß die Signatur wirklich dem vorgeblichen Besitzer g
ehört.
Haupt-Fingerabdruck  = E162 F504 A20C DF15 827F  718D 4B7C 549A 058F 8B6B
[root@centos mongodb]#
```

A.4.2.2.1.2 Windows

SHA256

MongoDB Inc. stellt für Downloadpakete, die direkt beim Hersteller geladen werden, einen SHA256-Signatur-Key zur Verfügung. Zusätzlich wird ein Windows Powershell-Skript veröffentlicht, dass den Vergleich des Keys mit dem des Installationspaketes ermöglicht und eine Eindeutige Ausgabe „True“ oder „False“ zurückgibt.

Powershell-Script

```
1  $sigHash = (Get-Content mongodb-win32-x86_64-2012plus-4.2.5-signed.msi.sha256| Out-String).SubString(0,64).ToUpper();
2  $fileHash = (Get-FileHash mongodb-win32-x86_64-2012plus-4.2.5-signed.msi).Hash.Trim(); `
3  echo $sigHash; echo $fileHash;
4  $sigHash -eq $fileHash
```

Ausgabe

```
Windows PowerShell
PS C:\Users\Nutzer\MagentaCLOUD\Thesis\Umsetzung\APP.4.3.A2\MongoDB> .\CheckSum.ps1
CC46897AD51F617D1946AD1B8F2EB50D215CAB1B4B2FE21FD55F6022C862C333
CC46897AD51F617D1946AD1B8F2EB50D215CAB1B4B2FE21FD55F6022C862C333
True
```

Angefügte Dateien:

- MongoDB_Setupintegritaet.docx
- Windows
  - CheckSum.ps1
  - mongodb-win32-x86_64-2012plus-4.2.5-signed.msi
  - mongodb-win32-x86_64-2012plus-4.2.5-signed.msi.sha256
- Linux
  - mongodb-macos-x86_64-4.2.8.tgz
  - mongodb-macos-x86_64-4.2.8.tgz.sig
  - server-4.2.asc

A.4.3 APP.4.3.A3

A.4.3.1 Einleitung

Diese Checkliste ist eine Sammlung von Konfigurationsvorschlägen zur Gewährleistung einer Basis-Härtung von Datenbanksystemen, insbesondere der Systeme „MySQL Community 8.0.18“, „PostgreSQL 12.2“ und „MongoDB Community Server 3.6.16“.

Die Vorschläge basieren auf den Arbeiten der Deutschen Telekom AG („Privacy and Security Assessment“, PSAsho), dem Center for Information Security („CIS Benchmarks“) und Empfehlungen der jeweiligen Systemhersteller.

Diese Liste ist im Rahmen der Master-Thesis „Untersuchung von Opensource Datenbanksystemen auf Härtungsmaßnahmen unter Betrachtung des IT-Grundschutz-Kompendium“ an der Hochschule Wismar von Henner Bendig entstanden.

A.4.3.2 Generelle Härtung

PSA

- ☐ Nicht benötigte (Default-)Datenbanken auf dem Datenbanksystem müssen gelöscht werden.
- ☐ Nicht benötigte (Default-)User und (Default-)Rollen müssen gelöscht werden.
- ☐ Auf einer Betriebssysteminstanz (mit physikalischer oder virtualisierter Hardware) dürfen nur Datenbankinstanzen betrieben werden, die
  - den gleichen Schutzbedarf haben,
  - unter einer einzigen Kundenhoheit stehen und
  - administrativ vom gleichen Personenkreis betrieben werden.
- ☐ (Default-)Passwörter auf Datenbanksystemen müssen geändert werden.
- ☐ Alle Datenbanksysteme müssen mit minimalen Rechten (least privilege principle) auf Betriebssystemebene aufgesetzt werden.

- ☐ Ein Datenbankdienst darf nicht mit Root-Rechten oder auf anderen administrativen Rechten des Betriebssystems laufen.
- ☐ Nicht benötigte, erweiterte SQL-Funktionen (z.B. T-SQL, PL/SQL, SQL PL, extended stored procedures) und/oder -Pakete aus den Datenbanksystemen müssen gelöscht werden.
- ☐ Datenbankfunktionen, die den Zugriff auf Dateien des Betriebssystems erlauben, müssen gelöscht oder deaktiviert werden.
- ☐ Datenbankfunktionen, die den Zugriff auf andere Netzdienste (z. B. SMTP, HTTP, SNMP, FTP etc.) erlauben, müssen gelöscht oder deaktiviert werden.
- ☐ Datenbankfunktionen, welche ein Zugriff auf Betriebssystemebene und/oder Netzwerkdienste ermöglichen, muss das Zugriffsrecht der Rollen/Gruppen „Public“ und/oder „Everyone“ entzogen werden.
- ☐ Die Betriebssystemrechte für die Dateien und Verzeichnisse der Datenbank (Programm-, Control-, Trace- und Logdateien) müssen exklusiv dem Betriebssystemkonto des Datenbanksystems zugeordnet werden.

A.4.3.3 MySQL Härtung

PSA

- ☐ Die Default Datenbanken wie „test“ müssen gelöscht werden.
- ☐ Es muss sichergestellt sein, dass keine Benutzerkonten ohne Benutzername (Anonymous Accounts) vorhanden sind.
- ☐ Auf einer Betriebssysteminstanz (Hardwareplattform oder Virtualisierungsgast) darf jeweils nur eine Instanz des Datenbanksystems installiert sein.
- ☐ (Default-) Passwörter von (SUPER-) User Accounts müssen geändert werden.
- ☐ Der Datenbank Dämon muss mit der Option --safe-user-create gestartet werden.
- ☐ Die Startoption old-passwords darf nicht verwendet werden.
- ☐ Die Startoption secure-auth muss verwendet werden.
- ☐ Die Variable MYSQL_PWD darf nicht bei der Passwort-Speicherung und/oder in Scripts verwendet werden.
- ☐ Falls Linux verwendet wird, muss der Datenbank Dämon mit einem dedizierten, nicht-administrativen Unix/Linux-Konto laufen, das sich auf dem System nur für die Maria / MySQL-DB verwendet wird.
- ☐ Falls Windows verwendet wird, muss die MySQL/Maria-DB mit eingeschränkten Privilegien unter einem Network-Service-Account laufen.
- ☐ Sofern nicht benötigt, muss die Symlink-Funktion deaktiviert werden.
- ☐ Das FILE-Privileg muss ausschließlich auf den Super User Account beschränkt sein.
- ☐ Die Schreib- und Leserechte für das FILE-Privileg müssen auf ein definiertes Verzeichnis begrenzt werden.
- ☐ Die Datenbank muss mit der Option --local-infile=0 gestartet werden
- ☐ Es muss sichergestellt sein, dass nur das MySQL/Maria-Benutzerkonto Lese- und Schreibrechte zu allen MySQL/Maria-Daten und deren Unterverzeichnissen sowie zu Logfiles und Datenbankdateien erhält.
- ☐ Es muss sichergestellt sein, dass nur die MySQL-Benutzer und berechtigte Administratoren Lese- und Schreibzugriff auf Query und Binary Log-Dateien besitzen.
- ☐ Falls Linux verwendet wird, muss sichergestellt sein, dass nur die MySQL/Maria-Benutzer Lese- und Schreibrechte auf die Datei “mysql_history“ besitzen.

- ☐ Es muss sichergestellt sein, dass nur administrative Konten Zugriff auf die “User-Table“ der MySQL/Maria-DB haben.
- ☐ Es muss sichergestellt sein, dass nur berechtigte Konten Zugriff auf die Datenbanktabellen (database table) der MySQL/Maria-DB haben.
- ☐ Das Berechtigungssystem der MySQL/Maria-DB darf nicht deaktiviert werden.
- ☐ Die GRANT-Option “WITH GRANT“ darf nicht eingesetzt werden.
- ☐ Vorwiegend administrative Privilegien müssen ausschließlich für DB-Administratorkonten vergeben werden (Least Privilege-Prinzip).
- ☐ Die “Privilege Tables“ muss nach jedem Upgrade überprüft werden.
- ☐ Die Startoption --allow-suspicious-udfs muss deaktiviert sein.
- ☐ Es muss sichergestellt sein, dass sich bei der Benutzer-Authentifizierung nicht mit Wildcards (“%“) im Hostnamen arbeiten lässt.
- ☐ Es muss sichergestellt sein, dass ein Login des Super Users nur an Localhost erfolgen kann.
- ☐ Die pro Benutzerkonto verfügbaren Ressourcen müssen einschränkt werden.

CIS

Operating System Level Configuration

- ☐ Linux: Place Databases on Non---System Partitions
- ☐ Linux: Use Dedicated Least Privileged Account for MySQL Daemon/Service
- ☐ Linux: Disable MySQL Command History
- ☐ Linux: Verify that the MYSQL_PWD Environment Variables is not in Use
- ☐ Linux: Disable interactive Login
- ☐ Linux: Verify that ‚MYSQL_PWD‘ is not set in users‘ profiles

Planning

- ☐ Dedicate Machine Running MySQL
- ☐ Do Not Specify Passwords in Command Line
- ☐ Do Not Reuse Usernames
- ☐ Do Not Use Default or Non-MySQL-specific Cryptographic Keys
- ☐ Set a Password Expiry Policy for Specific Users

General

- ☐ Ensure Latest Security Patches Are Applied
- ☐ Ensure the 'test' Database Is Not Installed
- ☐ Ensure 'allow-suspicious-udfs' Is Set to 'FALSE'
- ☐ Ensure ‚local_inffile‘ is disabled
- ☐ Ensure ‚mysqld‘ is not started with ‚—skip-grant-tables‘
- ☐ Ensure ‚—skip-symbolic-links‘ is enabled
- ☐ Ensure the ‚daemon_memcached‘ Plugin is disabled
- ☐ Ensure ‚secure_file_priv‘ is not empty
- ☐ Ensure ‚sql_mode‘ contains ‚STRICT_ALL_TABLES‘

MySQL Permissions

- ☐ Ensure only administrative User have full database access
- ☐ Ensure ‚file_priv‘ is not set to ‚Y‘ for non-administrative users
- ☐ Ensure ‚process_priv‘ is not set to ‚Y‘ for non-administrative users
- ☐ Ensure ‚super_priv‘ is not set to ‚Y‘ for non-administrative users
- ☐ Ensure ‚shutdown_priv‘ is not set to ‚Y‘ for non-administrative users

- ☐ Ensure ‚create_user_priv‘ is not set to ‚Y‘ for non-administrative users
- ☐ Ensure ‚grant_priv‘ is not set to ‚Y‘ for non-administrative users
- ☐ Ensure ‚repl_slave_priv‘ is not set to ‚Y‘ for non-slave users
- ☐ Ensure DML/DDL Grants are limited to specific databases and users

Auditing and Logging

- ☐ Ensure „log_error“ is not empty
- ☐ Linux: Ensure Log-Files are stored on a non-system partition
- ☐ Ensure ‚log_error_verbosity‘ is set to ‚1‘
- ☐ Ensure Audit Logging is enabled
- ☐ Ensure ‚log_raw‘ is set to ‚OFF‘

Authentication

- ☐ Linux: Ensure passwords are not stored in the Global Configuration
- ☐ Ensure ‚sql_mode‘ containts ‚NO_AUTO_CREATE_USER‘
- ☐ Ensure passwords are set for all MySQL-Accounts
- ☐ Ensure ‚default_password_lifetime‘ is less than or equal to ‚90‘
- ☐ Ensure password complexity is in place
- ☐ Ensure no users have wildcard hostnames
- ☐ Ensure No anonymous accounts exist

Network

- ☐ Ensure ‚have_ssl‘ is set to ‚YES‘
- ☐ Ensure ‚ssl_type‘ is set to ‚ANY‘, ‚X509‘ or ‚SPECIFIED‘ for all remote users

Replication

- ☐ Ensure replication traffic is secured
- ☐ Ensure 'MASTER_SSL_VERIFY_SERVER_CERT' Is Set to 'YES' or '1'
- ☐ Ensure ‚master_info_repository‘ is set to ‚TABLE‘
- ☐ Ensure ‚super_priv‘ is not set to ‚Y‘ for replication users
- ☐ Ensure no replication users have wildcard hostnames

Hersteller

- ☐ Do not ever give anyone (except MySQL root accounts) access to the user table in the mysql system database
- ☐ Do not grant more privileges than necessary. Never grant privileges to all hosts.
- ☐ Try mysql -u root. If you are able to connect successfully to the server without being asked for a password, anyone can connect to your MySQL server as the MySQL root user with full privileges
- ☐ Use the SHOW GRANTS statement to check which accounts have access to what. Then use the REVOKE statement to remove those privileges that are not necessary.
- ☐ Do not store cleartext passwords in your database. If your computer becomes compromised, the intruder can take the full list of passwords and use them. Instead, use SHA2() or some other oneway hashing function and store the hash value.
- ☐ To prevent password recovery using rainbow tables, do not use these functions on a plain password; instead, choose some string to be used as a salt, and use hash(hash(password)+salt) values.
- ☐ Do not choose passwords from dictionaries. Special programs exist to break passwords.

- ☐ Invest in a firewall. This protects you from at least 50% of all types of exploits in any software. Put MySQL behind the firewall or in a demilitarized zone (DMZ).
- ☐ MySQL uses port 3306 by default. This port should not be accessible from untrusted hosts
- ☐ Applications that access MySQL should not trust any data entered by users, and should be written using proper defensive programming techniques
- ☐ Do not transmit plain (unencrypted) data over the Internet. This information is accessible to everyone who has the time and ability to intercept it and use it for their own purposes. Instead, use an encrypted protocol such as SSL or SSH. MySQL supports internal SSL connections
- ☐ Account passwords can be expired
- ☐ Use the ‚validate password'-plugin
- ☐ Protect the plugin directory („plugin_dir" system variable)
- ☐ Do not use „insert" or „update" for settings passwords
- ☐ Protect the log-folder (audit log files)
- ☐ Never use the Server-Start-Option „—log-raw" in production
- ☐ Protect database backups
- ☐ Require all MySQL Accounts to have a password
- ☐ Make sure that the only user account with read or write privileges in the database directories ist he account, that is used for running mysqld
- ☐ Never run the MySQL Server the the root user
- ☐ Do not grant the FILE privilege to nonadministrative users
- ☐ Encrypt binary log files and relay log files.
- ☐ Do not grant the PROCESS or SUPER privilege to nonadministrative users
- ☐ Do not permit the use of symlinks to tables.
- ☐ Stored programs and views should be written using the security guidelines discussed in Stored Object Access Control.
- ☐ Use IP Addresses rather than host names
- ☐ Restrict the number of connecetions per user
- ☐ Disable or restrict files permitted for Local Data Loading
- ☐ Enable the strict SQL mode to tell the server tob e more restrictive
- ☐ Arrange the server to start and stop automatically when the system starts and stops

A.4.3.4 PostgreSQL Härtung

PSA

- ☐ Auf einer Betriebssysteminstanz (Hardwareplattform oder Virtualisierungsgast) darf jeweils nur eine Instanz des Datenbanksystems installiert sein.
- ☐ Windows: Default Passwörter müssen geändert werden.
- ☐ Passwörter müssen gehashed gespeichert und übertragen werden.
- ☐ Die Authentisierungsmethode "peer authentication" darf nicht für „nicht-administrative" bzw. "nicht technische" Accounts verwendet werden,
- ☐ Nur der Database Administrator (DBA) muss SUPERUSER, CREATEROLE oder CREATEDB Rechte besitzen.
- ☐ Es muss sichergestellt sein, dass nur der DBA Rechte an der Tabelle pg_catalog.pg_authid besitzt.
- ☐ Das Privileg rolcanlogin darf nur für bestimmte (d.h. nicht "all") Benutzer / Rollen gesetzt sein, die einen Eintrag in der pg_hba.conf haben.

- ☐ Es muss für jedweden Benutzer eine Authentisierungsmethode jenseits "trust" verwendet werden.
- ☐ Rechte auf Objekte dürfen nicht an PUBLIC vergeben werden
- ☐ Privilegien auf Objekte müssen ohne "with grant" Option vergeben werden.
- ☐ SQL Funktionen mit SECURITY DEFINER müssen sehr restriktiv verwendet werden.
- ☐ Falls nicht anders benötigt, dürfen nur c und internal als non-trusted procedural languages verwendet werden.
- ☐ Nicht benötigte prozedurale Sprachen müssen gelöscht werden.
- ☐ Das PostgreSQL "data_directory" Verzeichnis und Konfigurationsdateien müssen exclusiv dem Betriebssystem Account der Datenbank zugeweisen werden.
  Anderen Systemuser müssen die Zugriffsrechte auf diese Daten entzogen werden.

CIS

Installation

- ☐ Ensure installation binaries and packages are obtained from authorized repositories
- ☐ Ensure Installation of authorized Cummunity Packages
- ☐ Linux: Ensure systemd Service Files are enabled
- ☐ Ensure Data Cluster initialized successfully

Directory and File Permissions

- ☐ Linux: Ensure the file permissions mask is correct (077) for postgres-user
- ☐ Linux: Ensure the PostgreSQL „pg_wheel“ group membership is correct (only for superuser)

Logging

- ☐ Ensure the log destinations are set correctly
- ☐ Ensure the logging collector is enabled
- ☐ Ensure the log file destination directory is set correctly
- ☐ Ensure the filename pattern for log files is set correctly
- ☐ Ensure the log file permissions are set correctly
- ☐ Ensure 'log_truncate_on_rotation' is enabled
- ☐ Ensure the maximum log file lifetime is set correctly
- ☐ Ensure the maximum log file size is set correctly
- ☐ Ensure the correct syslog facility is selected
- ☐ Ensure the program name for PostgreSQL syslog messages is correct
- ☐ Ensure the correct messages are written to the server log
- ☐ Ensure the correct SQL statements generating errors are recorded
- ☐ Ensure 'debug_print_parse' is disabled
- ☐ Ensure 'debug_print_rewritten' is disabled
- ☐ Ensure 'debug_print_plan' is disabled
- ☐ Ensure 'debug_pretty_print' is enabled
- ☐ Ensure 'log_connections' is enabled
- ☐ Ensure 'log_disconnections' is enabled
- ☐ Ensure 'log_error_verbosity' is set correctly
- ☐ Ensure 'log_hostname' is set correctly (
- ☐ Ensure 'log_line_prefix' is set correctly

- ☐ Ensure 'log_statement' is set correctly
- ☐ Ensure 'log_timezone' is set correctly
- ☐ Ensure the PostgreSQL Audit Extension (pgAudit) is enabled

User Access and Authorization

- ☐ Ensure sudo is configured correctly
- ☐ Ensure excessive administrative privileges are revoked
- ☐ Ensure excessive function privileges are revoked
- ☐ Ensure excessive DML privileges are revoked
- ☐ Use pg_permission extension to audit object permissions (
- ☐ Ensure Row Level Security (RLS) is configured correctly
- ☐ Ensure the set_user extension is installed
- ☐ Make use of default roles

Postgres Settings

- ☐ Ensure 'Attack Vectors' Runtime Parameters are Configured
- ☐ Ensure 'backend' runtime parameters are configured correctly
- ☐ Ensure 'Postmaster' Runtime Parameters are Configured (
- ☐ Ensure 'SIGHUP' Runtime Parameters are Configured
- ☐ Ensure 'Superuser' Runtime Parameters are Configured
- ☐ Ensure 'User' Runtime Parameters are Configured (
- ☐ Ensure FIPS 140-2 OpenSSL Cryptography Is Used
- ☐ Ensure SSL is enabled and configured correctly
- ☐ Ensure the pgcrypto extension is installed and configured correctly

Replication

- ☐ Ensure a replication-only user is created and used for streaming replication
- ☐ Ensure base backups are configured and functional
- ☐ WAL archiving is configured and functional
- ☐ Ensure streaming replication parameters are configured correctly

Special configuration

- ☐ Ensure PostgreSQL configuration files are outside the data cluster
- ☐ Ensure PostgreSQL subdirectory locations are outside the data cluster
- ☐ Ensure the backup and restore tool, 'pgBackRest', is installed and configured
- ☐ Ensure miscellaneous configuration settings are correct

A.4.3.5 MongoDB

CSI

Installation and Patching

- ☐ Ensure the appropriate MongoDB software version/patches are installed

Authentication

- ☐ Ensure Authentication is configured
- ☐ Ensure that MongoDB does not bypass authentication via the localhost exception
- ☐ Ensure authentication is enabled in the shared cluster

Access Control

- ☐ Ensure that Role-based access control (RBAC) is enabled and configured
- ☐ Ensure that MongoDB only listens for network connections on authorized interfaces
- ☐ Ensure that MongoDB is run using a Least Privileges, dedicated service account
- ☐ Ensure that each role for each MongoDB database is needed and grants only
- ☐ the necessary privileges
- ☐ Review User-Defined Roles
- ☐ Review Superuser/Admin Roles

Data Encryption

- ☐ Ensure Encryption of Data in Transit TLS/SSL (Transport Encryption)
- ☐ Ensure Federal Information Processing Standard (FIPS) is enabled
- ☐ Ensure Encryption of Data at Rest

Auditing

- ☐ Ensure that system activity is audited
- ☐ Ensure that audit filters are configured properly
- ☐ Ensure that logging captures as much information as possible
- ☐ Ensure that new entries are appended to the end of the log file

Hersteller

Pre-/Installation

- ☐ Enable Access Control and Enforce Authentication
- ☐ Configure Role-Based Access Control
- ☐ Encrypt Communication (TLS/SSL)
- ☐ Encrypt and Protect Data
- ☐ Limit Network Exposure
- ☐ Audit System Activity
- ☐ Run MongoDB with a Dedicated User
- ☐ Run MongoDB with Secure Configuration Options

Ongoing / Periodic

- ☐ Periodically apply patches to your machine and review guidelines.
- ☐ Review policy/procedure changes, especially changes to your network rules to prevent inadvertent MongoDB exposure to the Internet.
- ☐ Review MongoDB database users and periodically rotate them.

Angefügte Dateien:

- Checkliste.docx
- DTAG_PSA_3_16_Datenbanksysteme.pdf

A.4.3.6 MySQL

A.4.3.6.1 Festlegen von Passwortrichtlinien in MySQL

Zur Umsetzung von Passwortrichtlinien bietet MySQL das optionale Plugin „Validate Password“ an. Mit Hilfe von diesem Plugin, kann die notwendige Komplexität der Passwörter festgelegt werden.

Die MySQL Richtlinien sind in drei Stufen unterteilt: „Low“, „Medium“ und „Strong“. Folgende Tabelle zeigt, wann welche Tests durchgeführt werden.

Stufe	Durchgeführte Tests
0 oder LOW	Länge
1 oder MEDIUM	Länge; numerisch, lowercase/uppercase, und Sonderzeichen
2 oder STRONG	Länge; numerisch, lowercase/uppercase, and Sonderzeichen; Wörterbuch-Prüfung

Die einzelnen Tests können noch individuell konfiguriert werden, d.h. es kann die Mindestlänge, die Anzahl von numerischen, klein- und groß-geschriebenen und Sonderzeichen angegeben werden. Außerdem kann das durchsuchte Wörterbuch definiert werden.

Plugin installieren

```
INSTALL PLUGIN validate_password SONAME 'validate_password.so';
```

Level der Passwortrichtlinie setzen

```
SET GLOBAL validate_password_policy = 2;
```

Setzen des Wörterbuchs

```
SET GLOBAL validate_password_dictionary_file='/Most-Popular-Letter-Pass
es.txt';
```

Aktuelle Konfiguration anzeigen

```
SHOW VARIABLES LIKE 'validate_password%';
```

Testnutzer mit schwachem Passworte anlegen

```
create user 'Tester'@'localhost' identified by 'Manager1;';
```

A.4.3.6.2 Verschlüsselung von Passwörtern in MySQL

MySQL bietet unterschiedliche, integrierte Verschlüsselungsmethoden an und empfiehlt diese zu nutzen, um Passwörter im System zu speichern. Standardmäßig wird das Plugin „caching_sha2_password“ genutzt, um die Passwörter im System zu speichern. Dies betrifft die konkreten MySQL-System-

Nutzer. Das Plugin verschlüsselt mehrfach nach SHA256, in Verbindung mit Zufallsdaten. Detailliertere Beschreibungen sind [128] zu entnehmen.

Wenn individuelle Tabellen für Nutzer angelegt werden, ist vom Hersteller empfohlen, die Verschlüsselung selbst vorzunehmen. Hierbei wird ebenfalls der Algorithmus SHA256, implementiert in der Systemfunktion SHA2, empfohlen. Hierbei sollte folgendes Schema eingesetzt werden: sha2(sha2(password)+salt) [30].

Testnutzer mit schwachem Passworte anlegen

```
create user 'Tester'@'localhost' identified by 'Manager1;';
```

Angefügte Dateien:

- MySQL_Quellen.zip
- MySQL_Passwortverschluesselung.docx
- MySQL_Passwortregeln.docx
- Most-Popular-Letter-Passes.txt

A.4.3.7 PostgreSQL

A.4.3.7.1 Festlegen von Passwortrichtlinien in PostgreSQL

Zum Setzen von Passwortregeln bietet PostgreSQL das Plugin „passwordcheck“ an. Dieses kann über die Einstellungen aktiviert werden.

Zur Nutzung von erweiterten Funktionalitäten kann das Modul „Cracklib“ in dem Plugin aktiviert werden. Auf einem RHEL-basierten System muss Cracklib zuvor installiert werden. Dazu im Terminal folgenden Befehl eingeben:

```
dnf --enablerepo=PowerTools install cracklib-devel
```

Nachdem cracklib korrekt installiert ist, muss das Plugin neu kompiliert werden.

Hierzu den Quellcode von PostgreSQL herunterladen und in den „contrib“-Ordner wechseln.

Beim allerersten Mal muss das System noch entsprechend konfiguriert werden, damit es kompilieren kann. Hierzu folgender Anleitung folgen: https://www.postgresql.org/docs/9.1/contrib.html

Wenn alles korrekt funktioniert, im Unterordner „/contrib/passwordcheck/“ in der „Makefile“ die Zeilen 8 und 9 für Cracklib einkommentieren (# entfernen).

Nun mit dem Befehl „make“ (oder „make -B“ wenn es nicht neu kompiliert) die Dateien neuerstellen. Nach erfolgreichem Erstellen mit „make install“ die Dateien ins System installieren.

Encryption setzen

Zur Nutzung der Passwortregeln, im System den Befehl:

```
ALTER SYSTEM SET shared_preload_libraries = '$libdir/passwordcheck'
;
```

eingeben. Dann den Server neustarten.

Testnutzer mit schwachem Passworte anlegen

```
CREATE User Tester WITH PASSWORD 'secret123';
```

Nutzer anzeigen

```
Select * from pg_authid;
```

Nutzer löschen

```
Delete user Tester;
```

Passwort ändern

```
ALTER ROLE postgres WITH PASSWORD 'secret123';
```

A.4.3.7.2 Verschlüsselung von Passwörtern in PostgreSQL

Standardmäßig hashed PostgreSQL die Systemnutzer-Passwörter mit dem MD5-Algorithmus. Der Hersteller empfiehlt aber auf die sichere Methode des SCRAM34-Mechanismus umzusteigen. Hinter dem SCRAM-Mechanismus verbirgt die ein SHA256-Hashing-Algorithmus. [129]

Encryption setzen

```
ALTER SYSTEM SET password_encryption = 'scram-sha-256';
```

[34] SCRAM: Salted Challenge Response Authentication Mechanism. Ein in der RFC5802 festgelegter Authentifizierungsmechanismus. [4]

Testnutzer mit schwachem Passworte anlegen

```
CREATE User Tester WITH PASSWORD 'secret123';
```

Nutzer anzeigen

```
Select * from pg_authid;
```

Nutzer löschen

```
Delete user Tester;
```

Passwort ändern

```
ALTER ROLE postgres WITH PASSWORD 'secret123';
```

Angefügte Dateien:

- PostgreSQL_Quellen.zip
- PostgreSQL_Passwortverschluesselung.docx
- PostgreSQL_Passwortregeln.docx

A.4.3.8 MongoDB

A.4.3.8.1 Verschlüsselung von Passwörtern in MongoDB

Standardmäßig authentifiziert MongoDB die Systemnutzer-Passwörter mit dem SCRAM35-Mechanismus umzusteigen. SCRAM nutzt bei MongoDB den SHA1- und den SHA256-Hashing-Algorithmus. Beide Varianten werden abgelegt. [29] [130]

Testnutzer mit schwachem Passworte anlegen

```
db.createUser(
  {
    user: "amueller",
    pwd: " a_mueller_123",
    roles: [
       { role: "read", db: "Uni" }
    ]
  }
)
```

[35] SCRAM: Salted Challenge Response Authentication Mechanism. Ein in der RFC5802 festgelegter Authentifizierungsmechanismus. [4]

Nutzer anzeigen

```
Show user
```

oder

```
db.runCommand(
   {
     usersInfo:  { user: "tester", db: "mt_db" },
     showCredentials: true
   }
)
```

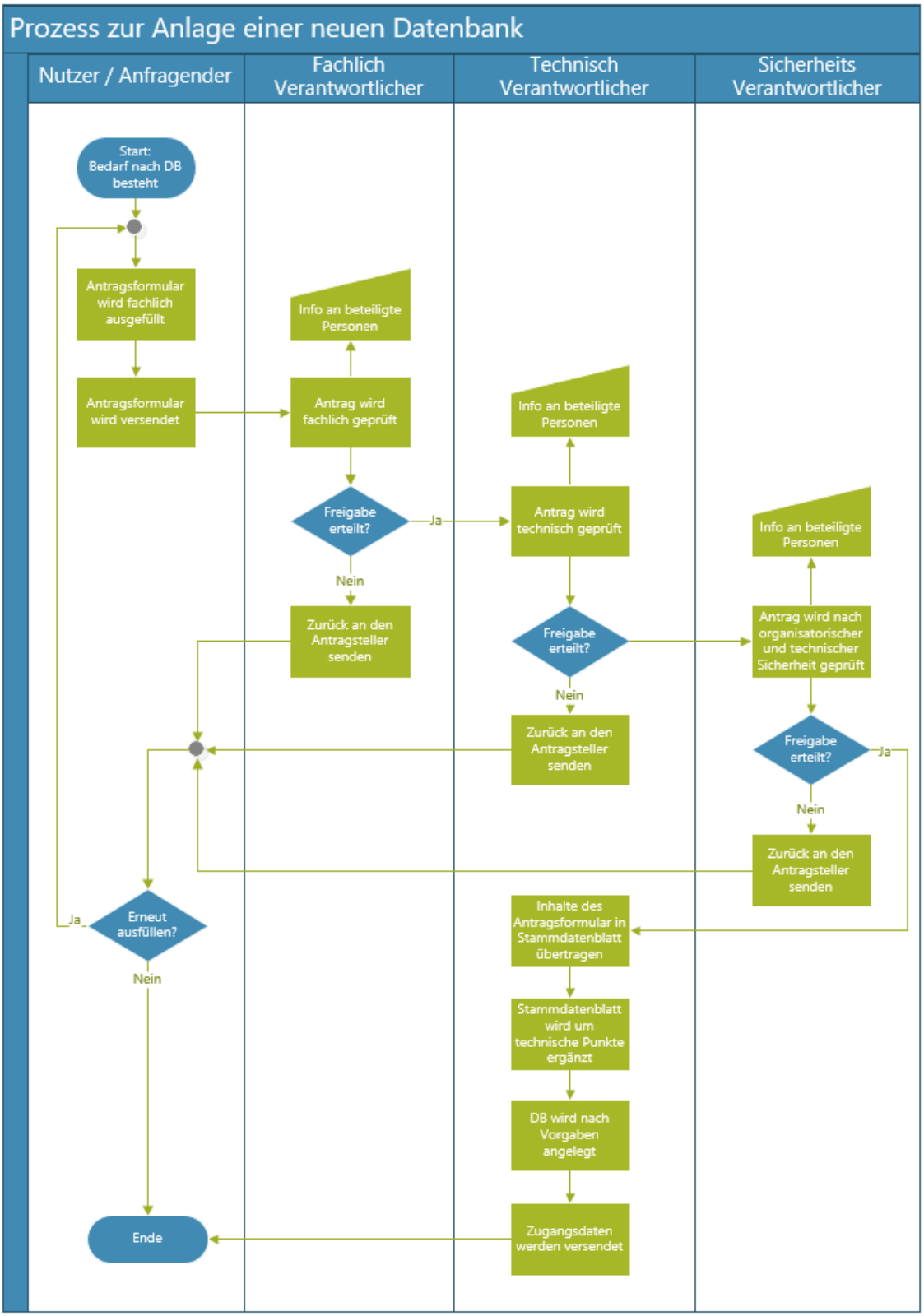
Angefügte Dateien:

- MongoDB_Quellen.zip
- MongoDB_Passwortverschluesselung.docx

A.4.4 APP.4.3.A4

Stammdatenblatt Datenbanken

Datenbankbezeichner		
Fachlich verantwortliche Person	Technisch verantwortliche Person	
Sicherheit verantwortliche Person		
Zugelassene Administratoren	*Berechtigung*	
Zweck der Datenbank / Zugehörigkeit		Kostenstelle
Laufzeit	Angelegt am	Löschung am
Datenbanktyp		
Hersteller	Modell	
Version	Letzter Patch	
Standort / Cloud Subscription		
Enthaltene DB-Links		
Ziel	*Zweck*	
Backups		
Rhythmus	*Aufbewahrungsfrist*	



Zur besseren Übersichtlichkeit wurde in diesem Prozess auf die Darstellung von beteiligten Dokumenten verzichtet.

Angefügte Dateien:

- DB_Stammdatenblatt.docx
- Prozess_AnlageNeueDB.vsdx

A.4.5 APP.4.3.A5

A.4.5.1 Testnutzer zur Umsetzung

Nutzer: amueller
Passwort: a_Mueller_123

Nutzer: ewagner
Passwort: e_Wagner_456

MySQL

```
create user 'amueller'@'localhost' identified by 'a_Mueller_123';
create user 'ewagner'@'localhost' identified by 'e_Wagner_456';
```

PostgreSQL

```
create user amueller with password 'a_Mueller_123';
create user ewagner with password 'e_Wagner_456';
```

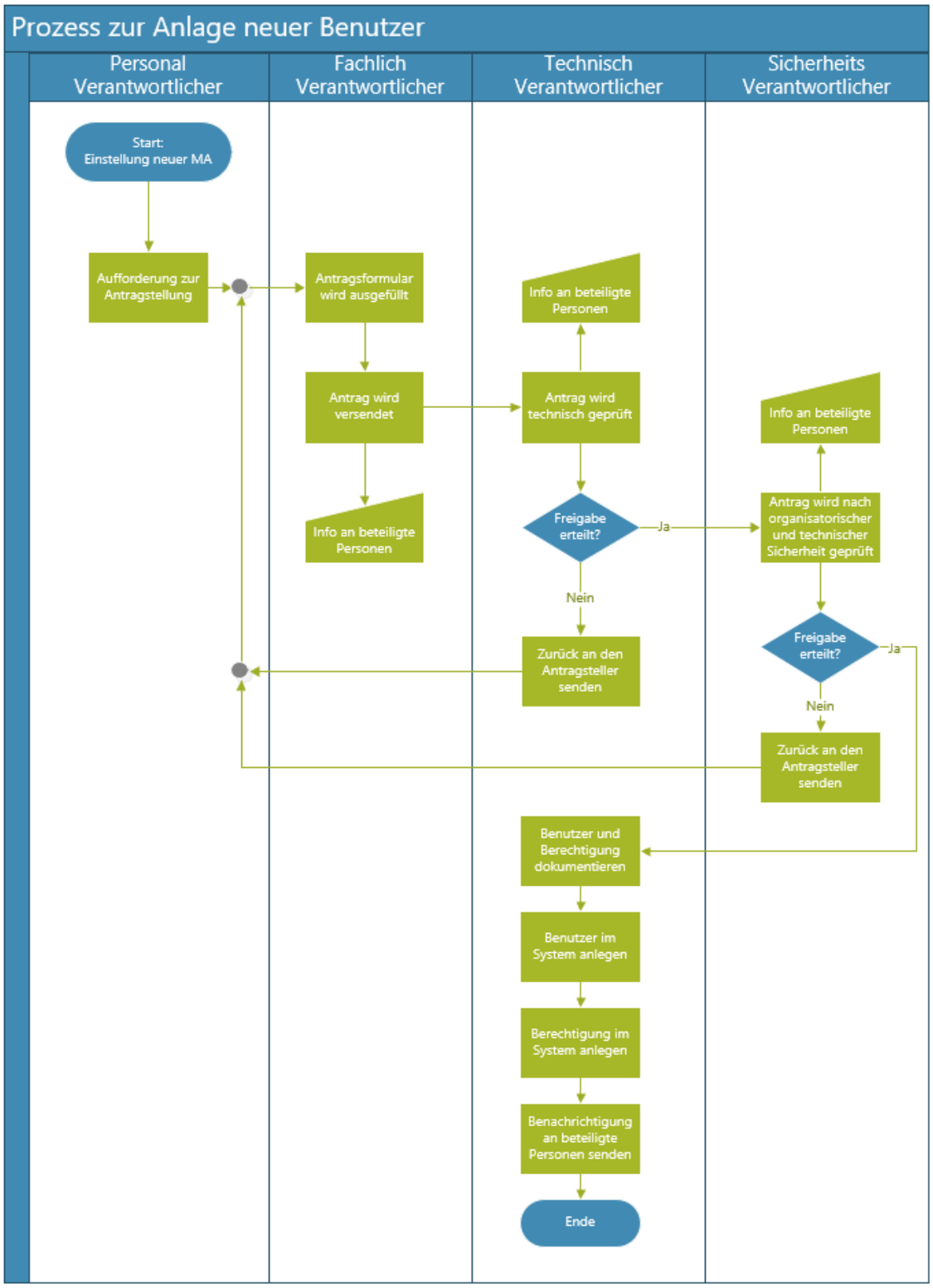
MongoDB

```
use admin
db.createUser(
   {
     user: "amueller",
     pwd: "a_Mueller_123",
     roles: [ ]
   }
)
db.createUser(
   {
     user: "ewagner",
     pwd: "e_Wagner_456",
     roles: [ ]
   }
)
```

A.4.5.2 Prozesse und Anträge

Antrag Neue Berechtigungen

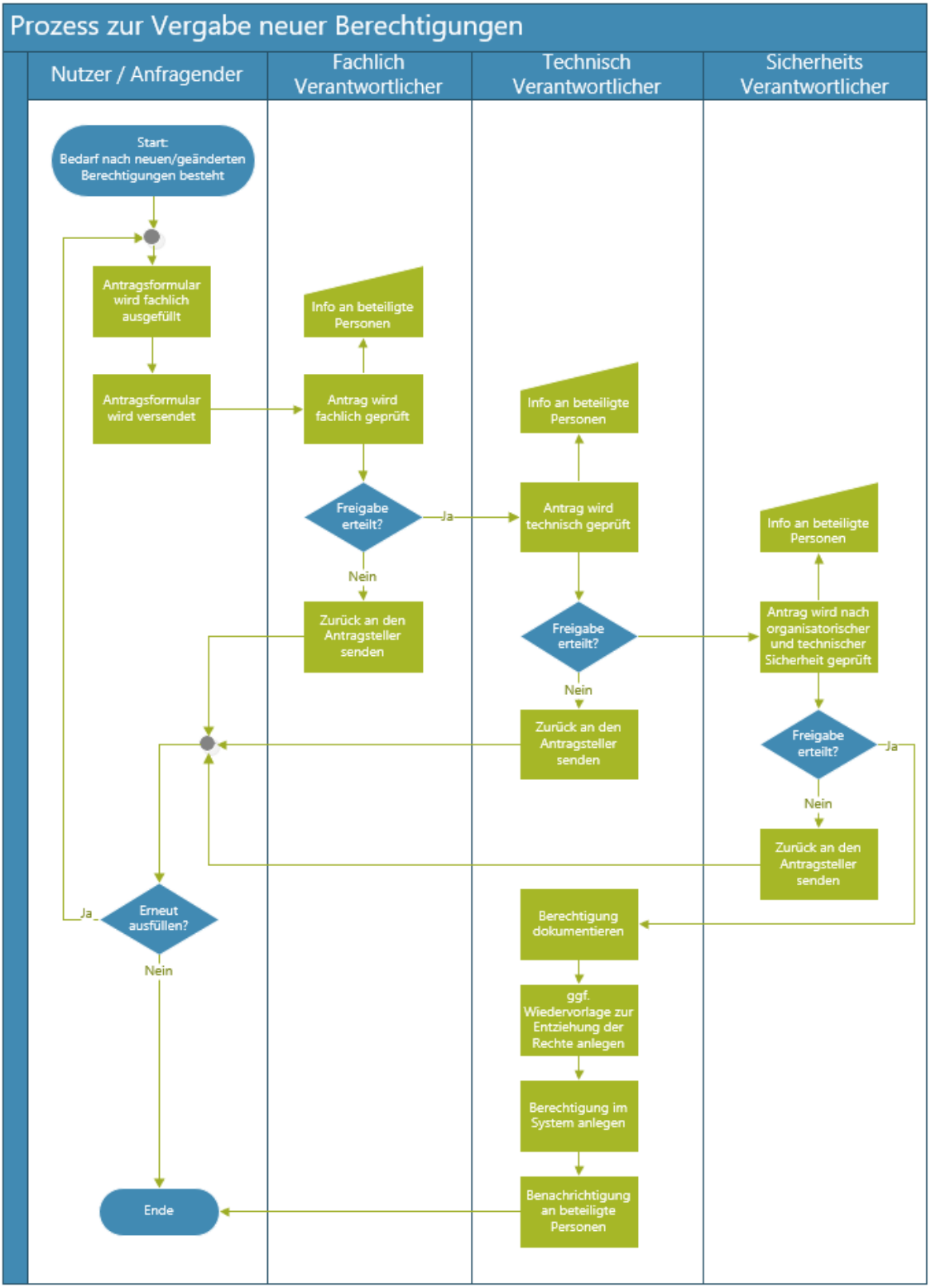
Antragsteller		
Fachlich verantwortliche Person	Technisch verantwortliche Person	
Sicherheit verantwortliche Person		
Begründung / Wozu wird die neue Berechtigung benötigt?		
Dauer der Berechtigung ☐ unbegrenzt	Von	Bis
Berechtigung		
Welche Datenbank?	Welche Tabelle / Informationen?	
Berechtigungsstufe ☐ Lesen ☐ Schreiben ☐ Administrative		



Zur besseren Übersichtlichkeit wurde in diesem Prozess auf die Darstellung von beteiligten Dokumenten verzichtet.

Antrag Berechtigung ändern

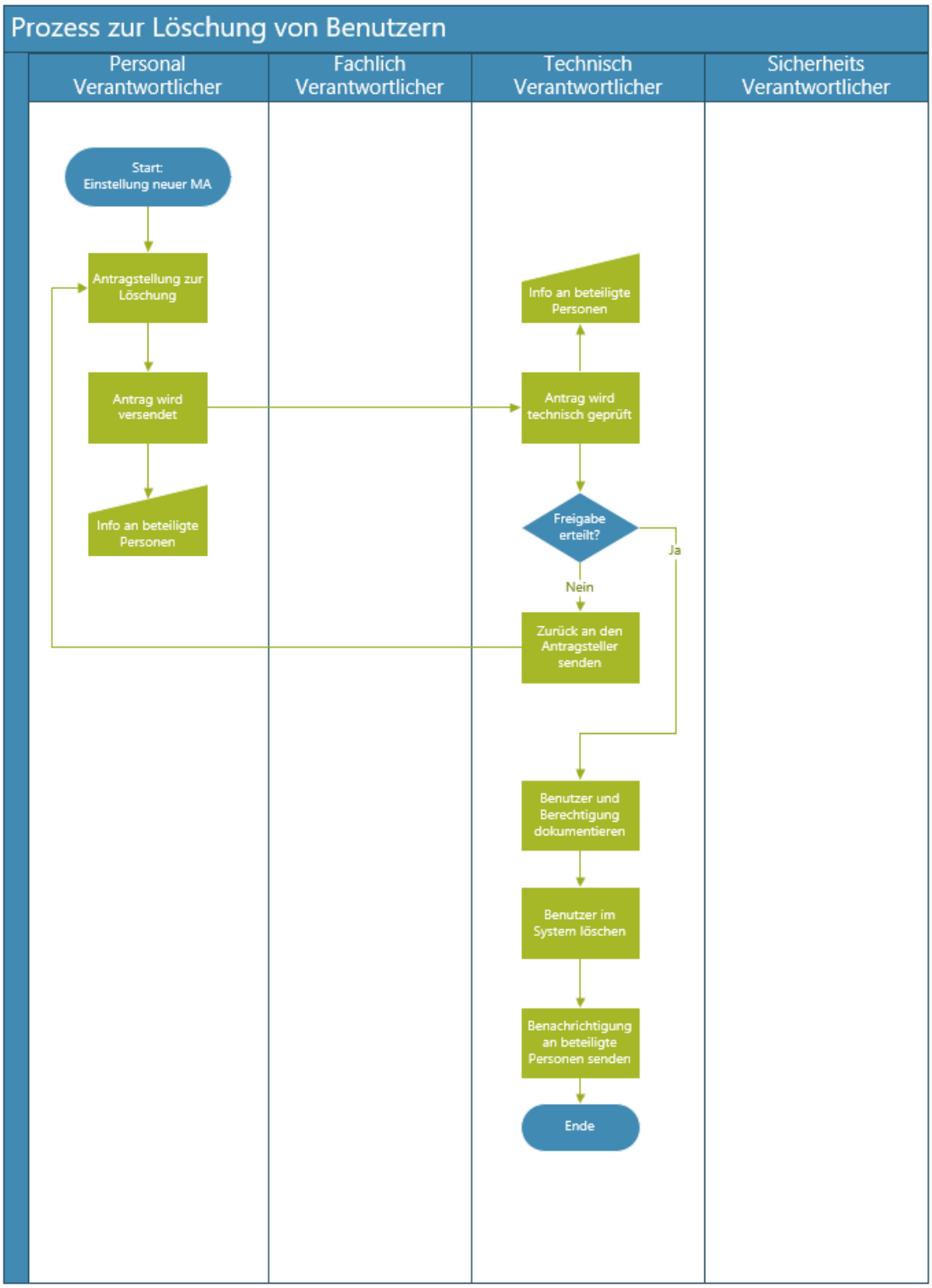
Antragsteller / Fachlich verantwortliche Person	
Sicherheit verantwortliche Person	Technisch verantwortliche Person
Benutzerdaten	
Nutzerreferenz zum Identity Provider (Nutzername, Kürzel, etc.)	
Geschäftsbereich	Abteilung
Berechtigung ändern	
Neue / Geänderte Nutzerprofile	
Nutzerprofil entziehen	



Zur besseren Übersichtlichkeit wurde in diesem Prozess auf die Darstellung von beteiligten Dokumenten verzichtet.

Antrag Benutzer entfernen

Antragsteller / Personal verantwortliche Person	
Fachlich verantwortliche Person	Technisch verantwortliche Person
Sicherheit verantwortliche Person	
Begründung / Warum wird der Nutzer gelöscht?	
Berechtigung	
Welche Datenbanken?	



Zur besseren Übersichtlichkeit wurde in diesem Prozess auf die Darstellung von beteiligten Dokumenten verzichtet.

Angefügte Dateien:

- Antrag_BenutzerEntfernen.docx
- Antrag_NeueGeaenderteBerechtigung.docx
- Antrag_NeuerBenutzer.docx
- Prozess_BenutzerEntfernen.vsdx
- Prozess_NeueGeaenderteBerechtigung.vsdx
- Prozess_NeuerBenutzer.vsdx

A.4.5.3 MySQL

A.4.5.3.1 Festlegen von Rollen und Rechten in MySQL

Implementierung der Rollen und Rechte

```
Create user 'amueller'@'localhost';
Create user 'ewagner'@'localhost';
create role 'PersonalEinsicht'@'localhost';
create role 'Pruefungsamt'@'localhost';
create role 'PersonalTariflich'@'localhost';
FLUSH PRIVILEGES;
grant select on Uni.Professoren to 'PersonalEinsicht'@'localhost';
grant select on Uni.Assistenten to 'PersonalEinsicht'@'localhost';
grant select, insert, update, delete on Uni.Assistenten to 'PersonalTar
iflich'@'localhost';
grant select, insert, update, delete on Uni.pruefen to 'Pruefungsamt'@'
localhost';
grant select, insert, update, delete on Uni.Assistenten to 'PersonalTar
iflich'@'localhost';
FLUSH PRIVILEGES;
grant 'PersonalEinsicht'@'localhost' to 'amueller'@'localhost';
grant 'Pruefungsamt'@'localhost' to 'amueller'@'localhost';
FLUSH PRIVILEGES;
grant 'PersonalEinsicht'@'localhost' to 'ewagner'@'localhost';
grant 'PersonalTariflich'@'localhost' to 'ewagner'@'localhost';
FLUSH PRIVILEGES;
SET DEFAULT ROLE ALL TO 'amueller'@'localhost';
SET DEFAULT ROLE ALL TO 'ewagner'@'localhost';
FLUSH PRIVILEGES;
```

Testen der Berechtigung

Benutzer	**Zugriff auf Tabelle**	**Berechtigung**	**Erwartet**	**Ergebnis**
amueller	Professoren	Lesen	Zugelassen	Zugelassen
	Professoren	Schreiben	Abgelehnt	Abgelehnt
	Assistenten	Lesen	Zugelassen	Zugelassen
	Assistenten	Schreiben	Abgelehnt	Abgelehnt
	pruefen	Lesen	Zugelassen	Zugelassen
	pruefen	schreiben	Zugelassen	Zugelassen
ewagner	Professoren	Lesen	Zugelassen	Zugelassen
	Professoren	Schreiben	Abgelehnt	Abgelehnt
	Assistenten	Lesen	Zugelassen	Zugelassen
	Assistenten	Schreiben	Zugelassen	Zugelassen
	pruefen	Lesen	Abgelehnt	Abgelehnt
	pruefen	schreiben	Abgelehnt	Abgelehnt

Test-Statements:

```
SELECT * FROM uni.professoren;
UPDATE uni.professoren SET raum = '666' WHERE NAME = 'Kant';

SELECT * FROM uni.assistenten;
UPDATE uni.assistenten SET fachgebiet = '666' WHERE NAME = 'Newton'
;

SELECT * FROM uni.pruefen;
UPDATE uni. pruefen SET note = 4.0 WHERE matrnr = 27550;
```

Eine weitere Rolle “PersonalProfessoren” wird angelegt:

```
create role 'PersonalProfessoren'@'localhost';
```

Die neue Rolle wird einer bestehenden Rolle zugeordnet:

```
grant 'PersonalTariflich'@'localhost' to 'PersonalProfessoren'@'localho
st';
```

Diese Zuweisung von einer Rolle zu einer anderen Rolle ist somit in MySQL möglich.

Es soll die Einschränkung definiert werden, dass die Rolle „PersonalTariflich“ nicht mit der Rolle „Prüfungsamt“ vergeben werden darf.

Das ist in MySQL derzeit nicht möglich.

Angehängte Datei:

- MySQL_RollenUndRechte.docx

A.4.5.4 PostgreSQL

A.4.5.4.1 Festlegen von Rollen und Rechten in PostgreSQL

Implementierung der Rollen und Rechte

```
create user amueller with password 'a_mueller_123';
create user ewagner with password 'e_wagner_456';

create role PersonalEinsicht;
create role Pruefungsamt;
create role PersonalTariflich;

grant select on professoren to PersonalEinsicht;
grant select on assistenten to PersonalEinsicht;
grant select, insert, update, delete on assistenten to PersonalTariflic
h;
grant select, insert, update, delete on pruefen to Pruefungsamt;
grant select, insert, update, delete on assistenten to PersonalTariflic
h;
```

```
grant PersonalEinsicht to amueller;
grant Pruefungsamt to amueller;
grant PersonalEinsicht to ewagner;
grant PersonalTariflich to ewagner;
```

Testen der Berechtigung

Benutzer	Zugriff auf Tabelle	Berechtigung	Erwartet	Ergebnis
amueller	Professoren	Lesen	Zugelassen	Zugelassen
	Professoren	Schreiben	Abgelehnt	Abgelehnt
	Assistenten	Lesen	Zugelassen	Zugelassen
	Assistenten	Schreiben	Abgelehnt	Abgelehnt
	pruefen	Lesen	Zugelassen	Zugelassen
	pruefen	schreiben	Zugelassen	Zugelassen
ewagner	Professoren	Lesen	Zugelassen	Zugelassen
	Professoren	Schreiben	Abgelehnt	Abgelehnt
	Assistenten	Lesen	Zugelassen	Zugelassen
	Assistenten	Schreiben	Zugelassen	Zugelassen
	pruefen	Lesen	Abgelehnt	Abgelehnt
	pruefen	schreiben	Abgelehnt	Abgelehnt

Test-Statements

```
SELECT * FROM professoren;
UPDATE professoren SET raum = '666' WHERE NAME = 'Kant';

SELECT * FROM assistenten;
UPDATE assistenten SET fachgebiet = '666' WHERE NAME = 'Newton';

SELECT * FROM pruefen;
UPDATE pruefen SET note = 4.0 WHERE matrnr = 27550;
```

Eine weitere Rolle “PersonalProfessoren” wird angelegt:

```
create role PersonalProfessoren;
```

Die neue Rolle wird einer bestehenden Rolle zugeordnet:

```
grant PersonalTariflich to PersonalProfessoren;
```

Diese Zuweisung von einer Rolle zu einer anderen Rolle ist in PostgreSQL möglich.

Es soll die Einschränkung definiert werden, dass die Rolle „PersonalTariflich“ nicht mit der Rolle „Prüfungsamt“ vergeben werden darf.

Das ist in PostgreSQL derzeit nicht möglich.

Angehängte Datei:

- PostgreSQL_RollenUndRechte.docx

A.4.5.5 MongoDB

Implementierung der Rollen und Rechte

```
db.createRole(
   {
     role: "PersonalEinsicht",
     privileges: [
       { resource: { db: "uni", collection: "professoren" }, actions: [
"find" ] },
       { resource: { db: "uni", collection: "assistenten" }, actions: [
"find" ] }
     ],
     roles: []
   }
)
db.createRole(
   {
     role: "Pruefungsamt",
     privileges: [
       { resource: { db: "uni", collection: "pruefen" }, actions: [ "fi
nd", "update", "insert" ] }
     ],
     roles: ["PersonalEinsicht"]
   }
)
db.createRole(
   {
     role: "PersonalTariflich",
     privileges: [
       { resource: { db: "uni", collection: "assistenten" }, actions: [
"find", "update", "insert" ] }
     ],
     roles: ["PersonalEinsicht"]
   }
)
db.grantRolesToUser( "amueller", [ "Pruefungsamt" ] ) db.grantRolesToUs
er( "ewagner", [ "PersonalTariflich" ] )
```

Testen der Berechtigung

Benutzer	Zugriff auf Tabelle	Berechtigung	Erwartet	Ergebnis
amueller	Professoren	Lesen	Zugelassen	Zugelassen
	Professoren	Schreiben	Abgelehnt	Abgelehnt
	Assistenten	Lesen	Zugelassen	Zugelassen
	Assistenten	Schreiben	Abgelehnt	Abgelehnt
	pruefen	Lesen	Zugelassen	Zugelassen
	pruefen	schreiben	Zugelassen	Zugelassen
ewagner	Professoren	Lesen	Zugelassen	Zugelassen
	Professoren	Schreiben	Abgelehnt	Abgelehnt
	Assistenten	Lesen	Zugelassen	Zugelassen
	Assistenten	Schreiben	Zugelassen	Zugelassen
	pruefen	Lesen	Abgelehnt	Abgelehnt
	pruefen	schreiben	Abgelehnt	Abgelehnt

Test-Statements:

```
db.professoren.find()
db.professoren.updateOne(
   {"PersNr": "2125"},
   {
     $set: {"Raum": "666"}
   }
)

db.assistenten.find()
db.assistenten.updateOne(
   { Name": "Newton"},
   {
     $set: {"Fachgebiet": "666"}
   }
)

db.pruefen.find()
db.pruefen.updateOne(
   {"MatrNr": 27550},
   {
     $set: {"Note": 4}
   }
)
```

Eine weitere Rolle “PersonalProfessoren” wird angelegt und die Rolle PersonalTariflich direkt zugeordnen:

```
db.createRole(
   {
     role: "PersonalProfessoren",
     privileges: [],
     roles: ["PersonalTariflich"]
   }
)
```

Diese Zuweisung von einer Rolle zu einer anderen Rolle ist in MongoDB möglich.

Es soll die Einschränkung definiert werden, dass die Rolle „PersonalTariflich“ nicht mit der Rolle „Prüfungsamt“ vergeben werden darf.

Das ist in MongoDB derzeit nicht möglich.

Angehängte Datei:

- MongoDB_RollenUndRechte.docx

A.4.6 APP.4.3.A6

A.4.6.1 MySQL

A.4.6.1.1 Setzen von Passwortrichtlinien in MySQL

MySQL bietet unterschiedliche Richtlinien an, um ein Passwortmanagement zu ermöglichen [36]. Hierzu zählen:

- Ablaufen von Passwörtern
- Passwort-Wiederbenutzung unterbinden
- Nutzung von Dual Passwörter
- Passwortkomplexität festlegen
- Zufälliges Generieren von Passwörtern
- Account sperren, bei zu häufiger Passwort Falscheingabe

Zur Vergleichbarkeit werden hiervon drei Anforderungen ausgesucht:

- Regelmäßiges Erneuern von Passwörtern
- Wiederbenutzung von alten Passwörtern
- Ungültig machen aller Passwörter im System

A.4.6.1.2 Regelmäßiges Erneuern von Passwörtern

Mit dem Befehl

```
SET PERSIST default_password_lifetime = 180;
```

Kann eine dauerhafte Lebenszeit von Passwörtern global eingestellt werden.

---

[36] MySQL Password Management; https://dev.mysql.com/doc/mysql-security-excerpt/8.0/en/password-management.html

Der Wert am Ende ist in Tagen angegeben.

A.4.6.1.3 Wiederbenutzung von alten Passwörtern

Mit dem Befehl

```
SET PERSIST password_history = 6;
```

Wird global eingestellt, dass die letzten sechs Passwörter nicht wiederverwendet werden dürfen.

A.4.6.1.4 Ungültig machen aller Passwörter im System

In MySQL kann das Passwort von einem einzelnen Nutzer ungültig gemacht werden, dazu dient der Befehl:

```
ALTER USER USERNAME@'localhost' PASSWORD EXPIRE;
```

Um alle Passwörter der Nutzer im System ungültig zu machen, muss dieser Befehl für jeden Nutzer einzeln aufgerufen werden, was einen hohen administrativen Aufwand bedeutet.

Ein weiterer Gedanke ist es, die Ablaufzeit auf einen Tag zu setzen. Ein Tag ist die kleinste, mögliche Zeit. Eine Dauer von Null hätte zu Folge, dass das Passwort niemals abläuft.

In jedem Fall sollte der Befehl

```
SET PERSIST password_require_current = ON;
```

ausgeführt werden. Dieser sorgt dafür, dass der Nutzer auch das alte Passwort eingeben muss, bevor er ein neues festlegen kann.

Eine weitere Möglichkeit ist das manuelle Setzen des abgelaufenen Passwortes in der MySQL-User—Liste:

```
Update mysql.user set password_expired = 'Y' WHERE USER = 'amueller
';
```

Angehängte Datei:

- MySQL_Passwortmanagement.docx

A.4.6.2 PostgreSQL

A.4.6.2.1 Setzen von Passwortrichtlinien in PostgreSQL

Es sollen Zur Vergleichbarkeit drei Anforderungen umgesetzt werden:

- Regelmäßiges Erneuern von Passwörtern
- Wiederbenutzung von alten Passwörtern
- Ungültig machen aller Passwörter im System

A.4.6.2.2 Regelmäßiges Erneuern von Passwörtern

PostgreSQL hat keine integrierte Funktion, um eine standardmäßiges Ablaufzeit festzulegen. Wenn Nutzer angelegt werden, kann ein Ablaufdatum für das Passwort festgelegt werden:

```
ALTER USER username VALID UNTIL 'Jan 31 2030';
```

Für Nutzer, bei denen das nicht angegeben wurde, kann der Wert mit folgender Abfrage erneut gesetzt werden:

```
UPDATE pg_authid
SET rolvaliduntil = 'Jan 31 2030'
WHERE rolname IN (
  SELECT rolname
  FROM pg_authid
  WHERE rolvaliduntil IS NULL
);
```

A.4.6.2.3 Wiederbenutzung von alten Passwörtern

Wird derzeit nicht in PostgreSQL unterstützt.

A.4.6.2.4 Ungültig machen aller Passwörter im System

In PostgreSQL können die Nutzerpasswörter nicht per sofort ungültig gemacht werden. Es ist stattdessen möglich, ein Ablaufdatum anzugeben. Um das Passwort von einem Nutzer für ungültig zu erklären, kann das Ablaufdatum auf ein in der Vergangenheit liegendes Passwort gesetzt werden:

```
ALTER USER USERNAME@'localhost' VALID UNTIL '01-01-2000';
```

Um alle Passwörter der Nutzer im System ungültig zu machen, muss dieser Befehl für jeden Nutzer einzeln aufgerufen werden, was einen hohen administrativen Aufwand bedeutet.

Problematisch bei dieser Lösung ist, dass der Nutzer nach dem Ablaufen des Passworts nicht mehr in der Lage ist, sich einzuloggen. Der Nutzer kann nur durch einen Administrator-Nutzer wieder aktiviert werden.

Angehängte Datei:

- PostgreSQL_ Passwortmanagement.docx

A.4.7 APP.4.3.A8

A.4.7.1 MySQL

A.4.7.1.1 Protokollierung im MySQL Server

MySQL bietet unterschiedliche Log-Funktionen an [44]:

- Error-Log: Zeichnet Probleme mit dem Starten, Ausführen und Stoppen des Dienstprogrammes auf.
- General Query Log: Zeichnet Verbindungen zum Server und erhaltene Statements vom Server auf
- Binary Log: Zeichnet Statements auf, die Änderungen an der Datenbasis vornehmen
- Relay Log: Zeichnet Änderungen auf, die von einem Replikationsserver gesendet werden
- Slow Query Log: Zeichnet Statements auf, die länger als eine definierte Zeit (*long_query_time*) zur Ausführung benötigen
- DDL Log: Zeichnet Metadaten auf, die die Datenstruktur verändern (DDL Statements)

In der Standardinstallation von MySQL ist keine dieser Log-Funktionalität aktiviert.

A.4.7.1.2 Protokollfunktionen in MySQL mit Audit Log (Percona LLC)

Da es unterschiedliche Derivate gibt, die auf der Basis von MySQL Community entwickelt wurden, und erweiterte Funktionalitäten bieten, wird hier nach Alternativen geschaut. Außerdem gibt es ein Plugin, welches von McAfee für MySQL entwickelt wurde, um diese Funktionalität zu bedienen. Daraus ergeben sich drei Alternativen:

Übersicht von MySQL Audit-Plugins

Anbieter	Plugin	Quelle
MariaDB Foundation	Server_Audit	MariaDB Audit Plugin
McAfee LLC	Libaudit_plugin	McAfee Knowledge Center
Percona LLC	Audit_log	Percona Server Audit Log

Alle drei Varianten werden mit einer Opensource-Lizenz angeboten, somit ist die volle Funktionalität einsehbar und das Risiko eine Schadsoftware einzuführen geringer.

Von den drei Alternativen lässt sich lediglich die Variante von Percona LLC in der aktuellen MySQL Server Version installieren. Die anderen sind nicht kompatibel. Außerdem wird offiziell nur das Betriebssystem Linux unterstützt.

A.4.7.1.3 Installation des Plugins

Zur Installation des Plugins muss die Plugin-Datei aus dem Server-Paket des „Percona Server for MySQL“-Paket gezogen werden. Das Paket kann unter folgenden Link bezogen werden:

https://www.percona.com/downloads/Percona-Server-LATEST/

Im Beispiel wird ein CentOS8-Server verwendet, daher wird die entspreche Software heruntergeladen. Nachdem Entpacken der Downloaddatei wird im Unterordner „/usr/lib64/mysql/plugin“ die Datei „audit_log.so“ herausgesucht.

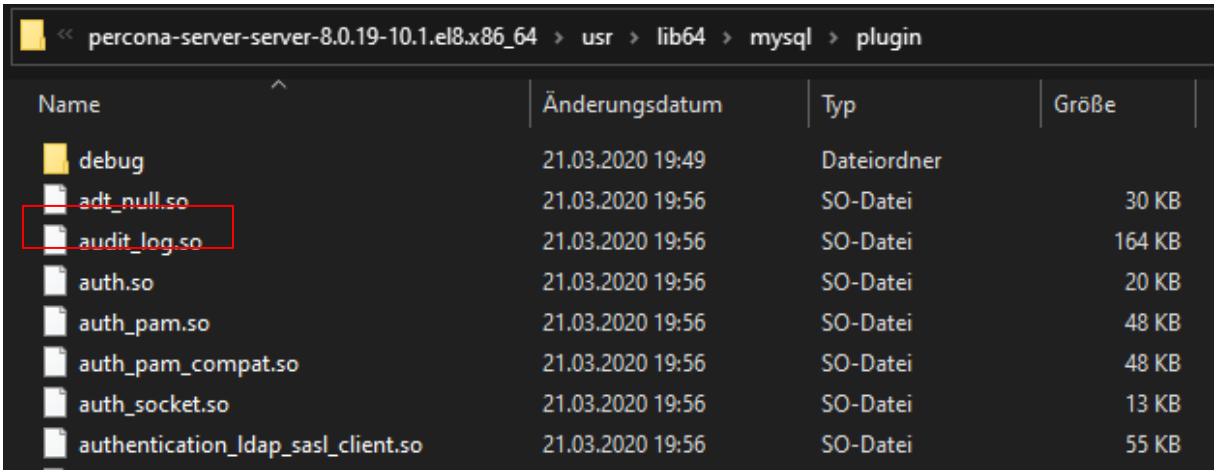


Abbildung 1 - Plugin-Verzeichnis der Percona-Installation mit Audit-log-Plugin

Die Installation des Plugins im MySQL Server erfolgt nach folgenden Schritten:

1) Beenden des MySQL-Services via Befehl:

```
sudo service mysqld stop
```

Kopieren der Plugin-Datei „audit_log.so“ in das Verzeichnis: `/usr/lib64/mysql/plugin/`

2) Anpassen der Config-Datei „my.cnf“:

   Einfügen der Zeilen:

```
plugin-load       = audit_log.so
audit_log_file    = /var/log/mysql/audit.log
audit_log_format  = NEW
audit_log_policy  = ALL
audit_log_handler = FILE
```

   Wobei die Angabe „audit_log_file“ den Speicherpfad der Log-Datei angibt.

1) Starten des MySQL-Services via: `sudo service mysqld start`
2) Wechseln in die mysql-Console und Installation des Plugins überprüfen:

```
select * from information_schema.PLUGINS where PLUGIN_NAME like '%a
udit%'\G
```

   Das Ergebnis sollte wie folgt aussehen:

```
mysql> select * from information_schema.PLUGINS where PLUGIN_NAME like '%audit%'\G
*************************** 1. row ***************************
           PLUGIN_NAME: audit_log
        PLUGIN_VERSION: 0.2
         PLUGIN_STATUS: ACTIVE
           PLUGIN_TYPE: AUDIT
   PLUGIN_TYPE_VERSION: 4.1
        PLUGIN_LIBRARY: audit_log.so
PLUGIN_LIBRARY_VERSION: 1.10
         PLUGIN_AUTHOR: Percona LLC and/or its affiliates.
    PLUGIN_DESCRIPTION: Audit log
        PLUGIN_LICENSE: GPL
           LOAD_OPTION: ON
1 row in set (0,00 sec)
```

Abbildung 2 - Überprüfung der Installation des Audit-Plugins

A.4.7.1.4 Konfiguration

Die aktuelle Konfiguration wird mit folgender Abfrage angezeigt:

```
mysql> show global variables like 'audit%';
+-----------------------------+--------------------------+
| Variable_name               | Value                    |
+-----------------------------+--------------------------+
| audit_log_buffer_size       | 1048576                  |
| audit_log_exclude_accounts  |                          |
| audit_log_exclude_commands  |                          |
| audit_log_exclude_databases |                          |
| audit_log_file              | /var/log/mysql/audit.log |
| audit_log_flush             | OFF                      |
| audit_log_format            | NEW                      |
| audit_log_handler           | FILE                     |
| audit_log_include_accounts  |                          |
| audit_log_include_commands  |                          |
| audit_log_include_databases |                          |
| audit_log_policy            | ALL                      |
| audit_log_rotate_on_size    | 0                        |
| audit_log_rotations         | 0                        |
| audit_log_strategy          | ASYNCHRONOUS             |
| audit_log_syslog_facility   | LOG_USER                 |
| audit_log_syslog_ident      | percona-audit            |
| audit_log_syslog_priority   | LOG_INFO                 |
+-----------------------------+--------------------------+
18 rows in set (0,01 sec)
```

Abbildung 3 - Anzeige der Audit-log Konfiguration

A.4.7.1.5 Protokoll-Beispiel

Der nachfolgende Auszug ist das Ergebnis folgender Ausführungen auf der Datenbank:

- Start des Datenbankservices
- Einloggen als „root“
- Wechseln zur Datenbank „Uni“
- Abfrage der Daten: Select * from Studenten;
- Ausloggen von der Datenbank

```
<?xml version="1.0" encoding="UTF-8"?>
<AUDIT>
<AUDIT_RECORD>
  <NAME>Audit</NAME>
  <RECORD>1_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:01Z</TIMESTAMP>
  <MYSQL_VERSION>8.0.20</MYSQL_VERSION>
  <STARTUP_OPTIONS></STARTUP_OPTIONS>
  <OS_VERSION>x86_64-Linux</OS_VERSION>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Connect</NAME>
  <RECORD>2_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:21Z</TIMESTAMP>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <USER>root</USER>
```

```
  <PRIV_USER>root</PRIV_USER>
  <OS_LOGIN></OS_LOGIN>
  <PROXY_USER></PROXY_USER>
  <HOST>localhost</HOST>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Query</NAME>
  <RECORD>3_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:21Z</TIMESTAMP>
  <COMMAND_CLASS>select</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT>select @@version_comment limit 1</SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Query</NAME>
  <RECORD>4_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>select</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT>SELECT DATABASE()</SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Init DB</NAME>
  <RECORD>5_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>error</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT></SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Query</NAME>
  <RECORD>6_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>show_databases</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT>show databases</SQLTEXT>
  <USER>root[root] @ localhost []</USER>
```

```
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Query</NAME>
  <RECORD>7_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>show_tables</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT>show tables</SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Field List</NAME>
  <RECORD>8_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>error</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT></SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Field List</NAME>
  <RECORD>9_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>error</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT></SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Field List</NAME>
  <RECORD>10_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>error</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT></SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
```

```
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Field List</NAME>
  <RECORD>11_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>error</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT></SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Field List</NAME>
  <RECORD>12_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>error</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT></SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Field List</NAME>
  <RECORD>13_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>error</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT></SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Field List</NAME>
  <RECORD>14_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:24Z</TIMESTAMP>
  <COMMAND_CLASS>error</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT></SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
```

```
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Query</NAME>
  <RECORD>15_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:29Z</TIMESTAMP>
  <COMMAND_CLASS>select</COMMAND_CLASS>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <SQLTEXT>select * from Studenten</SQLTEXT>
  <USER>root[root] @ localhost []</USER>
  <HOST>localhost</HOST>
  <OS_USER></OS_USER>
  <IP></IP>
  <DB></DB>
</AUDIT_RECORD>
<AUDIT_RECORD>
  <NAME>Quit</NAME>
  <RECORD>16_2020-05-03T11:28:01</RECORD>
  <TIMESTAMP>2020-05-03T11:28:32Z</TIMESTAMP>
  <CONNECTION_ID>8</CONNECTION_ID>
  <STATUS>0</STATUS>
  <USER>root</USER>
  <PRIV_USER>root</PRIV_USER>
  <OS_LOGIN></OS_LOGIN>
  <PROXY_USER></PROXY_USER>
  <HOST>localhost</HOST>
  <IP></IP>
  <DB>Uni</DB>
</AUDIT_RECORD>
```


Angehängte Datei:

- MySQL_Protokollierung.docx

A.4.7.2 PostgreSQL

A.4.7.2.1 Protokollfunktionen in PostgreSQL mit pgAudit

Der Defakto-Standard für Audit-Logging ist PostgreSQL Datenbanken ist pgAudit, welches von der Postgres Community entwickelt wurde.

A.4.7.2.2 Installation des Plugins

Zu Installation des Plugins wird im CentOS Terminal folgender Befehl ausgeführt:

```
dnf install pgaudit
```

In der Postgres-Config-Datei muss das Plugin zum Laden angegeben werden. Hierzu wird der Variable „Shared_preload_libraries“ der Pluginname mitgegeben.

Beispiel:

```
# - Shared Library Preloading -
shared_preload_libraries = 'pgaudit.so'
```

Anschließend kann muss der PostgreSQL Service neugestartet werden:

```
Systemctl restart postgresql-12
```

Nach dem Neustart in der psql-Konsole als Admin anmelden und folgenden Befehl ausführen:

```
CREATE EXTENSION pgaudit;
```

Zur Überprüfung der Installation des Plugins den Befehl

```
select * from pg_available_extensions;
```

Eingeben. Folgende Ausgabe sollte erscheinen.

```
postgres=# select * from pg_available_extensions;
        name        | default_version | installed_version |                              comment
--------------------+-----------------+-------------------+----------------------------------------------------------------------
 hstore             | 1.6             |                   | data type for storing sets of (key, value) pairs
 plpgsql            | 1.0             | 1.0               | PL/pgSQL procedural language
 hstore_plperl      | 1.0             |                   | transform between hstore and plperl
 adminpack          | 2.0             |                   | administrative functions for PostgreSQL
 hstore_plperlu     | 1.0             |                   | transform between hstore and plperlu
 amcheck            | 1.2             |                   | functions for verifying relation integrity
 cube               | 1.4             |                   | data type for multidimensional cubes
 insert_username    | 1.0             |                   | functions for tracking who changed a table
 autoinc            | 1.0             |                   | functions for autoincrementing fields
 bloom              | 1.0             |                   | bloom access method - signature file based index
 earthdistance      | 1.1             |                   | calculate great-circle distances on the surface of the Earth
 intagg             | 1.1             |                   | integer aggregator and enumerator (obsolete)
 btree_gin          | 1.3             |                   | support for indexing common datatypes in GIN
 file_fdw           | 1.0             |                   | foreign-data wrapper for flat file access
 fuzzystrmatch      | 1.1             |                   | determine similarities and distance between strings
 btree_gist         | 1.5             |                   | support for indexing common datatypes in GiST
 intarray           | 1.2             |                   | functions, operators, and index support for 1-D arrays of integers
 citext             | 1.6             |                   | data type for case-insensitive character strings
 dblink             | 1.2             |                   | connect to other PostgreSQL databases from within a database
 jsonb_plperl       | 1.0             |                   | transform between jsonb and plperl
 dict_xsyn          | 1.0             |                   | text search dictionary template for extended synonym processing
 lo                 | 1.1             |                   | Large Object maintenance
 dict_int           | 1.0             |                   | text search dictionary template for integers
 jsonb_plperlu      | 1.0             |                   | transform between jsonb and plperlu
 isn                | 1.2             |                   | data types for international product numbering standards
 ltree              | 1.1             |                   | data type for hierarchical tree-like structures
 moddatetime        | 1.0             |                   | functions for tracking last modification time
 pgrowlocks         | 1.2             |                   | show row-level locking information
 pageinspect        | 1.7             |                   | inspect the contents of database pages at a low level
 pg_trgm            | 1.4             |                   | text similarity measurement and index searching based on trigrams
 tcn                | 1.0             |                   | Triggered change notifications
 pgstattuple        | 1.5             |                   | show tuple-level statistics
 pg_buffercache     | 1.3             |                   | examine the shared buffer cache
 xml2               | 1.1             |                   | XPath querying and XSLT
 postgres_fdw       | 1.0             |                   | foreign-data wrapper for remote PostgreSQL servers
 pg_freespacemap    | 1.2             |                   | examine the free space map (FSM)
 pg_prewarm         | 1.2             |                   | prewarm relation data
 pg_visibility      | 1.2             |                   | examine the visibility map (VM) and page-level visibility info
 refint             | 1.0             |                   | functions for implementing referential integrity (obsolete)
 pgcrypto           | 1.3             |                   | cryptographic functions
 pg_stat_statements | 1.7             |                   | track execution statistics of all SQL statements executed
 seg                | 1.3             |                   | data type for representing line segments or floating-point intervals
 sslinfo            | 1.2             |                   | information about SSL certificates
 tablefunc          | 1.0             |                   | functions that manipulate whole tables, including crosstab
 tsm_system_rows    | 1.0             |                   | TABLESAMPLE method which accepts number of rows as a limit
 tsm_system_time    | 1.0             |                   | TABLESAMPLE method which accepts time in milliseconds as a limit
 unaccent           | 1.1             |                   | text search dictionary that removes accents
 uuid-ossp          | 1.1             |                   | generate universally unique identifiers (UUIDs)
 pgaudit            | 1.4             | 1.4               | provides auditing functionality
(49 Zeilen)
```

Abbildung 4 - PostgreSQL Extension mit installiertem Plugin "pgaudit"

A.4.7.2.3 Konfiguration

In der Postgres-Config wird folgende Konfiguration hinzugefügt:

```
pgaudit.log_catalog = on
pgaudit.log = 'all'
pgaudit.log_relation = 'on'
pgaudit.log_parameter = 'on'
```

Damit werden alle Events protokolliert.

Standardmäßig schreibt Postgres die Logs ins Datenverzeichnis in den Unterordner „log“.

A.4.7.2.4 Protokoll-Beispiel

Der nachfolgende Auszug ist das Ergebnis folgender Ausführungen auf der Datenbank:

- Start des Datenbankservices
- Einloggen als „postgres“
- Abfrage der Daten: Select * from Studenten;
- Ausloggen von der Datenbank

```
2020-05-03 16:11:35.655 CEST [5358] LOG:  Datenbanksystem wurde am 2
020-05-03 16:11:35 CEST heruntergefahren
2020-05-03 16:11:35.724 CEST [5355] LOG:  Datenbanksystem ist berei
t, um Verbindungen anzunehmen
2020-05-03 16:11:51.700 CEST [5374] LOG:  AUDIT: SESSION,1,1,READ,S
ELECT,,,SELECT NOW(),<none>
2020-05-03 16:11:51.718 CEST [5374] LOG:  AUDIT: SESSION,2,1,READ,S
ELECT,,,SELECT VERSION(),<none>
2020-05-03 16:11:51.746 CEST [5374] LOG:  AUDIT: SESSION,3,1,MISC,S
ET,,,SET statement_timeout TO 30000,<none>
2020-05-03 16:11:51.766 CEST [5374] LOG:  AUDIT: SESSION,4,1,READ,S
ELECT,,,SELECT EXTRACT(EPOCH FROM CURRENT_TIMESTAMP - pg_postmaster
_start_time())::INTEGER,<none>
2020-05-03 16:11:51.785 CEST [5374] LOG:  AUDIT: SESSION,5,1,MISC,S
ET,,,"SET search_path TO E'uni', E'public'",<none>
2020-05-03 16:11:51.821 CEST [5374] LOG:  AUDIT: SESSION,6,1,READ,S
ELECT,TABLE,uni.studenten,SELECT * FROM Studenten,<none>
2020-05-03 16:11:51.858 CEST [5374] LOG:  AUDIT: SESSION,7,1,READ,S
ELECT,,,SELECT 16475::regclass,<none>
2020-05-03 16:11:51.883 CEST [5374] LOG:  AUDIT: SESSION,8,1,READ,S
ELECT,,,SELECT 16475::regclass,<none>
2020-05-03 16:11:51.947 CEST [5374] LOG:  AUDIT: SESSION,9,1,READ,S
ELECT,,,SELECT 16475::regclass,<none>
2020-05-03 16:11:51.962 CEST [5374] LOG:  AUDIT: SESSION,10,1,READ,
SELECT,,,SELECT 16475::regclass,<none>
2020-05-03 16:12:38.739 CEST [5355] LOG:  schnelles Herunterfahren v
erlangt
```

```
2020-05-03 16:12:39.174 CEST [5355] LOG:  etwaige aktive Transaktio
nen werden abgebrochen
```

Angehängte Datei:

- PostgreSQL_Protokollierung.docx

A.4.8 APP.4.3.A9

A.4.8.1 MySQL

A.4.8.1.1 Datensicherung mit MySQL Server

In diesem Dokument wird beschrieben, ob und wie im Oracle MySQL Community Server die folgenden Datensicherungsarten möglich sind:

- Volle Datenbank-Sicherung
- Inkrementelle Datenbank-Sicherung
- Sicherung der Transaktionen

Zusätzlich wird das System dahingegen überprüft, ob aus dem DBS heraus eine zeitgesteuerte Sicherung möglich ist.

A.4.8.1.2 Volle Datensicherung

MySQL bietet zur Datensicherung zwei Dienstprogramme: „mysqldump“ und „mysqlpump“. Zwar ist „mysqldump“ eine neuere Version von „mysqldump“, die mehr Funktionen mit sich bringt, so referenziert Oracle in seinen Manuals zu MySQL meistens die ältere „mysqldump“-Version. Daher wird sich im Folgenden auf diese Variante bezogen.

Eine volle Datensicherung aller Datenbank kann über folgenden Befehl erfolgen:

```
mysqldump --all-databases --single-transaction --quick --lock-table
s=false > full-backup-$(date +%F).sql -u root -p
```

Damit wird ein volles Backup mit dem Dateinamen „full-backup-YYYY-MM-DD.sql“ erzeugt.

Zum Wiedereinspielen eines solchen SQL-Format Backups ist folgender Befehl im Terminal abzusetzen:

```
mysql < full-backup-2020-05-06.sql
```

Damit wird die Sicherung aus der angegebenen Datei ins System eingespielt.

A.4.8.1.3 Inkrementelles Backup

Offiziell unterstützt MySQL Community Server kein inkrementelles Backup – die Enterprise-Version hingegen schon.

Mithilfe von Binary-Logs kann aber trotzdem eine inkrementelle Sicherung durchgeführt werden. Binary Logs sichern speicherverändernde Vorgänge, wie z.B. Update-, Alter- und Delete-Statements. Hierzu müssen die Binary-Logs in der Config-Datei aktiviert werden:

```
[mysqld]
log_bin               = /var/log/mysql/mysql-bin.log      # Speiche
rort und Dateiname
expire_logs_days      = 2                          # Ablaufdatum de
r Logs
max_binlog_size       = 100M                       # Erlaubte Größe
pro Inkrement
binlog-do-db   = Uni                         # Zu sichernde Datenbank
```

Nach einem Server-Neustart wird im Log-Ordner die erste Inkrement-Datei, sowie eine Index-Datei erzeugt. Die Index-Datei enthält Informationen über alle erzeugten Inkremente. Fortlaufend werden alle Inkremente mit einer eindeutigen Nummer versehen (mysql-bin.000001, mysql-bin.000002, …).

Neue Inkremente werden erzeugt, wenn:

- Der Server (neu-)startet
- Die Logs geflusht werden
- Die maximale Dateigröße für ein Inkrement erreicht wurde

Das Zurückspielen von Binary-Logs in das Live-System erfolgt über den Befehl:

```
mysqlbinlog mysql-bin.000001 | mysql -u root -p
```

Damit wird der Inhalt der angegebenen Log-Datei über den root-Nutzer in die Datenbank gespielt.

A.4.8.1.4 Sicherung der Transaktionen

Mithilfe von Binary-Logs sind alle speicherverändernden Transaktionen gesichert. MySQL bietet mit den „General Logs“ noch eine weitere Speicherung von Transaktionen an. Hierbei werden alle Vorgänge, die das DBS betreffen, gespeichert. D.h. auch alle Select-Statements, aber auch alle Connect- und Disconnect-Versuche. Somit wird die Zieldatei des General Logs schnell sehr groß und es werden Daten erzeugt, die für den weiteren Betrieb nicht relevant sind. Auf diesen zwei Gründen ist der General-Log nicht für den Dauerbetrieb geeignet, sondern für Fehlersuche und andere administrative Arbeiten.

Zur Aktivierung der des General Logs müssen in der MySQL-Configdatei folgende Attribute gesetzt werden:

```
general_log_file      = /var/log/mysql/mysql.log   # Pfad und Datei
ziel
general_log           = 1                                  # General
Log aktivieren
```

Aus den zuvor genannten Gründen wird diese Funktion in der bereitgestellten Beispieldatenbank nicht aktiviert.

A.4.8.1.5 Zeitgesteuerte Sicherung

MySQL bietet aus dem DBS heraus keine Funktionen für eine automatische, zeitgesteuerte Sicherung.

Angehängte Datei:

- MySQL_Datensicherung.docx
- full-backup-2020-05-06.sql
- mysql-bin.index
- mysql-bin.000001
- mysql-bin.000002

A.4.8.2 PostgreSQL

A.4.8.2.1 Datensicherung mit PostgreSQL Server

In diesem Dokument wird beschrieben, ob und wie in PostgreSQL Server die folgenden Datensicherungsarten möglich sind:

- Volle Datenbank-Sicherung
- Inkrementelle Datenbank-Sicherung
- Sicherung der Transaktionen

Zusätzlich wird das System dahingegen überprüft, ob aus dem DBS heraus eine zeitgesteuerte Sicherung möglich ist.

A.4.8.2.2 Volle Datensicherung

Zur vollen Datensicherung bietet PostgreSQL zwei Dienstprogramme an: „pg_dump“[37] um eine einzelne Datenbank als SQL-File zu sichern (bzw. „pg_dumpall“[38] um alle Datenbanken in einem Cluster zu sichern) und „pg_basebackup“[39] um alle Daten als Binärdateien zu sichern.

Eine volle Datensicherung mittels „pg_dumpall“ aller Datenbank kann über folgenden Befehl erfolgen:

```
pg_dumpall > full-backup-$(date +%F).sql
```

Damit wird ein volles Backup mit dem Dateinamen „full-backup-YYYY-MM-DD.sql“ erzeugt.

Zum Wiedereinspielen eines solchen SQL-Format Backups ist folgender Befehl im Terminal abzusetzen:

```
psql -f full-backup-2020-05-08.sql postgres
```

Damit wird die Sicherung aus der angegebenen Datei ins System eingespielt.

Die Binärdateien können über folgenden Befehl gesichert werden:

```
pg_basebackup -D /usr/local/pgsql/data -Ft  # erstellt ein Backup al
s tar gepackt
```

---

37 https://www.postgresql.org/docs/10/app-pgdump.html

38 https://www.postgresql.org/docs/9.1/app-pg-dumpall.html

39 https://www.postgresql.org/docs/10/app-pgbasebackup.html

Das Backup ist eine Sicherung aller notwendigen Dateien des „data“-Verzeichnis der Postgres-Instanz. Um dieses wieder her zu stellen, muss die erstellte Backup-Tar-Datei in das „data“-Verzeichnis entpackt werden. Hierfür ist es wichtig, dass der postgres-Service beendet wurde, außerdem sollte das Entpacken über den „postgres“-Nutzer im System erfolgen, damit die Berechtigungen korrekt gesetzt sind.

```
tar -xf base.tar -C /postgres/
```

A.4.8.2.3 Sicherung der Transaktionen

Zur Point-In-Time (PIT) Wiederherstellung bietet PostgreSQL die sogenannten „Write Ahead Logs“ (WAL). Hier wird zwischen zwei unterschiedlichen WAL-Arten unterschieden: „Internal WAL“ und „Archive WAL“.

Die Funktion für „internal WAL“ ist standardmäßig aktiviert und speichert die Log-Dateien im „data“-Verzeichnis im Unterordner „pg_wal“. Hier werden alle Änderungen zur Datenbasis („UPDATE“, „ALTER“, „DELETE“, …) gesichert. Standardmäßig hat eine WAL-Datei eine Größe von 16MB, anschließend wird eine neue angelegt. Bei der Ausführung von „pg_backbackup“ werden auch die WAL-Dateien gesichert. Diese sind zur konsistenten Speicherung der Daten, während einer Transaktion. D.h. hier werden die Daten aller Transaktionen zwischengesichert, die in einer festgelegten Zeitspanne („checkpoint_timeout“) passieren. Die Daten werden bei einem Server-Neustart mit den Datenbasis in der Datenbank validiert.

Die „Archive WAL“-Funktion ist zur dauerhaften Speicherung der Transaktionen und einer Wiederherstellung hieraus geeignet. Um diese Funktion zu aktivieren, muss in der postgres.config folgende Einstellungen vorgenommen werden:

```
archive_mode = 'on'
archive_command = 'cp %p /var/lib/pgsql/12/archives/%f'
# gibt an, wo die Archive gesichert werden
```

Um eine Wiederherstellung zu starten, muss der Server im Recovery-Modus gestartet werden. Hierzu wird eine Datei mit dem Namen „recovery.signal“ im „data“-Verzeichnis der Installation angelegt. Wird der Server nun gestartet, aktiviert sich der Wiederherstellungsmodus und liest die Archiv-Dateien aus dem Verzeichnis ein.

A.4.8.2.4 Zeitgesteuerte Sicherung

PostgreSQL bietet aus dem DBS heraus keine Funktionen für eine automatische, zeitgesteuerte Sicherung.

Angehängte Datei:

- PostgreSQL_Datensicherung.docx
- full-backup-2020-05-08.sql
- base.tar
- pg_wal.tar

A.4.8.3 MongoDB

A.4.8.3.1 Datensicherung mit MongoDB Server

In diesem Dokument wird beschrieben, ob und wie im MongoDB Community Server die folgenden Datensicherungsarten möglich sind:

- Volle Datenbank-Sicherung
- Inkrementelle Datenbank-Sicherung
- Sicherung der Transaktionen

Zusätzlich wird das System dahingegen überprüft, ob aus dem DBS heraus eine zeitgesteuerte Sicherung möglich ist.

A.4.8.3.2 Volle Datensicherung

Das Dienstprogramm der MongoDB, um Datensicherungen zu erzeugen, wird „mongodump“[40] genannt.

Eine volle Datensicherung aller Datenbank kann über folgenden Befehl erfolgen:

```
mongodump --out /var/log/mongodb/backup/
```

[40] https://docs.mongodb.com/manual/reference/program/mongodump/

```
# erzeugt eine Sicherung am angegeben Verzeichnis
```

Zum Wiedereinspielen eines solchen BSON-Format Backups ist folgender Befehl im Terminal abzusetzen:

```
mongorestore /var/log/mongodb/backup/
# lädt eine Sicherung am angegeben Verzeichnis
```

A.4.8.3.3 Inkrementelles Backup

In der freien Version unterstützt MongoDB kein inkrementelles Backup. In zusätzlichen, kostenpflichtigen Diensten (z.B. MongDB Atlas [53], MongoDB Cloud Manager [54]) bieten erweiterte Backupmethoden, wie Point-in-Time Wiederherstellung. [41]

A.4.8.3.4 Sicherung der Transaktionen

Eine Sicherung der Transaktionen als solche ist derzeit nicht implementiert. Ein Weg, um alle Transaktionen zu sichern gibt es dennoch. MongoDB unterstützt das Erstellen von Replikationen für den Betrieb verteilter Datenbank-Cluster. Der Datenaustausch erfolgt über Log-Dateien, genannt „Oplogs“ (Operative Logs). In diesen Oplogs sind alle ändernden Vorgänge (Transaktionen) gespeichert. Im Normalfall werden die diese von dem Master-Server an seine Replikationen verteilt, so dass alle den gleichen Datenstand haben. Wenn bei einem Einzel-Server die Replikation aktiviert wird, werden diese Oplogs ebenso gesichert. Mithilfe dieser Dateien ist es möglich, eine Point-in-Time (PIT)-Wiederherstellung zu ermöglichen. Dazu dient wiederrum das Dienstprogramm „mongorestore“.

Zum Aktivieren der Oplogs muss der MongoDB-Server mit dem Parameter „replSet=rs0“ gestartet werden. Damit wird angegeben, dass dieser Server die Replikation 0 ist. Wichtig ist, hierbei nochmal explizit die Konfigurationsdatei mit anzugeben, da es vorkommen kann, dass der Server ohne eingestellte Konfiguration als neue Instanz startet.

```
mongod --config=/etc/mongo.config --replSet=rs0
```

[41] https://docs.mongodb.com/manual/core/backups/

Nun die MongoDB-Shell mit dem Befehl „mongo“ starten. Nach dem Einloggen sollte die Shell folgenden Präfix haben:

```
---

rs0:PRIMARY>
```

Hiermit ist angezeigt, dass der Nutzer sich in der Replikation „rs0“ angemeldet hat, welche sich im Primary-Node befindet.

Zum Aktivieren der Oplogs nun folgenden Befehl eingeben:

```
rs.initiate()
```

Es sollte eine Rückmeldung mit dem Status „OK“ kommen:

```
rs0:PRIMARY> rs.initiate()
{
        "operationTime" : Timestamp(1589095960, 1),
        "ok" : 0,
```

Zum Überprüfen ob die Oplogs erzeugt werden, folgende Befehle eingeben:

```
use local
db.oplog.rs.find();
```

Als Rückmeldung sollte in etwa folgende Ausgabe erscheinen:

```
rs0:PRIMARY> use local
switched to db local
rs0:PRIMARY> db.oplog.rs.find();
{ "ts" : Timestamp(1589093669, 1), "h" : NumberLong(0), "v" : 2, "op" : "n", "ns" : "", "wall" : ISODate("2020-05-
{ "ts" : Timestamp(1589093669, 3), "t" : NumberLong(1), "h" : NumberLong(0), "v" : 2, "op" : "c", "ns" : "config.$
0T06:54:30.208Z"), "o" : { "create" : "transactions", "idIndex" : { "v" : 2, "key" : { "_id" : 1 }, "name" : "_id_
{ "ts" : Timestamp(1589093670, 1), "t" : NumberLong(1), "h" : NumberLong(0), "v" : 2, "op" : "n", "ns" : "", "wall
```

Nachdem Ausloggen aus der MongoDB-Shell (Befehl: „exit“) können die Oplogs mittels den Dienstprogrammes „mongodump“ exportiert werden:

```
mongodump -d local -c oplog.rs -o ~/oplog
```

Hierbei muss angemerkt werden, dass nur die Oplogs nur je Datenbank exportiert werden können und nicht für alle zugleich. Der Parameter „-d“ gibt den Datenbanknamen an. Mit diesem Befehl wird die Datenbank „local“ in den Unterordner „/oplog“ exportiert.

Über das Dienstprogramm „bsondump“ können die Oplogs betrachtet werden:

```
bsondump ~/oplog/oplog.bson
```

Die Ausgabe sieht wie folgt aus:

```
[root@centos ~]# bsondump ~/oplog2/oplog.bson
{"ts":{"$timestamp":{"t":1589093669,"i":1}},"h":{"$numberLong":"0"},"v":{"$numberInt":"2"},"op":"n","ns":"","wall"
:{"$date":{"$numberLong":"1589093669867"}},"o":{"msg":"initiating set"}}
{"ts":{"$timestamp":{"t":1589093669,"i":3}},"t":{"$numberLong":"1"},"h":{"$numberLong":"0"},"v":{"$numberInt":"2"}
,"op":"c","ns":"config.$cmd","ui":{"$binary":{"base64":"sik97t18QWq6v61nGS0D+w==","subType":"04"}},"wall":{"$date"
:{"$numberLong":"1589093670208"}},"o":{"create":"transactions","idIndex":{"v":{"$numberInt":"2"},"key":{"_id":{"$n
umberInt":"1"}},"name":"_id_","ns":"config.transactions"}}}
```

In dieser Datei sind alle Vorgänge gesichert. Wichtig sind hierbei die Variablen „t“ und „i“, da diese Zusammen der eindeutige Identifizierter des Vorgangs sind. Bei der Wiederherstellung werden diese Parameter angegeben und alle Vorgänge bis zu den gewählten werden ins System übertragen.

Zur Wiederherstellung der Vorgänge wird nun das Programm mongorestore genutzt:

```
mongorestore --oplogReplay --oplogLimit 1484002910:1 ~/oplog
```

Der Parameter „oplogReplay“ gibt an, dass die Oplogs wieder eingespielt werden sollen. Das Oplog-Limit sind die Werte „t“ und „i“ die zuvor aus dem BSONdump herausgelesen wurden. Das Format ist „t:i“.

A.4.8.3.5 Zeitgesteuerte Sicherung

MongoDB bietet aus dem DBS heraus keine Funktionen für eine automatische, zeitgesteuerte Sicherung.

Angehängte Datei:

- MongoDB_Datensicherung
- backup.zip
- oplog.zip

A.4.9 APP.4.3.A12

Stammdatenblatt Datenbanken

Datenbankbezeichner		
Fachlich verantwortliche Person	Technisch verantwortliche Person	
Sicherheit verantwortliche Person		
Zugelassene Administratoren	*Berechtigung*	
Zweck der Datenbank / Zugehörigkeit		Kostenstelle
Laufzeit	Angelegt am	Löschung am
Datenbanktyp		
Hersteller	Modell	
Version	Letzter Patch	
Standort / Cloud Subscribtion		
Eingesetzte Konfiguration:		
Abweichungen zur Standard-Konfiguration		
Enthaltene DB-Links		
Ziel	*Zweck*	
Backups		
Rhythmus	*Aufbewahrungsfrist*	

Angehängte Dateien:

- DB_Stammdatenblatt.docx
- MySQL / my.cnf
- PostgreSQL / postgresql.conf
- PostgreSQL / pg_hba.conf
- PostgreSQL / pg_ident.conf
- MongoDB / mongod.conf

A.4.10 APP.4.3.A13

A.4.10.1.1 Restriktive Handhabung von Datenbanklinks

Datenbank-Links sind ein Konzept zur unidirektionalen Verbindung von einem physikalischen Datenbankserver mit einem zweiten. Diese Funktion ist vor allem bekannt aus dem DBS Oracle. Hierbei werden in einer Datenbank feste Pseudonyme für die externen Quellen angelegt, über das auf die Daten per Queries zugegriffen werden kann. Diese Links können verschiedene Restriktionslevel aufweisen: private (nur der anlegende User kann den Link sehen und nutzen), public (alle Nutzer der Datenbank können den Link sehen und nutzen) und global (der DB-Link kann über die Datenbank hinaus eingesehen und genutzt werden). [66] In anderen Datenbanksystemen ist diese Funktionalität zumeist nicht implementiert. Von den untersuchten Datenbanken weißt nur PostgreSQL eine entsprechende Funktion auf, die sich über ein Plugin „dblink“ nachinstallieren lässt. Mithilfe dieses Plugins ist es aber lediglich möglich eine Query in folgendem Format auf einem Remotesystem auszuführen:

```
SELECT * FROM dblink('dbname=postgres options=-csearch_path=',
                     'select proname, prosrc from pg_proc')
  AS t1(proname name, prosrc text) WHERE proname LIKE 'bytea%';
```

Hierbei gibt es keine dauerhafte Verbindung zum Remotesystem, die mitgesichert werden kann oder muss. Auch eine Unterscheidung von Sichtbarkeiten ist hiermit nicht gegeben.

Eine Möglichkeit, eine solche Query dauerhaft in der PostgreSQL-Datenbank zu verankern, ist das Anlegen einer View, mit einer dieser Query als Quelle. Views werden standardmäßig mit dem System gesichert. [131]

Angehängte Dateien:

- PostgreSQL_DB-Links.docx

A.4.11 APP.4.3.A14

A.4.11.1 MySQL

A.4.11.1.1 Überprüfung einer Datensicherung mit MySQL Server

Für MySQL-Server wurden die Sicherungsarten „mysqldump“, „binlog“ und „General Log“ identifiziert. Für „mysqldump“ existiert ein impliziter Test, indem das Backup in ein MySQL-System eingespielt wird, „binlog“ bietet eine Checksummenüberprüfung mit der zyklischen Redundanzprüfung CRC32. Die „General Logs“ bieten keine Validierungsoption.

A.4.11.1.1.1 Mysqldump

Das Dienstprogramm „mysqldump“ erzeugt Textdateien im SQL-Format. Es existiert derzeit keine Funktionalität, um die Integrität der Sicherung explizit zu testen. Um die generelle Funktionalität der Sicherung zu überprüfen, ist ein impliziter Test, die Backupdatei in ein Testsystem einzuspielen.

Zum Wiedereinspielen eines solchen SQL-Format Backups ist folgender Befehl im Terminal abzusetzen:

```
mysql < full-backup-2020-05-06.sql
```

Damit wird die Sicherung aus der angegebenen Datei ins System eingespielt. Sollte sich der Server nun fehlerfrei starten lassen, ist die Sicherungsdatei an sich funktional. Die Datenintegrität ist hiermit noch nicht überprüft.

A.4.11.1.1.2 binlog

Das Programm „binlog“ bietet eine integrierte Checksummenüberprüfung via CRC32. Standardmäßig ist die Generierung dieser Prüfsumme aktiviert und wird daher erzeugt. Mit folgendem Befehl kann ein die Prüfung ausgeführt werden, wobei „binlog.000002“ die zu prüfende Backupdatei angibt.

```
mysqlbinlog --verify-binlog-checksum binlog.000002
```

Ein Fehlerfreier Durchlauf sieht wie folgt aus:

```
[root@centos mysql]# mysqlbinlog --verify-binlog-checksum binlog.000002
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=1*/;
/*!50003 SET @OLD_COMPLETION_TYPE=@@COMPLETION_TYPE,COMPLETION_TYPE=0*/;
DELIMITER /*!*/;
# at 4
#200502 12:34:02 server id 1  end_log_pos 124 CRC32 0x7aa8a6cc  Start: binlog v
4, server v 8.0.17 created 200502 12:34:02 at startup
ROLLBACK/*!*/;
BINLOG '
mkytXg8BAAAAeAAAAHwAAAAAAAQAOC4wLjE3AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAACaTK1eEwANAAgAAAAABAAEAAAAYAAEGggAAAAICAgCAAAACgoKKioAEjQA
CgHMpqh6
'/*!*/;
# at 124
#200502 12:34:02 server id 1  end_log_pos 155 CRC32 0xdd7300f8  Previous-GTIDs
# [empty]
# at 155
#200502 12:52:23 server id 1  end_log_pos 178 CRC32 0x85cdeb1f  Stop
SET @@SESSION.GTID_NEXT= 'AUTOMATIC' /* added by mysqlbinlog */ /*!*/;
DELIMITER ;
# End of log file
/*!50003 SET COMPLETION_TYPE=@OLD_COMPLETION_TYPE*/;
/*!50530 SET @@SESSION.PSEUDO_SLAVE_MODE=0*/;
[root@centos mysql]#
```

A.4.11.1.1.3 General Log

Keine Integritätsprüfung vorhanden.

Angehängte Dateien:

- MySQL_Datensicherung_Validierung.docx

A.4.11.2 PostgreSQL

A.4.11.2.1 Datensicherung mit PostgreSQL Server

Die Sicherungsprogramme „pg_dump“ bzw. „pg_dumpall“ und mittels WAL-Dateien bietet für PostgreSQL umfangreiche Sicherungsmaßnahmen. Für „pg_dump“ existiert ein impliziter Test, indem das Backup in ein PostgreSQL-System eingespielt wird. WAL-Dateien enthalten zwar eine CRC32-Prüfsumme, jedoch existiert derzeit kein Weg, diese Dateien manuell gegen diese Prüfsumme zu testen.

A.4.11.2.1.1 Pg_dump

Das Dienstprogramm „pg_dump“ erzeugt Textdateien im SQL-Format. Es existiert derzeit keine Funktionalität, um die Integrität der Sicherung explizit zu testen. Um die generelle Funktionalität der Sicherung zu überprüfen, ist ein impliziter Test, die Backupdatei in ein Testsystem einzuspielen.

Zum Wiedereinspielen eines solchen SQL-Format Backups ist folgender Befehl im Terminal abzusetzen. „full-backup-2020-05-08.sql“ gibt hierbei die Sicherungsdatei an.

```
psql -f full-backup-2020-05-08.sql postgres
```

Damit wird die Sicherung aus der angegebenen Datei ins System eingespielt. Sollte sich der Server nun fehlerfrei starten lassen, ist die Sicherungsdatei an sich funktional. Die Datenintegrität ist hiermit noch nicht überprüft.

A.4.11.2.1.2 Write Ahead Log

Derzeit keine Validierung implementiert.

Angehängte Dateien:

- PostgreSQL_Datensicherung_Validierung.docx

A.4.12 APP.4.3.A16

A.4.12.1 MySQL

A.4.12.1.1 Verschlüsselung der Datenbankanbindung zum MySQL Server

MySQL implementiert das TSL-Protokoll auf Basis der Bibliothek OpenSSL [73]in Version 1.1.1. Von MySQL werden folgende TSL-Versionen unterstützt: TLSv1, TLSv1.1, TLSv1.2 und TLSv1.3. Die jeweiligen Versionen können vom Datenbankadministrator in der Konfiguration festgelegt werden.

A.4.12.1.1.1 Prüfen des aktuellen SSL/TLS-Status

Zum Prüfen des aktuellen Stati, als root im MySQL-Serverservice anmelden. Anschließend folgenden Befehl ausführen:

```
mysql> show variables like '%ssl%';
```

Die Ausgabe kann wie folgt aussehen:

Wenn SSL/TLS aktiv ist	Wenn SSL/TLS nicht aktiv ist

`mysql> show variables like '%ssl%';`
`+--------------------+-----------------+`
`| Variable_name | Value |`
`+--------------------+-----------------+`
`| have_openssl | YES |`
`| have_ssl | YES |`
`| mysqlx_ssl_ca | |`
`| mysqlx_ssl_capath | |`
`| mysqlx_ssl_cert | |`
`| mysqlx_ssl_cipher | |`
`| mysqlx_ssl_crl | |`
`| mysqlx_ssl_crlpath | |`
`| mysqlx_ssl_key | |`
`| ssl_ca | ca.pem |`
`| ssl_capath | |`
`| ssl_cert | server-cert.pem |`
`| ssl_cipher | |`
`| ssl_crl | |`
`| ssl_crlpath | |`
`| ssl_fips_mode | OFF |`
`| ssl_key | server-key.pem |`
`+--------------------+-----------------+`
`17 rows in set (0,01 sec)`
	`mysql> show variables like '%ssl%'`
	`-> ;`
	`+--------------------+-----------`
	`| Variable_name | Value |`
	`+--------------------+-----------`
	`| have_openssl | DISABLED |`
	`| have_ssl | DISABLED |`
	`| mysqlx_ssl_ca | |`
	`| mysqlx_ssl_capath | |`
	`| mysqlx_ssl_cert | |`
	`| mysqlx_ssl_cipher | |`
	`| mysqlx_ssl_crl | |`
	`| mysqlx_ssl_crlpath | |`
	`| mysqlx_ssl_key | |`
	`| ssl_ca | |`
	`| ssl_capath | |`
	`| ssl_cert | |`
	`| ssl_cipher | |`
	`| ssl_crl | |`
	`| ssl_crlpath | |`
	`| ssl_fips_mode | OFF |`
	`| ssl_key | |`
	`+--------------------+-----------`
	`17 rows in set (0.00 sec)`

Mit dem Befehl

```
\s
```

Kann der aktuelle Status samt Verschlüsselungsverfahren angezeigt werden:

Wenn SSL/TLS aktiv ist	Wenn SSL/TLS nicht aktiv ist

`mysql> \s`
`--------------`
`mysql  Ver 8.0.20 for Linux on x86_64 (My`
 	
`Connection id:  10`
`Current database:`
`Current user:  root@localhost`
`SSL:  Cipher in use is`
`Current pager:  stdout`
`Using outfile:  ''`
`Using delimiter:  ;`
`Server version:  8.0.20 MySQL Comm`
`Protocol version:  10`
`Connection:  127.0.0.1 via TCP`
`Server characterset:  utf8mb4`
`Db  characterset:  utf8mb4`
`Client characterset:  utf8mb4`
`Conn.  characterset:  utf8mb4`
`TCP port:  3306`
`Binary data as:  Hexadecimal`
`Uptime:  1 min 27 sec`
 	
`Threads: 2  Questions: 11  Slow queries:`
`3  Queries per second avg: 0.126`
`--------------`
	`mysql> \s`
	`--------------`
	`mysql  Ver 8.0.20 for Linux on x86_64`
	
	`Connection id:  8`
	`Current database:`
	`Current user:  root@localhost`
	`SSL:  Not in use`
	`Current pager:  stdout`
	`Using outfile:  ''`
	`Using delimiter:  ;`
	`Server version:  8.0.20 MySQL C`
	`Protocol version:  10`
	`Connection:  Localhost via`
	`Server characterset:  utf8mb4`
	`Db  characterset:  utf8mb4`
	`Client characterset:  utf8mb4`
	`Conn.  characterset:  utf8mb4`
	`UNIX socket:  /var/lib/mysql`
	`Binary data as:  Hexadecimal`
	`Uptime:  1 min 30 sec`
	
	`Threads: 2  Questions: 6  Slow queries`
	`les: 53  Queries per second avg: 0.066`
	`--------------`

A.4.12.1.1.2 Generieren von Schlüsseln

Zum Generieren der nötigen Schlüssel, liefert der MySQL-Server ein Programm mit, welches die Zertifikate und Schlüssel generiert. Mit dem Terminal sollte in den Data-Ordner der MySQL-Instanz gewechselt werden. Dann werden die Dateien dort mit folgendem Befehl erzeugt:

```
sudo mysql_ssl_rsa_setup --uid=mysql
```

der Parameter „--uid=mysql“ gibt an, für welchen Linux-User die Dateien erzeugt werden sollen, damit diese die richtigen Berechtigungen haben.

Es werden drei Dateien erzeugt:

Dateiname	*Nutzen*
ca-key.pem	Privater Schlüssel der Zertifizierungs-Autorität
ca.pem	Zertifikat der Zertifizierungs-Autorität
server-key.pem	Privater Schlüssel des Servers
client-key.pem	Privater Schlüssel des Clients

A.4.12.1.1.3 Konfigurieren der Verbindung

In der Mysql-config-Datei müssen folgende Einstellungen vorgenommen werden:

```
# SSL/TSL
ssl
ssl-ca=/var/lib/myqsl/ca-cert.pem
ssl-cert=/var/lib/mysql/server-cert.pem
ssl-key=/var/lib/mysql/server-key.pem
```

Neure Versionen (ab 5.7) von MySQL erkennen die Existenz der passenden Zertifikate automatisch und setzen diese Einstellungen beim Systemstart.

Mit dem Setzen der Einstellung

```
require_secure_transport = ON
```

werden nur noch verschlüsselte Verbindungen vom Server zugelassen. Verbindungen ohne gültiges Zertifikat werden abgelehnt.

Angehängte Dateien:

- MySQL_VerschluesselteVerbindung.docx
- cert.zip

A.4.12.2 PostgreSQL

A.4.12.2.1 Verschlüsselung der Datenbankanbindung zum PostgreSQL Server

Der PostgreSQL-Server nutzt OpenSSL und greift hierbei auf die installierte Version des Hostsystems zurück. Die unterstützen SSL/TLS-Versionen sind daher abhängig von der installierten OpenSSL-Version [80] [81]. Vorausgesetzt das die aktuelle Version 1.1.1 installiert ist, unterstützt Postgres die Versionen TLSv1, TLSv1.1, TLSv1.2 und TLSv1.3 [82].

A.4.12.2.1.1 Prüfen des aktuellen SSL/TLS-Status

Zum Prüfen des aktuellen Status, kann in der *postgresql.conf* nachgeschaut werden.

Im Abschnitt „- SSL –" sollte das standardmäßig der Parameter „ssl=off“ gesetzt sein.

- „ssl=on“ bedeutet, SSL ist aktiviert
- „ssl=off“ bedeutet, SSL ist deaktiviert

Zusätzlich sollte in der Datei *pg_hba.conf* geprüft werden, für welche Nutzer SSL verpflichtend eingestellt wurde.

Beispiel:

```
hostssl all             remoteuser      0.0.0.0/0               scram-sha-256 clientcert=1
```

In dieser Zeile ist für den User „remoteuser“ von jeder eingehenden IP-Adresse ein Client-Zertifikat verlangt.

A.4.12.2.1.2 Generieren von Schlüsseln

PostgreSQL bietet keine Dienstprogramme zur Erzeugung von Zertifikaten. Es können die Tools von OpenSSL genutzt werden. Der Vorgang wird an dieser Stelle nicht weiter beschrieben.

Es sollten mehrere Dateien erzeugt werden:

Dateiname	*Nutzen*
ca-key.pem	Privater Schlüssel der Zertifizierungs-Autorität

ca.pem	Zertifikat der Zertifizierungs-Autorität
server-key.pem	Privater Schlüssel des Servers
client-key.pem	Privater Schlüssel des Clients

Im Beispiel werden die Dateien in den Ordner */etc/ssl/postgresql/* hinterlegt.

A.4.12.2.1.3 Konfigurieren der Verbindung

In der *postgresql.conf* müssen folgende Einstellungen gesetzt werden:

```
# - SSL -

ssl = on
ssl_ca_file = '/etc/ssl/postgresql/ca-cert.pem'
ssl_cert_file = '/etc/ssl/postgresql/server-cert.pem'
#ssl_crl_file = ''
ssl_key_file = '/etc/ssl/postgresql/server-key.pem'
#ssl_ciphers = 'HIGH:MEDIUM:+3DES:!aNULL' # allowed SSL ciphers
#ssl_prefer_server_ciphers = on
#ssl_ecdh_curve = 'prime256v1'
#ssl_min_protocol_version = 'TLSv1'
#ssl_max_protocol_version = ''
#ssl_dh_params_file = ''
#ssl_passphrase_command = ''
#ssl_passphrase_command_supports_reload = off
```

.

In der Abbildung sind noch weitere Einstellungen zu sehen, die in dem Beispiel auskommentiert sind. Hier können u.a. bestimmte Verschlüsselungsverfahren und TSL-Versionen vorausgesetzt werden.

Nach dem Speichern der Datei, muss der Service neugestartet werden.

Angehängte Dateien:

- PostgreSQL_VerschluesselteVerbindung.docx
- cert.zip

A.4.12.3 MongoDB

A.4.12.3.1 Verschlüsselung der Datenbankanbindung zum MongoDB Server

Der MongoDB-Server greift auf installierte, native SSL/TLS-Bibliotheken des Hostsystems zu, wobei es dies abhängig vom Betriebssystem macht. Bei Microsoft Windows wird „Secure Channel“ (Schannel) genutzt, bei Linux/BSD-Systemen „OpenSSL“ und bei macOS „Secure Transport“. [83] Entsprechend eines Linux Systems mit installierter OpenSSL-Version 1.1.1, werden die Versionen TLSv1, TLSv1.1, TLSv1.2 und TLSv1.3 unterstützt [82].

A.4.12.3.1.1 Prüfen des aktuellen SSL/TLS-Status

Zum Prüfen des aktuellen Status, kann in der *mongodb.conf* nachgeschaut werden.

Hier muss für den Bereich „net“ eine Untergruppe „tls“ geben, die die entsprechenden Verweise auf eine Certificate Autorithy und den Server-Key hält.

Beispiel:

```
net:
  tls:
    mode: requireTLS
    certificateKeyFile: /etc/ssl/mongodb/mongodb.pem
    CAFile: /etc/ssl/mongodb/rootCA.pem
```

A.4.12.3.1.2 Generieren von Schlüsseln

MongoDB bietet keine Dienstprogramme zur Erzeugung von Zertifikaten. Es können die Tools von OpenSSL genutzt werden. Der Vorgang wird an dieser Stelle nicht weiter beschrieben.

WICHTIG: Bei der Erzeugung des Schlüsselpaares für den Server muss der anzugebene „Common Name“ der IP-Adresse bzw. dem Hostnamen des Servers entsprechen! Ansonsten kann kein Client die Verbindung aufnehmen!

Es sollten mehrere Dateien erzeugt werden:

Dateiname	*Nutzen*
rootCA.key	Privater Schlüssel der Zertifizierungs-Autorität
rootCA.pem	Zertifikat der Zertifizierungs-Autorität
mongodb.pem	Privater Schlüssel des Servers
mongodb.key	Privater Schlüssel des Clients

Im Beispiel werden die Dateien in den Ordner */etc/ssl/mongodb/* hinterlegt.

A.4.12.3.2 Konfigurieren der Verbindung

In der *mongodb.conf* müssen folgende Einstellungen gesetzt werden:

```
net:
  tls:
    mode: requireTLS
    certificateKeyFile: /etc/ssl/mongodb/mongodb.pem
    CAFile: /etc/ssl/mongodb/rootCA.pem
```

Für den „mode“ können unterschiedliche Einstellungen gewählt werden, z.B. allowTLS um generell die Nutzung von Zertifikaten zu bevorzugen. Über die Einstellung „disabledProtocols“ können z.B konkrete TLS-Versionen deaktiviert werden.

Nach dem Speichern der Datei, muss der Service neugestartet werden.

Zum Verbinden per Commandline:

```
mongo --tls --host 192.168.178.37 --tlsCertificateKeyFile /etc/ssl/
mongodb/client.pem --tlsCAFile /etc/ssl/mongodb/rootCA.pem
```

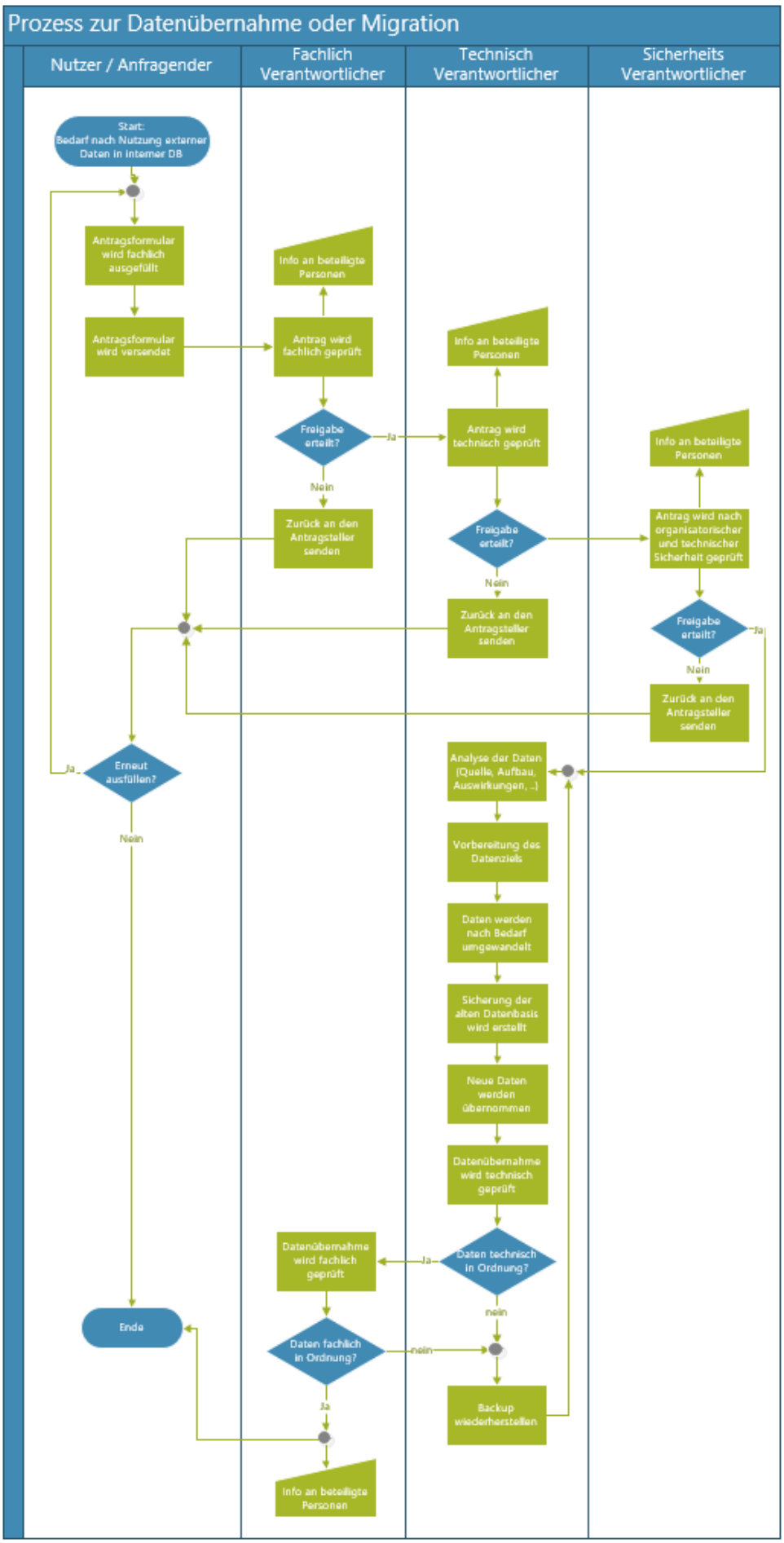
Angehängte Dateien:

- MongoDB_VerschluesselteVerbindung.docx
- cert.zip

A.4.13 APP.4.3.A17

Antrag Datenübernahme / Migration

Kurzbeschreibung der Datenübernahme / Migration	
Fachlich verantwortliche Person	Technisch verantwortliche Person
Sicherheit verantwortliche Person	
Zweck/Begründung der Datenübernahme	Kostenstelle
Quelle der Daten	Quellenformat
In welche Datenbank?	
Datenbankbezeichner	Zugriffsberechtigungen
Nur auszufüllen, wenn die Daten regelmäßig übernommen werden sollten	
Rhytmus	*(Wunsch-)Uhrzeit der Übernahme*
Täglich am: Wöchentlich am: Monatlich am: Jährlich am:	



Zur besseren Übersichtlichkeit wurde in diesem Prozess auf die Darstellung von beteiligten Dokumenten verzichtet.

Angehängte Dateien:

- Antrag_DatenuebernahmeMigration.docx
- Prozess_DatenuebernahmeMigration.vsdx

A.4.14 APP.4.3.A18

A.4.14.1 MySQL

A.4.14.1.1 Monitoring des MySQL Server

Zur Umsetzung dieser Anforderung wird zuerst geschaut, ob es integrierte Mittel gibt, um das Monitoring zu ermöglichen. Sollte es keine direkte, freiverfügbare Lösung geben, wird nach einer alternativen freien Variante, vorzugsweise ebenso als Opensource-Version, gesucht.

Folgende Funktionen werden untersucht:

- Allgemeines Monitoring des Hostsystems, wie z.B. CPU-Auslastung und Speicherreserven
- Datenbankmonitoring, wie z.B. Querydauer
- Festlegen von Schwellwerten für Alarmierungen
- Alarmierung außerhalb der Monitoring-Software, wie z.B. E-Mail, SMS, o.ä.

In der MySQL Community Edition ist mit der „MySQL Workbench“ ein kostenfreies Monitoring-Tool integriert. Hiermit können Hardware- und Datenbankparameter überprüft werden. Es ist aber keine Alarmierungsfunktion vorhanden. Abbildung 5 und Abbildung 6 zeigt die verfügbaren Ansichten. In der kostenpflichtigen Enterprise-Variante ist das Tool „MySQL Enterprise Monitor“ inkludiert, welches laut Funktionsliste alle o.g. Kriterien erfüllt [86].

Weitere Alternativen sind z.B.

Anbieter	Produktname	Lizenz	Quelle
Zenoss Inc.	Zenpack „MySQL“	Kommerziell	LINK
Percona LLC	Percona Monitoring and Management „MySQL“	„Opensource“	LINK
SolarWinds	Server & Application Monitor	Kommerziell	LINK

Paessler	MySQL Monitoring tool	Kommerziell	LINK

Daher wurde das freie Tool „Percona Monitoring and Management MySQL“ untersucht.

A.4.14.1.2 Percona Monitoring and Management

Zur Nutzung des „Percona Monitoring and Management“ (PMM) sind zwei Komponenten notwendig:

- PMM Server
- PMM Client

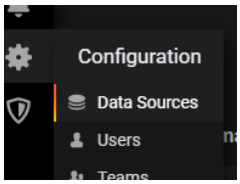
Der PMM-Server kann per Docker Image, per Virtuelle Maschine oder als Amazon Web Service installiert werden. Der Server basiert auf der Opensource-Software „Grafana“, einer Daten-Analyse und Visualisierungssoftware.

Der PMM Client muss in der Umgebung installiert werden, auf der der MySQL-Service installiert ist.

A.4.14.1.3 Server einrichten

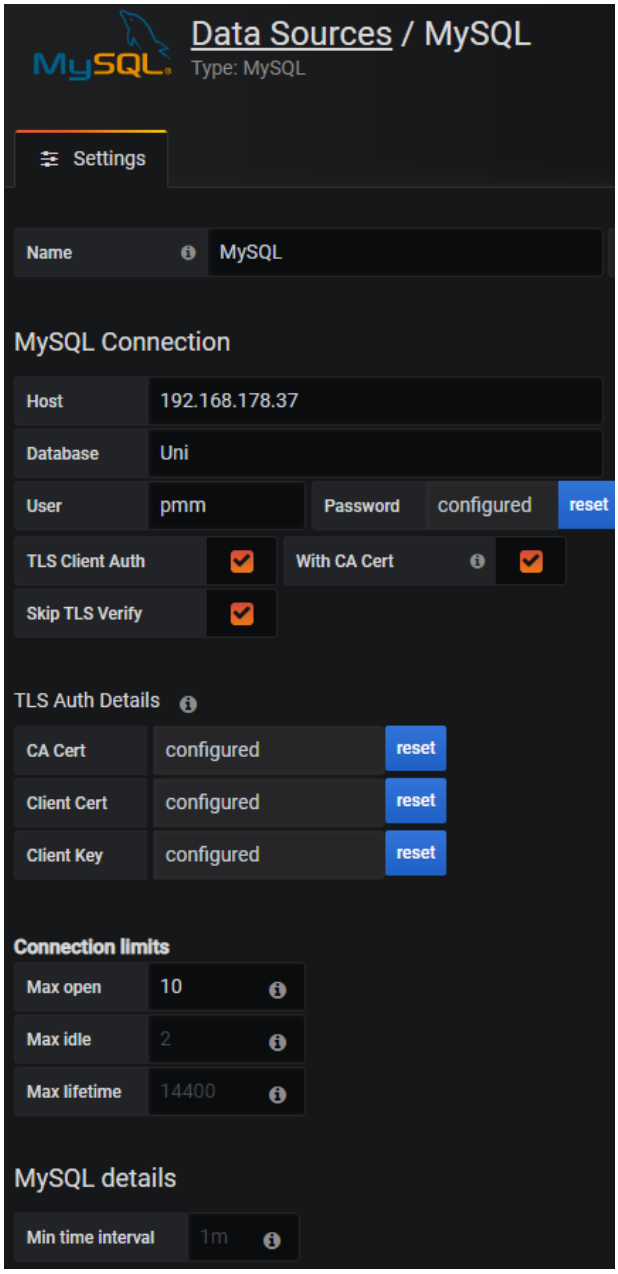
Es sollte ein dedizierter Nutzer in für das Monitoring der MySQL-Datenbank angelegt werden. Das Anlegen des Nutzers sollte, nach dem Hersteller Percona, wie folgt vergeben werden:

```
create user 'pmm'@'%' IDENTIFIED WITH mysql_native_password BY 'pas
s_W0RD1337' WITH MAX_USER_CONNECTIONS 10;
GRANT SELECT, PROCESS, SUPER, REPLICATION CLIENT, RELOAD ON *.* TO '
pmm'@'%';
GRANT SELECT, UPDATE, DELETE, DROP ON performance_schema.* TO 'pmm'
@'%';
```



In der Benutzeroberfläche des PMM-Servers mit den Standardlogindaten (Nutzer: admin, Passwort: admin) anmelden und die Datenquelle eingerichten:

Dafür in der linken Menüleiste bei „Configuration“ auf „Datasources klicken.



Auf der Folgeseite über „Add data source“ klicken und in der Folgesuche nach der dem Typ „MySQL“ suchen.

In der anschließenden Einstellungsseite wird die Verbindung zum Server aufgebaut. Dafür die MySQL-Logindaten nutzen, die zuvor in der MySQL-Datenbank angelegt wurden. In der folgenden Abbildung wird sich mit einem lokalen Server und mit der Datenbank „Uni“ verbunden.

A.4.14.1.4 Client installieren

Auf dem Hostsystem des MySQL DBS muss nun noch der PMM Client installiert werden. Auf einem RHEL/CentOS-System kann hierzu das Installationsprogramm „yum“ genutzt werden. Im Terminal hierzu folgenden Befehl eingeben und die Installation bestätigen

```
sudo yum install https://repo.percona.com/yum/percona-release-lates
t.noarch.rpm
```

Um die Verbindung zum PMM-Server herzustellen, dient der folgende Befehl:

```
pmm-admin config --server-insecure-tls --server-url=https://admin:a
dmin@192.168.178.38:443
```

Wobei der Parameter „--server-insecure-tls“ dazu dient, dass die genutzten TLS-Zertifikate akzeptiert werden. Diese wurden zu Testzwecken selbst-signiert, welches standardmäßig nicht akzeptiert werden würde.

Der Parameter „- -server-url“ gibt die IP-Adresse des PMM-Servers, den entsprechenden Port und die Logindaten am System an.

Nach dem Starten die Services, muss noch das MySQL-Plugin aktiviert werden:

```
pmm-admin add mysql --query-source=slowlog --username=pmm --passwor
d= pass_W0RD1337
```

Die angegebenen Logindaten entsprechen den Daten, die für den PMM-Server in dem MySQL-DBS angelegt wurden.

A.4.14.1.5 Funktionsumfang des Monitorings

A.4.14.1.5.1 Allgemeines Monitoring des Hostsystems, wie z.B. CPU-Auslastung und Speicherreserven

Das PMM MySQL bietet von sich aus ein Hardwaremonitoring für CPU-Auslastung, Disk Space, System Uptime und einige mehr. Ein Ausschnitt darauf zeigt Abbildung 7.

A.4.14.1.5.2 Datenbankmonitoring, wie z.B. Querydauer

Das PMM bietet mit dem MySQL-Plugin eine detaillierte Einsicht in das DBS. Hier werden z.B. Werte wie Server-Uptime, DB-Version, belegter Buffer, aktive Connections, abgebrochene Connections und einige mehr angezeigt. Es können auch Testqueries definiert werden, die eine Performancemessung des Systems ermöglichen. Ein Einblick in das DBS-Monitoring bietet Abbildung 8.

A.4.14.1.5.3 Festlegen von Schwellwerten für Alarmierungen

Für jeden überwachten Wert kann ein Graphpanel angelegt werden. Die Graphpanel haben die Funktionalität, Benachrichtigungen einstellen zu können, wie in Abbildung 9 gezeigt. Hier kann eine neue Regel angelegt werden und nach den Anforderungen des Administrators ein Event ausgeführt werden.

A.4.14.1.5.4 Alarmierung außerhalb der Monitoring-Software, wie z.B. E-Mail, SMS, o.ä.

Die Alarmierung aus der Software heraus kann über unterschiedliche Kanäle erfolgen, wie z.B. Email, diverse Chats oder andere Verwaltungssoftware, wie Kafka. Alle Kanäle sind Abbildung 10 zu entnehmen.

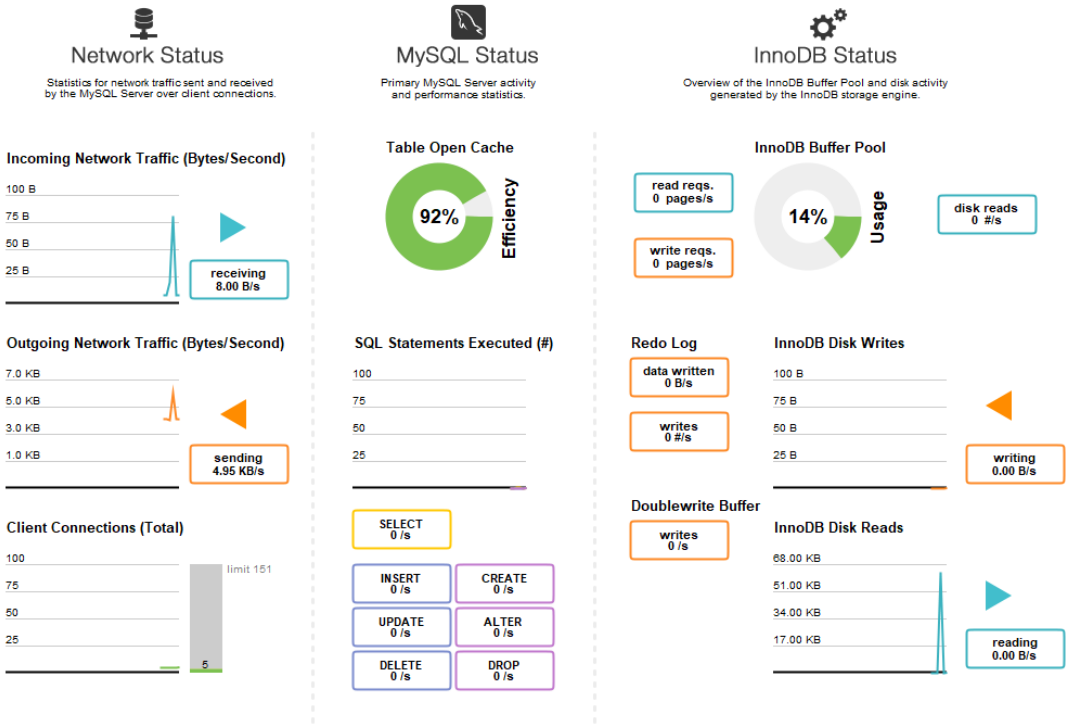


Abbildung 5 - Performance Dashboard in der MySQL Workbench

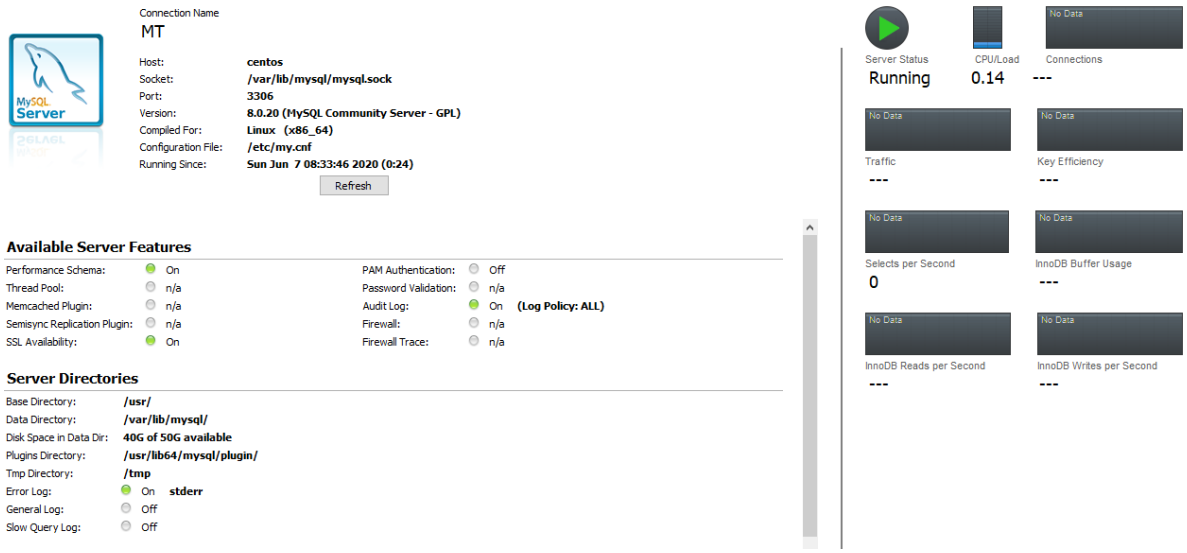


Abbildung 6 - Server Status in MySQL Workbench

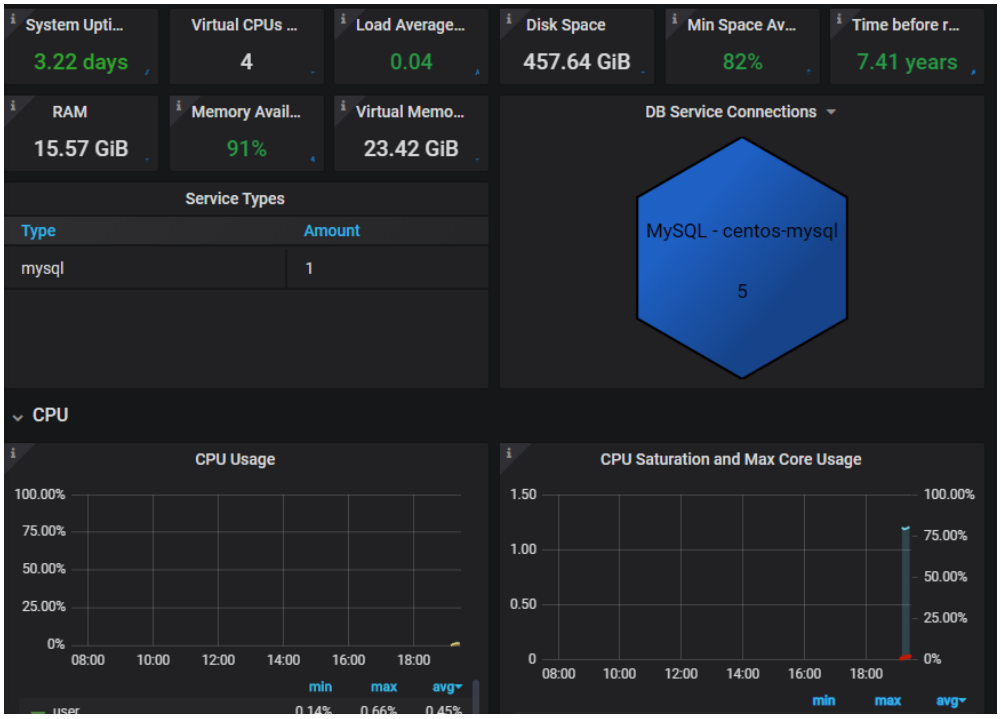


Abbildung 7 - Hardwaremonitoring mit PMM MySQL

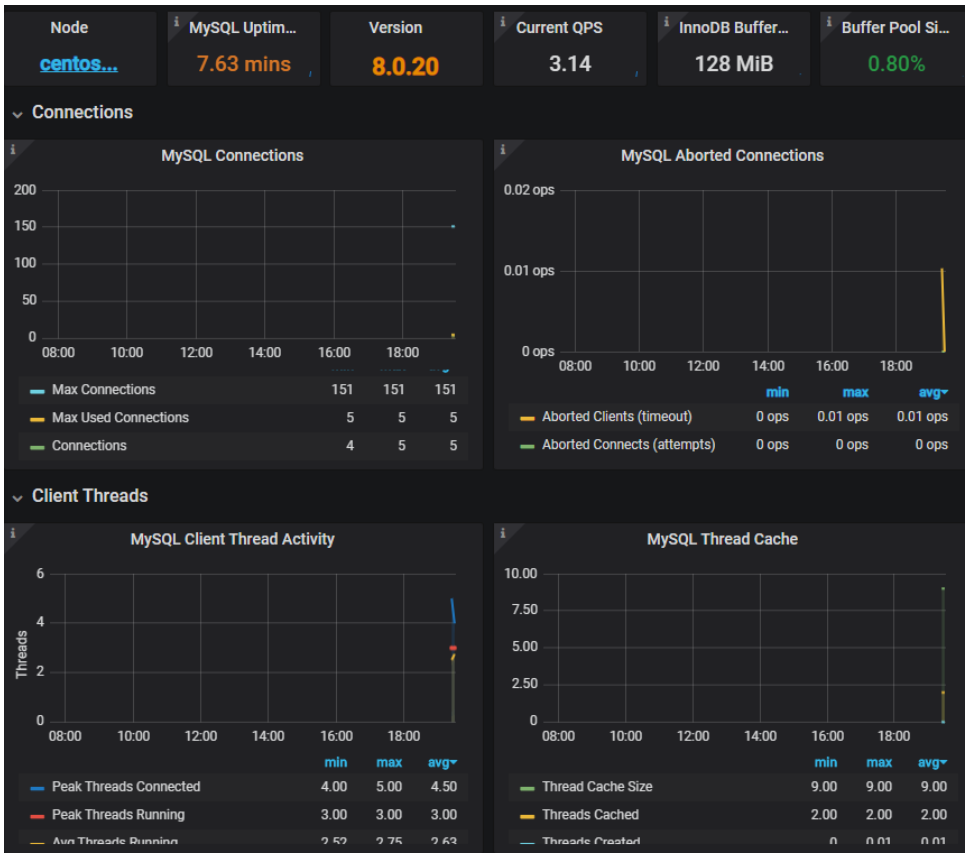


Abbildung 8 - DBS Monitoring mit PMM MySQL

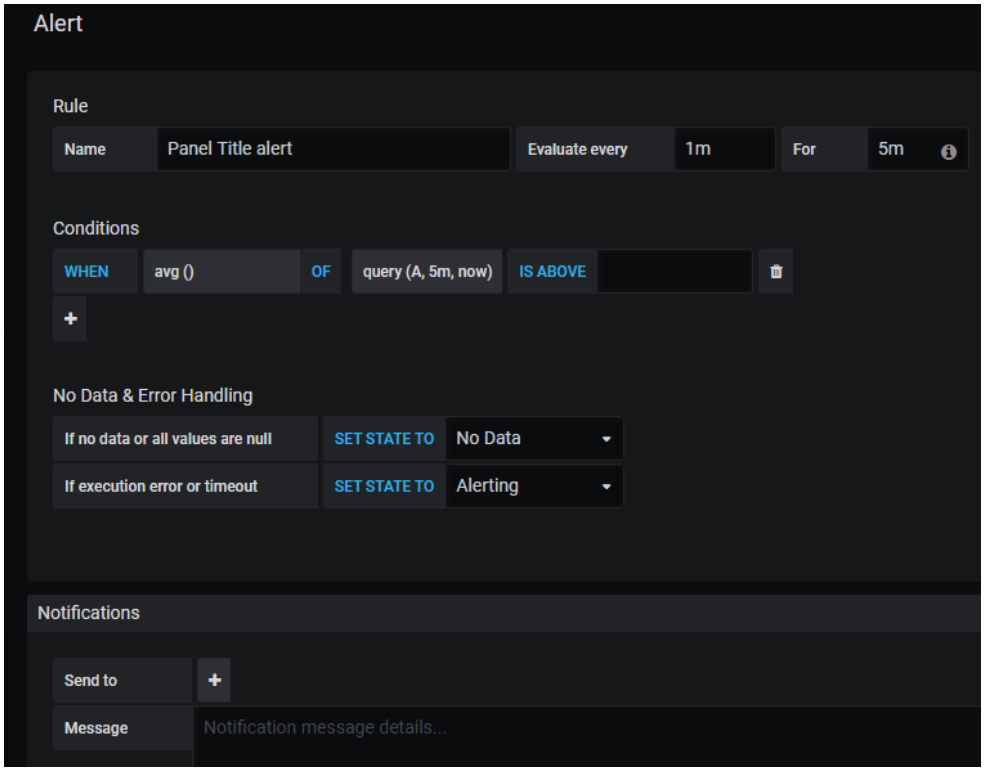


Abbildung 9 - Benachrichtigung für Panel hinzufügen

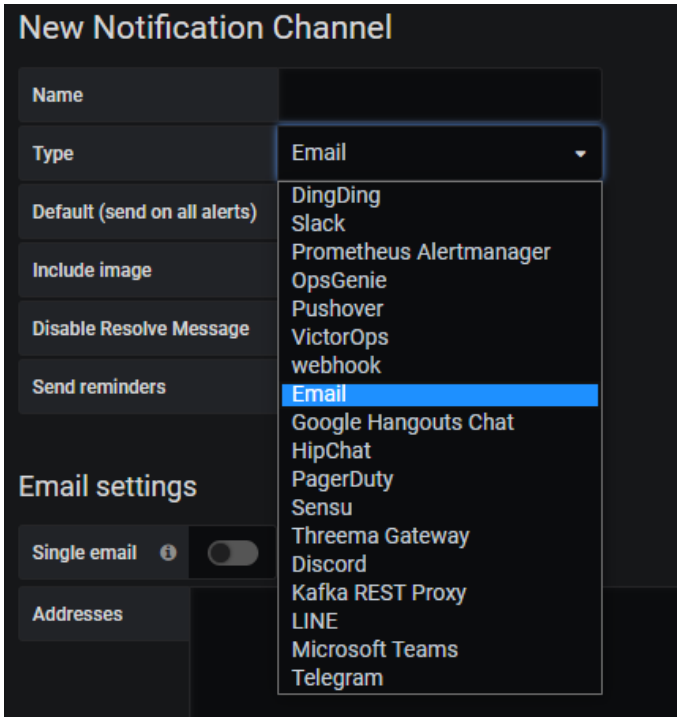


Abbildung 10 - Übersicht der Benachrichtigungsmöglichkeiten

Angehängte Dateien:

- MySQL_Monitoring.docx

A.4.14.2 PostgreSQL

A.4.14.2.1 Monitoring des PostgreSQL Server

Zur Umsetzung dieser Anforderung wird zuerst geschaut, ob es integrierte Mittel gibt, um das Monitoring zu ermöglichen. Sollte es keine direkte, freiverfügbare Lösung geben, wird nach einer alternativen freien Variante, vorzugsweise ebenso als Opensource-Version, gesucht.

Folgende Funktionen werden untersucht:

- Allgemeines Monitoring des Hostsystems, wie z.B. CPU-Auslastung und Speicherreserven
- Datenbankmonitoring, wie z.B. Querydauer
- Festlegen von Schwellwerten für Alarmierungen
- Alarmierung außerhalb der Monitoring-Software, wie z.B. E-Mail, SMS, o.ä.

In der PostgreSQL Community Edition ist kein Monitoring-Tool integriert.

In der PostgreSQL Sever ist mit dem Tool „pgAdmin“ ein kostenfreies Administrationsprogramm integriert. Hiermit können Hardware- und Datenbankparameter überprüft werden. Es ist aber keine Alarmierungsfunktion vorhanden. Abbildung 5 zeigt die verfügbare Ansicht. In der kostenpflichtigen Enterprise-Variante „EnterpriseDB Postgres“ ist das Tool „EDB Postgres Enterprise Manager[42]“ inkludiert, welches laut Funktionsliste alle o.g. Kriterien erfüllt

---

[42] https://www.enterprisedb.com/enterprise-postgres/edb-postgres-enterprise-manager-pem

Weitere Alternativen sind z.B.

Anbieter	Produktname	Lizenz	Quelle
Open PostgreSQL Monitoring Development Group (OPMDG)	Open PostgreSQL Monitoring	„Opensource“	LINK
Percona LLC	Percona Monitoring and Management „PostgreSQL“	„Opensource“	LINK
RapidLoop, Inc.	pgDash	Kommerziell	LINK
Duboce Labs, Inc.	pganalyze	Kommerziell	LINK

Da das Tool „Open PostgreSQL Monitoring“ auf der proprietären Software „Nagios XI“[43] aufbaut, wird diese nicht betrachtet. Daher wurde das freie Tool „Percona Monitoring and Management PostgreSQL“ untersucht.

A.4.14.2.2 Percona Monitoring and Management

Zur Nutzung des „Percona Monitoring and Management“ (PMM) sind zwei Komponenten notwendig:

- PMM Server
- PMM Client

Der PMM-Server kann per Docker Image, per Virtuelle Maschine oder als Amazon Web Service installiert werden. Der Server basiert auf der Opensource-Software „Grafana“, einer Daten-Analyse und Visualisierungssoftware.

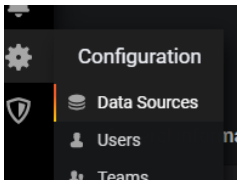
Der PMM Client muss in der Umgebung installiert werden, auf der der PostgreSQL-Service installiert ist.

[43] https://www.nagios.com/products/nagios-xi/

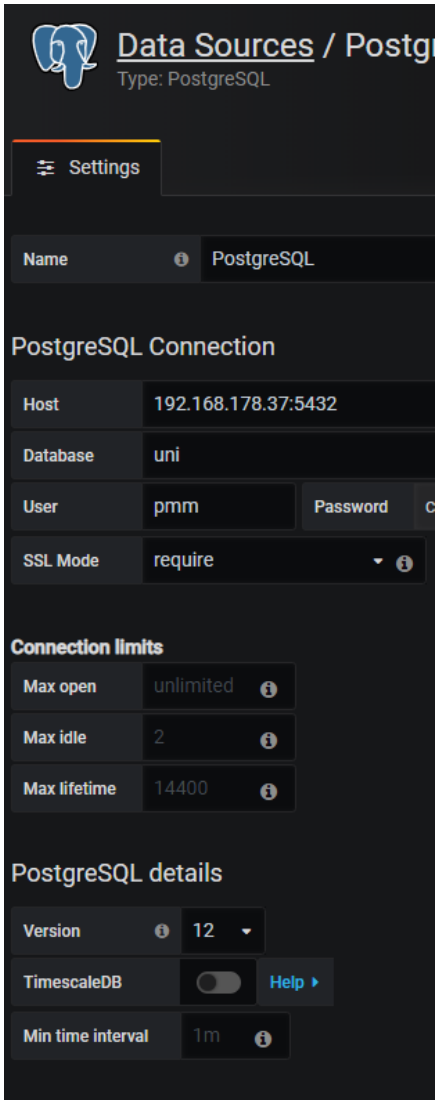
A.4.14.2.3 Server einrichten

Es sollte ein dedizierter Nutzer in für das Monitoring der MySQL-Datenbank angelegt werden. Das Anlegen des Nutzers sollte, nach dem Hersteller Percona, wie folgt vergeben werden:

```
psql -c "CREATE USER pmm WITH ENCRYPTED PASSWORD 'pass_W0RD1337'"
psql -c "GRANT pg_monitor to pmm"
```



In der Benutzeroberfläche des PMM-Servers mit den Standardlogindaten (Nutzer: admin, Passwort: admin) anmelden und die Datenquelle eingerichten:



Dafür in der linken Menüleiste bei „Configuration“ auf „Datasources klicken.

Auf der Folgeseite über „Add data source“ klicken und in der Folgesuche nach der dem Typ „PostgreSQL“ suchen.

In der anschließenden Einstellungsseite wird die Verbindung zum Server aufgebaut. Dafür die Postgres-Logindaten nutzen, die zuvor in der PostgreSQL-Datenbank angelegt wurden. In der folgenden Abbildung wird sich mit einem lokalen Server und mit der Datenbank „Uni“ verbunden.

A.4.14.2.4 Client installieren

Auf dem Hostsystem des PostgreSQL DBS muss nun noch der PMM Client installiert werden. Auf einem RHEL/CentOS-System kann hierzu das Installationsprogramm „yum“ genutzt werden. Im Terminal hierzu folgenden Befehl eingeben und die Installation bestätigen

```
sudo yum install https://repo.percona.com/yum/percona-release-lates
t.noarch.rpm
```

Um die Verbindung zum PMM-Server herzustellen, dient der folgende Befehl:

```
pmm-admin config --server-insecure-tls --server-url=https://admin:a
dmin@192.168.178.38:443
```

Wobei der Parameter „--server-insecure-tls“ dazu dient, dass die genutzten TLS-Zertifikate akzeptiert werden. Diese wurden zu Testzwecken selbst-signiert, welches standardmäßig nicht akzeptiert werden würde.

Der Parameter „- -server-url“ gibt die IP-Adresse des PMM-Servers, den entsprechenden Port und die Logindaten am System an.

```
echo "host    all             pmm        192.168.80.20/32        md
5" >> $PGDATA/pg_hba.conf
$ psql -c "select pg_reload_conf()"
```

Query Analytics aktivieren

```
psql -c "CREATE DATABASE pmm_user"
psql -c -d pmm_user "CREATE EXTENSION pg_stat_statements"
---
psql
/c pmm
CREATE EXTENSION pg_stat_statements
---
psql -c "ALTER SYSTEM SET shared_preload_libraries TO 'pg_stat_stat
ements'"
```

Anschließend den Server neustarten, um die Einstellungen zu übernehmen.

Abschließend noch die Service-Connection zum Server aktvieren

```
pmm-admin add postgresql --username=pmm --password=pass_W0RD1337 po
stgresql-12 192.168.178.37:5432 --server-url=https://admin:admin@19
2.168.178.38:443 --server-insecure-tls
```

Die angegebenen Logindaten entsprechen den Daten, die für den PMM-Server in dem PostgreSQL-DBS angelegt wurden.

A.4.14.2.5 Funktionsumfang des Monitorings

A.4.14.2.5.1 Allgemeines Monitoring des Hostsystems, wie z.B. CPU-Auslastung und Speicherreserven

Das PMM PostgreSQL bietet von sich aus ein Hardwaremonitoring für CPU-Auslastung, Disk Space, System Uptime und einige mehr. Ein Ausschnitt darauf zeigt Abbildung 12.

A.4.14.2.5.2 Datenbankmonitoring, wie z.B. Querydauer

Das PMM bietet mit dem PostgreSQL-Plugin eine detaillierte Einsicht in das DBS. Hier werden z.B. Werte wie Server-Uptime, DB-Version, belegter Buffer, aktive Connections, abgebrochene Connections und einige mehr angezeigt. Es können auch Testqueries definiert werden, die eine Performancemessung des Systems ermöglichen. Ein Einblick in das DBS-Monitoring bietet Abbildung 13.

A.4.14.2.5.3 Festlegen von Schwellwerten für Alarmierungen

Für jeden überwachten Wert kann ein Graphpanel angelegt werden. Die Graphpanel haben die Funktionalität Benachrichtigungen einstellen zu können, wie in Abbildung 14 gezeigt. Hier kann eine neue Regel angelegt werden und nach den Anforderungen des Administrators ein Event ausgeführt werden.

A.4.14.2.5.4 Alarmierung außerhalb der Monitoring-Software, wie z.B. E-Mail, SMS, o.ä.

Die Alarmierung aus der Software heraus kann über unterschiedliche Kanäle erfolgen, wie z.B. Email, diverse Chats oder andere Verwaltungssoftware, wie Kafka. Alle Kanäle sind Abbildung 15 zu entnehmen.

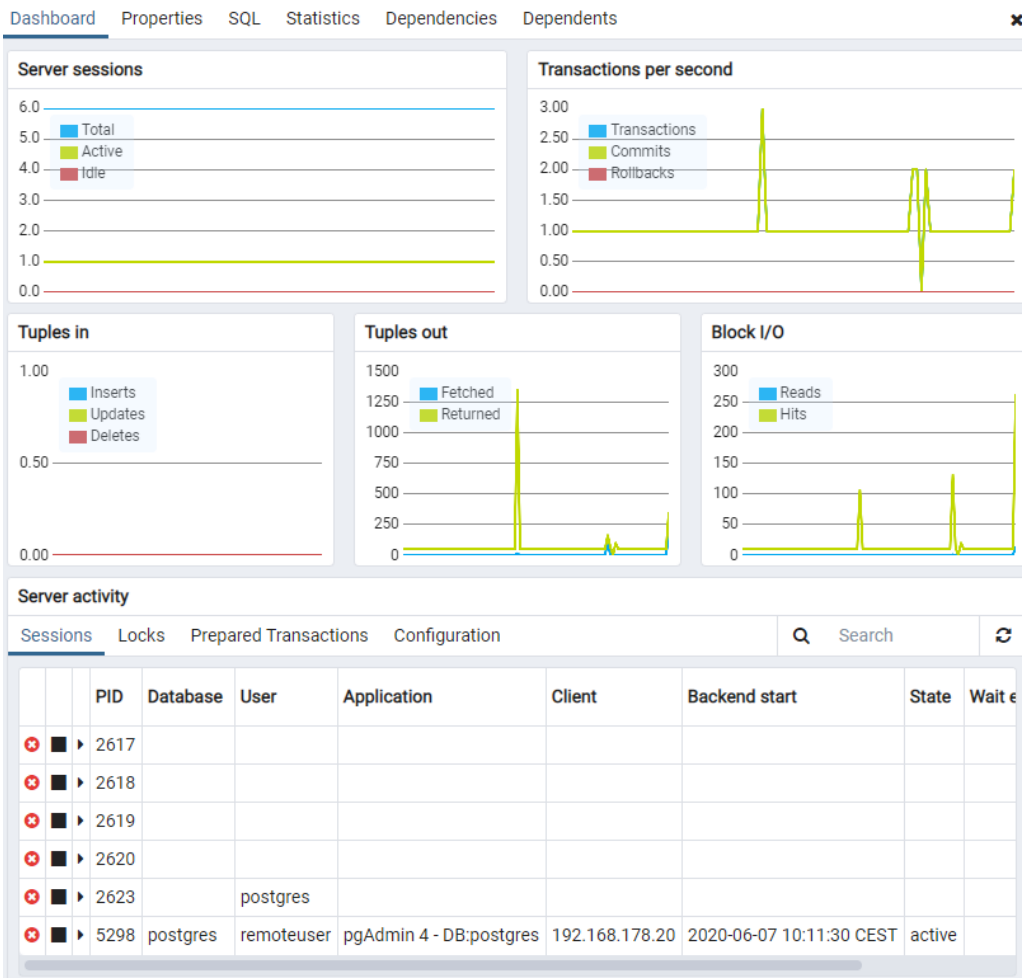


Abbildung 11 - Monitoring mit pgAdmin

Abbildung 12 - Hardwaremonitoring mit PMM PostgreSQL

Abbildung 13 - DBS Monitoring mit PMM PostgreSQL

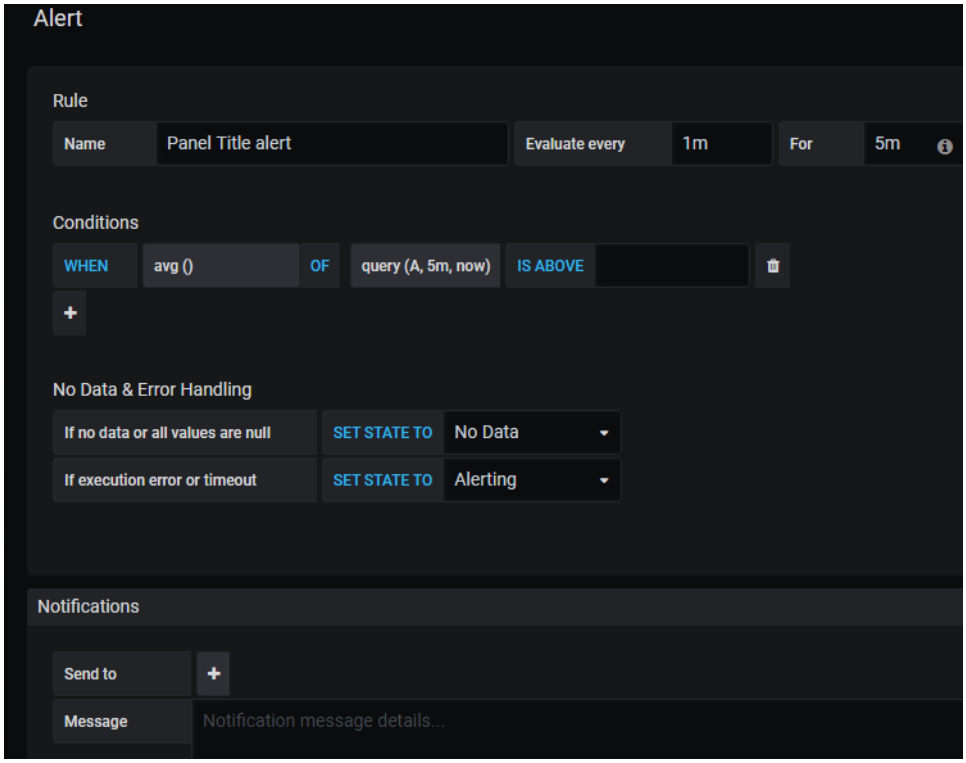


Abbildung 14 - Benachrichtigung für Panel hinzufügen

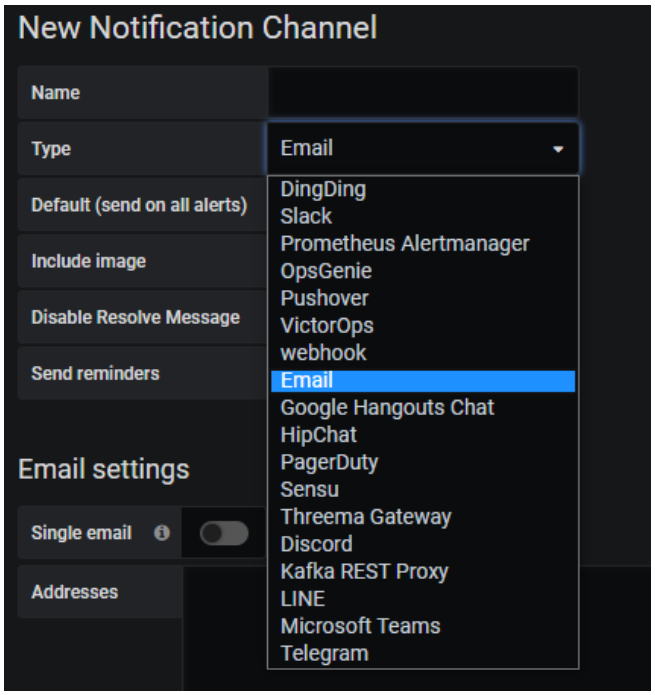


Abbildung 15 - Übersicht der Benachrichtigungsmöglichkeiten

Angehängte Dateien:

- PostgreSQL_Monitoring.docx

A.4.14.3 MongoDB

A.4.14.3.1 Monitoring des MongoDB Servers

Zur Umsetzung dieser Anforderung wird zuerst geschaut, ob es integrierte Mittel gibt, um das Monitoring zu ermöglichen. Sollte es keine direkte, freiverfügbare Lösung geben, wird nach einer alternativen freien Variante, vorzugsweise ebenso als Opensource-Version, gesucht.

Folgende Funktionen werden untersucht:

- Allgemeines Monitoring des Hostsystems, wie z.B. CPU-Auslastung und Speicherreserven
- Datenbankmonitoring, wie z.B. Querydauer
- Festlegen von Schwellwerten für Alarmierungen
- Alarmierung außerhalb der Monitoring-Software, wie z.B. E-Mail, SMS, o.ä.

Für alle MongoDB Instanzen, einschließlich der Community-Variante, bietet MongoDB eine freie Monitoring-Lösung an. Diese kann mit den entsprechenden Rechten aktiviert werden und wird auf einem Cloudserver des Herstellers gehostet. Jeder mit Zugriff auf die URL kann die Daten des Servers einsehen. Die Daten sind jeweils für die letzten 24 Stunden einsehbar. Darüber hinaus ist kein Monitoring möglich. Es werden einige Hardwaredaten und einige Datenbank-Daten aufgenommen, wie z.B. CPU-Auslastung, Netzwerktraffic, aktive Operationen (lesend und schreibend) und Anzahl gescannter Dokumente. [87] Es werden keine Funktionen für Alarmierungen bereitgestellt. Abbildung 16 zeigt einen Ausschnitt der bereitgestellten Monitoringsoftware. Alternativ können vom Hersteller MongoDB Inc. Auch kostenpflichtige Dienste gekauft werden, z.B. MongoDB Cloud Manager oder MongoDB Ops Manager. Diese Dienste bieten weitere Funktionalitäten. [88]

Da nicht alle Funktionen in der freien Monitor-Lösung vorhanden sind, wurde nach Alternativen hierzu gesucht. Das sind z.B.

Anbieter	Produktname	Lizenz	Quelle
Dynatrace LLC	MongoDB monitoring	Kommerziell	LINK
Datadog, Inc.	Datadog	Kommerziell	LINK
Perconca LLC	Percona Monitoring and Management	„OpenSource“	LINK
SolarWinds Worldwide LLC	Database Performance Monitor	Kommerziell	LINK

Daher wurde das freie Tool „Percona Monitoring and Management MongoDB“ zusätzlich untersucht.

A.4.14.3.2 Percona Monitoring and Management

Zur Nutzung des „Percona Monitoring and Management“ (PMM) sind zwei Komponenten notwendig:

- PMM Server
- PMM Client

Der PMM-Server kann per Docker Image, per Virtuelle Maschine oder als Amazon Web Service installiert werden. Der Server basiert auf der Opensource-Software „Grafana“, einer Daten-Analyse und Visualisierungssoftware.

Der PMM Client muss in der Umgebung installiert werden, auf der der MongoDB-Service installiert ist.

A.4.14.3.3 Server einrichten

Es sollte ein dedizierter Nutzer in für das Monitoring der MongoDB-Datenbank angelegt werden. Das Anlegen des Nutzers sollte, nach dem Hersteller Percona, wie folgt vergeben werden:

```
db.getSiblingDB("admin").createUser({
    user: "mongodb_exporter",
    pwd: "s3cR#tpa$$worD",
    roles: [
        { role: "clusterMonitor", db: "admin" },
        { role: "read", db: "local" }
    ]
})
```

Außerdem sollen zur Query Analytic folgende Einstellungen in der mongod.config gesetzt werden:

```
operationProfiling:
   slowOpThresholdMs: 200
   mode: slowOp
   slowOpSampleRate: 1.0
```

Anschließend muss der Sever neugestartet werden.

A.4.14.3.4 Client installieren

Auf dem Hostsystem des MongoDB DBS muss nun noch der PMM Client installiert werden. Auf einem RHEL/CentOS-System kann hierzu das Installationsprogramm „yum“ genutzt werden. Im Terminal hierzu folgenden Befehl eingeben und die Installation bestätigen

```
sudo yum install https://repo.percona.com/yum/percona-release-lates
t.noarch.rpm
```

Um die Verbindung zum PMM-Server herzustellen, dient der folgende Befehl:

```
pmm-admin config --server-insecure-tls --server-url=https://admin:a
dmin@192.168.178.38:443
```

Wobei der Parameter „--server-insecure-tls“ dazu dient, dass die genutzten TLS-Zertifikate akzeptiert werden. Diese wurden zu Testzwecken selbst-signiert, welches standardmäßig nicht akzeptiert werden würde.

Der Parameter „- -server-url“ gibt die IP-Adresse des PMM-Servers, den entsprechenden Port und die Logindaten am System an.

Nach dem Starten die Services, muss noch das MongoDB-Plugin aktiviert werden:

```
pmm-admin add mongodb --username=mongodb_exporter --password=s3cR#t
pa$$worD mongo 127.0.0.1:27017
```

Die angegebenen Logindaten entsprechen den Daten, die für den PMM-Server in der MongoDB-DB angelegt wurden.

Wichtig: in der Testinstallation war es nicht möglich, eine Verbindung mit SSL/TSL-Verschlüsselung aufzubauen, weil die PMM-Serverversion keine MongoDB-Konfiguration vorsieht und daher keine SSL/TSL-Verschlüsselung eingerichtet werden kann.

A.4.14.3.5 Funktionsumfang des Monitorings

A.4.14.3.5.1 Allgemeines Monitoring des Hostsystems, wie z.B. CPU-Auslastung und Speicherreserven

Das PMM MongoDB bietet von sich aus ein Hardwaremonitoring für CPU-Auslastung, Disk Space, System Uptime und einige mehr. Ein Ausschnitt darauf zeigt Abbildung 17.

A.4.14.3.5.2 Datenbankmonitoring, wie z.B. Querydauer

Das PMM bietet mit dem MongoDB-Plugin eine detaillierte Einsicht in das DBS. Hier werden z.B. Werte wie Server-Uptime, DB-Version, belegter Buffer, aktive Connections, abgebrochene Connections und einige mehr angezeigt. Es können auch Testqueries definiert werden, die eine Performancemessung des Systems ermöglichen. Ein Einblick in das DBS-Monitoring bietet Abbildung 18.

A.4.14.3.5.3 Festlegen von Schwellwerten für Alarmierungen

Für jeden überwachten Wert kann ein Graphpanel angelegt werden. Die Graphpanel haben die Funktionalität Benachrichtigungen einstellen zu können. Hier kann eine neue Regel angelegt werden und nach den Anforderungen des Administrators ein Event ausgeführt werden.

A.4.14.3.5.4 Alarmierung außerhalb der Monitoring-Software, wie z.B. E-Mail, SMS, o.ä.

Die Alarmierung aus der Software heraus kann über unterschiedliche Kanäle erfolgen, wie z.B. Email, diverse Chats oder andere Verwaltungssoftware, wie Kafka. Alle Kanäle sind Abbildung 20 zu entnehmen.

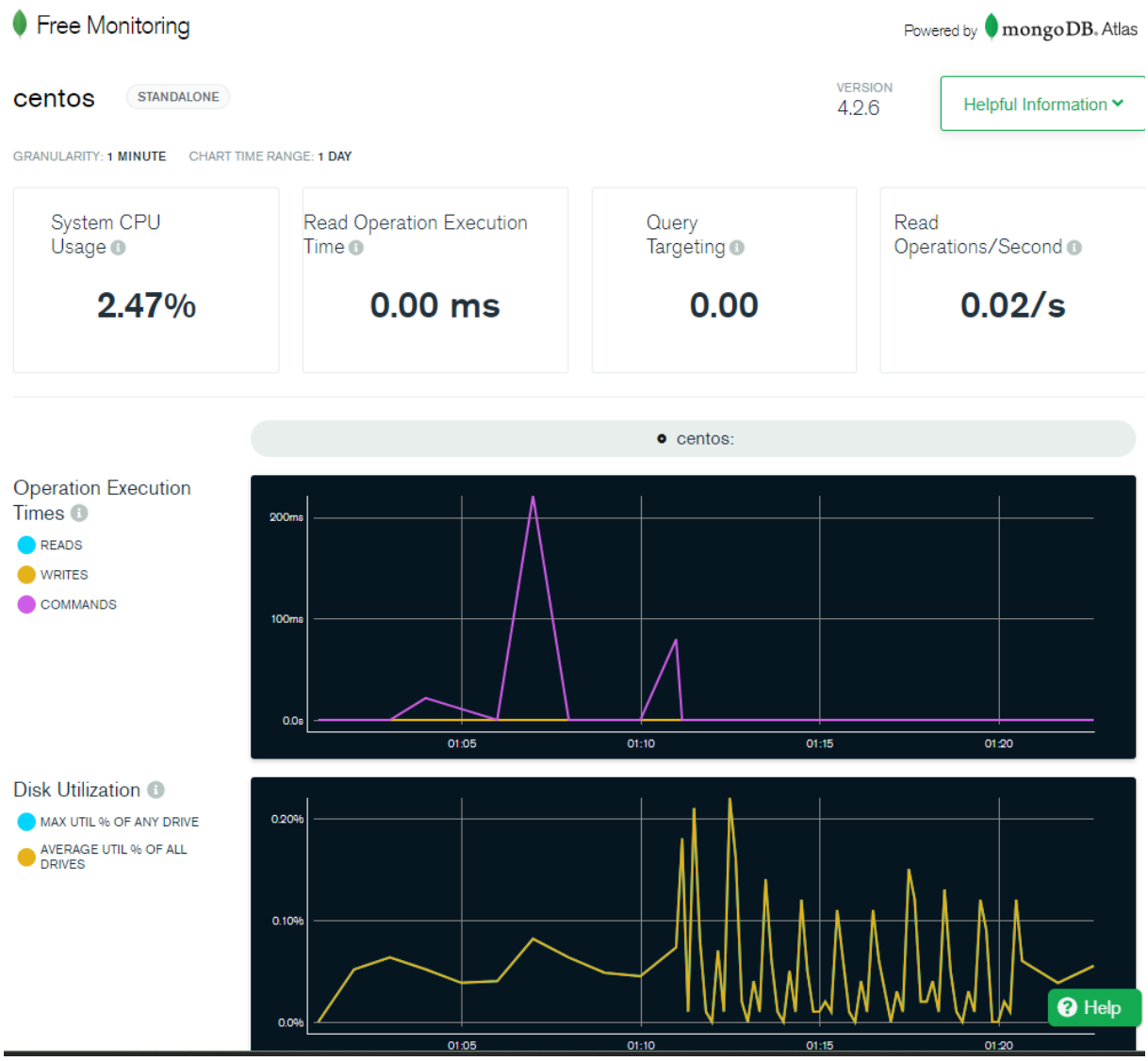


Abbildung 16 - Ausschnitt aus dem Free MongoDB Monitoring

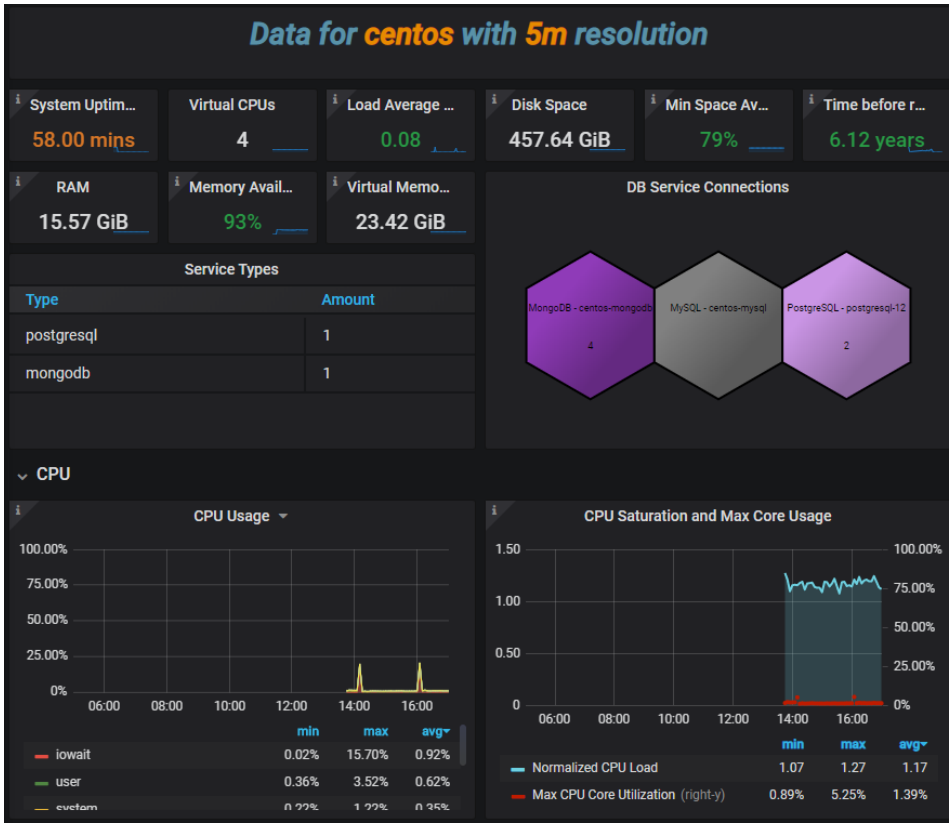


Abbildung 17 - Hardwaremonitoring mit PMM MongoDB

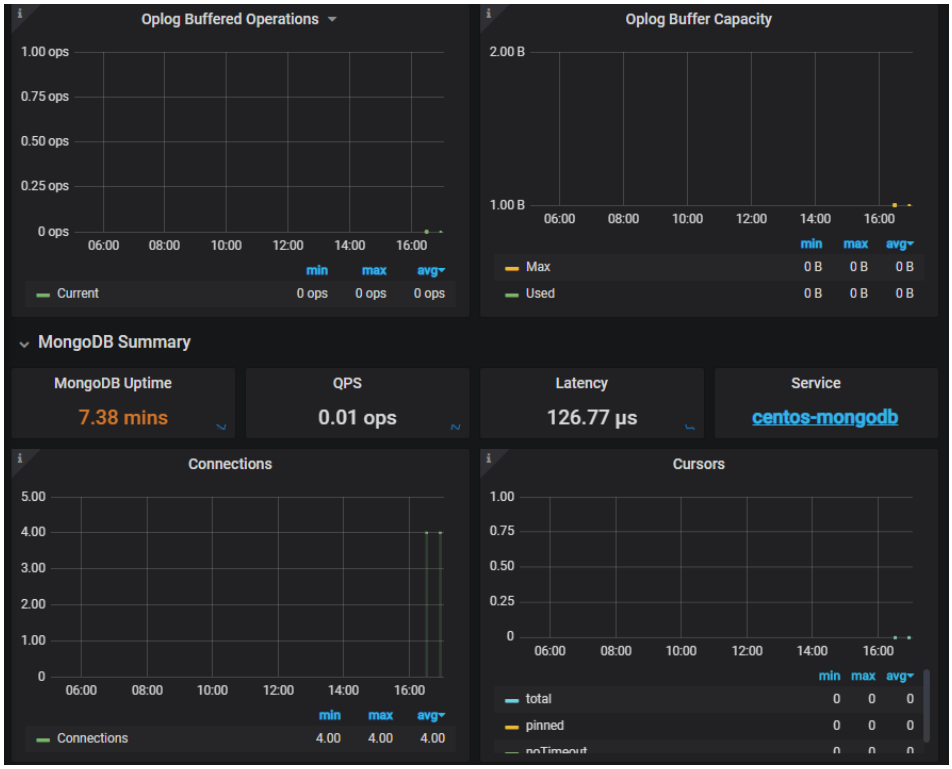


Abbildung 18 - DBS Monitoring mit PMM MongoDB

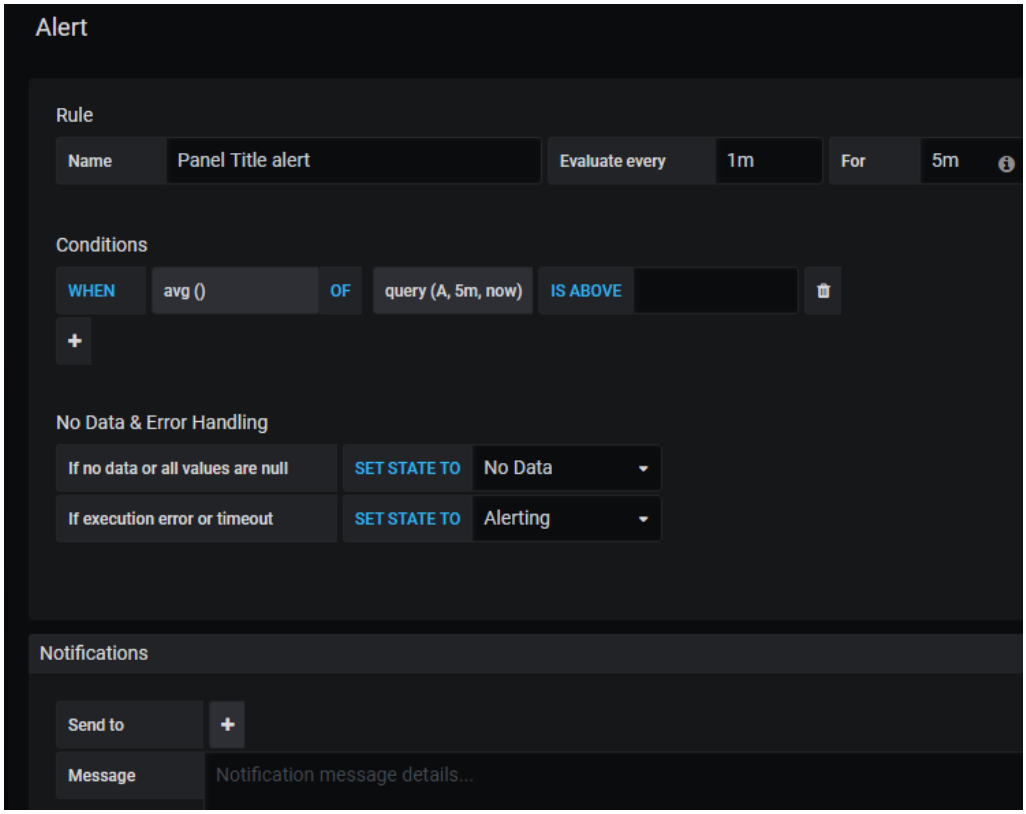


Abbildung 19 - Benachrichtigung für Panel hinzufügen

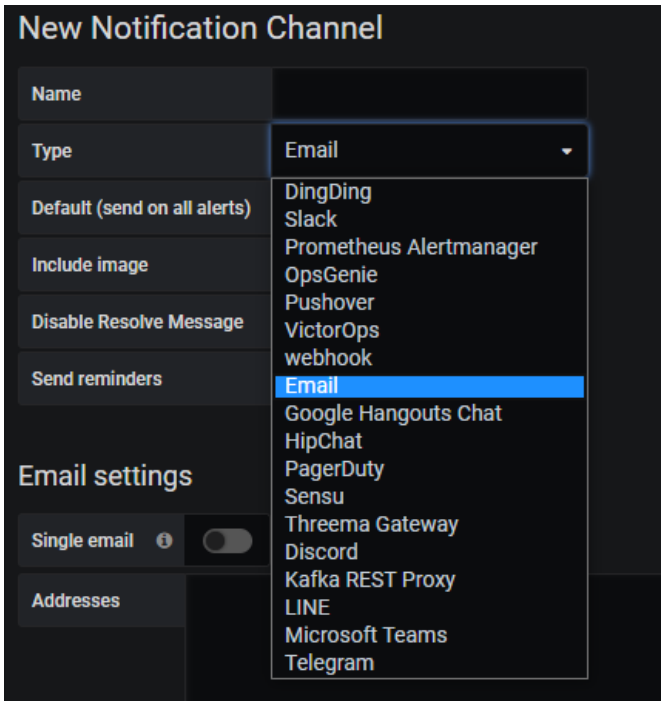


Abbildung 20 - Übersicht der Benachrichtigungsmöglichkeiten

Angehängte Dateien:

- MongoDB_Monitoring.docx

A.4.15 APP.4.3.A19

- Mozilla_SQL_StyleGuide.pdf

A.4.16 APP.4.3.A20

- ITGS-Check_APP.4.3 Relationale Datenbanksysteme.odt

A.4.17 APP.4.3.A21

A.4.17.1 MySQL

A.4.17.1.1 Einsatz von Datenbank Security Tools für MySQL

In der Anforderung „Einsatz von Datenbank Security Tools“ werden Sicherheitstools für die Datenbankysteme gefordert, die die folgenden Funktionen bereitstellen:

- Erstellung einer Übersicht über alle Datenbanksysteme,
- erweiterte Konfigurationsmöglichkeiten und Rechtemanagement der Datenbanken,
- Erkennung und Unterbindung von möglichen Angriffen (z. B. Brute Force Angriffe auf ein Benutzerkonto, SQLInjection) und
- Auditfunktionen (z. B. Überprüfung von Konfigurationsvorgaben).“

Derzeit existieren keine Informationssicherheitsprodukte für Datenbanken, die Opensource oder frei verfügbar sind. Einzelne Funktionen können über andere Systeme auf Opensource-Basis erfüllt werden. Im Folgenden werden Umsetzungsvorschläge für die Einzelfunktionen aufgezeigt.

A.4.17.1.1.1 Erstellung einer Übersicht über alle Datenbanksysteme

Mit dem Tool „Percona Monitoring and Management“, wird eine Übersicht über alle konfigurierten Datenbanksysteme erstellt. Außerdem kann hier der aktuelle Status des Systems mit abgefragt und individuelle Dashboards konfiguriert werden.

A.4.17.1.1.2 Erweiterte Konfigurationsmöglichkeiten und Rechtemanagement

Die von Oracle bei dem MySQL Community Server mitgelieferte „MySQL Workbench“, ist ein Grafiktools, zur Verwaltung und Überwachung von MySQL-

Datenbanken. Mit diesem Tool können z.B. Konfigurationsdateien angepasst werden, wie in Abbildung 21 gezeigt, und Benutzer verwaltet werden. Dies ist in Abbildung 22 gezeigt. Das ist möglich, ohne direkt die MySQL-Configdatei anzupassen

A.4.17.1.1.3 Erkennung und Unterbindung von möglichen Angriffen

Für MySQL gibt es derzeit kein freies Programm auf dem Markt, welches eine Angriffserkennung und -unterbindung ermöglicht.

A.4.17.1.1.4 Auditfunktionen

Ein Audit-Logging wurde auf Grundlage eines Drittanbieter-Plugins in Kapitel 3.1.8 im Hauptdokument dieser Arbeit beschrieben. Mithilfe des Plugins kann eine Audit-Taugliches Log-Funktionalität ermöglicht werden. Dieses Plugin nimmt auch Konfigurationsänderungen auf, die während der Laufzeit vorgenommen werden. Ein geeignetes Log-Management Tool, wie z.B. Graylog[44], kann darauf konfiguriert werden, solche Änderungen zu erkennen und zu melden.

---

[44] https://www.graylog.org/

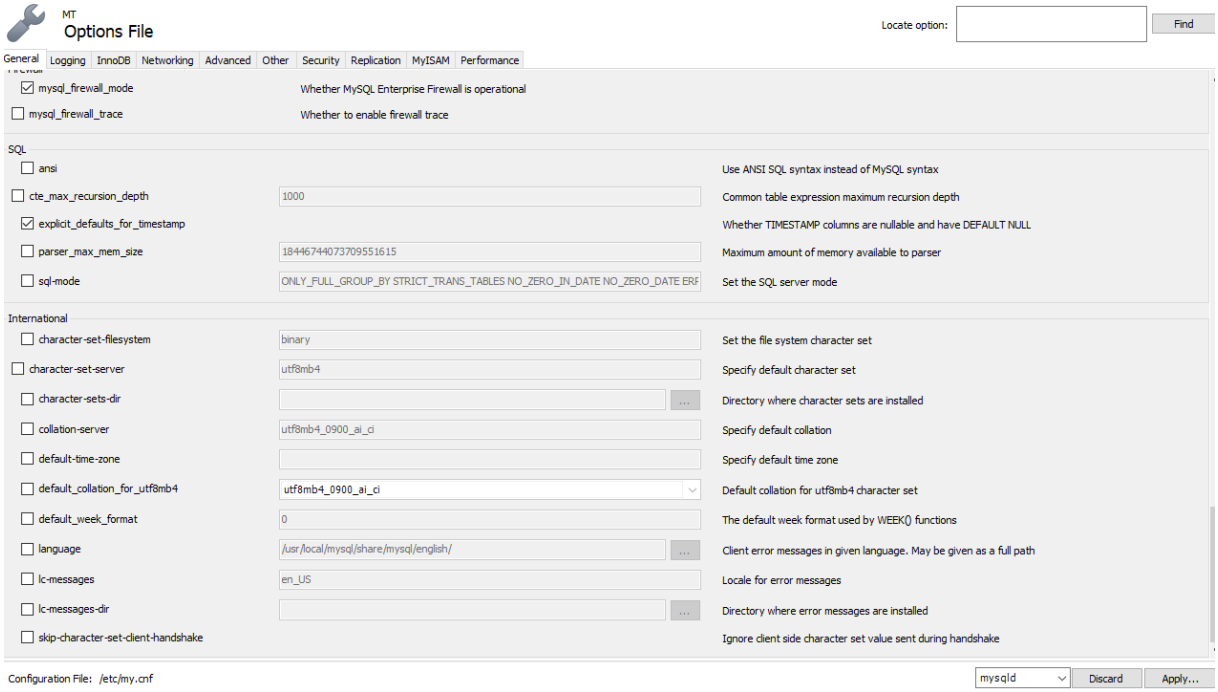


Abbildung 21 - Grafische Konfiguration in der MySQL Workbench

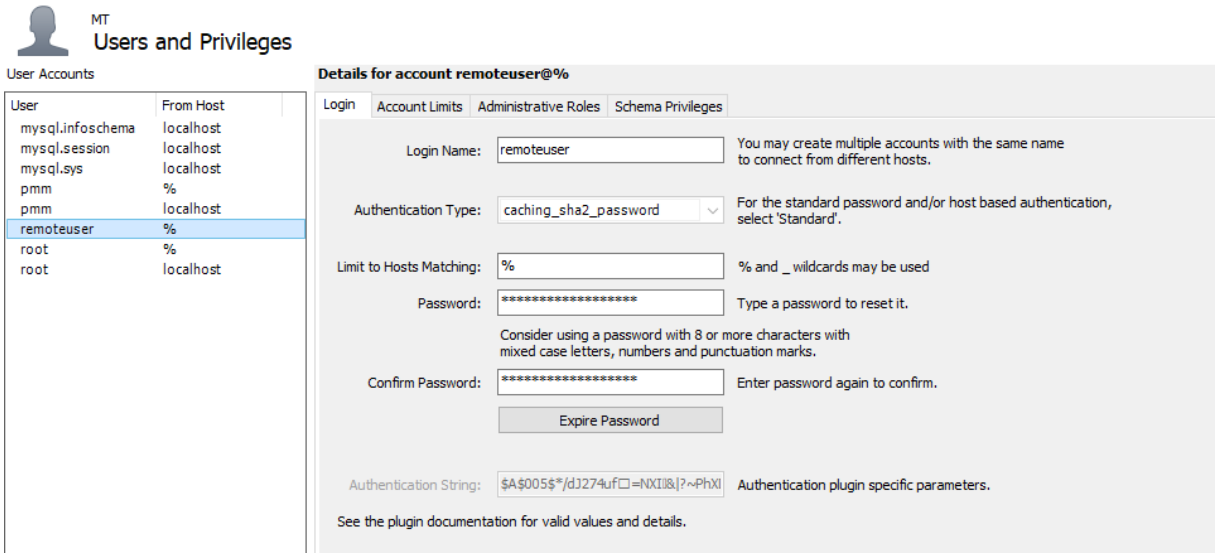


Abbildung 22 - Grafische Verwaltung von Benutzern

Angehängte Dateien:

- MySQL_SecurityTools.docx

A.4.17.2 PostgreSQL

A.4.17.2.1 Einsatz von Datenbank Security Tools für PostgreSQL

In der Anforderung „Einsatz von Datenbank Security Tools“ werden Sicherheitstools für die Datenbankysteme gefordert, die die folgenden Funktionen bereitstellen:

- Erstellung einer Übersicht über alle Datenbanksysteme,
- erweiterte Konfigurationsmöglichkeiten und Rechtemanagement der Datenbanken,
- Erkennung und Unterbindung von möglichen Angriffen (z. B. Brute Force Angriffe auf ein Benutzerkonto, SQLInjection) und
- Auditfunktionen (z. B. Überprüfung von Konfigurationsvorgaben).“

Derzeit existieren keine Informationssicherheitsprodukte für Datenbanken, die Opensource oder frei verfügbar sind. Einzelne Funktionen können über andere Systeme auf Opensource-Basis erfüllt werden. Im Folgenden werden Umsetzungsvorschläge für die Einzelfunktionen aufgezeigt.

A.4.17.2.1.1 Erstellung einer Übersicht über alle Datenbanksysteme

Mit dem Tool „Percona Monitoring and Management“, wird eine Übersicht über alle konfigurierten Datenbanksysteme erstellt. Außerdem kann hier der aktuelle Status des Systems mit abgefragt und individuelle Dashboards konfiguriert werden.

A.4.17.2.1.2 Erweiterte Konfigurationsmöglichkeiten und Rechtemanagement

Die bei PostgreSQL Server optional mitgelieferte Software „pgAdmin“, ist ein Grafiktool, zur Verwaltung und Überwachung von PostgreSQL-Datenbanken. Mit diesem Tool können z.B. aktuelle Konfigurationen eingesehen, aber nicht angepasst werden. Abbildung 21 zeigt die Konfigurationsansicht. Zusätzlich bietet pgAdmin die Funktion, SQL-Statements abzusetzen, so dass über das Programm auch Einstellungen zur Laufzeit angepasst werden können, allerdings ohne grafische Unterstützung. Benutzer können über das Programm verwaltet werden. Dies ist in Abbildung 22 gezeigt.

A.4.17.2.1.3 Erkennung und Unterbindung von möglichen Angriffen

Für MySQL gibt es derzeit kein freies Programm auf dem Markt, welches eine Angriffserkennung und -unterbindung ermöglicht.

A.4.17.2.1.4 Auditfunktionen

Ein Audit-Logging wurde auf Grundlage eines Drittanbieter-Plugins in Kapitel 3.1.8 im Hauptdokument dieser Arbeit beschrieben. Mithilfe des Plugins kann eine Audit-Taugliches Log-Funktionalität ermöglicht werden. Dieses Plugin nimmt auch Konfigurationsänderungen auf, die während der Laufzeit vorgenommen werden. Ein geeignetes Log-Management Tool, wie z.B. Graylog[45], kann darauf konfiguriert werden, solche Änderungen zu erkennen und zu melden.

[45] https://www.graylog.org/

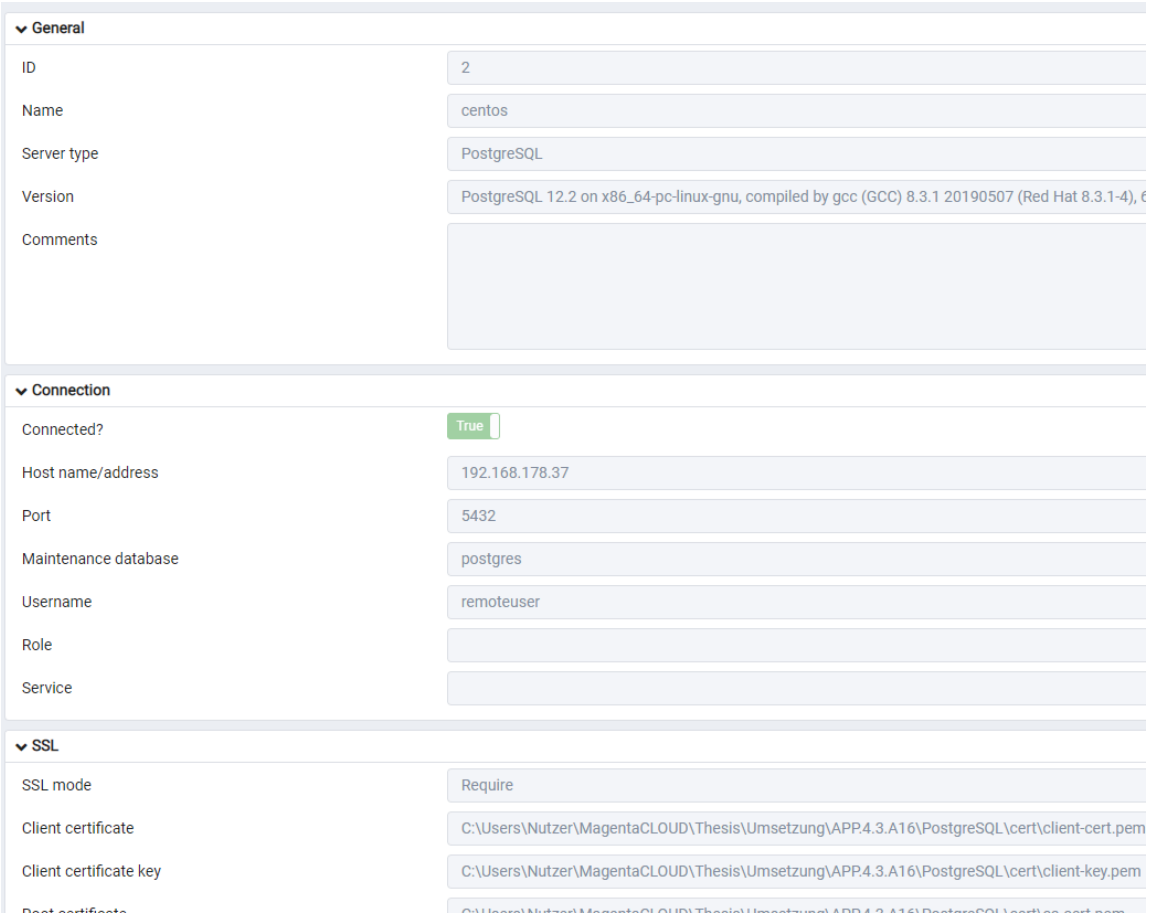
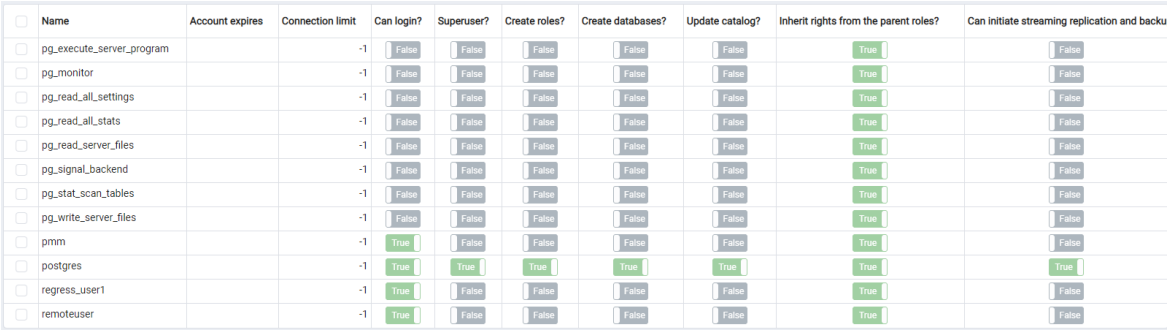


Abbildung 23 – Grafische Ansicht der Konfiguration in pgAdmin



Name	Account expires	Connection limit	Can login?	Superuser?	Create roles?	Create databases?	Update catalog?	Inherit rights from the parent roles?	Can initiate streaming replication and backu
pg_execute_server_program		-1	False	False	False	False	False	True	False
pg_monitor		-1	False	False	False	False	False	True	False
pg_read_all_settings		-1	False	False	False	False	False	True	False
pg_read_all_stats		-1	False	False	False	False	False	True	False
pg_read_server_files		-1	False	False	False	False	False	True	False
pg_signal_backend		-1	False	False	False	False	False	True	False
pg_stat_scan_tables		-1	False	False	False	False	False	True	False
pg_write_server_files		-1	False	False	False	False	False	True	False
pmm		-1	True	False	False	False	False	True	False
postgres		-1	True	True	True	True	True	True	True
regress_user1		-1	True	False	False	False	False	True	False
remoteuser		-1	True	False	False	False	False	True	False

Abbildung 24 - Grafische Verwaltung von Benutzern

Angehängte Dateien:

- PostgreSQL_SecurityTools.docx

A.4.17.3 MongoDB

A.4.17.3.1 Einsatz von Datenbank Security Tools für MongoDB

In der Anforderung „Einsatz von Datenbank Security Tools“ werden Sicherheitstools für die Datenbankysteme gefordert, die die folgenden Funktionen bereitstellen:

- Erstellung einer Übersicht über alle Datenbanksysteme,
- erweiterte Konfigurationsmöglichkeiten und Rechtemanagement der Datenbanken,
- Erkennung und Unterbindung von möglichen Angriffen (z. B. Brute Force Angriffe auf ein Benutzerkonto, SQLInjection) und
- Auditfunktionen (z. B. Überprüfung von Konfigurationsvorgaben).“

Derzeit existieren keine Informationssicherheitsprodukte für Datenbanken, die Opensource oder frei verfügbar sind. Einzelne Funktionen können über andere Systeme auf Opensource-Basis erfüllt werden. Im Folgenden werden Umsetzungsvorschläge für die Einzelfunktionen aufgezeigt.

A.4.17.3.1.1 Erstellung einer Übersicht über alle Datenbanksysteme

Mit dem Tool „Percona Monitoring and Management“, wird eine Übersicht über alle konfigurierten Datenbanksysteme erstellt. Außerdem kann hier der aktuelle Status des Systems mit abgefragt und individuelle Dashboards konfiguriert werden.

A.4.17.3.1.2 Erweiterte Konfigurationsmöglichkeiten und Rechtemanagement

Derzeit gibt es kein Administrationstool, welches die Konfiguration und das Rechtemanagement mit einer grafischen Benutzeroberfläche unterstützt. Es gibt aber viele Programme, die die Verwaltung von Collections und Documents ermöglicht, z.B. Robo3T[46] oder NoSQLBooster[47].

---

[46] https://robomongo.org/

[47] https://nosqlbooster.com/

A.4.17.3.1.3 Erkennung und Unterbindung von möglichen Angriffen

Für MongoDB gibt es derzeit kein freies Programm auf dem Markt, welches eine Angriffserkennung und -unterbindung ermöglicht.

A.4.17.3.1.4 Auditfunktionen

Ein Audit-Logging wurde auf Grundlage eines Drittanbieter-Plugins in Kapitel 3.1.8 im Hauptdokument dieser Arbeit beschrieben. Mithilfe des Plugins kann eine Audit-Taugliches Log-Funktionalität ermöglicht werden. Dieses Plugin nimmt auch Konfigurationsänderungen auf, die während der Laufzeit vorgenommen werden. Ein geeignetes Log-Management Tool, wie z.B. Graylog[48], kann darauf konfiguriert werden, solche Änderungen zu erkennen und zu melden.

Angehängte Dateien:

- MongoDB_SecurityTools.docx

A.4.18 APP.4.3.A23

- Archivierungskonzept_der_SUB_Hamburg.pdf
- Archivierungskonzept_InMi_NRW.pdf

A.4.19 APP.4.3.A25

A.4.19.1 MySQL

- MySAT-master.zip

A.4.19.2 PostgreSQL

- DbDat-master.zip

A.4.19.3 MongoDB

- mongoaudit-master.zip

---

[48] https://www.graylog.org/

10 Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich die hier vorliegende Arbeit selbstständig, ohne unerlaubte fremde Hilfe und nur unter Verwendung der in der Arbeit aufgeführten Hilfsmittel angefertigt habe.

München,

Ort, Datum (Henner Bendig)