Hochschule Wismar
University of Applied Sciences: Technology, Business and Design
Fakultät für Ingenieurswissenschaften, Bereich Elektrotechnik und Informatik

Bachelor-Thesis

„SICHERHEITSANALYSE VON MICROSOFT ACCESS“

Unter Berücksichtigung regulatorischer Anforderungen im Kredit- und Finanzdienstleistungswesen, Banking 4.0 & der Cloud-Technologie

Abschlussarbeit zur Erlangung des Grades eines

Bachelor of Engineering (B. Eng.)

in IT-Forensik

der Hochschule Wismar

Frankfurt am Main, den 24. August 2023

Eingereicht von: Nils Majewski
Studiengang: IT-Forensik

Erstgutachter: Prof. Dr. Antje Raab-Düsterhöft
Zweitgutachter: Prof. Dr. habil. Andreas Ahrens

Sicherheitsanalyse von Microsoft Access – Unter Berücksichtigung regulatorischer Anforderungen im Kredit- und Finanzdienstleistungswesen, Banking 4.0 & der Cloud-Technologie

Aufgabenstellung

Ziel der vorliegen Arbeit ist eine Sicherheitsanalyse von Microsoft Access:

Neben der Frage nach der Sicherheit von Microsoft Access ist auch die Frage zu beantworten, ob und, wenn ja, wann Microsoft Access eine Alternative zu „professionellen“ Datenbanksystemen sein kann.

Für die Sicherheitsanalyse sind Kriterien zu recherchieren und festzulegen, anhand die Bewertung erfolgen wird. Dabei wird auch eine Gegenüberstellung mit mindestens einem anderen, im Rahmen der Bachelor-Thesis auszuwählenden, Datenbanksystem durchgeführt.

Aus dem Ergebnis der Sicherheitsanalyse und des Vergleichs werden etwaige Handlungsempfehlungen für den Einsatz beziehungsweise Umgang mit Microsoft Access abgeleitet.

Kurzreferat

Thema: Sicherheitsanalyse von Microsoft Access

Microsoft Access, ein dateibasiertes Datenbanksystem, ist Teil der weitverbreiteten Microsoft 365-Suite und unter den Top 10 der weltweit beliebtesten Datenbanksysteme vertreten. Aufgrund der Beliebtheit von Access ist neben einer Bestandsaufnahme in Sachen Sicherheit auch wichtig zu wissen, ob und, gegebenenfalls wann Access eine Alternative zu „professionellen“ Datenbanksystemen sein kann.

Zur Beantwortung der Fragen und zur Identifikation potenzieller Handlungsempfehlungen zur Verbesserung der IT-Sicherheit wurde eine Sicherheitsanalyse auf Basis von Anforderungen aus dem Kredit- und Finanzdienstleistungswesen, als Vertreter kritischer Infrastruktur, durchgeführt.

Beim methodischen Vorgehen wurden neben Empfehlungen von Microsoft aktuelle (regulatorische) Anforderungen der Aufsichtsbehörden und gängige Sicherheitsstandards in Form von Bausteinen aus dem IT-Grundschutz, technische Sicherheitsanforderungen der Telekom AG und Cheat Sheets des Open Web Application Security Project berücksichtigt. Außerdem floss die Vision Banking 4.0 sowie die Schnittstelle zu Excel in die Analyse mit ein und es fand ein Vergleich mit der Microsoft Azure SQL-DB statt. Mit der Forensik-Plattform Autopsy wurde die Auffindbarkeit von Access-Dateien bestätigt.

Im Ergebnis wurden in Access massive Sicherheitslücken und technologische Nachteile gefunden. Auch das Azure-Cloud-Umfeld ist derzeit kein Garant für Sicherheit, wobei der Ansatz und Funktionsumfang durchweg überzeugend sind.

Diese Ausarbeitung zeigt zudem, dass in der Regel Alternativen zu Access zu suchen sind, die als Synergieeffekt neben der Erhöhung der Sicherheit auch den technologischen Wandel und die Zentralisierung der Datenhaltung unterstützen.

Abstract

Subject: Security Analysis of Microsoft Access

Microsoft Access, a file-based database system, constitutes a component of the widely pervasive Microsoft 365 Suite and is featured among the top 10 globally acclaimed database systems. Given the prevalence of Access, it becomes imperative, in addition to conducting a comprehensive security assessment, to ascertain whether and, if applicable, when Access could serve as an alternative to „professional“ database systems.

To address these inquiries and discern potential recommendations for enhancing IT security, a security analysis was conducted, grounded in requisites drawn from the realm of credit and financial services, representative of critical infrastructure. The methodological approach considered not only Microsoft's recommendations but also contemporary (regulatory) mandates from supervisory authorities and prevailing security standards, embodied as elements derived from the „IT-Grundschutz“ framework, technical security requisites from Telekom, or Cheat Sheets provided by the Open Web Application Security Project. Furthermore, the vision of Banking 4.0 and the interface with Excel were incorporated into the analysis, and a comparison with the Microsoft Azure SQL-DB was undertaken. The Autopsy forensic platform validated the traceability of Access files.

As a result, substantial security vulnerabilities and technological drawbacks were identified within Access, and it is noteworthy that the Azure cloud environment presently does not guarantee security. Nonetheless, the approach and functional scope remain consistently compelling.

This exposition also elucidates that it is generally advisable to explore alternatives to Access, which, as a synergistic effect, not only heighten security but also support technological advancement and centralized data management.

Inhaltsverzeichnis

1. Einleitung

Dieses Kapitel gibt dem Leser eine Einführung sowie einen allgemeinen Überblick über das in dieser Arbeit behandelte Thema und die behandelte Fragestellung.

1.1. Relevanz & wissenschaftlicher Mehrwert

Büroanwendungen wie *Microsoft* (MS) 365 als wichtigster Vertreter mit Word, Excel, Access oder PowerPoint gehören gemäß *Bundesamt für Sicherheit in der Informationstechnik* (BSI) in jeder Organisationsgröße inner- sowie außerhalb professioneller Informationstechnologie (IT)-Abteilungen zu den am häufigsten genutzten Anwendungsprogrammen. Diese Tools können leicht von Mitarbeitenden ohne fundiertes, technisches Know-how durch selbstentwickelte Makros erweitert werden. Dieser Umstand macht sie zu einem beliebten Angriffsziel und anfällig für die Erweiterung mit Schadcode [5]/[189]/[197]/[199]/[221]. Eine etwas in die Jahre gekommene aber mindestens genauso aktuelle Studie aus 2014/2015 von *Statista* zeigt, dass von 1.107 befragten Arbeitnehmenden[1], die mindestens 50 % ihrer Arbeitszeit am Computer verbringen, bei 92 % die Office-Suite von MS eingesetzt wird. Ein durchschnittlicher Büroarbeiter verbringt im Schnitt 72 % seiner Zeit mit Office-Standardsoftware. Pro Monat entstehen dabei durchschnittlich 600 E-Mails, 15 Tabellenkalkulationen, 20 Dokumente und 4 Präsentationen. Auch im Jahr 2020 kommt die Office-Suite noch auf 85 % Marktanteil (1.023 befragte Arbeitnehmende) [254]:

[1] Um für die Berücksichtigung des Gleichstellungsgedankens zu sensibilisieren, wird in der vorliegenden Thesis abwechselnd die männliche und weibliche Form verwendet. Gemeint sind jeweils beide Geschlechter.

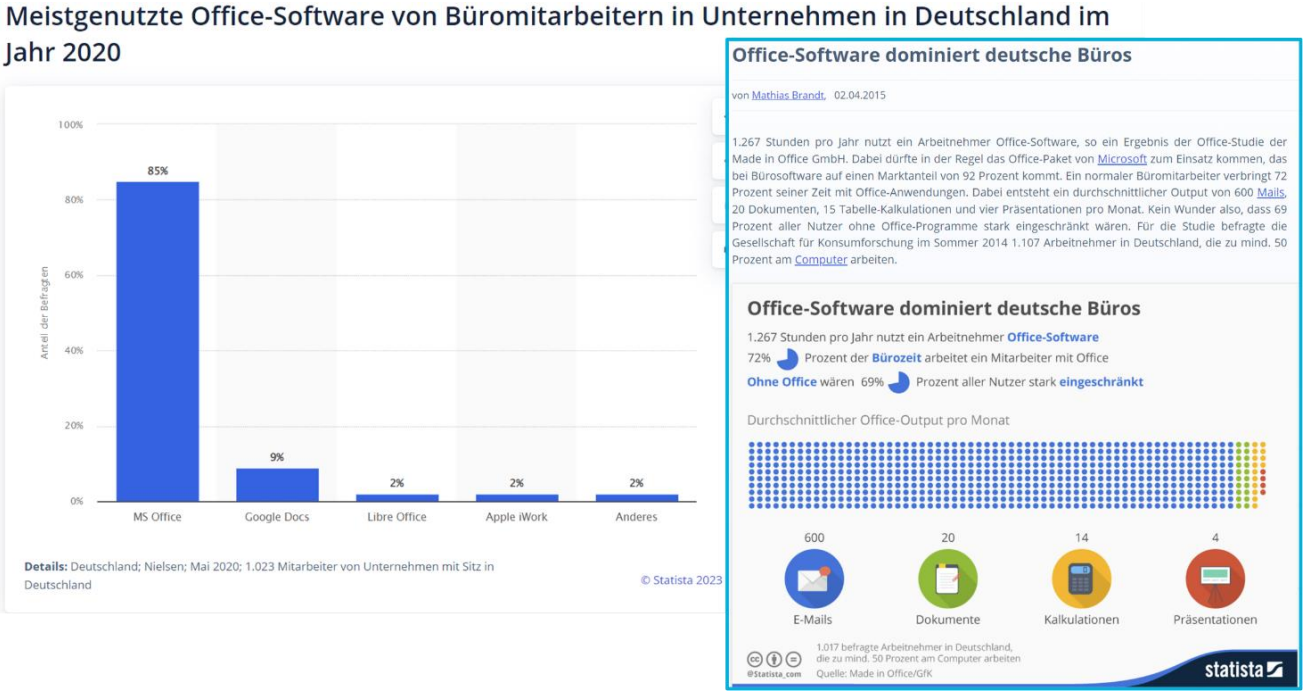


Bild 1: Links: Studie aus 2020 zur Office-Nutzung (Desktop & Online-Version) bei Arbeitnehmern (Statista) [255]; Rechts: Studie aus 2014/2015 zur Office-Nutzung bei Arbeitnehmern (Statista) [254]

Büroanwendungen werden auch in Sektoren der kritischen Infrastrukturen wie Energie, IT, Telekommunikation, Transport und Verkehr, Gesundheit, Wasser, Ernährung sowie im Finanz- und Versicherungswesen eingesetzt, die für die Aufrechterhaltung wichtiger gesellschaftlicher Funktionen unerlässlich sind. Daten und deren Analyse werden für den Unternehmenserfolg zunehmend wichtiger und stellen einen entscheidenden Wettbewerbsvorteil dar. Die Anfänge der Datenanalyse liegen mit „Business Intelligence“ in den 1990er Jahren. Aufgrund des Standes der Technik, der begrenzten Datenverfügbarkeit und der Analysemethoden sind diese jedoch nicht mit heutigen Verfahren vergleichbar. Speicherplatz wird vermehrt billiger, die Rechenleistung nimmt weiter zu, es findet eine „Datafizierung aller Lebensbereiche“ hin zu einer „datengetriebenen Kultur“ statt, in der zunehmend mehr Dinge digital vermessen und Entscheidungen zunehmend auf Grundlage von Daten getroffen werden. Die Menge an Daten verdoppelt sich in immer kürzeren Abständen. Die Industrialisierung schreitet immer weiter voran und die Fortschritte bei der künstlichen Intelligenz werden in naher Zukunft zunehmend zu mehr autonom handelnden IT-Systemen führen (*siehe auch Kapitel „4.1.9 Banking 4.0“*). Da Excel in der Regel zur Datenspeicherung und -verwaltung ungeeignet (*siehe Kapitel „3.5 Auswahl Office-Vergleichsprogramm zur Schnittstellenanalyse (Access Excel)“*) und anstelle dessen ein Datenbanksystem (DBS) zu präferieren ist, liegt der Fokus dieser Arbeit auf Access. MS 365 bietet mit Access eine dateibasierte, relationale DBS-Alternative zu Excel. Access belegt unter den populärsten Datenbankmanagementsystemen (DBMSs) gemäß Datenbank (DB)-Engines im August 2023 Platz 9 von insgesamt 420. Im Ranking unter den relationalen DBSs landet Access auf Platz 6 von 169. *DB-Engines* ist die Hauptquelle von einem *Statista*-Ranking über die weltweit beliebtesten DBMSs aus Februar 2023 [16]/[34]/[35]/[49]/[128]/[150]/[252]:

420 systems in ranking, August 2023

Rank Aug 2023	Rank Jul 2023	Rank Aug 2022	DBMS	Database Model	Score Aug 2023	Score Jul 2023	Score Aug 2022
1.	1.	1.	Oracle	Relational, Multi-model	1242.10	-13.91	-18.70
2.	2.	2.	MySQL	Relational, Multi-model	1130.45	-19.89	-72.40
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	920.81	-0.78	-24.14
4.	4.	4.	PostgreSQL	Relational, Multi-model	620.38	+2.55	+2.38
5.	5.	5.	MongoDB	Document, Multi-model	434.49	-1.00	-43.17
6.	6.	6.	Redis	Key-value, Multi-model	162.97	-0.80	-13.43
7.	↑8.	↑8.	Elasticsearch	Search engine, Multi-model	139.92	+0.33	-15.16
8.	↓7.	↓7.	IBM Db2	Relational, Multi-model	139.24	-0.58	-17.99
9.	9.	9.	Microsoft Access	Relational	130.34	-0.38	-16.16
10.	10.	10.	SQLite	Relational	129.92	-0.27	-8.95
11.	11.	↑13.	Snowflake	Relational	120.62	+2.94	+17.50
12.	12.	↓11.	Cassandra	Wide column, Multi-model	107.38	+0.86	-10.76
13.	13.	↓12.	MariaDB	Relational, Multi-model	98.65	+2.55	-15.24
14.	14.	14.	Splunk	Search engine	88.98	+1.87	-8.46
15.	↑16.	15.	Amazon DynamoDB	Multi-model	83.55	+4.75	-3.71
16.	↓15.	16.	Microsoft Azure SQL Database	Relational, Multi-model	79.51	+0.55	-6.67
17.	17.	17.	Hive	Relational	73.35	+0.48	-5.31
18.	18.	↑22.	Databricks	Multi-model	71.34	+2.87	+16.72
19.	19.	↓18.	Teradata	Relational, Multi-model	61.31	+1.06	-7.76
20.	20.	↑24.	Google BigQuery	Relational	53.90	-1.52	+3.87

include secondary database models — 169 systems in ranking, August 2023

Rank Aug 2023	Rank Jul 2023	Rank Aug 2022	DBMS	Database Model	Score Aug 2023	Score Jul 2023	Score Aug 2022
1.	1.	1.	Oracle	Relational, Multi-model	1242.10	-13.91	-18.70
2.	2.	2.	MySQL	Relational, Multi-model	1130.45	-19.89	-72.40
3.	3.	3.	Microsoft SQL Server	Relational, Multi-model	920.81	-0.78	-24.14
4.	4.	4.	PostgreSQL	Relational, Multi-model	620.38	+2.55	+2.38
5.	5.	5.	IBM Db2	Relational, Multi-model	139.24	-0.58	-17.99
6.	6.	6.	Microsoft Access	Relational	130.34	-0.38	-16.16
7.	7.	7.	SQLite	Relational	129.92	-0.27	-8.95
8.	8.	↑9.	Snowflake	Relational	120.62	+2.94	+17.50
9.	9.	↓8.	MariaDB	Relational, Multi-model	98.65	+2.55	-15.24
10.	10.	10.	Microsoft Azure SQL Database	Relational, Multi-model	79.51	+0.55	-6.67
11.	11.	11.	Hive	Relational	73.35	+0.48	-5.31
12.	12.	↑14.	Databricks	Multi-model	71.34	+2.87	+16.72
13.	13.	↓12.	Teradata	Relational, Multi-model	61.31	+1.06	-7.76
14.	14.	↑16.	Google BigQuery	Relational	53.90	-1.52	+3.87
15.	15.	15.	FileMaker	Relational	53.85	+0.53	+0.73
16.	16.	↓13.	SAP HANA	Relational, Multi-model	50.66	-0.07	-4.30
17.	17.	17.	SAP Adaptive Server	Relational, Multi-model	43.69	+0.82	-1.12
18.	18.	↑22.	Microsoft Azure Synapse Analytics	Relational	26.97	-0.65	+4.51
19.	19.	19.	Firebird	Relational	25.99	-0.17	+1.05
20.	20.	20.	Informix	Relational, Multi-model	22.62	+0.69	-1.54

Bild 2: Links: DB-Engines Ranking über die weltweit beliebtesten DBMSs aus August 2023 [35]; Rechts: DB-Engines Ranking über die weltweit beliebtesten relationalen DBMSs aus August 2023 [34]

Aber auch die IT-Sicherheit gewinnt zunehmend an Bedeutung. Angriffe werden ausgeklügelter, das technisches Know-how bei Angreifenden wächst und neue, technische Fortschritte schaffen neue Bedrohungsszenarien. Gemäß MS und ihrer „Zero-Trust-Strategie“ („niemals vertrauen und immer überprüfen“, *siehe Kapitel „Anlage 1: Zero-Trust-Strategie“*) ist es nicht mehr zeitgemäß, dass Unternehmen den Netzwerkzugriff nur mit Virtual Private Networks (VPN) und Firewalls schützen. Aufgrund der Möglichkeiten durch neue Technologien wie Cloud-Computing oder Homeoffice muss mehr für die Sicherheit getan werden. Es gilt jede beim Datenaustausch beteiligte Schicht („Identity“, „Endpoints“, „Applications“, „Network“, „Infrastructure“, „Data“) zu schützen, da sie jederzeit ein potenzielles Angriffsziel für den Abfluss sensibler Daten darstellen kann. Da nicht davon ausgegangen werden kann, dass alle Kommunikationspartner vertrauenswürdig sind, muss jede Anfrage auch in einem internen Netzwerk so überprüft werden, als ob sie aus einem öffentlichen Netzwerk stammt [174]/[193, S. 1]/[223, S. 1].

Es ist daher unerlässlich zu wissen, ob die gespeicherten Daten in einer Access-Anwendung sicher abgelegt sind und, ob Access im Vergleich zu „professionellen“ Client-Server-DBSs eine Chance hat. Welche potenziellen Schwachstellen hat Access und welche Maßnahmen können ergriffen werden, um die Sicherheit zu erhöhen? Stehen nach einem erfolgreichen Angriff genügend verlässliche Quellen zur Spurensicherung zur Verfügung? Gemäß BSI IT-Grundschutz-Baustein „APP.4.3 Relationale Datenbanken“ (*siehe Kapitel „3.2 Identifikation von gängigen Sicherheitsstandards für relationale DBSs (Kriterienanalyse)“*) hat jede Institution eine nachvollziehbare und verpflichtend einzuhaltende Sicherheitsrichtlinie für DBSs, mit Anforderungen und Vorgaben zum sicheren Betreiben, zu erstellen. Abweichungen von dieser Sicherheitsrichtlinie müssen dokumentiert und der Einhaltungsgrad in Stichprobenkontrollen überprüft werden [8]/[9]. Die vorliegende Arbeit führt eine Sicherheitsanalyse von MS Access als Vertreter dateibasierter DBSs durch und gibt ergebnisabhängige Handlungsempfehlungen für die Nutzung.

Darüber hinaus werden zumindest am Rande auch aktuelle technische Entwicklungen wie Cloud-Computing oder künstliche Intelligenz (KI) einbezogen.

1.2. Abgrenzung

Um den Rahmen der vorliegenden Arbeit nicht überzustrapazieren und zur Fokussierung werden folgende Punkte abgegrenzt:

- Der Fokus liegt auf MS Access Desktop-Anwendungen in den 2007-2016 Dateiformaten .accdb und .accde. Die Web-App-Möglichkeiten wie der Organisations-App-Store in MS Teams oder die SharePoint-Integration (veraltet und von MS nicht mehr empfohlen) werden in der Sicherheitsanalyse nicht berücksichtigt.
- Das überholte Dateiformat .mdb von Access inklusive der dort verfügbaren Sicherheit auf Benutzerebene (User-Level-Security) bleibt unberücksichtigt.
- Es erfolgt keine Analyse der Kommunikation zwischen Dateiserver und Client. Es wird angenommen, dass die Kommunikation verschlüsselt erfolgt.
- Die Sicherheitsanalyse erfolgt mit den Standardkonfigurationen der Computer- und Benutzerrichtlinien inklusive MS 365, dem Trust-Center, der Windows Registry, dem Windows EventLog und der Computerverwaltung.
- Es findet keine detaillierte Analyse der Überwachungsfunktionalitäten des Betriebs- und Dateisystems wie Überwachungsmöglichkeiten des New Technology File System (NTFS) statt. Hierfür bedarf es einer separaten Betrachtung.
- Es bleiben Risiken unberücksichtigt, die durch Fehler in der Implementierung entstehen. Die Gründe hierfür sind vielfältig und reichen von Zeitdruck bei der Implementierung über eine ungenügende Testphase, bis hin zu ungenügender Qualität des Testdatenbestands. Ausgeschlossen wird außerdem das Risiko, wenn eine Access-Datei mit kritischen Daten auf Netzlaufwerken abgelegt und „vergessen“ wird [33]. An dieser Stelle sei auf die Richtlinie zur Durchführung einer Risikoanalyse vom BSI verwiesen [14].
- Es erfolgt keine Definition von Risiko, Kritikalität oder „kritischem Zweck“. Die Sicherheitsanalyse erfolgt unabhängig von einer Kritikalitäts- beziehungsweise Schutzbedarfseinschätzung. Aus diesem Grund erfolgt auch keine Gewichtung der einzelnen Bewertungskriterien, da diese sich von Anwendungsfall zu Anwendungsfall unterscheiden kann. Darüber hinaus werden Fachbegriffe aus dem Kredit- und Finanzdienstleistungswesen nur oberflächlich erklärt, da der Fokus auf den technischen Inhalten liegt.
- Bei der Schnittstellenanalyse wird sich beim Office-Vergleichsprogramm Excel auf die Dateitypen .xlsx und .xlsm beschränkt. Für das Excel-Binärformat .xlsb bedarf es einer separaten Analyse [63].
- Bei der Auswahl des Vergleichs-DBS wird sich auf relationale DBSs aus dem MS-Umfeld beschränkt.
- Die Wahl des Programms zur Schnittstellenanalyse wird aufgrund der weiten Verbreitung auf MS 365 eingegrenzt.
- Es erfolgt keine datenschutzrechtliche Analyse bis ins kleinste Detail und keine Analyse des Auftragsverarbeitungsvertrages „MS Products and Services Data Protection Addendum“.

- Aufgrund des Deprecated-Status der MS Joint Engine Technology (JET) für Access wird keine detaillierte Analyse durchgeführt.
- Es erfolgt keine Analyse des Funktionsumfangs aller Aktionen, Programmiertools oder Makroaktionen, die ohne Erteilen eines expliziten vertrauenswürdigen Status genutzt werden können [187].
- Es erfolgt keine Betrachtung der unterschiedlichen Kaufmodelle für die Azure Structured Query Language (SQL)-DB.
- Die Azure SQL-DB wird lediglich als Vergleichs-DBS, während der Kriterienanalyse herangezogen, es erfolgt jedoch keine detaillierte Sicherheitsanalyse. Der Fokus liegt auf dem Funktionsumfang.
- Es findet kein Vergleich zwischen kostenlosen und kostenpflichtigen forensischen Plattformen statt.

1.3. Anwendungsszenario

Zum Herstellen eines Praxisbezugs wird sich auf das Kredit- und Finanzdienstleistungswesens als Vertreter von kritischer Infrastruktur beschränkt (*siehe Kapitel „1.1 Relevanz & wissenschaftlicher Mehrwert“*).

Ausgangspunkt ist eine einfache Access-Datei zur Datenhaltung oder eine mittels der dedizierten MS 365-Programmiersprache Visual Basic for Applications (VBA) individualisierte MS Access-Büroanwendung, die außerhalb einer professionellen IT-Abteilung von einem Nicht-IT-Mitarbeitenden mit wenig IT-Know-how erstellt und betrieben wird (dezentrale IT) [30]. Derartige Anwendungen werden gemäß aktuellen regulatorischen Anforderungen auch als „Individuelle Datenverarbeitung (IDV)-Anwendungen“ bezeichnet. Sie werden explizit dem Bereich der Anwendungsentwicklung zugeordnet, sodass Anforderungen aus der professionellen Anwendungsentwicklung, wie die Einhaltung von Programmierrichtlinien oder das verpflichtende Durchlaufen eines geregelten Software-Entwicklungsprozesses, gelten (*siehe Kapitel „2.6 Aktueller Stand: Regulatorische Anforderungen im Kredit- & Finanzdienstleistungswesen“ sowie „Allgemeiner Teil (AT) 7.2 Technisch-organisatorische Ausstattung“ Teilziffer (Tz) 5 Mindestanforderungen an das Risikomanagement (MaRisk)“*) [21, S. 19]/[22, S. 23].

Neben Access setzen IDV-Anwendungen häufig auch auf MS Excel als Plattform (*siehe Kapitel „3.5 Auswahl Office-Vergleichsprogramm zur Schnittstellenanalyse (Access Excel)“*). Autark aber auch in unterschiedlichsten Kombinationen wie als Excel-Frontend zur Nutzerinteraktion sowie Ablageort der Logik und einem Access-Backend zur Datenhaltung („Access Excel“). Die Anwendungsbereiche sind vielseitig und reichen vom Reporting (Aggregation von Eingabedaten) über bloße Arbeitserleichterungen als Workaround zur Automatisierung von manuellen Schritten oder Optimierung von Teilprozessschritten.

Dieses Bild bestätigt sich auch außerhalb des Kredit- und Finanzdienstleistungssektors bei Vergleichsplattformen für Unternehmenssoftware wie *GetApp*, *TrustRadius*, *Capterra* oder *SOURCEFORGE*. Hier beschriebene Anwendungsfälle reichen von der Verwaltung von über 20.000 Artikeln eines Online-Handels, über eine Kunden-

kontaktverwaltung eines Buchladens bis hin zur Nutzung in einem Call-Center, um Anrufe und zugehörige Gesprächsergebnisse zu verwalten [24]/[46]/[246]/[264].

Da mit Access komplexe Geschäftsanwendungen mit einem Graphical User Interface (GUI) erstellt werden können, sind auch aus dem Web-Anwendungsumfeld bekannte SQL-Injection-Angriffe, die im Jahr 2022 den ersten Platz unter den Bedrohungen für Web-Applikationen belegten, zu berücksichtigen [253]. In diesem Anwendungsszenario geht es jedoch nicht um einen Angriff auf eine Web-Anwendung über das Internet, sondern um einen böswilligen Mitarbeitenden, der nur über ein in die Access-Datei integriertes Formular (GUI) mit Eingabetext-Steuerelementen Zugriff auf die Daten besitzt und eine Datenmanipulation oder einen Datenabzug vornehmen möchte. Sind die Daten in der Access-Datei verschlüsselt, bleibt nur der Weg über das GUI *(sollten etwaige in dieser Ausarbeitung aufgezeigte Schwachstellen unbekannt sein)*.

Büroanwendungen wie MS Access werden häufig von Nicht-IT-Spezialisten mit VBA-Programmcode oder über die Oberfläche erstellte Abfragen erweitert. Damit dabei möglichst keine Bedrohungen aus einem falschen Umgang entstehen, sind eine möglichst intuitive sowie nutzerfreundliche Unterstützung in Form von nutzbaren, gehärteten VBA-Funktionen zur SQL-Datenabfrage oder über die Oberfläche konfigurierbare Datenüberprüfungsfunktionalitäten wichtig. Fatal ist, wenn sich Nutzende aufgrund mangelnder IT-Kenntnisse oder intransparenten Herstellerversprechungen in falscher Sicherheit wiegen.

1.4. Struktur der vorliegenden Arbeit

Insgesamt wird auf vermeidbare Wiederholungen, in dieser Ausarbeitung so gut es geht verzichtet. Daher erfolgen teilweise Verweise auf Folgekapitel, um den Schwerpunkt der jeweiligen Kapitel nicht zu verzerren und einen Bezug zwischen den Kapiteln herzustellen. Die verbleibenden Redundanzen sind absichtlich für ein besseres Verständnis eingefügt. Um Bilder auch in der Druckversion erkennen zu können, wurden sie absichtlich vergrößert. Dies kann jedoch zu beabsichtigten Lücken im Text führen, wenn das Bild auf die nächste Seite umgebrochen wird. Ausgangspunkt der vorliegenden Arbeit ist das in *Kapitel „1.3 Anwendungsszenario“* skizzierte Anwendungsszenario. Der auf das Kredit- und Finanzdienstleistungswesen gesetzte Fokus dient als Rahmen für die durchzuführenden Schritte.

Aufgrund des gesteckten Rahmens erfolgt in *Kapitel „2 Kontext & Begriffsbestimmungen“* die Analyse des aktuellen Stands der regulatorischen Anforderungen im Kredit- und Finanzdienstleistungswesen sowie ein Blick in die mögliche Zukunft in Form von der Vision Banking 4.0: Der Trend geht zu autonom handelnden IT-Systemen, der maßgeblich von den Entwicklungen im Umfeld der künstlichen Intelligenz abhängt. Darüber hinaus sind in diesem Kapitel Informationen zur eingesetzten Software, relevanten Begriffen zu relationalen DBSs sowie Beispielen für Datenzugriffe zu finden. Da es sich bei Access um ein dateibasiertes DBS handelt, wird auch der Unterschied zu Client-Server-Ansätzen aufgezeigt.

Der weitere Aufbau der aufeinander aufbauenden Kapitel ist sequenziell gestaltet und beginnt mit der Vorbereitung in *Kapitel „3 Vorbereitung“*. Aus den während der Vorbereitung identifizierten regulatorischen Anforderungen sowie Anforderungen resultierend aus der Vision Banking 4.0 werden im ersten Schritt Anforderungen an ein relationales DBS im Kredit- und Finanzdienstleistungswesen ermittelt. Ohne zu viel vorwegzunehmen sei an dieser Stelle darauf hingewiesen, dass die Gestaltung der IT-Systeme und zugehörigen IT-Prozesse auf der Basis von gängigen Standards erfolgt, die im weiteren Verlauf ausgewählt werden. Auf Grundlage der ausgewählten Sicherheitsstandards werden Bewertungskriterien festgelegt, die schließlich für die Kriterienanalyse als Teil der Sicherheitsanalyse verwendet werden. Das ausgewählte Vergleichs-DBS (Azure SQL-DB) ist für die Kriterienanalyse relevant, das Office-Vergleichsprogramm (Excel) für die Schnittstellenanalyse und das forensische Tool (Autopsy) für die forensische Analyse.

Nachdem alle Vorbereitungsschritte erfüllt sind, folgt in *Kapitel „4 Durchführung der Sicherheitsanalyse“* die eigentliche Sicherheitsanalyse bestehend aus vier Teilen. Die Sicherheitsanalyse hat das Ziel konkrete Schwachstellen, Bedrohungen sowie Risiken bei der Verwendung von MS Access zu identifizieren. Die konkreten Bewertungskriterien der bereits erwähnten Kriterienanalyse resultieren aus den Anforderungen an ein relationales DBS im Kredit- und Finanzdienstleistungswesen. Während der Kriterienanalyse wird geprüft, ob alle Sicherheitsanforderungen, die nach den geltenden Normen ermittelt werden, erfüllt sind. Die Kriterienanalyse wird jeweils für MS Access und das Vergleichs-DBS durchgeführt, um eine Gegenüberstellung beider Technologien herauszuarbeiten. In der Schnittstellenanalyse wird das ausgewählte Office-Vergleichsprogramm analysiert. Das Ziel ist herauszufinden, ob das Vergleichsprogramm ein geeignetes, erstes Angriffsziel darstellt, um hierüber erste Informationen für eine darauffolgende Kompromittierung des dahinterliegenden MS Access-Backend zu erhalten. In der Dateiformatanalyse wird mit Hilfe eines Hexadezimal-Editors geprüft, ob Access Schwachstellen, wie das Ablegen von Zeichenketten im Klartext, aufweist. Schwerpunkt der forensischen Analyse ist das Auffinden aller in einem präparierten Image versteckten Access-Dateien mittels der während der Vorbereitungsphase ausgewählten Forensik-Software. Dabei gilt es auch potenzielle Verbesserungsvorschläge zu identifizieren.

In *Kapitel „5 Auswertung der Sicherheitsanalyse“* erfolgt die Auswertung der Sicherheitsanalyse sowie die Identifikation gefundener Bedrohungen und Risiken.

Kapitel „6 Handlungsempfehlungen & Ausblick“ gibt anhand den in der Sicherheitsanalyse aufgedeckten Bedrohungsszenarien konkrete Handlungsempfehlungen zur Steigerung der IT-Sicherheit (Ergebnis der Sicherheitsanalyse), inklusive einem Ausblick.

Kapitel „*7 Fazit*“ fasst die Erkenntnisse zusammen und gibt eine Antwort auf die Frage der Aufgabenstellung, ob und gegebenenfalls wann Access eine Alternative zu „professionellen“ DBSs sein kann. Als Zusammenfassung für die Ergebnisse der Sicherheitsanalyse in Form der Handlungsempfehlungen werden konkrete technisch-organisatorische Maßnahmen (TOM) für die aufgedeckten Bedrohungsszenarien genannt.

2. Kontext & Begriffsbestimmungen

In diesem Kapitel werden nach der vorangegangenen Einführung und Rahmensetzung relevante Begriffe und Details erläutert sowie ein tieferer Einstieg in die Thematik gegeben.

2.1. Auszug relevanter Begriffe im relationalen DBS-Umfeld

MS Access sowie das ausgewählte Vergleichs-DBS (*siehe Kapitel „3.4 Auswahl Vergleichs-DBS zur Kriterienanalyse (Azure SQL-DB)“*) basieren jeweils auf dem relationalen Ansatz (zur Datenmodellierung wird unter anderem das „Entity-Relationship-Modell“ eingesetzt).

Im relationalen Modell werden logisch zusammengehörende Datensätze (Zeilen), auch „Tupel“ oder „Entitäten“ genannt, in einer Tabelle, auch als „Relation“, „Entitätstyp“ oder „Entitätsmenge“ bezeichnet, gespeichert. Eine Entität ist dabei mit einem konkreten Objekt aus der realen Welt, wie einem konkreten Mitarbeitenden, gleichzusetzen. Eine Entität besitzt über den zugrundeliegenden Entitätstypen Eigenschaften („Attribute“), wie einen Vor- oder Nachnamen sowie einen eindeutigen Identifikator („Primärschlüssel“). Der Primärschlüssel wird aus einem oder mehreren Attributen gebildet. Ein Beispiel für einen Entitätstypen ist eine Tabelle für alle Mitarbeitenden-Entitäten. Je Attribut des Entitätstyps wird eine Spalte in der Tabelle angelegt. Eine Relation wird durch einen Namen, wie „`T_Mitarbeiter`“, und seine Attribute eindeutig beschrieben. Die konkreten Ausprägungen der einzelnen Entitätstyp-Attribute einer Entität, wie „Max“ und „Mustermann“, werden „Attributwerte“ genannt. Über Beziehungen können Zusammenhänge zwischen Entitäten verschiedener Entitätstypen mittels Fremdschlüsselbezügen realisiert werden. Auch logisch zusammengehörende Beziehungen lassen sich in einem „Beziehungstyp“, auch als „Beziehungsmenge“ bezeichnet, zusammenfassen. Der Beziehungstyp wird über die Menge der beteiligten Relationen und deren jeweils relevante Attribute klassifiziert. Über „Kardinalitäten“ wird angegebenen, wie viele Entitäten je Entitätstyp bei einer Beziehung beteiligt sind. Mitarbeitende können nur in einer Abteilung arbeiten aber eine Abteilung kann aus mehreren Mitarbeitenden bestehen (n:1-Beziehung). Die Attribute liegen in atomarer Form, also pro Attribut nur ein konkreter und nicht weiter zerlegbarer Attributwert, vor („Normalisierung“, Vor- und Nachname werden nicht in einem Attribut „Name“ abgelegt). Beziehungen zwischen den einzelnen Entitäten unterschiedlicher Relationen werden über Attributwerte hergestellt oder bilden bei n:m-Beziehungen eine eigene Beziehungsrelation (Primärschlüssel- und Fremdschlüsselbezüge). Da Relationen Mengen sind, ist die Reihenfolge der Tupel nicht zwingend festgelegt:

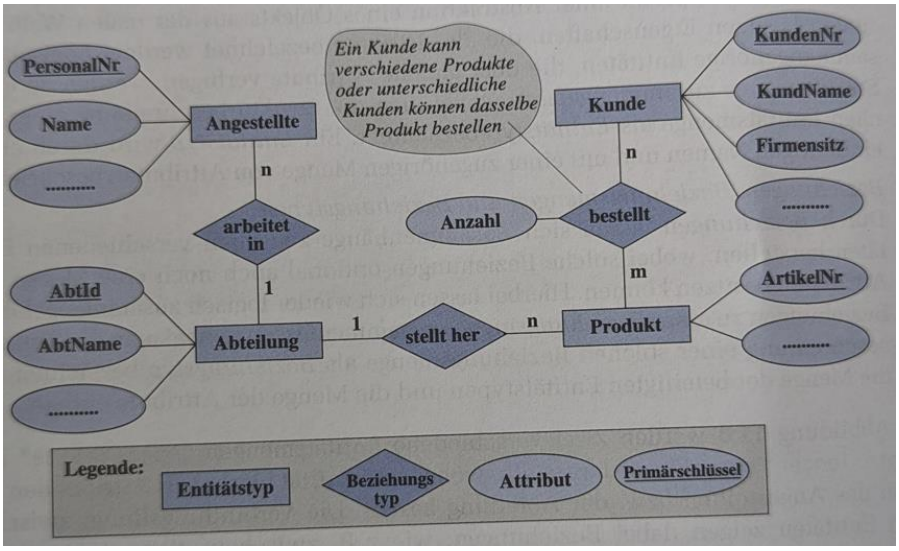


Bild 3: Übersicht relevanter Begriffe aus dem DBS-Umfeld anhand eines Entity-Relationship-Modells – Tabelle/Relation/Entitätstyp/Entitätsmenge (wie „Angestellte“), zum Entitätstyp gehörende Eigenschaften/Attribute (wie „Vor-„ oder „Nachname“, Spalte), Datensatz/Tupel/Entität (Zeile) mit Attributwerten (wie „Max“, „Mustermann oder „Erika“, „Musterfrau“) und Beziehungstypen zwischen unterschiedlicher Entitätstypen [57, S. 524]

Eine oder mehrere gemäß Regeln in einer Beziehung miteinander stehende Relationen mit den enthaltenen Daten werden in einer DB gespeichert [57, S. 523-525].

Die Kombination aus einer oder mehreren DBs mit dem zugehörigen DBMS (im relationalen Ansatz Relationales Datenbankmanagementsystem (RDBMS)) wird DBS genannt.

DBSs werden zur konsistenten sowie geordneten Speicherung von großen Datenmengen verwendet. DBSs besitzen Merkmale/Anforderungen wie das Verhindern oder Kontrollieren von Datenredundanzen, das Gewährleisten der Datenintegrität sowie der Datensicherheit, das Ermöglichen eines effizienten sowie parallelen Datenzugriffs oder eine Zentralisierung der Datenablage. Sie ermöglichen die Wiederherstellbarkeit nach einem unerwarteten Fehler und unterstützen die Unabhängigkeit von Anwendungsprogramm und Datenhaltung.

Das DBMS ist verantwortlich, die zuvor beschriebenen Merkmale eines DBS zu erfüllen. Ein DBMS stellt die Schnittstelle zwischen der DB und den Benutzern sowie nutzenden Programmen dar. Nutzende kommunizieren stets mit dem DBMS und niemals direkt mit der DB. Dabei sorgt das DBMS unter anderem bei Datenabfragen über eine Abfragesprache (bei RDBMS SQL) für einen effizienten Datenzugriff, ermöglicht die Anlage neuer DBs und gewährleistet die Wiederherstellbarkeit nach einem Fehler. Es regelt die zentrale Zugriffssteuerung wie die Verwaltung von Benutzern oder die Vergabe von Zugriffsrechten. Die Administration mehrerer DBs kann zentral über ein DBMS erfolgen. Das DBMS nutzt die von der DB-Engine angebotenen Funktionalitäten.

Die DB-Engine ist der zentrale Dienst zum Speichern und Verarbeitern von Daten. Dabei stellt sie einen kontrollierten Datenzugriff und die Verarbeitung der Transaktionen (mehrere aufeinanderfolgende Aktionen, die in einem Zusammenhang stehen und daher

gemeinsam als einzelne Arbeitseinheit ausgeführt werden) sicher. Beschrieben werden die einzelnen DBs eines DBS mittels Data Dictionary. Das Data Dictionary beschreibt die Struktur der DBs, also wie die Daten hier gespeichert werden.

Strenggenommen handelt es sich nach den vorherigen Ausführungen bei Access um ein DBMS und nur in Kombination mit der DB inklusive Daten um ein DBS. Zur Vereinfachung und, um auch die eigentliche Datenhaltung einzuschließen, wird in der vorliegenden Arbeit nicht scharf zwischen beiden Begriffen unterschieden und beide Ausdrücke daher teilweise als Synonym verwendet [34]/[35]/[98]/[230]/[265, S. 525-532].

Die nachfolgende linke Grafik verbildlicht die Rolle des DBMS als zentraler Zugriffspunkt für Nutzende. Die Rechte Illustration zeigt die dreischichtige Architektur eines DBS:

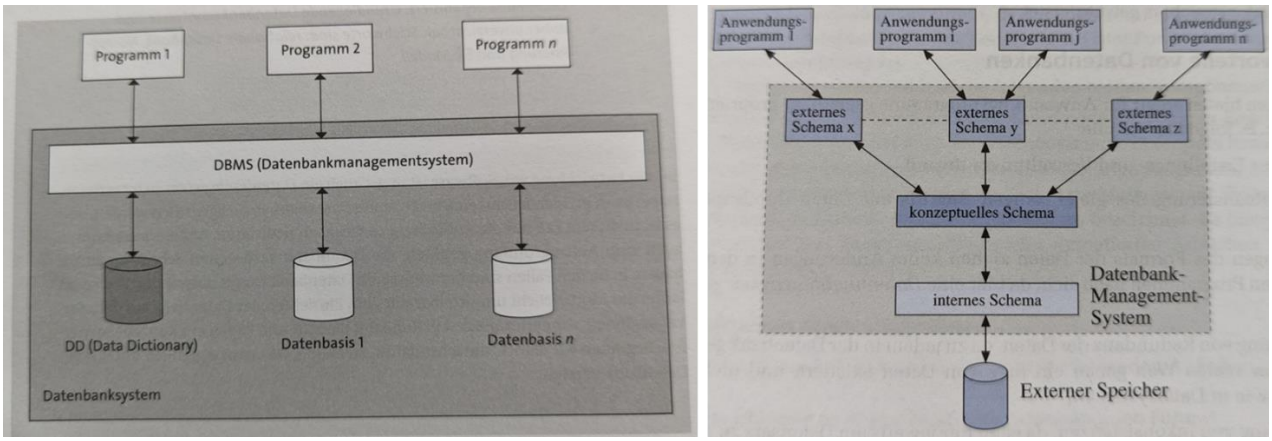


Bild 4: Links: Aufbau eines DBS und Rolle des DBMS [265, S. 526]; Rechts: Drei-Schichten-Architektur eines DBS [57, S. 522]

Bei der untersten Schicht handelt es sich um die physikalische Schicht oder auch interne Schicht genannt, welche die auf Speichermedien persistierten Rohdaten umfasst beziehungsweise wie die Daten abgelegt sind. Die Daten werden in der untersten Schicht in einer physikalischen Struktur (strukturierte Sammlung von Bits) auf dem Speichermedium gespeichert, um einen effizienten Datenzugriff zu ermöglichen. Beispiele für die physikalische Struktur von persistierten Daten sind Hash-Tabellen, balancierte Bäume oder verkettete Listen.

Die mittlere, konzeptionelle Schicht stellt die logische Sicht auf die Daten und somit das Modell (vergleiche Entity-Relationship-Modell) der jeweiligen DB dar. Im relationalen Ansatz wird die logische Sicht mittels miteinander zusammenhängender Relationen inklusive der Attribute repräsentiert. Es werden alle Daten in der DB beschrieben. Nutzende benötigen Kenntnisse über diese Schicht, um in der externen Schicht Abfragen erstellen zu können.

Die oberste, externe Schicht umfasst die für jeden Anwendungsfall individuellen Sichten auf die Daten (Benutzersichten), die für Nutzende (Views) oder von diesen über SQL-Abfragen zur Verfügung gestellt werden.

Hauptvorteil an der Schichtentrennung in einem DBS ist, dass die unterschiedlichen Schichten unabhängig voneinander sind. Durch die mehrschichtige Architektur können

die einzelnen Komponenten wie das DBMS ausgetauscht werden, ohne dabei die in den DBs gespeicherten Daten anpassen zu müssen. Die gespeicherten Daten können statt in einer Hash-Tabelle auch in einer Baumstruktur gespeichert werden, ohne dass diese Änderung das DBMS oder das verwendete SQL-Statement zur Datenabfrage beeinflusst [57, S. 522-524]/[258, S. 117 ff.].

Ein DBS stellt eine Datenquelle dar, zu der auch die benötigten Verbindungsinformationen zum Zugriff auf die Daten der Datenquelle gehören. Beispiele für Datenquellen sind MS Access, MS SQL Server, Oracle RDBMS, eine Kalkulationstabelle oder auch eine Textdatei. Beispiele für Verbindungsinformationen sind Server-Name, DB-Name, Port, Anmelde-Identifikator (ID) und Kennwort oder vergleichbare Merkmale zur Authentisierung sowie verschiedene Open Database Connectivity (ODBC)-Treiberoptionen, die beschreiben, wie die Verbindung mit der Datenquelle hergestellt wird. Nutzt eine Anwendung wie MS Access die ODBC-Architektur, stellt sie zum ODBC-Treibermanager, der auf dem jeweiligen Betriebssystem installiert ist, eine Verbindung her. Dieser verwendet wiederrum einen spezifischen ODBC-Treiber, wie den MS SQL ODBC-Treiber, um die Verbindung mit einer konkreten Datenquelle herzustellen (*siehe Kapitel „2.2 Beispiele für Datenzugriffe in Access“*). Unter Windows kann der Anwender mit dem System-Programm „ODBC-Datenquellen“ weitere Informationen zum Treiber einsehen. In Access werden ODBC-Datenquellen genutzt, um Verbindungen mit zu Access externen Datenquellen herzustellen, die keine integrierten Treiber besitzen [217].

2.2. Beispiele für Datenzugriffe in Access

Um auf Daten in Office-Dateien wie einer Access- oder Excel-Datenquelle (*siehe Kapitel „2.1 Auszug relevanter Begriffe im relationalen DBS-Umfeld“*) zuzugreifen oder zu bearbeiten, unterstützt Access diverse Schnittstellen [198]. Mit diesen Schnittstellen beziehungsweise Application Programming Interfaces (API)-Spezifikationen/Standards erfolgt ein einheitlicher Datenzugriff sowie eine standardisierte Datenmanipulation für unterschiedliche Datenquellen wie DBSs oder auch Excel-Arbeitsmappen. Zwei dieser standardisierten DBMS-API-Spezifikationen für relationale und nichtrelationale Zugriffe, Object Linking and Embedding, Database (OLE DB) sowie ODBC, werden nachfolgend vorgestellt.

OLE DB und ODBC sind Protokolle mit jeweils unterschiedlichen Ansätzen zur Kommunikation innerhalb eines Prozesses aber auch zur Interprozesskommunikation (Datenaustausch), wobei OLE DB aktueller ist [217].

Beide Schnittstellen stellen jeweils eine Abstraktionsebene in Form eines Treibers (bei ODBC) oder Anbieters (bei OLE DB) je DBMS dar, welche die DBMS spezifischen Befehle für den eigentlichen Zugriff kapseln und Nutzenden einen einheitlichen, DBMS unspezifischen Zugriff anbieten und das unabhängig von der jeweiligen verwendeten Programmiersprache oder des Betriebssystems. Dadurch müssen sich Softwareentwickler nur einen API-Standard merken und nicht die einzelnen DBMS spezifischen Zugriffe, die auch von Programmiersprache und Betriebssystem abhängig sind. Anwendungen können die angebotenen Schnittstellen wie ODBC oder OLE DB nutzen, ohne sie in die jeweilige Applikation kompilieren zu müssen. OLE DB-Anbieter sowie ODBC-

Treiber liegen je Datenquelle in Form einer Dynamic Link Library (DLL) vor, die zum Zugriff auf das jeweilige DBMS benötigt wird. Vorteil daran ist, dass der Quellcode nicht für jede einzelne Datenquelle neu kompiliert werden muss, denn die Verknüpfung zwischen einer Applikation und einer konkreten Datenquelle ist in die jeweiligen ODBC-Treiber und OLE DB-Anbieter ausgelagert. Treiber und Anbieter dienen als Vermittler (Provider). Die Applikation ist der Verbraucher (Consumer), sie stellt eine Anfrage an den Vermittler, der bei einer lesenden Anfrage eine Kopie der angefragten Daten an die Applikation zurückgibt. Dieser Umstand ermöglicht Cross-Plattform-Support, durch den Anwendungen auch über diverse ODBC-Treiber und OLE DB-Anbieter dynamisch neue Datenquellen hinzugefügt werden können, ohne für jede einzelne Datenquelle den Quellcode der Anwendung neu kompilieren zu müssen. Dieser Umstand erleichtert die Wartung, hat einen positiven Effekt auf die Erweiterbarkeit und erleichtert eine Migration in die Cloud (*siehe Kapitel „2.7 Die Zukunft: Banking 4.0 inklusive Historie“*). Zum Ändern der Datenquelle wird die jeweilige DLL ausgetauscht, die Anweisungen innerhalb der Anwendung ändern sich aufgrund der DBMS übergreifenden Standardisierung dabei nicht. Durch die DLL werden die DBMS-Spezifika der Datenquelle, die sich ändern kann, gemäß des Clean Code Open Closed Principle (OCP) von dem getrennt, was beständig bleibt (Anwendungslogik). Die Anwendung ist offen für Erweiterungen, allerdings geschlossen gegenüber Modifikationen. Als Beispiel für ein Cross Platform-DB-Tool ist DbVisualizer zu nennen, welches im Rahmen der Kriterienanalyse zum Einsatz kommt (*siehe Kapitel „4.1 Kriterienanalyse (Access & Vergleichs-DBS)“*). OLE DB wurde von MS entwickelt und verwendet für den standardisierten Zugriff auf Datenquellen eine Sammlung von Component Object Model (COM)-Schnittstellen (Interfaces), über welche diverse Funktionen angeboten werden. Die COM-Bibliothek ist auch ein von MS entwickeltes, objektorientiertes Modell in Form eines Binärstandards. Es definiert, wie Objekte innerhalb eines Prozesses oder zwischen mehreren Prozessen miteinander kommunizieren, und zwar unabhängig von der jeweiligen Programmiersprache, der das Objekt zugrunde liegt. Die Interfaces geben ähnlich wie ein Vertrag vor, wie Anwendungen mit den COM-Objekten kommunizieren können. Jedes Interface besitzt eine eindeutige ID (Globally Unique Identifier (GUID)), den Abbreviation Forinterface Identifier (IID). Ein COM-Interface wird von einem COM-Objekt implementiert, welches wiederrum die konkreten Funktionen und Daten kapselt, die über das zugehörige Interface angeboten werden. Die Interprozesskommunikation ist im COM über betriebssystemunterstützten Shared Memory realisiert [26]/[92]/[125]/[131]/[132]/[214]/[239].

MS hat im Jahr 2011 OLE DB for SQL Server für obsolet (depracated) erklärt, diese Entscheidung aber 2017 widerrufen und 2018 den Nachfolger OLE DB for SQL Server (SQL OLE DB) in der dritten Generation herausgebracht [83]/[123]/[126].

MS hat die Ablösung von OLE DB for SQL Server damit begründet, dass ODBC der de facto Industriestandard für den nativen, relationalen Datenzugriff ist, der auf allen Plattformen einschließlich der MS Cloud-Lösung Azure SQL verfügbar ist. ODBC ist an *International Organization for Standardization* (ISO)/*International Electrotechnical Commission* (IEC) 9075-3:1995 (Call-Level Interface (SQL/CLI)) angelehnt. Hauptgründe für die Beliebtheit des ODBC-Protokolls sind neben der weiten Verbreitung aufgrund des Industriestandard-Status die einfache Verwendbarkeit und der unkomplizierte Aufbau der Programmierschnittstelle. OLE DB ist deutlich komplexer, bietet jedoch

einen erweiterten Funktionsumfang wie die Nutzung zusätzlicher Optimierungsoptionen zum Ausführen asynchroner Vorgänge. Die ODBC-Schnittstelle wird zentral vom Windows-Betriebssystem bereitgestellt. Die DBMS unabhängigen Funktionen der ODBC-API werden in DBMS spezifischen Treibern verwendet, welche für das jeweilige DBMS programmiert werden. Im Gegensatz zu OLE DB rufen Anwendungen die Funktionen der DBMS spezifischen Treiber-DLLs nicht über das COM, sondern über einen Treibermanager auf, um unabhängig vom zugrundeliegenden DBMS auf die Daten in der jeweiligen Datenquelle zuzugreifen. Für jede Datenquelle beziehungsweise für jedes DBMS muss es einen eigenen Treiber geben. Ein Treibermanager koordiniert die Kommunikation zwischen Applikation und Treiber (*siehe Kapitel „2.1 Auszug relevanter Begriffe im relationalen DBS-Umfeld“*) [127]/[130]/[231]/[237]/[275, S. 476]:

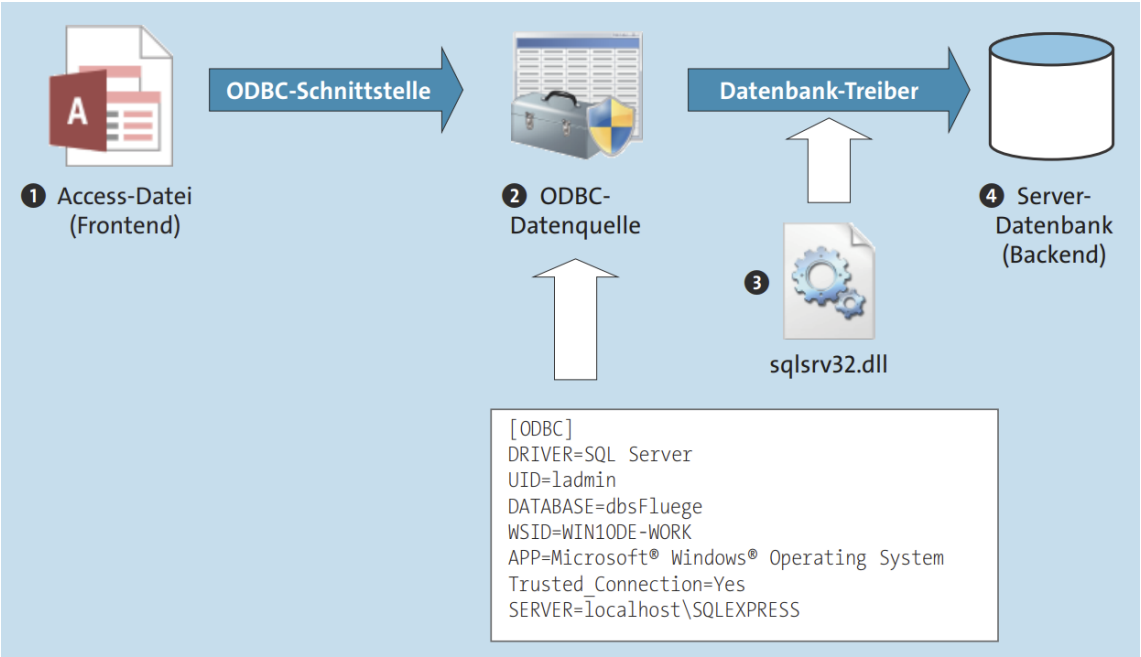


Bild 5: Zugriff mittels Access-Frontend auf eine externe Datenquelle via ODBC-Schnittstelle des Windows-Betriebssystems. 1) Frontend greift auf ODBC-Schnittstelle des Windows-Betriebssystems zu. 2) Über ODBC-Schnittstelle und zugehörige Verbindungsinformationen lädt das Betriebssystem den zugehörigen Treiber. 3) Der Treiber übernimmt die Kommunikation 4) mit dem DBMS [275, S. 476]

VBA unterstützt zwei Programmierobjektmodelle, welche jeweils auf ODBC oder OLE DB aufsetzen, um sich mit anderen Datenquellen wie dem SQL Server zu verbinden. Die Schnittstellentechnologie Data Access Object (DAO) baut auf ODBC auf, ActiveX Data Objects (ADO) auf OLE DB (*siehe Kapitel „2.4 Informationen zu Access, JET, ACE/ADE & .accdb-Dateiformat“*):

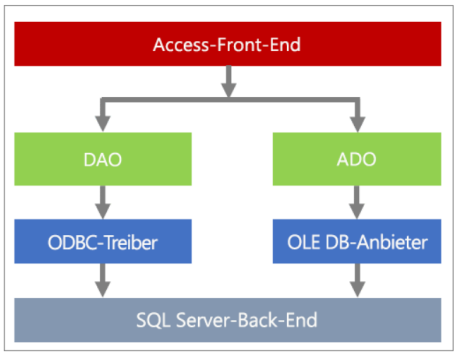


Bild 6: Möglichkeiten zum Verbinden von Access mit einer externen Datenquelle mittels DAO (ODBC) und ADO (OLE DB) in VBA [214]

DAO ist die Standardoption in Access und wird intern für alle Objekte wie Formulare, Berichte, Abfragen verwendet. MS empfiehlt die Verwendung von ADO für den SQL Server. ADO bietet eine höhere Anwenderfreundlichkeit, einen größeren Funktionsumfang, und eine höhere Verarbeitungsgeschwindigkeit. Zudem unterstützt ADO die Erstellung von webbasierten Anwendungen und hat einen geringeren Arbeitsspeicher- sowie Festplattenspeicherbedarf [214].

2.3. Informationen zu MS, Access & weitere eingesetzte Software

Nachdem einige relevante Begriffsbestimmungen erfolgt sind, wird in diesem Kapitel eine Übersicht über die in dieser Ausarbeitung verwendete Software wie Access oder das ausgewählte Vergleichs-DBS zur Kriterienanalyse gegeben. Da der Schwerpunkt auf MS-Produkten liegt, erfolgt auch eine Vorstellung von MS.

MS hat 1985 das erste GUI veröffentlicht, um die Bedienung des Betriebssystems zu erleichtern. Zunächst ging es darum die Darstellung von Programmen zu vereinheitlichen und die Verwendung von Peripheriegeräten wie Drucker und Bildschirme benutzerfreundlich zugänglich zu machen. Komplexe Technologien sollten für jeden und nicht nur Experten zugänglich sein. Erst durch diesen Ansatz wurde der Computer massentauglich und ermöglichte Unternehmen ihre Produktivität durch den Einsatz von Computern zu steigern. MS berücksichtigte dabei von Anfang an die Bedürfnisse von Geschäftskunden, zu denen auch die IT-Sicherheit gehörte.

Auch beim Internet gehörte MS zu den Vorreitern. Bereits Mitte der 90er Jahre stellte MS über ihr Betriebssystem „Windows“ relevante Technologien, Anwendungen und Dienste (Services) für die Nutzung des Internets zur Verfügung. Damit die Nutzung des Internets unter der Prämisse größtmöglicher Sicherheit erfolgt, wurde von Bill Gates im Jahre 2002 die Initiative „Trustworthy Computing“ ins Leben gerufen, welche die Bedeutung höchster Sicherheitsstandards in MS-Anwendungen hervorhebt und laut MS bis heute die MS-Produktlandschaft beeinflusst. Dazu sagte Bill Gates am 17.01.2002:

„However, even more important than any of these new capabilities is the fact that it is designed from the ground up to deliver Trustworthy Computing. What I mean by this is that customers will always be able to rely on these systems to be available and to secure their information. Trustworthy Computing is computing that is as available, reliable and secure as electricity, water services and telephony [271].“

Auch heute noch kann MS an seine vergangenen Erfolge anknüpfen. Nach Angaben von MS läuft das mittlerweile durch Windows 11 abgelöste Betriebssystem Windows 10 derzeit noch auf mehr als 1,3 Milliarden Geräten. Heutige Betriebssysteme haben sich zu offenen Plattformen entwickelt, auf denen Anwendungen und Services unterschiedlichster Hersteller integriert und bereitgestellt werden können [74]/[271].

Aber nicht nur im Betriebssystem-Umfeld ist MS Marktführer, auch im Bereich der Bürosoftware ist MS mit der Office-Suite (MS 365, ehemals Office 365) führend (*siehe Kapitel „1.1 Relevanz & wissenschaftlicher Mehrwert“*) [257]. Die MS 365-Suite umfasst Programme wie Word, Outlook, OneDrive, Teams, PowerPoint, Excel sowie Access. In Kürze werden die Office-Programme um KI erweitert („Copilot“) [188].

In dieser Ausarbeitung liegt der Fokus auf Access. Excel wird im Rahmen der Schnittstellenanalyse (*siehe Kapitel „4.2 Schnittstellenanalyse (Access Excel)“*) verwendet und besitzt teilweise vergleichbare Funktionen, wird jedoch für andere Anwendungsfälle empfohlen (*siehe Kapitel „3.5 Auswahl Office-Vergleichsprogramm zur Schnittstellenanalyse (Access Excel)“*). Einer der Hauptunterschiede zwischen Excel und Access ist, dass Access ein dateibasiertes, relationales DBS ist und im Gegensatz zu anderen Office-Programmen nur auf dem Computer zur Verfügung steht. Excel dagegen ist ein Tabellenkalkulationsprogramm, die Datenerfassung sowie -verarbeitung erfolgt in Tabellenform.

Wie bereits erwähnt, können mit Access (und Excel) auch komplexe Geschäftsanwendungen über die in der Oberfläche integrierten Designwerkzeuge und mit der dedizierten Programmiersprache VBA erstellt werden. Dieser Umstand stellt jedoch kein Alleinstellungsmerkmal dar, da dies auch mit anderen DBSs wie Oracle und der dedizierten Programmiersprache PL/SQL möglich ist [238]. Durch diese Funktionalität kann Access neben einem einfachen Backend zur Datenhaltung auch als Frontend (Präsentationsschicht inklusive Logik) verwendet werden (*siehe Kapitel „2.1 Auszug relevanter Begriffe im relationalen DBS-Umfeld“*) [204].

Hinter Access steht ein Konzept, das aus sechs Bausteinen besteht [160]/[187]/[192]/-[211]/[275, S. 18, S. 21-26, S.819, S. 855]:

1) Daten in Tabellen (Relationen): Es werden Regeln für die Datenablage in den Tabellen und Beziehungen zwischen Relationen festgelegt, um die Daten zu strukturieren und Inkonsistenzen zu vermeiden.
2) Abfragen: Mittels Abfragen lassen sich Daten aus einer oder mehreren Tabellen abrufen, auswerten, aggregieren.
3) Formulare: Werden zur Erstellung von Benutzeroberflächen verwendet. Auf ihnen können Steuerelemente und Grafiken angeordnet werden, sie werden vorrangig zur Benutzerführung, Datenanzeige und -bearbeitung genutzt.
4) Berichte: Im Gegensatz zu Formularen steht bei Berichten die Datenauswertung in Form eines Snapshots des aktuellen DB-Zustands im Fokus, es ist keine Datenerfassung möglich.
5) Programmierung mittels (Daten)Makros oder VBA: Werden zur Automatisierung von Aufgaben verwendet. Makros werden über die Oberfläche erstellt, indem die benötigte Logik über den „Makro-Generator“ in Form von Makroaktionen

zusammengestellt wird. Im Unterschied dazu können mit Datenmakros Ereignissen in Tabellen, ähnlich zu Triggern, Logik hinzufügt werden. In beiden Fällen muss kein Quellcode geschrieben werden. Mit zunehmender Komplexität werden (Daten)Makros schnell unübersichtlich. VBA ist eine ereignisgesteuerte Programmiersprache mit rudimentärer Unterstützung der objektorientierten Programmierung wie in Form von Klassenmodulen. VBA kann alles was Makros können und besitzt darüber hinaus zusätzliche, mächtigere Funktionalitäten. VBA ist desktopzentriert. Das bedeutet, dass VBA mit dem Desktop eines Benutzers interagieren kann. VBA hat die gleiche Sicherheitsfreigabe wie die zugrunde liegende Datei und erhält somit unter Umständen Vollzugriff auf den Desktop.

6) Anwendungsprogramme: Verbinden von Formularen und Berichten, um die jeweilige Access-Datei in ein eigenständiges Programm zu verwandeln.

Für weitere Details zum Access-Dateiformat siehe *Kapitel „2.4 Informationen zu Access, JET, ACE/ADE & .accdb-Dateiformat“*.

Im Vergleich zu Access und Excel ist Azure SQL ein Sammelbegriff bestehend aus SQL Server auf virtuellen Azure-Computern, Azure SQL Managed Instance sowie Azure SQL-DB. Dabei handelt es sich um drei Bereitstellungsmethoden, die sich darin unterscheiden, wie die DB-Engine des MS SQL Servers in der MS Azure-Cloud, in Form eines Client-Server-DBS (*siehe Kapitel „2.5 Unterschied dateibasierte & Client-Server DBS“*), bereitgestellt wird. Für weitere Details sei auf das *Kapitel „3.4 Auswahl Vergleichs-DBS zur Kriterienanalyse (Azure SQL-DB)“* verwiesen. Hier wird die Azure SQL-DB als Vergleichs-DBS für die Kriterienanalyse in *Kapitel „4.1 Kriterienanalyse (Access & Vergleichs-DBS)“* ausgewählt. Bei Azure SQL handelt es sich um reine DBSs, es können anders als in Access keine in das DBS integrierten Geschäftsanwendungen mittels VBA oder einer anderen Programmiersprache erstellt werden. Language Extensions, wie die „Common Language Runtime“-Integration, zum Ausführen von externen .NET-Code, wie C#, steht nur in der Managed Instance zur Verfügung. Beim SQL Server auf virtuellen Azure-Computern (Infrastructure-as-a-Service) wird der SQL Server in einer virtuellen Maschine in der Azure-Cloud betrieben. MS verwaltet in der Infrastructure-as-a-Service-Option lediglich die Infrastruktur wie die Hardware, auf der die virtuelle Maschine läuft. Nutzende wählen und verwalten in der virtuellen Maschine das Betriebssystem inklusive der zu installierenden Software sowie den SQL Server. Bei der Azure SQL Managed Instance (Platform-as-a-Service) wird die Infrastruktur sowie das Betriebssystem von MS verwaltet und somit eine Plattform angeboten, welche den bei Nutzenden anfallenden Verwaltungsaufwand reduziert. Nutzende sind lediglich für die Verwaltung des SQL Servers zuständig und erhalten im Gegensatz zur Infrastructure-as-a-Service-Option einige Vorteile wie automatische Versions-Updates des Betriebssystems und des SQL Servers, DB-Sicherungen, einen erweiterten Funktionsumfang sowie zusätzliche Tools wie Entwicklungs- und Testumgebungen oder Analysemöglichkeiten, die nur über die Azure-Cloud außerhalb der virtuellen Maschine zur Verfügung stehen:

Infrastructure as a service (IaaS) is a type of cloud computing service that offers essential compute, storage, and networking resources on demand, on a pay-as-you-go basis. IaaS is one of the four types of cloud services, along with software as a service (SaaS), platform as a service (PaaS), and serverless.

Migrating your organization's infrastructure to an IaaS solution helps you reduce maintenance of on-premises data centers, save money on hardware costs, and gain real-time business insights. IaaS solutions give you the flexibility to scale your IT resources up and down with demand. They also help you quickly provision new applications and increase the reliability of your underlying infrastructure.

IaaS lets you bypass the cost and complexity of buying and managing physical servers and datacenter infrastructure. Each resource is offered as a separate service component, and you only pay for a particular resource for as long as you need it. A cloud computing service provider like Azure manages the infrastructure, while you purchase, install, configure, and manage your own software—including operating systems, middleware, and applications.

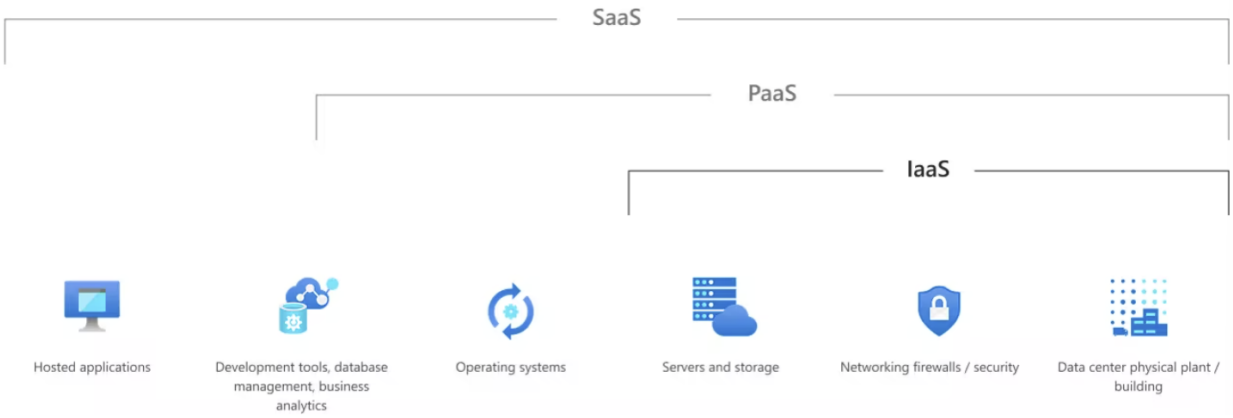


Bild 7: Unterschied Infrastructure-as-a-Service und Platform-as-a-Service [222]

Bei der Azure SQL-DB (Platform-as-a-Service) wird der bei Nutzenden anfallende Verwaltungsaufwand weiter reduziert, sie sind nur noch für das Datenmanagement und die Überwachung verantwortlich. Hier verwaltet MS die Infrastruktur, das Betriebssystem und den SQL Server. Nutzende müssen sich in der angebotenen Plattform nur noch um die Verwaltung der DBs im SQL Server kümmern [1, S. 4, S. 32]/[93]/[220]/[222]:

You	Microsoft
Capacity planning	Hardware, datacenter, virtualization
Migration	Operating system
Monitoring	SQL installation, configuration, patches
Performance tuning	Backup and restore
Database-level configuration	High availability and disaster recovery
Database maintenance	Security
Fixing outages	Scaling
Database design	Auditing
Automation	
Cost optimization	

Bild 8: Aufgabenverteilung zwischen Nutzenden der Azure SQL-DB und MS im Rahmen des Platform-as-a-Service- beziehungsweise Database-as-a-Serivce-Angebots [1, S. 2]

Nachfolgend eine Übersicht über die eingesetzte Software:

Tabelle 1: Überblick über genutzte Software

Genutzte Software	Relevante Kapitel	Kurzbe-schreibung/ Benötigt für	Version
Windows 10 Education (64-Bit)	*Übergreifend*	Betriebssys-tem.	10.0.19045 Build 19045.3324
Windows 11 Education (64-Bit)	*Übergreifend*	Betriebssys-tem.	10.0.22621 Build 22621.2134
MS Access für MS 365 (64-Bit)	*Übergreifend*	Relationales, dateibasiertes DBS (Haupt-gegenstand der vorliegen-den Thesis).	Aktueller Kanal: 2307 (Build 16.0.16626.20170)
MS Excel für MS 365 (64-Bit)	*4.4 Forensische Ana-lyse (Access), 4.2 Schnittstellenanalyse (Access Excel)*	Tabellenkal-kulationssoftw are.	Aktueller Kanal: 2307 (Build 16.0.16626.20170)
MS Azure SQL-DB	*3.4 Auswahl Ver-gleichs-DBS zur Kriterienanalyse (Azure SQL-DB), 4.1 Kriterienanalyse (Ac-cess & Vergleichs-DBS)*	Gehört zur Produktfamilie Azure SQL und stellt als Platform-as-a-Service-Op-tion die MS SQL Server-DB-Engine in der MS Azure-Cloud zur Verfü-gung. Nutzende sind nur für die Datenver-waltung zuständig.	MS SQL Azure (RTM) - 12.0.2000.8 Jul 8 2023 12:00:47 Copyright (C) 2022 Microsoft Corporation
MS Azure Data Studio	*4.1 Kriterien-analyse (Ac-cess & Ver-gleichs-DBS)*	SQL Client und Database Management Software.	1.44.1
DbVisualizer	*4.1 Kriterienanalyse (Access & Ver-gleichs-DBS)*	SQL Client und Database Management Software. Ein-geschränkter Funktionsum-fang in der	23.2.2

Genutzte Software	Relevante Kapitel	Kurzbe-schreibung/ Benötigt für	Version
		kostenfreien Variante.	
WinRAR	*4.2 Schnittstellen-analyse (Access Excel)*	Packpro-gramm zur Datenkom-pression.	6.22
FEX Ima-ger™	*4.4 Forensische Ana-lyse (Access)*	Kostenloses Programm zum Erstellen forensischer Images.	2.2.1.283 *DiskPart: 10.0.19041.964*
Autopsy (Windows-Version)	*3.6 Auswahl forensi-sches Tool, 4.4 Forensische Analyse (Access)*	Open Source digitale Fo-rensik-Plattform mit GUI (basiert auf Sleuth Kit).	4.20.0
Visual Stu-dio Code inklusive „Hex Edi-tor"-Erweiterung von MS	*4.2 Schnittstellen-analyse (Access Excel), 4.3 Dateifor-matanalyse mittels Hexadezimal-Editor (Access)*	Quelltext-Edi-tor inklusive Hexadezimal-Editor.	1.79.2 (User Setup) 1.9.11 (Hex Editor)
Visual Stu-dio 2022	*4.1.1 SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen*	Integrierte Entwicklungs-umgebung. Benötigt zur Erstellung der eigenentwi-ckelten Web-Anwendung „DemoApp-ForSqlInjectio nAttacksOnA-zureSqlDb".	17.7.0
Eigenentwi-ckelte Web-Anwendung „DemoApp-ForSqlInjecti onAttacksO-nAzureSqlD b" (Razor, C#)	*4.1.1 SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen*	Diese Demo-Web-Anwen-dung dient zum Testen von SQL-Injection-An-griffen auf die Azure SQL-DB.	ASP.NET Core Web App (.NET 7.0) *Verwendete Bibliotheken:* *- Microsoft.Extensions.Configuration (7.0.0)* *- Serilog (3.0.1)* *- Serilog.Extensions.Logging (7.0.0)* *- Serilog.Sinks.File (5.0.0)* *- System.Data.SqlClient (4.8.5)*

2.4. Informationen zu Access, JET, ACE/ADE & .accdb-Dateiformat

2007 ist das .mdb-Dateiformat durch das .accdb-Dateiformat ersetzt worden. Mit Einführung des .accdb-Dateiformats sind neben einigen Produktverbesserungen wie Optimierungen an der Dateiverschlüsselungsfunktionalität auch einige Änderungen an der internen Architektur des Dateiformats vorgenommen worden. Vor 2007 hat das Access-DMBS als DB-Engine zur Datenspeicherung und für Datenabfragen auf die 1992 eingeführte MS JET gesetzt, die auch seit Windows 2000 Teil des Betriebssystems ist. Ab 2007 ist die JET-Engine zur Datenspeicherung und für Datenabfragen dann im .accdb-Dateiformat durch die MS Access Connectivity Engine (ACE), auch als Microsoft Access Database Engine (ADE) bezeichnet, ersetzt worden. Somit hat MS Access eine eigenständige DB-Engine erhalten. Die ACE basiert auf der JET und ist abwärtskompatibel zur JET. Zur Vereinfachung wird nachfolgend nicht zwischen DB-Engine und DBMS unterschieden [161]/[166]/[258, S. 113 ff.].

Im Internet finden sich zahlreiche Quellen zur JET, wie der „Microsoft Jet Database Engine Programmer's Guide“ von Dan Haught aus dem Jahr 1997. MS empfiehlt dieses Buch als zentrales Nachschlagewerk für alle JET-Nutzenden sowie Entwickler von VBA-Anwendungen in Office Produkten wie Access oder Excel [124].

Eine Recherche zur ACE sowie Details zum .accdb-Dateiformat haben keine detaillierten technischen Dokumentationen ergeben, sodass keine konkreten Aussagen zur ACE-Architektur sowie zum .accdb-Dateiformat getroffen werden können. Entsprechend MS-Angaben wird die ADE benötigt, um den Datentransfer zwischen MS Office-Anwendungen, wie Access im .accdb-Dateiformat oder Excel im .xlsx-Dateiformat, Nicht-MS Office-Anwendungen wie MS SQL Server sowie Textdateien über Schnittstellen, wie OLE DB oder ODBC beziehungsweise die Programmier-Objektmodelle DAO und ADO (*siehe Kapitel „2.2 Beispiele für Datenzugriffe in Access“*), zu ermöglichen [194]/[198].

Wie in der JET ist das DAO-Objekt auch bei der ACE eines der Hauptkomponenten, um über VBA (Hostsprache der JET und der ACE für „Database Application Programming“) mit der Engine zu kommunizieren (*siehe Kapitel „2.2 Beispiele für Datenzugriffe in Access“*). Das DAO stellt dabei die objektorientierte Schnittstelle zum DBMS dar, über die Nutzende Data Definition Language (DDL) sowie Data Manipulation Language (DML)-Abfragen in Form von Funktionen ausführen können. Der Nachfolger von DAO für den universellen Datenzugriff ist die ADO-Schnittstelle der JET. Wird über die vom DAO angebotene Funktion „`OpenRecordset`“ eine `SELECT`-Abfrage ausgeführt, wird das Ergebnis gemäß objektorientiertem Ansatz als Recordset-Objekt an den Aufrufer zurückgegeben. Auch für andere SQL-Befehle wie `CREATE TABLE` („`CreateTableDef`“) werden über das DAO-Objekt äquivalent funktionierende Funktionen angeboten.

Wird das in *Kapitel „2.1 Auszug relevanter Begriffe im relationalen DBS-Umfeld“* skizzierte Drei-Schichten-Modell eines DBS auf Access abstrahiert, kann über die Jet-Hostsprache VBA eine externe Schicht angefragt werden, da sie eine individuelle Sicht auf die Daten ermöglicht. Die JET übernimmt die Ausführung der über die jeweilige Schnittstelle angefragten individuellen Sicht (hier in Form von VBA). Dabei kümmert sich die JET um den Zugriff auf die strukturiert gespeicherten Daten in der jeweiligen Access-DB-Datei (physikalische Schicht). Durch dieses Vorgehen werden die Interna der

physikalischen Schicht vor den Softwareentwickelnden versteckt, denn sie kommunizieren nur mit dem JET-DBMS über die hier angebotenen, standardisierten Schnittstellen wie VBA. Wird zur Generierung einer externen Sicht von VBA zu Abfragen gewechselt, bleibt die konzeptionelle und physikalische Schicht von der Änderung unberührt, da der Aufruf weiterhin über eine von der JET angebotenen Schnittstelle erfolgt [258, S. 115-123]:

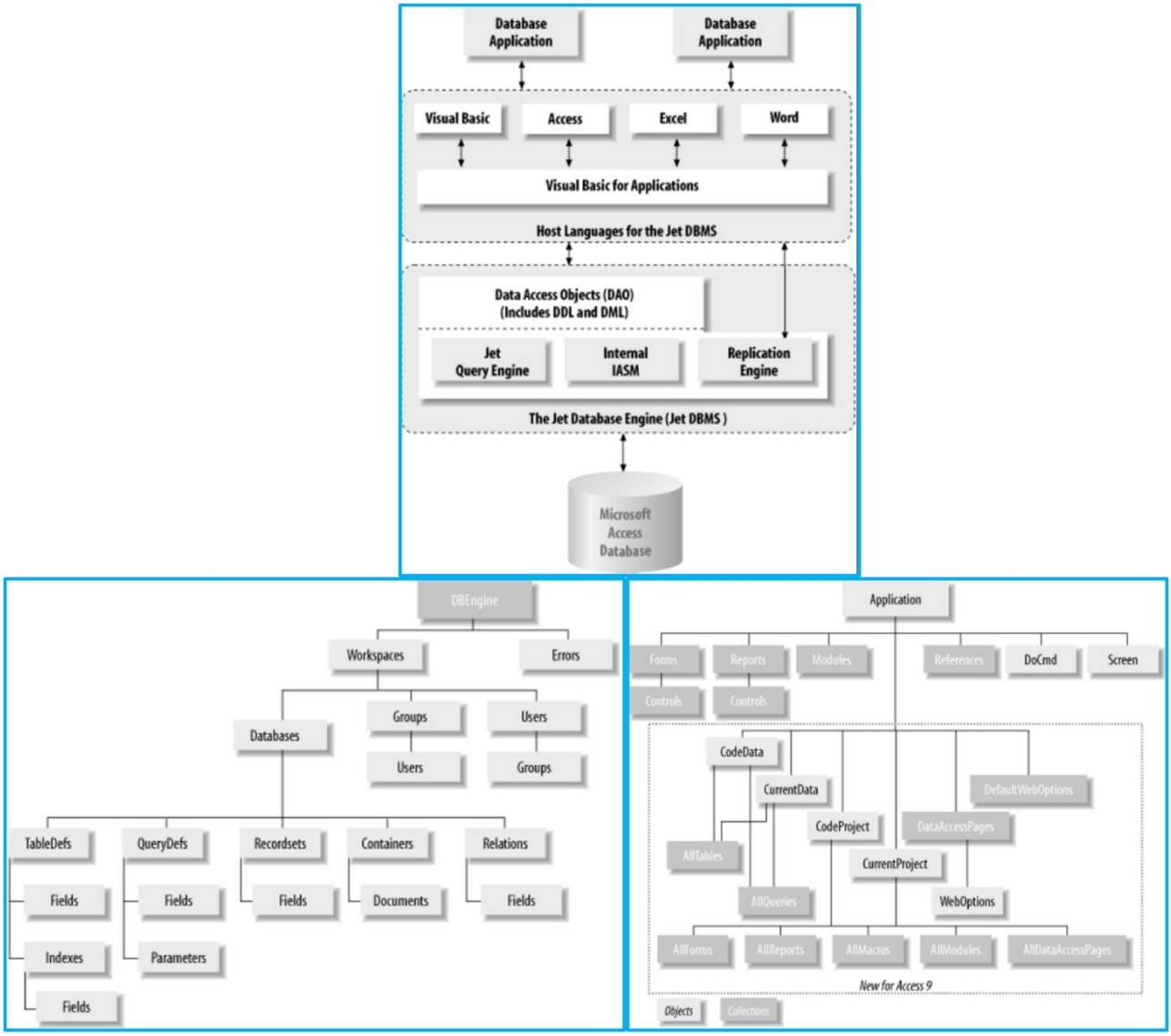


Bild 9: Oben: Architektur des JET-DBMS und Beziehungen zwischen dem Drei-Schichten-Modell; Unten links: DAO-Objekt-(Listen-)Hierarchie, Funktionalität der DB-Engine; Unten rechts: Applikation-Objekt-(Listen-)Hierarchie der jeweils geöffneten Access-Datei. Das „Forms"-Objekt hält eine Objekt-Liste mit allen Formularen, die in der aktuell geöffneten Datei gespeicherten sind. „Modules" enthält eine Objekt-Liste mit allen zur Datei gehörenden VBA-Modulen. Mittels DoCmd-Objekt können wie bei DB-Engine.QueryDef parametrisierte Abfragen ausgeführt werden [258, S. 119, S. 207, S. 209]

Bei Access, Excel und Word handelt es sich um die Anwendungs-Schicht (Host), die mittels Hostsprache VBA auf das JET-DBMS zugreift. Da die Office-Programme unterschiedliche Anwendungsfälle abdecken, unterscheiden sich die jeweiligen VBA-Implementierungen der Programme. Das JET-DBMS enthält das DAO-Objekt mit

den darüber angebotenen Funktionen sowohl für DML als auch für DDL-Zwecke als einheitliche Programmier-Schnittstelle. Ähnliches gilt für das ADO-Objekt oder die Verwendung der ODBC-Schnittstelle der JET.

Die JET besteht aus mehreren DLLs, über die Funktionen des DBMS bereitgestellt werden. Die „Jet Query Engine“ übersetzt die individuellen Abfragen in Access SQL. Danach kompiliert und optimiert es die Eingaben und führt die Access SQL-Abfrage letztendlich aus. Aufgabe der Internal Indexed Sequential Access Method (ISAM)-Komponente ist das Speichern und Abfragen der Daten von der physikalischen Access-DB-Datei auf dem jeweiligen Speichermedium gemäß der zuvor generierten Access SQL-Abfrage. Die „Replication Engine“ kümmert sich um die ständige Synchronisation von Kopien der Access-DB-Datei auf unterschiedlichen Systemen.

ISAM ist eines von insgesamt drei Hauptvorgehensweisen für den lesenden und schreibenden Dateizugriff und nutzt ein indexbasiertes Verfahren. Daneben gibt es sequenzielle und direkte Verfahren. Sequenzielle Verfahren stellen die einfachste Lösung dar. Dabei werden die Daten in der Datei als ein Datensatz eingelesen, über den dann von Anfang bis Ende iteriert wird, bis das gesuchte Element gefunden ist. Die Daten werden jeweils in der abgelegten Reihenfolge gelesen, was sich aufgrund der vielen Zugriffe auf das Dateisystem negativ auf die Performance auswirkt. Bei direkten Verfahren erfolgt der direkte Zugriff auf die Daten abhängig von der jeweiligen Position (physikalische Adresse) in der Datei. Dieses Vorgehen beschleunigt den Datenzugriff im Vergleich zum sequenziellen Ansatz, da weniger Daten durchsucht und somit weniger Zugriffe auf das Speichermedium benötigt werden. Indexbasierte Dateizugriffe kombinieren die Vorteile aus sequenziellen und direkten Verfahren. Sie finden aufgrund ihrer Effizienz auch in Dateisystemen Anwendung (vergleiche NTFS und Master File Table($MFT) in *Kapitel „4.4 Forensische Analyse (Access)“*). Bei indexbasierten Verfahren wird eine Indexdatei oder -tabelle erstellt, in der Schlüssel in sortierter Form als Suchkriterien mit zugehörigen Pointern (physikalische Adressen) als Wert abgelegt sind. Im DBS-Umfeld sind die Schlüsselwerte in der Regel alle Attributwerte des (zusammengesetzten) Primärschlüssels aller Tupel des jeweiligen Entitätstyps. Die Pointer zeigen auf den korrespondierenden Datensatz. Die Zugriffe auf alle zu einem Index gehörenden Datensätze erfolgt sequenziell. Da die Datensätze in der Indexdatei oder -tabelle sortiert sind, kann die Suche nach allen Datensätzen mit demselben Index abgebrochen werden, wenn beim Iterieren über die Indexdatei ein Schlüsselwertwechsel festgestellt wird. Bei Dateisystemen kann mit einem Index bestehend aus Dateinamen und Pfad auf den Dateiinhalt zugegriffen werden. Alle Primärschlüssel werden in Access automatisch als Index angelegt. Darüber hinaus kann je Relation ein Sekundärindex angelegt werden. Wie die Indexdatei konkret im .accdb-Dateiformat der ACE abgebildet ist, kann aufgrund fehlender Quellen abschließend nicht beantwortet werden [3]/[258, S. 31-32, S. 115-123]:

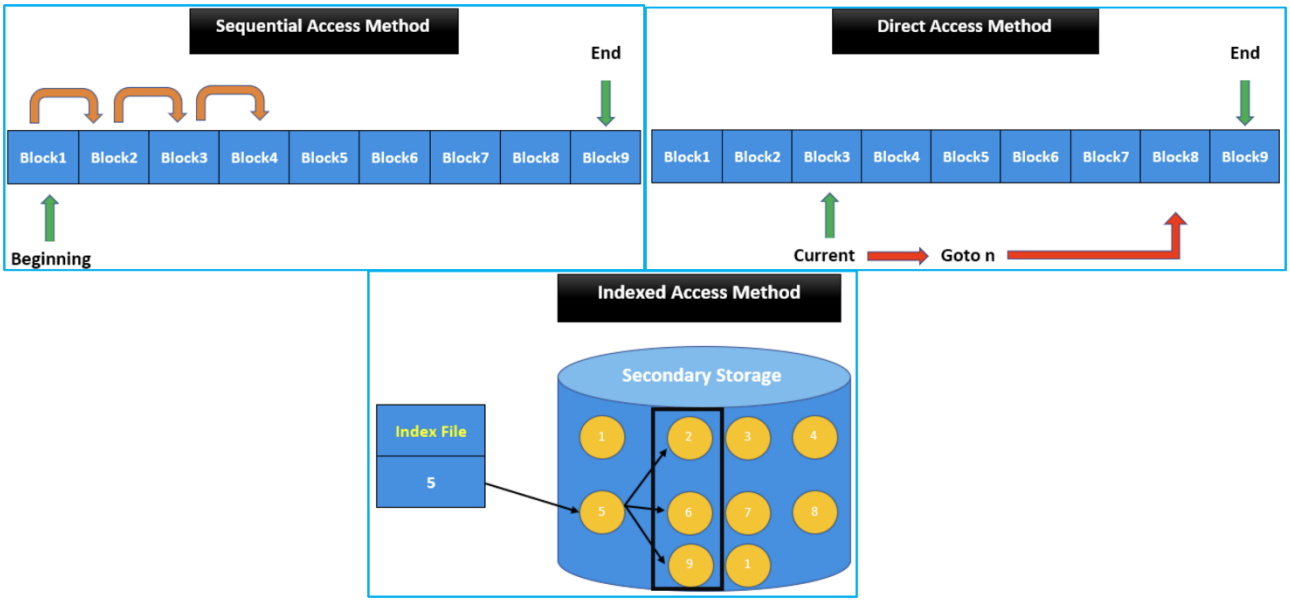


Bild 10: Links: Sequenzieller Dateizugriff; Rechts: Direkter Dateizugriff; Unten: Indexbasierter Dateizugriff [3]

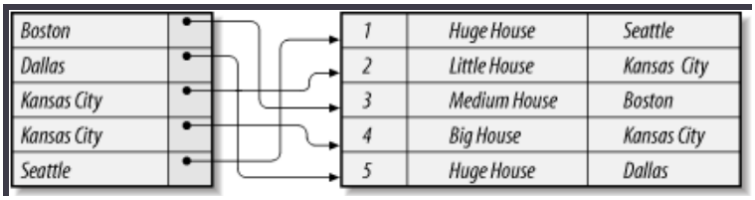


Bild 11: Links: Nach Index („City“) sortierte Indexdatei; Rechts: Indexierte Access-Relation anhand „City“-Attribut. Die Indices werden mittels Pointern den zugehörigen Datensätzen zugeordnet [258, S. 32]

Zum Ausführen von MS Access-Anwendungen wird die Laufzeitumgebung „MS 365 Access Runtime“ benötigt, in der die ADE (auch ACE genannt) 2016 inkludiert ist. Basis für Access 365 ist Access 2016, da in diesem Jahr die letzten Änderungen am Dateiformat vorgenommen wurden. Alle darauffolgenden Updates beinhalteten „lediglich“ Funktionserweiterungen auf Basis von Office 2016. Aus diesem Grund ist auf der Access-Oberfläche sowie unter dem Kopfreiter „*Datei → Optionen → Allgemein → Standarddateiformat für leere Datenbank*“ die Angabe „Access 2007-2016“ zu finden. Außerdem beginnt die aktuelle Versionsnummer mit „16“ [101]:

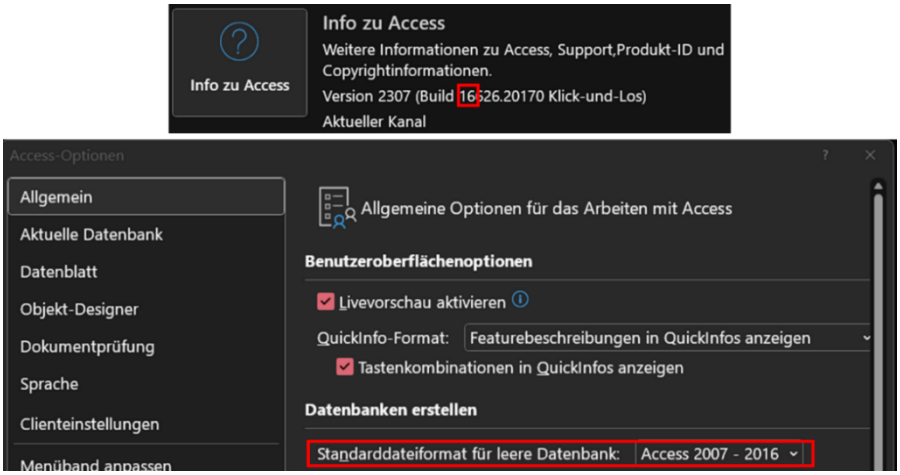


Bild 12: Access 2016 als Grundlage für Access 365 – Zugehörige Informationen

Die Access Runtime kann kostenlos für das Windows-Betriebssystem heruntergeladen werden. Mit der Microsoft 365 Access Runtime können Access-Anwendungen an Nutzende verteilt und von diesen genutzt werden, die kein Microsoft Office oder eine Access-Vollversion auf ihrem Endgerät installiert haben. Neben der Einsparung von Lizenzkosten fehlen einige sicherheitskritische Funktionalitäten wie alle Werkzeuge zum Erstellen einer DB inklusive Navigationsbereich und Menüband. Sobald jedoch eine Vollversion von Access zur Verfügung steht, ist dieser Schutz hinfällig (*siehe Kapitel „4.1.2 Konfiguration“*) [194]/[198]/[275, S. 26].

2.5. Unterschied dateibasierte & Client-Server DBSs

Da es sich bei MS Access um ein dateibasiertes DBS handelt und das Vergleichs-DBS (*siehe Kapitel „3.4 Auswahl Vergleichs-DBS zur Kriterienanalyse (Azure SQL-DB)“*) auf einer Client-Server-Architektur basiert, erfolgt nachfolgend jeweils eine Definition von dateibasiertem DBS und DBS mit Client-Server-Architektur. Außerdem werden die mit dateibasierten DBSs zusammenhängenden Begriffe „Datei“ sowie „Dateisystem“ erläutert. Bei dateibasierten DBSs handelt es sich um Dateien, die in Dateisystemen auf Speichermedien wie Netzlaufwerken abgelegt werden. Bei Client-Server DBSs handelt es sich um eigenständige Server (Begriff umfasst Hard- und Software), die auf Anfragen von unterschiedlichen Clients warten, diese entgegennehmen und verarbeiten.

Eine Datei ist ein Datenpaket beziehungsweise eine Sammlung von zusammengehörenden Daten, die in der einfachsten Form als lineare Folge von Bytes („Byte-Stream“) auf einem Speichermedium, wie einer Festplatte als Sekundärspeicher, abgelegt sind. Sie besitzen einen Namen und werden durch ein Dateisystem verwaltet. Mit einer Datei können Daten dauerhaft gespeichert werden. Der Arbeitsspeicher ist hierfür ungeeignet, da er klein ist und nach dem Herunterfahren bereinigt wird. Eine Datei wird durch ihren Dateinamen eindeutig bezeichnet, der als Suffix optional eine Dateierweiterung wie „.accdb“ besitzt. Dateiname und -erweiterung sind durch einen Punkt voneinander getrennt. Durch die angegebene Dateierweiterung weiß das Betriebssystem mit welchem Anwendungsprogramm die Datei geöffnet werden soll. Die Dateierweiterung

ist zu unterscheiden von der Dateisignatur im Datei-Header (*siehe Kapitel „4.4 Forensische Analyse (Access)“*) [57, S. 449]/[243, S. 206]/[245, S. 288, S. 316].

Bei einem Dateisystem handelt es sich um ein dauerhaftes Ablage- und Verwaltungssystem von Dateien auf Sekundärspeichern. Es ist verantwortlich dafür, die auf dem jeweiligen Speichermedium abgelegten Daten/Dateien in geeigneter Form zugänglich zu machen, ohne dass sich Nutzende um die Details der internen Datenorganisation kümmern müssen. Ein Dateisystem dient als Vermittlungsebene zwischen dem Betriebssystem und den in Dateien gespeicherten Informationen auf einem Sekundärspeicher. Darüber hinaus bietet ein Dateisystem auch einen Schutzmechanismus der gewährleistet, dass Dateien nur von berechtigten Nutzern gelesen und/oder verändert werden können (Dateiberechtigungen/Benutzerrechte) [50, S. 337-338]/[57, S. 449]/[243, S. 207].

Dateibasierte DBSs, wie MS Access oder SQLite, sind von „normalen“ Dateiformaten wie Graphics Interchange Format (GIF), Joint Photographic Experts Group (JPEG) oder Extensible Hypertext Markup Language (XHTML) zu unterschieden, da sie mehr als ein Objekt speichern. Bei GIF und JPEG ist das Objekt jeweils das gespeicherte Bild, bei XHTML der gespeicherte Text. Im Gegensatz dazu zählen Dateiformate wie MS Access, MS Word, MS Excel, Electronic Publication (EPUB), Portable Document Format (PDF), GIT oder SQLite zu den „Anwendungsformaten“. Genutzt werden sie, um den Anwendungsstatus zu speichern oder Informationen zwischen Programmen auszutauschen. Anwendungsformate bestehen in der Regel aus einer Datei, um den Verwaltungsaufwand zu minimieren und sie effizient transportieren (kopieren, verschieben, an E-Mails anhängen) zu können. Sie liegen im Binärformat vor und können aus mehreren Objekten inklusive zugehörigen Beziehungen bestehen. Eine EPUB-Datei besteht aus Text (XHTML) und Bildern (GIF/JPEG), ein dateibasiertes DBS enthält Objekte wie mehrere Relationen mit zugehörigen Attributen, Datentypen, Constraints, Indices sowie Beziehungen. Konkret handelt es sich bei MS Access-Dateien wie .accdb oder .accde um ein „benutzerdefiniertes Anwendungsformat“, da die Dateien im Binärformat vorliegen und Lese- sowie Schreibvorgänge auf den Inhalt nur über dedizierte Programme wie MS Access oder DbVisualizer (eingeschränkter Funktionsumfang), die extra für das Anwendungsformat entwickelt wurden, möglich sind. Über Kommandozeilenbefehle oder reine Text-Editoren ist hingegen kein ordnungsgemäßer Lese- und Schreibzugriff möglich. Anzumerken ist, dass die Grenze zwischen Datei- und Anwendungsformaten fließend und die Definition oft kontextabhängig ist, denn für ein Bild-Editor ist ein GIF oder JPEG auch ein Anwendungsformat [249].

MS Access ist darüber hinaus von Flat Files abzugrenzen, wobei viele Nachteile von Flat Files auch für Access-Dateien passen. Dazu zählen eine eingeschränkte Sicherheit aufgrund fehlender Funktionen, wie eine Benutzerauthentifizierung (Sicherheit auf Benutzerebene) sowie mangelhafte Transparenz, da die Dateien schnell über Netzwerke oder per E-Mail ausgetauscht und kopiert werden können. Außerdem besitzt keine zentrale Stelle Kenntnis über die enthaltenen Daten. Flat Files enthalten Datensammlungen, die über Text-Editoren gelesen und verarbeitet werden können. Ähnlich wie Relationen einer DB besitzen sie eine einheitliche Struktur, enthalten Zeilen und Spalten,

die durch ein Trennzeichen voneinander abgetrennt sind. Eines der bekanntesten Beispiele ist eine Comma-Separated Values (CSV)-Datei [31]/[33].

Beim Öffnen einer Datei wählt das Betriebssystem das der Dateierweiterung im Dateinamen zugeordnete Anwendungsprogramm aus und führt es lokal auf dem Host aus. Dabei werden folgende Schritte durchgeführt [36]:

1) Nach dem Klick des Nutzers auf eine Datei, wird über eine C-Bibliothek eine Anfrage an das Betriebssystem gesendet die Datei zu öffnen.
2) Der Computer prüft, ob das Öffnen der Datei mit dem Programm erlaubt ist, dass der Dateierweiterung im Dateinamen zugeordneten (dasselbe gilt beim Erstellen oder Löschen).
3) Das Betriebssystem prüft, ob das Öffnen der Datei gemäß den zugeordneten Rechten des Benutzenden erlaubt ist (dasselbe gilt beim Erstellen oder Löschen).
4) Bei Fehlern in den vorherigen Prüfungen unter 2) und 3) wird dem Nutzenden eine Fehlermeldung angezeigt. Sind beide vorherigen Prüfungen erfolgreich, startet das Betriebssystem die Anwendung, welche der Dateierweiterung im Dateinamen zugeordnet ist. Bei unbekannter Dateierweiterung müssen Nutzende manuell die zugehörige Anwendung auswählen.
5) Das Betriebssystem prüft, ob das Öffnen im Schreibzugriff erlaubt ist. Ist die Prüfung erfolgreich, wird die Datei im Schreibzugriff geöffnet.
6) Der Computer allokiert Random-Access Memory (RAM) für die zu öffnende Datei und das Programm.
7) Der Computer greift gemäß Pfad über das Dateisystem auf die Datei zu und kopiert den Inhalt (Byte-Stream) in den zuvor allokierten RAM.
8) Die Anwendung trennt den Byte-Stream in einzelne Bytes, interpretiert diese und zeigt Nutzenden den jeweiligen Inhalt wie ASCII-Zeichen an.
9) Ist der Dateierweiterung im Dateinamen ein falsches Anwendungsprogramm zugeordnet, wird die Datei zwar geöffnet, der Dateiinhalt wird allerdings von der Anwendung so interpretiert und angezeigt, wie sie programmiert wurde. Wird ein Bild in einem Text-Editor geöffnet, zeigt es anstelle dem Bild eine kryptische ASCII-Zeichensammlung gemäß den jeweiligen Byte-Reihenfolgen an.

Bei Client-Server-Architekturen stellt ein zentraler Dienstleistungsrechner/Computer (Server) Anwendern (Clients) eine oder mehrere Anwendungen, Ressourcen, Funktionen oder sonstige Dienstleistungen an zentraler Stelle für mehrere Nutzende (Clients) zur Verfügung. Üblicherweise erfolgt die Kommunikation zwischen Clients und Servern über ein Netzwerk:

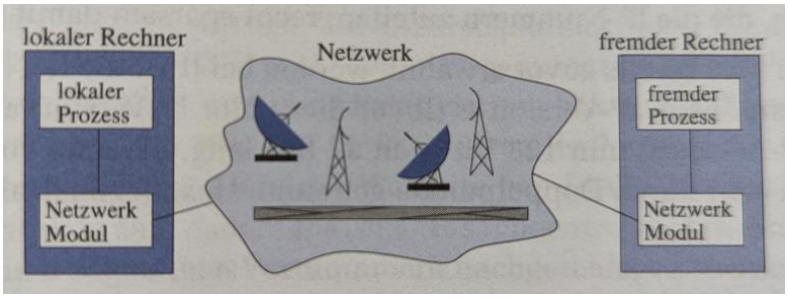


Bild 13: Kommunikationsmodell in einem Netzwerk zwischen Client (lokaler Rechner) und Server (fremder Rechner) [57, S. 467]

Der Server-Begriff umfasst neben der Hardware auch die zur Verfügung gestellte Software, die auf dem zentralen Dienstleistungsrechner in Form von Prozessen im Hintergrund laufen und darauf warten, gemäß eingehender Anfragen die jeweiligen Dienstleistungen bereitzustellen (im Unix-Umfeld auch „Daemon“ genannt und im Windows-Umfeld „Dienst/Service“). Beispiele für Server: DBS-Server, Web-Server, Druck-Server oder Dateiserver. Auf einem DBS-Server läuft ein DBS, dass von Clients im Netzwerk angesprochen werden kann.

Auch der Client-Begriff umfasst Soft- sowie Hardware und ist ein Prozess. Der Client nimmt die vom Server angebotene Dienstleistung in Anspruch, indem er mittels Verbindungsinformationen auf die Datenquelle oder Dienstleistung zugreift (*siehe Kapitel „2.1 Auszug relevanter Begriffe im relationalen DBS-Umfeld“*). Der Client kommuniziert mit dem Server über die vom Server angebotenen Schnittstellen. Die standardisierte Kommunikation wird dabei über Protokolle geregelt. Beispiele für DBS-Server-Clients: Excel-Anwendungen, die über OLE DB oder ODBC auf den Server zugreifen oder DBS-Management Software wie DbVisualizer oder das SQL Server Management Studio von MS [57, S. 476]/[245, S. 198-199].

Zum Öffnen von Dateien benötigen Nutzende Zugriff auf den Ablageort, Dateien können bereits mit Lesezugriff kopiert werden. Außerdem fehlen wichtige Sicherheitsfunktionalitäten wie eine Benutzerauthentifizierung. Der zugehörige Quellcode müsste in jede Datei integriert werden, da Nutzende direkt auf die Dateien zugreifen können. Im Gegensatz dazu kommunizieren Nutzende (Client) mit einem Server in einer Transmission Control Protocol (TCP)/Internetprotokoll (IP)-Kommunikation über Sockets (Berkeley Socket API), wobei Server sowie Client jeweils einen Socket besitzen. Sockets stellen die plattformunabhängigen, standardisierten Endpunkte einer Netzwerkverbindung dar und bestehen jeweils aus IP-Adresse des Hosts sowie der Portnummer des jeweiligen Anwendungs-Prozesses auf dem Host. Datagram Sockets verwenden das verbindungslose Transportprotokoll User Datagram Protocol (UDP), Stream Sockets das verbindungsorientierte Transportprotokoll TCP. Ein Socket ist die Schnittstelle zwischen der Anwendung und der Transportschicht im Betriebssystem. Möchte eine Anwendung mit einer anderen Anwendung Daten über das Netzwerk austauschen, schreibt sie die Daten in ihren lokalen Socket. Die Gegenseite kann dann wiederrum die zuvor hinterlegten Informationen aus ihrem lokalen Socket lesen. Im Gegensatz zu Pipes ist ein Socket stets bidirektional (geeignet für Lese- sowie Schreibzugriffe). Eine konkrete Netzwerkverbindung wird über das verwendete Protokoll, Quell-IP-Adresse (lokaler

Host), Quell-Portnummer (lokaler Prozess), Ziel-IP-Adresse (fremder Host), Ziel-Portnummer (fremder Prozess) beschrieben. Dieses Vorgehen bietet im Vergleich zu Dateien zusätzliche Sicherheit, da der Nutzende nur über den Socket Zugriff auf den DBS-Server besitzt und eingehende Anfragen vom Server-Prozess (Daemon oder Dienst genannt) analysiert und abgelehnt werden können [25, S. 173]/[57, S. 467]/[245, S. 638]. Durch dieses Vorgehen können umfassende (sicherheitsrelevante) Funktionalitäten wie zur Überwachung, Wiederherstellung, Authentifizierung und Autorisierung implementiert werden, da Nutzende nur über eine Zwischenschicht auf die Daten beziehungsweise das DBMS zugreifen können. Beim DBMS handelt es sich um eine Softwarekomponente, welche die Daten, DBs und Zugriffe von Clients verwaltet (*siehe Kapitel „2.1 Auszug relevanter Begriffe im relationalen DBS-Umfeld“*).

Mit einem Client-Server-DBS werden die Daten unabhängig von den Anwendungen verwaltet. Damit ist es möglich die Daten von den Benutzern zu isolieren, Transparenz zu schaffen, Nutzenden Berechtigungen gemäß ihrem Aufgabenbereich zuzuweisen (Autorisierung) und den Zugriff zu kontrollieren. Außerdem wird die Datenverwaltung und -speicherung zentralisiert, somit Redundanzen sowie Inkonsistenzen vermieden und gleichzeitig Speicherplatz gespart.

Dateien sind nur begrenzt dazu geeignet Daten über einen längeren Zeitraum aufzubewahren und konsistent zu speichern. Auch, weil die zuvor erwähnte Transparenz verlorengeht, die Datenauffindbarkeit erschwert und die Gefahr der redundanten Datenablage entsteht. Zur Vermeidung dieser Probleme werden Client-Server-DBS eingesetzt, die auf eigenständigen Servern betrieben werden.

Dateien und Zugriffe auf sie werden dagegen über das Betriebssystem mit eingeschränkter Sicherheitsfunktionalität wie Dateisystemberechtigungen verwaltet, es gibt keine weitere Softwareebene dazwischen. Große Datenbestände können in viele Access-Dateien münden, die es zu verwalten gilt und teilweise aufgrund der dezentralen Ablage unbekannt sein können. Abhängig von der Bewertung der Datenschutz- und Informationssicherheitsrelevanz sowie Nutzung von Verschlüsselungsalgorithmen zur Verschlüsselung der Daten in der Datei kann daraus ein hohes Risiko entstehen [196]. Nachteilig an Dateien ist neben der Gefahr der einfachen Duplikation und Schaffung von Redundanzen, dass jede Anwendung das jeweilige Speicherformat einschließlich der Namenskonventionen der Daten bestimmen kann, die Daten hängen so eng mit der Software zusammen. Möchten andere Anwendungen die Daten nutzen, benötigen sie die Implementierungsdetails der Anwendung, welche die Daten erzeugt hat. Da Access Multi-User Support bietet, stellen parallele Zugriffe von mehreren Clients bis zu einem gewissen Grad kein Problem dar. Auch die Erstellung von Funktionen für das Anlegen, Ändern, Suchen oder Löschen sind problemlos via VBA möglich [33]/[48]/[50, S. 337-338]:

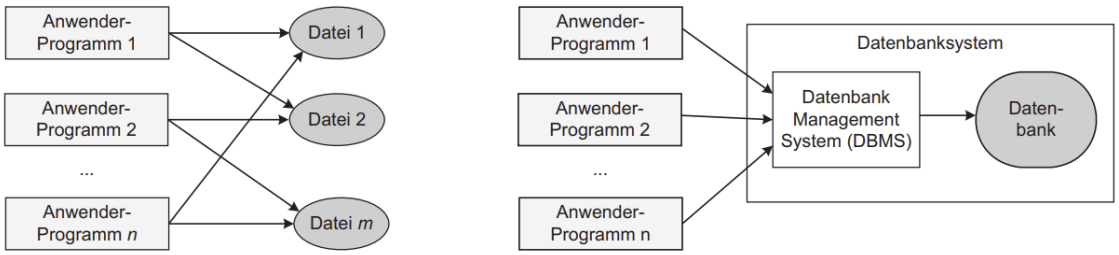


Bild 14: Unterschied beim Zugriff auf ein dateibasiertes DBS und Client-Server-DBS – Links: Zugriff auf MS Access-Dateien; Rechts: Zugriff auf ein Client-Server-DBS [50, S. 337]

2.6. Aktueller Stand: Regulatorische Anforderungen im Kredit- & Finanzdienstleistungswesen

Der in *Kapitel „1.3 Anwendungsszenario“* skizzierte Kontext für das Kredit- und Finanzdienstleistungswesen erfordert einige Begriffsdefinitionen und Einordnungen derzeitiger rechtlicher und organisatorischer Vorgaben, die in diesem Kapitel vorgenommen werden. Die Erkenntnisse dieses Kapitels fließen in die gestellten Anforderungen an ein relationales DBS im Kredit- und Finanzdienstleistungswesen ein (*siehe Kapitel „3.1 Anforderungen an ein relationales DBS im Kredit- & Finanzdienstleistungswesen (Kriterienanalyse)“*).

Nur in einem stabilen Finanzsystem sind eine möglichst kostengünstige Transformation und Bereitstellung finanzieller Mittel möglich. Daher ist ein funktionierendes, verlässliches, effizientes und stabiles Bankenwesen inklusive einer Bankenaufsicht wie der *Bundesanstalt für Finanzdienstleistungsaufsicht* (BaFin) sehr wichtig für die (gesamtwirtschaftliche) Leistungsfähigkeit einer Volkswirtschaft. Bankenaufsichten wie die BaFin sorgen dafür, dass das Bankensystem verlässlich, effizient und stabil läuft. Um den Lesefluss nicht zu stören, ist die weitere Definition einer Bankenaufsicht ans Ende dieses Kapitels ausgelagert.

Das Kreditwesengesetz (KWG) gibt als wesentliche rechtliche Grundlage die wichtigsten Ziele und Aufgaben der Bankenaufsicht sowie Regeln für Kredit- und Finanzdienstleistungsinstitute vor, die bei Gründung sowie Betreiben der Geschäfte zu beachten sind. Die Regeln sind darauf ausgerichtet Fehlentwicklungen vorzubeugen, die das reibungslose Funktionieren des Bankensystems stören könnten. Der Kontrollumfang einer Bank hängt von Art und Umfang ihrer Geschäfte ab. Die Aufsicht richtet grundsätzlich ihr Hauptaugenmerk darauf, dass Institute genügend Eigenkapital und Liquidität vorhalten und angemessene Risikokontrollmechanismen installiert haben.

Das KWG wurde erstmals im Jahr 1961 veröffentlicht. Neben dem KWG gibt es noch Spezialgesetze wie das Depotgesetz oder das Bausparkassengesetz, die hier nicht weiter betrachtet werden. Gemäß § 25a Abs. 1 KWG muss ein Institut über eine ordnungsgemäße Geschäftsorganisation verfügen, welche die Einhaltung der vom Institut zu beachtenden gesetzlichen Bestimmungen und der betriebswirtschaftlichen Notwendigkeiten gewährleistet. Die Verantwortung für eine ordnungsgemäße Geschäftsorganisation trägt die Geschäftsleitung. Zu einer ordnungsgemäßen Geschäftsorganisation zählen dabei Aspekte wie ein angemessenes und wirksames

Risikomanagement, also das Management der für das jeweilige Institut wesentlichen Risiken zu denen auch operationelle Risiken zählen. Ein operationelles Risiko kann durch unterschiedliche Ereignisse entstehen wie die Unangemessenheit oder das Versagen von internen Verfahren, Menschen oder Systemen (Hardware und Software). Ein operationelles Risiko kann demnach auch durch Sicherheitslücken in der verwendeten Software wie MS Access entstehen. Das Risikomanagement umfasst Punkte wie die Planung, Festlegung, Umsetzung, Beurteilung sowie Anpassung von Strategien, zu der auch eine Geschäftsstrategie zur nachhaltigen Entwicklung des Instituts oder eine IT-Strategie (vergleiche MaRisk AT 4.2) gehören sowie die Einrichtung von internen Kontrollverfahren wie einer Internen Revision oder interne Kontrollsysteme wie eine Risikocontrolling oder Compliance-Funktion. Außerdem zählen zum Risikomanagement eine angemessene personelle sowie technisch-organisatorische Ausstattung (vergleiche MaRisk AT 7.2) des Instituts.

Zu einer angemessenen technisch-organisatorischen Ausstattung zählt auch, dass IT-Systeme die Integrität, die Verfügbarkeit, die Authentizität sowie die Vertraulichkeit der Daten sicherstellen müssen. Die Ausgestaltung ist dabei auf gängige Standards abzustellen. Vertraulichkeit, Verfügbarkeit, Integrität, Authentizität und Verbindlichkeit stellen Schutzziele der Informationssicherheit dar. Authentizität verifiziert, dass die Information oder Identität echt sind. Die Nachrichtenauthentizität garantiert dem Empfänger einer Nachricht, dass die Nachricht von dem stammt, der sich als Sender ausweist. Die Teilnehmerauthentizität gewährleistet, dass Teilnehmer ihre Identität zweifelsfrei nachweisen können. Authentizität von Daten setzt Integrität voraus, da veränderte Daten nicht authentisch sein können. Daher kann Authentizität als Bestandteil von Integrität gesehen werden. Datenintegrität gibt dem Empfänger Sicherheit, dass Daten oder Funktionen von Systemen nicht unbemerkt verändert wurden. Im Gegensatz zur Verbindlichkeit ist eine Nachricht auch dann authentisch, wenn sich der Empfänger überzeugen kann, dass die Information vom vermeintlichen Sender stammt, dies aber Dritten gegenüber nicht beweisen kann. Im Rahmen der Verbindlichkeit kann ein Empfänger gegenüber Dritten die Authentizität einer Nachricht beweisen einschließlich wer der Autor ist. Der Autor kann nicht abstreiten, dass die Nachricht von ihm stammt. Verbindlichkeit setzt Authentizität und Integrität voraus. Die Vertraulichkeit gewährleistet, dass Informationen nur durch berechtigte Personen einsehbar sind, Unbefugte Dritte dürfen den Inhalt der Kommunikation nicht erfahren. Die Verfügbarkeit besagt, dass die Daten für Befugte jederzeit abrufbar und für Systeme jederzeit nutzbar sind.

Um den Umgang mit dem KWG sowie den hier enthaltenen Anforderungen zu erleichtern, veröffentlichen Aufsichtsbehörden diverse Rundschreiben wie die MaRisk oder Bankaufsichtliche Anforderungen an die IT (BAIT), die auf dem KWG aufbauen. MaRisk sowie BAIT geben auf Grundlage von § 25a Abs. 1 KWG jeweils einen praxisorientierten Rahmen vor, um die Anforderungen dieses Paragraphen angemessen zu erfüllen [11]/[18]/[21, S. 3]/[22, S. 6, S. 12]/[23].

Die derzeit aktuelle Version der MaRisk ist Rundschreiben 05/2023 (BA) vom 29.06.2023 in der siebten Novelle. Die BAIT liegt derzeit im Rundschreiben 10/2017 (BA) in der Fassung vom 16.08.2021 vor. Beide Rundschreiben werden von der BaFin herausgegeben.

In der MaRisk wird ein flexibler und praxisnaher Rahmen für die Ausgestaltung des Risikomanagements auf Grundlage des § 25a Abs. 1 des KWG gegeben [22, S. 6]. Die

MaRisk wird teilweise durch die BAIT konkretisiert [21, S. 3]. Zielgruppe nach AT 2.1 MaRisk ist das Kredit- und Finanzdienstleistungswesen (vergleiche § 1 Abs. 1b KWG). Die Reglementierungen sollen dazu beitragen, dass in den Instituten Missständen entgegengewirkt wird, welche die Sicherheit der den Kredit- und Finanzdienstleistungsinstituten (vergleiche § 1 Abs. 1, 1a, 1b KWG) anvertrauten Vermögenswerte gefährden, die ordnungsgemäße Durchführung der Bankgeschäfte oder Finanzdienstleistungen beeinträchtigen oder erhebliche Nachteile für die Gesamtwirtschaft herbeiführen können [22, S. 8]. Das Rundschreiben umfasst Themenfelder wie die Gesamtverantwortung der Geschäftsleitung (AT 3 MaRisk) oder Allgemeine Anforderungen an das Risikomanagement (AT 4 MaRisk). Aber auch Themen wie Datenmanagement, Datenqualität und die Aggregation von Risikodaten (AT 4.3.4 MaRisk) oder die bereits erwähnte Technisch-organisatorische Ausstattung (AT 7.2 MaRisk) sind hier enthalten.

Die BAIT richtet sich primär an die Geschäftsleitung, allerdings ergibt sich der Anwenderkreis auch aus dem AT 2.1 MaRisk. Die BAIT repräsentiert die Erwartungshaltung der Aufsicht in Bezug auf die IT-Sicherheit. Die BAIT gibt auf Grundlage von § 25a Abs. 1 KWG in Form eines Rundschreibens einen flexiblen und praxisnahen Rahmen für die technisch-organisatorische Ausstattung (vergleiche § 25a Abs. 1 Nr. 4 KWG) der Institute, zu der auch das Management der IT-Ressourcen sowie das IT-Risikomanagement zählen. Die BAIT präzisiert darüber hinaus die Anforderungen des § 25b KWG (Auslagerung von Aktivitäten und Prozessen) sowie die Anforderungen aus den MaRisk mittels der Themenfelder IT-Strategie (Kapitel 1), Informationsrisikomanagement (Kapitel 3), Informationssicherheitsmanagement (Kapitel 4), Identitäts- und Rechtemanagement (Kapitel 6) oder IT-Projekte und Anwendungsentwicklung (Kapitel 7), um nur einige Beispiele zu nennen [21, S. 3].

Hauptaufgaben der BaFin sind gemäß § 6 Abs. 2 KWG Missständen im Kredit- und Finanzdienstleistungswesen entgegenzuwirken, welche die Sicherheit der den Instituten anvertrauten Vermögenswerte gefährden, die ordnungsmäßige Durchführung der Bankgeschäfte oder Finanzdienstleistungen beeinträchtigen oder erhebliche Nachteile für die Gesamtwirtschaft herbeiführen können. Die relevanten Institute müssen Rahmenbedingungen erfüllen und ihre Bücher offenlegen. Entdeckt die BaFin Defizite kann sie je nach Tatbestand aufsichtliche Maßnahmen ergreifen, welche die Geschäftspolitik eines Instituts betreffen. Bei Prüfungen dienen MaRisk sowie BAIT oft als Referenz [18]. Die *Deutsche Bundesbank* ist die unabhängige Zentralbank der Bundesrepublik Deutschland und seit 1999 Teil des Eurosystems, in dem sie zusammen mit den anderen nationalen Zentralbanken und der *Europäischen Zentralbank* für die gemeinsame Währung (Euro) verantwortlich ist. Hauptziele sind hier unter anderem die Gewährleistung der Preisstabilität im Euroraum und die Mitwirkung an der nationalen Aufsicht über Kreditinstitute (einheitliche europäische Bankenaufsicht) [20]. BaFin und *Deutsche Bundesbank* teilen sich die Aufgaben in der Bankenaufsicht, die Zusammenarbeit ist in § 7 Abs. 1 KWG geregelt. Die BaFin übt als verantwortliche Verwaltungsbehörde gemäß § 6 Abs. 1 KWG die Aufsicht über die Institute nach Maßgabe des KWG aus. Die *Deutsche Bundesbank* übernimmt gemäß § 7 Abs. 1 KWG die laufende Überwachung, wertetet während der Aufsicht einzureichende Berichte sowie Meldungen aus und prüft, ob Eigenkapitalausstattung und Risikosteuerungsverfahren angemessen sind [18].

2.7. Die Zukunft: Banking 4.0 inklusive Historie

Um auch aktuelle technische Entwicklungen wie Cloud-Computing oder KI zumindest am Rande zu berücksichtigen, soll die Zukunft des Bankings (Banking 4.0) miteinbezogen werden. Neben einer Begriffsbestimmung findet sich in diesem Kapitel auch eine zugehörige Historie vorheriger Evolutionsstufen. Die Erkenntnisse dieses Kapitels fließen in die gestellten Anforderungen an ein relationales DBS im Kredit- und Finanzdienstleistungswesen mit ein (*siehe Kapitel „3.1 Anforderungen an ein relationales DBS im Kredit- & Finanzdienstleistungswesen (Kriterienanalyse)“*).

Die Industrialisierung und somit auch die vollständige (Prozess)Automatisierung macht auch vor dem Kredit- und Finanzdienstleistungssektor und ihrer IT keinen Halt. Der Einsatz von Technologie wird zunehmend wichtiger, es entstehen neue Geschäftsmodelle die profitabler und widerstandsfähiger gegen Risiken sowie Bedrohungen aus dem Cyber-Raum sind. Hierfür werden IT sowie relevante Prozesse zunehmend verbessert, automatisiert und industrialisiert.

Automatisierung ist die Anwendung von Technologie, mit denen Angestellte in einem Unternehmen Computer konfigurieren können, um einen Geschäftsvorgang zu interpretieren und zu verarbeiten, Daten zu transformieren, eine Reaktion auf Geschäftsvorfälle auszulösen oder mit anderen Systemen zu kommunizieren. Auch aufgrund fehlender Banken-KI in der IT sowie hohen technischen Schulden/Missständen (Technical Debt) erleben Banken aktuell eine abnehmende Rendite, da sie sich bis dato nur auf Automatisierung verlassen haben. An der abnehmenden Rendite ist erkennbar, dass Automatisierung allein jedoch keine nachhaltige Lösung ist.

Technische Schulden stellen zusätzliche Kosten dar, welche die Institute investieren müssen, um ein Geschäftsmodell mit vollem Potenzial zu entwickeln (*„vergleiche Chronologie weiter unten → Banking 3.0“*). Technische Schulden entstehen bei zu knapper Projektbudgetierung oder schlechter Design-, Architektur-, Implementierungs- oder auch Technologieauswahl. Der optimale Weg zur Vermeidung von Technischen Schulden ist das Verfolgen der Vision Banking 4.0. Technische Projekte ohne diese Vision sind in der Regel nicht zukunftsweisend.

Aus Industrialisierungs- und Banking 4.0-Sicht stellt Automatisierung daher nur eine Notlösung dar, um einzelne, manuelle Aufgaben zu automatisieren. Industrialisierung und Banking 4.0 stehen jedoch für mehr als nur Automatisierung (*„vergleiche Chronologie weiter unten → Banking 4.0“*). Industrialisierung bedeutet, dass Maschinen oder Systeme in Form eines Self-Services (hoch) komplexe Aufgaben vollständig übernehmen und dabei sogar mit Menschen über menschliche Kommunikationskanäle kommunizieren (Verarbeitung von unstrukturierten Daten wie Chat/Text, Dokument, Stimme, Video). Dabei greifen die Systeme auf Banken-KI zurück. Sogar das Risiko wird mittels vorhersagender Funktionalitäten von in jedem Prozess eingebauten Systemen selbst verwaltet und gesteuert. Industrialisierung wird durch Anwendung von Technologie ermöglicht, indem nicht nur Automatisierung stattfindet, sondern auch Wissen der Finanzdienstleistungsbranche in der Technologie abgebildet wird. Industrialisierung beruht unter anderem auf Standardisierung, Konsistenz, Wiederbenutzung, Flexibilität,

Robustheit/Ausfallsicherheit, Selbstkorrektur, Selbstversorgung, Echtzeitverarbeitung sowie einem hohen Volumen/-Durchsatz (Massenverarbeitung).

Primärziel ist dabei eine Maschinenschnittstelle in Bank-Prozesse zu implementieren, um manuelle Arbeitsschritte in durch Maschinen vollautomatisierte Prozesse und somit vollautomatisch generierte Ergebnisse zu verwandeln. Infolgedessen werden die Möglichkeiten und Kapazitäten der Institute und ihrer Systeme erhöht. Es können mit zunehmender Industrialisierung immer komplexere, unstrukturiertere Prozesse unterstützt werden.

Je Banking-Evolutionsstufe steigt die Automatisierung und Industrialisierung weiter, wobei in Banking 3.0 viele manuelle Eingriffe absichtlich nicht automatisiert wurden. Aus diesem Umstand ergeben sich hohe Kosten sowie geringe Erfahrungen in den jeweiligen Technologien, die Kehrseite der Automatisierung (technische Schulden). Technische Schulden machen Investitionen in Industrialisierung und Automatisierung unattraktiv und nicht nachhaltig, da sie erst beglichen werden müssen, um ein Geschäftsmodell mit vollem Potenzial zu entwickeln. Dieser Teufelskreis verlangsamt den Wandel zu Banking 4.0 [232, S. 6-10, 12, 19].

Weitere Details können der nachfolgenden Chronologie entnommen werden:

Banking 1.0 (circa 1980-2000): Erste Evolutionsstufe. Hauptmerkmal ist die erstmalige Verwendung von Technologie zur Automatisierung. Dabei wurde auf Großrechner (Mainframes) gesetzt [232, S. 1].

Banking 2.0 (circa 1995-2010): Startete mit dem Einsatz von interoperablen, offenen Systemen. Dabei wurden auch zunehmend Programmiersprachen der 4. Generation wie SQL eingesetzt, um unter anderem Kernbanksysteme zur Abbildung der Kernprozesse einer Bank oder Kreditmanagementsysteme zu implementieren. Programmiersprachen der 4. Generation (deklarative Programmierung) eröffneten das Zeitalter von Banking 2.0. Im Gegensatz zu Programmiersprachen der 3. Generation (imperative Programmierung zu denen auch objektorientierte Sprachen wie Java, C# zählen) werden mit der 4. Generation deutlich weniger Codezeilen benötigt. In der 3. Generation steht das „wie“ im Vordergrund, eines der Merkmale sind Kontrollstrukturen. In der 4. Generation steht das „was“ beziehungsweise die Aufgabenbeschreibung im Vordergrund [232, S. 1-2]/[269].

Banking 3.0 (circa 1998-2019): Banking 3.0 startete kurz vor der Jahrtausendwende und wurde maßgeblich durch die Einführung des Internets und mobiler Technologien beeinflusst. Ungefähr im Jahr 2005 war das Internet allgegenwärtig, es digitalisierte Bezahlvorgänge und führte zu Self-Service-Payments sowie -Transkationen. Hiermit können Kunden ihre Anliegen, wie eine Überweisung auf elektronischem Weg, selbstständig und unabhängig von Öffnungszeiten oder Service-Mitarbeitenden bearbeiten. Circa im Jahr 2000 wurden elektronische Bezahlvorgänge noch in Stapelverarbeitung (Batch-Modus) ausgeführt, bereits im Jahr 2020 sind circa 90 % der weltweiten Transaktionen in Echtzeit verarbeitet worden. Auch der Start des mobilen Bankings zur Abwicklung von Bankgeschäften um das Jahr 2005 gehört zu dieser Evolutionsstufe. Im Banking 3.0 liegt der Fokus auf der Automatisierung von manuellen Schritten mit

geringer Komplexität sowie hohem (Massenverarbeitung, hohe Güteranzahl) bis mittlerem Volumen, wie die Verarbeitung von Geschäftsvorfällen, Zahlungen oder Analysen. Applikationen sind auf einer Stand-alone-Basis mit monolithischer Architektur implementiert worden, was zu immensen Redundanzen bei Daten (über 80 %) und in Applikationen (über 50 %, inklusive Quellcode) führt. Komplexe Aufgaben wie die Risikovorhersage oder die Verwendung von KI mit Bankenwissen oder manuelle Aufgaben mit kleinem Volumen (Gegenteil von Massenverarbeitung, sind je Programm individuell), wie die Datenerfassung in Systemen durch Lesen oder Screening von Dokumenten, E-Mails, Bildschirmen oder Aufgaben die häufigen Änderungen ausgesetzt sind, wurden nicht automatisiert, da Banking 3.0-Anwendungen aufgrund ihrer Architektur sehr unflexibel sind und Änderungen daher ohnehin schon sehr lange dauern. Bei Umsetzung der zuvor genannten Punkte wären Change-Zyklen kaum noch steuerbar. Rückblickend hat die Beibehaltung manueller Aufgaben mit kleinem Volumen technische Missstände verstärkt. Auch der Einsatz von „Robotic Process Automation“ ist Bestandteil von Banking 3.0, um manuelle Aufgaben als Workaround zu automatisieren. Da im Banking 3.0 die Priorität auf einer schnellen Produkteinführungszeit sowie Kostenoptimierung liegt, wird oft nur unzureichend digitalisiert oder die Programme werden unzureichend flexibel und konfigurierbar umgesetzt. Auf Self-Services wird nur bei taktischer Relevanz gesetzt. Dies hat zur Folge, dass derzeit noch viele manuelle Eingriffe, Abstimmungen und Instandhaltungsmaßnahmen notwendig sind und es Kredit- und Finanzdienstleistungsinstituten daher schwerfällt, neue Dienste in die Systemlandschaft zu integrieren oder Anpassungen vorzunehmen (hierunter fallen auch Fintech Geschäftsmodelle (innovative Finanzdienstleistungen) wie Blockchain und Smart Contracts und Cyber Security-Analysen). Die technischen Missstände und Limitierungen der vorhandenen technischen Möglichkeiten verhindern es Geschäftsmodelle schnell zu verbessern und auf neue Möglichkeiten anzupassen. Dieser Missstand kann nur durch Industrialisierung und somit Transformation zu Banking 4.0 behoben werden. Der grundlegende Paradigmenwechsel benötigt jedoch enorme Investitionen in Technologie, um die technischen Schulden zu begleichen, ohne kurzfristige Geschäftsvorteile, was unter Umsatzdruck eine echte Herausforderung ist, und die Gewinnspanne reduziert [40]/[232, S. 2-6, S. 120].

Banking 4.0 (Vision der Finanzdienstleistungsbranche. Der Trend ist bereits heute erkennbar, die genaue Entwicklung allerdings unmöglich vorherzusagen): Soll die Missstände von Banking 3.0 lösen. Im Banking 4.0-Ansatz wird die Microservice-Architektur verfolgt (eine Anwendung besteht aus vielen voneinander unabhängigen, aufgabenbezogenen Diensten/Services (Prozesse), die miteinander kombiniert werden). Hierdurch werden folgende Eigenschaften erfüllt, um Redundanzen zu vermeiden: Reduzierung der Applikationsgröße, Standardisierung, Flexibilität und lose Kopplung. Unternehmensergebnisse werden zunehmend robuster sowie skalierbarer, Prozesse vollautomatisiert inklusive integrierter, Banken-KI und den Kunden sowie Angestellten wird eine zunehmende digitale Erfahrung in Form von Banking-as-a-Service angeboten. Mittels Banking-as-a-Service können Unternehmen aus Kundensicht zu eigenen, kleinen Banken werden, ohne sich dabei um Themen wie die Regulierung Gedanken machen zu müssen. Die Regulierung erfolgt weiterhin bei den Finanzinstituten. Das in Echtzeit zu verarbeitende Datenvolumen von strukturierten und unstrukturierten Daten

nimmt verstärkt zu, was eine Industrialisierung von Daten-Plattformen zur Verarbeitung verlangt. In dessen Folge wird neben der Banken-KI (Banking Artificial Intelligence (AI)-as-a-Service) auch der Cloud-Bereich zunehmend wichtiger, was im Gegenteil zu Banking 3.0 zunehmend flexible Änderungen oder die Resistenz/Härtung des Cyber-Raums ermöglicht. Aber auch die Verwendung von öffentlichen Daten und eine intelligente, lose Kopplung mit (internen) Datenquellen (*siehe Kapitel „2.1 Auszug relevanter Begriffe im relationalen DBS-Umfeld“*) werden zunehmend relevant. Durch Banking 4.0 werden Banken mittels Banking-as-a-Service zunehmend zum Anbieter, Vertreiber und Wegbereiter für finanzielle- und nichtfinanzielle Dienstleistungen. Zur Industrialisierung und Etablierung von Banking-as-a-Service wird in Banking 4.0 zunehmend auf APIs gesetzt, die standardisierte Zugriffe auf Dienste sowie als Art Integrations-Framework zur Integration von Applikationen und Daten verwendet werden. Die Eigenschaften von APIs fördern die lose Kopplung zwischen den einzelnen Anwendungskomponenten sowie eine Schichtentrennung. Über die entstehenden Schnittstellen können durch die zunehmende Industrialisierung Bankenaufsichtsbehörden (*siehe Kapitel „2.6 Aktueller Stand: Regulatorische Anforderungen im Kredit- & Finanzdienstleistungswesen“*), Mitarbeitenden und dem Management die unterschiedlichsten Plattformen in Kombination mit KI angeboten werden, darunter Risiko-Reporting-as-a-Service, Compliance-Reporting-as-a-Service, (reguläres) Reporting-as-a-Service. Hauptziel von Banking 4.0 ist es durch die Industrialisierungsmaßnahmen das Finanzinstitut in eine intelligente Maschinen-Schnittstelle für Kunden, Partner und Mitarbeitende zu verwandeln. Dabei sollen Mitarbeitende ihr Tagesgeschäft im Optimalfall vollautomatisiert und mit möglichst wenigen manuellen Eingriffen/Interaktionen bearbeiten, was auch als „Durchgehende Datenverarbeitung“ (Straight Through Processing oder auch Interoperabilität) bezeichnet wird [41]/[67]/[232, S. 6-9]/[268, S. 5].

Den Anlagen können in *Kapitel „Anlage 2: Banking 3.0 & 4.0“* weitere Grafiken zum Thema entnommen werden.

3. Vorbereitung

Nachdem die relevanten Begriffe und Details geklärt sind, kann mit der Vorbereitung der Sicherheitsanalyse als erste Phase begonnen werden. In diesem Kapitel werden alle Schritte durchgeführt, die für eine erfolgreiche Umsetzung der im nächsten Kapitel folgenden Sicherheitsanalyse erforderlich sind.

3.1. Anforderungen an ein relationales DBS im Kredit- & Finanzdienstleistungswesen (Kriterienanalyse)

Basierend auf dem in *Kapitel „1.3 Anwendungsszenario“* skizzierten Anwendungsfall erfolgt in diesem Kapitel die Ermittlung von Sicherheitsanforderungen an relationale DBSs (*siehe Kapitel „2.1 Auszug relevanter Begriffe im relationalen DBS-Umfeld“*) im Kredit- und Finanzdienstleistungswesen. Die Anforderungen werden aus den aktuellen Vorgaben der MaRisk und der BAIT (*siehe Kapitel „2.6 Aktueller Stand: Regulatorische Anforderungen im Kredit- & Finanzdienstleistungswesen“*) sowie der Vision Banking 4.0 nach *Kapitel „2.7 Die Zukunft: Banking 4.0 inklusive Historie“* abgeleitet. Aus dem in diesem Kapitel ermittelten Anforderungen ergibt sich gemäß § 25a Abs. 1 Satz 3 Nr. 4 KWG in Verbindung mit AT 7.2 Tz 2 MaRisk, dass für die Ausgestaltung der IT-Systeme und der zugehörigen IT-Prozesse gängige Standards zu verwenden sind [21, S. 3]. Daher erfolgt in *Kapitel „3.2 Identifikation von gängigen Sicherheitsstandards für relationale DBSs (Kriterienanalyse)“* die Auswahl gängiger Standards, aus denen dann in *Kapitel „3.3 Auswahl von Bewertungskriterien für die Kriterienanalyse“* die Selektion konkreter Bewertungskriterien erfolgt. Anhand der selektierten Bewertungskriterien erfolgt im letzten Schritt die Kriterienanalyse (*siehe Kapitel „4.1 Kriterienanalyse (Access & Vergleichs-DBS)“*). Konkrete Umsetzungsvorgaben machen weder MaRisk noch BAIT und lassen so Interpretationsspielraum, um die Anforderungen zu erfüllen.

Die MaRisk fordert im AT 4.3.4, dass die Datenstruktur und Datenhierarchie von Risikodaten wie Ausfallrisiken oder Bonitätsbeurteilungen von Kunden gewährleisten müssen, dass die Daten zweifelsfrei identifiziert, zusammengeführt und ausgewertet werden können und zeitnah zur Verfügung stehen. Soweit möglich, sind einheitliche Namenskonventionen und Datenkennzeichnungen festzulegen und institutsintern zu kommunizieren. Bei unterschiedlichen Namenskonventionen und Kennzeichnungen ist sicherzustellen, dass die Daten automatisiert ineinander überleitbar sind.

In AT 4.3.5 Tz 3 wird gefordert, dass geeignete Verfahren zur Sicherstellung der Qualität von Daten, die Modellen zugrunde liegen (Modelldaten), zu implementieren sind. Insbesondere sollen Qualitätsmängel in den zugrunde liegenden Daten erkannt und bereinigt werden können. Mittels eines statistischen oder mathematischen Modells können Eingabedaten zu quantitativen Schätzungen verarbeitet werden.

AT 7.2 MaRisk fordert unter anderem, dass IT-Prozesse, IT-Systeme (Hard- und Software) und sonstige, relevante Komponenten die Einhaltung der Integrität, Verfügbarkeit, Authentizität und Vertraulichkeit der Daten sicherstellen müssen. Dabei sind gemäß § 25a Abs. 1 Satz 3 Nr. 4 KWG in Verbindung mit AT 7.2 Tz 2 MaRisk für die

Ausgestaltung der IT-Systeme und der zugehörigen IT-Prozesse gängige Standards, wie der IT-Grundschutz des BSI oder die internationalen Sicherheitsstandards ISO/IEC 270XX der ISO, zu verwenden [21, S. 3].

Darüber hinaus ist eine angemessene IT-Berechtigungsvergabe einzurichten. Dabei ist zur Verbesserung des IT-Risikobewusstseins sicherzustellen, dass jeder Mitarbeitende nur über die Rechte verfügt, die zur Erfüllung einer konkreten Aufgabe benötigt werden (Need-to-Know-Prinzip). Das Need-to-Know-Prinzip gilt nur für natürliche Personen. In den Anforderungen an die Aufbau- und Ablauforganisation im Besonderen Teil (BTO Tz 9 MaRisk) sowie in BAIT 6.2 wird bei IT gestützter Verarbeitung eine Funktionstrennung (Segregation of Duties) durch geeignete Verfahren und Schutzmaßnahmen gefordert. Nach dem Grundsatz der Funktionstrennung dürfen Mitarbeitende keine miteinander unvereinbaren Aufgaben, wie die Datenerfassung und die damit verbundene Prüfung einschließlich der Freigabe, wahrnehmen [19]/[260].

Die Zusammenfassung von Berechtigungen in einem Rollenmodell ist möglich aber für diese Sicherheitsanalyse nicht weiter relevant. Der Umstand, dass die Eignung der IT-Systeme und der zugehörigen Prozesse regelmäßig von den fachlich und technisch zuständigen Mitarbeitenden zu überprüfen ist, unterstreicht den Nutzen dieser Sicherheitsanalyse von MS Access.

Die Anforderungen aus dem AT 7.2 MaRisk gelten in Abhängigkeit von der Kritikalität der unterstützten Geschäftsprozesse und der Bedeutung der Anwendungen für diese Prozesse auch für die von Mitarbeitenden des jeweiligen Fachbereichs entwickelten oder betriebenen Anwendungen (IDV, *siehe Kapitel „1.3 Anwendungsszenario“*). Derartige Anwendungen werden oftmals von Mitarbeitenden mit geringem IT-Kenntnissen erstellt und erfordern daher einen möglichst intuitiven Umgang sowie technische Unterstützung bei der Entwicklung. Die Festlegung von Maßnahmen zur Gewährleistung der Datensicherheit hat sich am Schutzbedarf der verarbeiteten Daten zu orientieren [21, S. 3]/[22, S. 23]. Um zusätzliche Komplexität zu vermeiden, bleiben Unterschiede im Schutzbedarf unberücksichtigt.

Aber auch in den BAIT wird in Kapitel 7 (IT-Projekte und Anwendungsentwicklung) von der Anwendungsentwicklung gefordert, dass je Schutzbedarf angemessene Vorkehrungen zu treffen sind, damit auch nach jeder Produktivstellung einer Anwendung die Vertraulichkeit, Integrität, Verfügbarkeit und Authentizität der zu verarbeitenden Daten nachvollziehbar sichergestellt ist (7.8 BAIT). Auch hier zeigt sich, dass für die Aufsichtsbehörden die Gewährleistung der Datensicherheit respektive der Schutz der Daten im Vordergrund steht. Als geeignete Vorkehrungen werden die Überprüfung der Eingabedaten, Systemzugangskontrollen, eine Benutzerauthentifizierung, eine Transaktionsautorisierung, eine Protokollierung der Systemaktivitäten, Prüfpfade (Audit Logs), die Verfolgung von sicherheitsrelevanten Ereignissen oder die Behandlung von Ausnahmen genannt [21, S. 22].

Aber auch die Integrität der Anwendung (insbesondere des Quellcodes, also der Logik) ist in geeigneter Weise sicherzustellen (7.9 BAIT). Dabei sind Vorkehrungen zu treffen, die versehentliche oder absichtliche Manipulationen an der Anwendung transparent machen.

In Kapitel 6 (Identitäts- und Rechtemanagement) der BAIT wird ein Berechtigungskonzept gefordert und das bereits in den MaRisk vorausgesetzte Need-to-Know-Prinzip zur Vergabe von Berechtigungen nach dem Sparsamkeitsgrundsatz durch das Least-Privilege-Prinzip erweitert (6.2 BAIT). Im Gegensatz zum Need-to-Know-Prinzip bezieht das Least-Privilege-Prinzip zusätzlich auch nicht-menschliche Benutzende wie Systeme, Anwendungen oder andere vernetzte Geräte mit ein [260].

Dabei können Zugangs- und Zugriffsberechtigungen auf IT-Systeme auf allen Ebenen eines IT-Systems, wie dem Betriebssystem, der DB oder Anwendung, umgesetzt sein (6.2 BAIT). Die Zugriffe und Zugänge müssen jederzeit, zweifelsfrei und idealerweise automatisiert einer handelnden Person zugeordnet werden können (6.3 BAIT) und es sind je nach Schutzbedarf und bei besonders kritischen Benutzer- und Zugriffsrechten Prozesse zur Protokollierung und Überwachung (Monitoring) einzurichten. Dabei muss in einem Audit überprüfbar sein, dass die Berechtigungen nur wie vorgesehen eingesetzt werden (6.7 BAIT). Die im Folgenden nicht weiter berücksichtigten Verfahren zur Einrichtung, Änderung, Deaktivierung oder Löschung von Zugriffsrechten müssen durch Genehmigungs- und Kontrollprozesse sicherstellen, dass die Vorgaben des Berechtigungskonzepts eingehalten werden.

Eine Umgehung der Vorgaben anhand der jeweiligen Berechtigungskonzepte soll durch flankierende TOM wie die Auswahl angemessener und starker Authentifizierungsverfahren, die Implementierung einer Richtlinie zur Wahl sicherer Passwörter, die Verschlüsselung von Daten oder die Implementierung einer manipulationssicheren Protokollierung vorgebeugt und bestenfalls verhindert werden (6.8 BAIT) [21, S 15-17].

In Kapitel 8.7 BAIT werden Vorgaben für Verfahren zur Datensicherung (Datensicherungskonzept) gefordert [21, S 25].

Die Zukunft anhand der Vision von Banking 4.0 und die damit verbundene Industrialisierung ergänzen Themen wie die Verarbeitung großer Datenmengen, Skalierbarkeit, Cloud-Fähigkeit sowie die Zentralisierung der Datenhaltung, um die KI bestmöglich mit Bankenwissen versorgen zu können oder neues Wissen mit Data Mining-Ansätzen zu generieren [70, S. 41-42]. Aber auch die Säulen der Industrialisierung wie Standardisierung, Wiederverwendung, Flexibilität, Robustheit/Ausfallsicherheit, eine lose Kopplung, Schichtentrennung, die Schaffung von API-Lösungen sowie die Reduzierung von technischen Missständen und Fehlentwicklungen werden wichtiger.

Abschließend werden die zuvor in diesem Kapitel identifizierten Anforderungen an ein relationales DBS im Kredit- und Finanzdienstleistungswesen in Themenbereiche überführt (in Anlehnung an die rechtlichen Anforderungen der Datenschutz-Grundverordnung (DS-GVO), die im Standard-Datenschutzmodell in Gewährleistungsziele überführt werden) [38, S. 24 ff.]. Den Themenfeldern werden dann anhand der nachfolgend ausgewählten Sicherheitsstandards konkrete Bewertungskriterien zugeordnet:

Tabelle 2: Anforderungen an ein DBS im Kredit- und Finanzdienstleistungswesen – Übersicht der ermittelten Themenbereiche

Nummer	Themenbereich
1	SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen
2	Konfiguration
3	Kryptographie
4	Passwörter & Authentifizierung
5	Logging & Auswertungsmöglichkeiten
6	Berechtigungen & Autorisierung
7	Datenschutzkonformer Zugriff
8	Datensicherung
9	Banking 4.0

3.2. Identifikation von gängigen Sicherheitsstandards für relationale DBSs (Kriterienanalyse)

Gemäß den in *Kapitel „3.1 Anforderungen an ein relationales DBS im Kredit- & Finanzdienstleistungswesen (Kriterienanalyse)“* identifizierten Anforderungen sind für die Ausgestaltung der IT-Systeme und der zugehörigen Prozesse gängige Standards wie der IT-Grundschutz des BSI oder die internationalen Sicherheitsstandards ISO/IEC 270XX der ISO zu verwenden. Daher erfolgt in diesem Kapitel die Auswahl gängiger Standards, aus denen dann in *Kapitel „3.3 Auswahl von Bewertungskriterien für die Kriterienanalyse“* die konkreten Bewertungskriterien selektiert werden. Die abschließende Verwendung der ausgewählten Bewertungskriterien erfolgt während der Sicherheitsanalyse in *Kapitel „4.1 Kriterienanalyse (Access & Vergleichs-DBS)“* im Rahmen der Kriterienanalyse von Access sowie dem ausgewählten Vergleichs-DBS (*siehe Kapitel „3.4 Auswahl Vergleichs-DBS zur Kriterienanalyse (Azure SQL-DB)“*). Das BSI ist gemäß § 1 BSI-Gesetz eine Bundesbehörde im Geschäftsbereich des *Bundesministeriums des Innern, für Bau und Heimat*. Das BSI ist die zentrale Stelle für Informationssicherheit auf nationaler Ebene. Es fördert die Sicherheit in der IT mit dem Ziel, die Verfügbarkeit, Integrität und Vertraulichkeit von Informationen und deren Verarbeitung zu gewährleisten (§ 3 BSI-Gesetz). Der IT-Grundschutz des BSI beinhaltet alles, was mit dem Thema Informationssicherheit in Zusammenhang steht. Im Mittelpunkt des IT-Grundschutzes stehen die IT-Grundschutz-Bausteine. In jedem Baustein wird ein Thema wie „Office-Anwendungen“ oder „relationale DBSs“ zu allen relevanten Sicherheitsaspekten analysiert. Der erste Teil eines Bausteins enthält mögliche Gefährdungen, der zweite Teil wichtige Sicherheitsanforderungen [8].

Da der IT-Grundschutz des BSI Maßnahmen der ISO/IEC-Normen 27001 und 27002 interpretiert und die ISO/IEC-Normen somit im IT-Grundschutz aufgehen, werden die ISO/IEC-Normen bei der Quellenauswahl nicht weiter berücksichtigt. Der IT-Grundschutz gibt eine konkrete Hilfestellung für die generischen ISO/IEC-Anforderungen [17]. Ähnlich verhält es sich mit dem Standard of Good Practice For Information Security (SOGP) vom *Information Security Forum* (ISF). Der SOGP interpretiert neben ISO/IEC 27002 noch weitere Standards wie die *Cloud Security Alliance* (CSA) Cloud Control Matrix, die allerdings auch auf ISO/IEC-Standard 27001 und weiteren

Normen der ISO/IEC-27000-Standardserie aufbaut [58]/[90]. Daher werden auch das SOGP und sonstige Standards, die auf der ISO/IEC-27000-Standardserie aufbauen nicht weiter berücksichtigt.

Das ISO ist eine unabhängige Nichtregierungsorganisation und der weltweit größte Entwickler freiwilliger internationaler Standards. Die IEC ist eine weltweit führende Organisation für die Veröffentlichung von internationalen Standards für elektrische, elektronische und verbundene Technologien [60]/[61].

Die ISO/IEC-27000-Normenreihe umfasst unzählige Kontrollmechanismen und globale Sicherheitsstandards, die Organisationen aller Art und Größen dabei unterstützen ihre Informationsbestände sicher zu verwalten. Sie bieten einen Rahmen für Richtlinien und Verfahren, die alle rechtlichen, physischen und technischen Kontrollen umfassen, welche an den Prozessen zum Informationsrisikomanagement einer Organisation beteiligt sind [115].

ISO/IEC 27001 unterstützt Unternehmen bei der Erfüllung regulatorischer und gesetzlicher Anforderungen an die Informationssicherheit. Sie gibt Empfehlungen zum Aufbau eines Informationssicherheitsmanagementsystems (Information Security Management System (ISMS)) einschließlich zur Gründung, Überwachung, Aufrechterhaltung und kontinuierlicher Weiterentwicklung. Die Einrichtung eines ISMS ist abhängig von dynamischen Faktoren wie Unternehmensanforderungen, Sicherheitsansprüche oder den Prozessen. Anforderungen an Dokumentation, Zugriffssteuerung, Sicherheit und Verantwortlichkeiten in Form von Best Practices werden ebenso wie Begriffsbestimmungen in diesem Standard behandelt. Ziel des ISMS ist es die Informationssicherheit unter ausdrückliche Verwaltungskontrolle zu bringen und dabei die Vertraulichkeit, Integrität, Authentizität und Verfügbarkeit von Daten durch Anwendung eines Risikomanagementprozesses zu gewährleisten, um die auftretenden Risiken angemessen zu managen. Damit die Geschäftsziele erreicht werden können, stellt das ISMS sicher, dass die gespeicherten und verarbeiteten Daten verantwortungsvoll und sicher aufbewahrt werden und vor Bedrohungen sowie Schäden geschützt sind. Die Azure-Cloud, MS 365 sowie die Power Plattform sind ISO/IEC 27001-zertifiziert [65]/[66]/[115].

Im Gegensatz dazu dient die ISO/IEC-Norm 27002 als Referenz, um Kontrollen zur Behandlung von Informationssicherheitsrisiken im ISMS oder allgemein anerkannte Kontrollen im Rahmen der Informationssicherheit zu definieren und zu implementieren. Informationssicherheit wird durch die Implementierung geeigneter Kontrollen, Richtlinien, Regeln und Prozesse erreicht, welche Risiken identifizieren sowie mitigieren. Um die unternehmensspezifischen Sicherheitsziele zu erreichen, müssen die implementierten Kontrollen, Richtlinien und Prozesse überwacht, überprüft und bei Bedarf verbessert werden. Dafür ist das in ISO/IEC 27001 spezifizierte ISMS verantwortlich, da es die Informationsrisiken in einem Unternehmen ganzheitlich betrachtet sowie koordiniert und darauf aufbauend (Informationssicherheits)Kontrollen definiert und durchführt [66].

Aus den oben genannten Gründen ergeben sich folgende Sicherheitsstandards für die Auswahl der Bewertungskriterien für die Kriterienanalyse:

Tabelle 3: Übersicht über ausgewählte Sicherheitsstandards für die Kriterienanalyse

1. IT-Grundschutz Kompendium (Edition 2023) des BSI. Nachfolgend berücksichtigte System-Bausteine [8]/[9]:
APP.1.1 Office-Produkte *(gemäß diesem IT-Grundschutz-Baustein muss bei Verwendung von integrierten DBSs, wie Access in MS Office oder auch Base in LibreOffice, der Baustein „APP.4.3 Relationale Datenbanken“ berücksichtigt werden)*
APP.4.3 Relationale Datenbanken
APP.6 Allgemeine Software
2. Empfehlungen des BSI zur sicheren Konfiguration von Microsoft-Office-Produkten (Access 2013/2016/2019) *(hierauf wird unter anderem im IT-Grundschutz-Baustein „APP.1.1 Office-Produkte“ verwiesen [5])*
3. Folgende technische Sicherheitsanforderungen der Deutsche Telekom AG (Edition Dezember 2022), die für deren Produkte und Prozesse gelten *(im IT-Grundschutz-Baustein „APP.4.3 Relationale Datenbanken“ wird auf die Telekom-Sicherheitsanforderungen verwiesen [259]):*
3_16_Datenbanksysteme_v4.1
3_30_Microsoft_SQL_Server_i36
3_50_Kryptographische_Algorithmen_und_Sicherheitsp_v6.1
3_60_PostgreSQL_Datenbanken_v4.1
4. Folgende Cheat Sheets der Cheat Sheet Serie von der *Non-Profit-Organisation Open Web Application Security Project* (OWASP) [241]:
Database Security Cheat Sheet
Query Parameterization Cheat Sheet
SQL Injection Prevention Cheat Sheet

3.3. Auswahl von Bewertungskriterien für die Kriterienanalyse

Den in *Kapitel „3.1 Anforderungen an ein relationales DBS im Kredit- & Finanzdienstleistungswesen (Kriterienanalyse)“* definierten Themenbereichen werden in diesem Kapitel konkrete Bewertungskriterien zugeordnet (siehe *Kapitel „3.3.2 Festlegung von relevanten Bewertungskriterien“*). Quelle für die Auswahl sind die in *Kapitel „3.2 Identifikation von gängigen Sicherheitsstandards für relationale DBSs (Kriterienanalyse)“* identifizierten Sicherheitsstandards. Um den Fokus auf Access als dateibasiertes DBS nicht zu verlieren und den Umfang der vorliegenden Arbeit nicht zu sprengen, erfolgt ein begründeter Ausschluss von grundsätzlich wichtigen Bewertungskriterien, die jedoch für die Sicherheitsanalyse eines dateibasierten DBS wie Access ungeeignet sind (*siehe Kapitel „3.3.1 Ausschluss von für dateibasierte DBSs unpassende Bewertungskriterien“*).

Eine Übersicht mit den festgelegten Themenbereichen findet sich am Ende des *Kapitels „3.1 Anforderungen an ein relationales DBS im Kredit- & Finanzdienstleistungswesen (Kriterienanalyse)“*.

3.3.1. Ausschluss von für dateibasierte DBSs unpassende Bewertungskriterien

Die nachfolgende Auflistung beinhaltet grundsätzlich wichtige aber für ein dateibasiertes DBS wie Access ungeeignete Sicherheitsanforderungen mit einer kurzen Begründung für den Ausschluss. Einige Punkte aus dem Web-Anwendungs- respektive Client-Server-Umfeld wie der Zugriff auf das Dateisystem oder das Ausführen von Betriebssystembefehlen sind für dateibasierte DBSs wie Access unpassend und werden daher nicht berücksichtigt. Die in diesem Kapitel ausgeschlossenen Bewertungskriterien können zusammen mit den in *Kapitel „3.3.2 Festlegung von relevanten Bewertungskriterien“* ausgewählten Bewertungskriterien für die Sicherheitsanalyse von Client-Server-DBS wie der Azure SQL-DB genutzt werden:

Tabelle 4: Übersicht über ausgeschlossene, für dateibasierte DBSs unpassende Bewertungskriterien

2. Themenbereich: Konfiguration
Bewertungskriterien
Der Zugriff auf das Dateisystem muss per Default deaktiviert sein.
Das Ausführen von Betriebssystembefehlen muss per Default deaktiviert sein.
Default-DBs müssen gelöscht werden können.
Default-Benutzer wie das DB-Administratorkonto (sa) und -Rollen müssen gelöscht werden können.
Begründung für den Ausschluss
Access-Anwendungen laufen unter dem jeweils angemeldeten Windows-Nutzer mit seinen jeweiligen Rechten, über die mit den zugewiesenen Rechten auf das Dateisystem zugegriffen oder Betriebssystem-Befehle ausgeführt werden können (desktopzentriert) [211]. Default-Benutzer sind nicht relevant, da MS Access im Dateiformat .accdb oder .accde keine Sicherheit auf Benutzerebene unterstützt, die es bis zur Einführung im Jahr 2007 in den Vorgänger-Dateiformaten wie .mdb, .mde oder .ade noch gab. Anzumerken ist, dass diese Funktionalität nicht der Datensicherheit, sondern zur Benutzerführung diente. Die VBA-Funktion „`Application.CurrentUser`“ wird im aktuellen Dateiformat .accdb nur noch aus Abwärtskompatibilitätsgründen unterstützt und liefert als Rückgabewert stets „`Admin`“. ACHTUNG: Die Dateiformate vor dem Jahr 2007 besitzen unter anderem eine schwache Verschlüsselungsfunktionalität.
3. Themenbereich: Kryptographie
Bewertungskriterien
Die Kommunikation beziehungsweise der Datenaustausch zwischen Anwendung und DBS muss verschlüsselt erfolgen und per Default aktiviert sein.
Begründung für den Ausschluss
Access ist ein dateibasiertes DBS, der zugehörige Prozess läuft lokal auf dem Client und auf keinem Server. Daher wird dieser Punkt in der Access-Sicherheitsanalyse ausgeschlossen (*siehe Kapitel „2.5 Unterschied dateibasierte & Client-Server DBS“*). Es erfolgt keine Analyse der Kommunikation zwischen Dateiserver und Client (*siehe Kapitel „1.2 Abgrenzung“*).
4. Themenbereich: Passwörter & Authentifizierung
Bewertungskriterien

Nutzende müssen nach einem konfigurierbaren Intervall automatisch vom System zur Passwortänderung aufgefordert werden.
Begründung für den Ausschluss
Regelmäßige Passwortänderungen werden weder von MS, BSI noch Geheimdiensten empfohlen. Regelmäßiges Ändern kann bei Nutzenden zu vorhersagbaren Passwortvergabe-Schemata führen und die wiederrum zu schwachen Passwörtern. Ein gutes Passwort kann über Jahre sicher verwendet werden, da Brute-Force-Angriffe hier in der Regel sehr zeitaufwändig sind [15]/[55]/[135]. Aus diesem Grund werden keine Anforderungen für automatische Passwortänderungen aufgenommen. Sie sind aus Sicht des Verfassers generell bei einer Sicherheitsanalyse nicht zu berücksichtigen.
6. Themenbereich: Berechtigungen & Autorisierung
Bewertungskriterien
Befehle müssen über einen Verfahrensuser ausgeführt werden können, der mit minimalen Berechtigungen ausgestattet ist.
Begründung für den Ausschluss
Als dateibasiertes DBS wird eine Access-Datei unter dem jeweiligen Benutzer, inklusive seinen zugeteilten Rechten ausgeführt. Dabei ist anzumerken, dass es Vorbehalte gegen die Nutzung von Verfahrensusern gibt und an deren Stelle die Nutzung von Active Directory (AD) in Verbindung mit der Vergabe von individuellen Zugriffsrechten je Benutzer empfohlen wird. Im SQL Server ist das unter anderem über den Windows Authentication Mode und im Azure SQL-DB unter anderem über die Azure AD-Authentifizierung (Umbenennung in „Microsoft Entra ID“ geplant) möglich. Durch dieses Vorgehen sind auch Direktzugriffe auf das DBS über andere Tools und Systeme wie DbVisualizer geschützt [28]. *Der Windows-Modus ist ein eng mit dem Windows-Betriebssystem und AD verzahntes Sicherheitsmodell. Mit dem (Azure) AD wird ein zentraler Ort für die einheitliche Verwaltung von Benutzern sowie zugeteilten Berechtigungen für diverse Dienste wie DB-Nutzer geschaffen. Der Zugriff wird über vorhandene Windows-User sowie -Gruppen gesteuert, die für die einzelnen Dienste und DBs mit den jeweils notwendigen Zugriffsrechten ausgestattet werden. User-Namen und zugehörige Passwörter müssen so nicht über das Netzwerk übertragen werden, es wird der aktuell am System angemeldete Nutzende verwendet und es wird auch eine tokenbasierte Authentifizierung sowie Single Sign-on-Funktionalität unterstützt [87]/[163].*

3.3.2. Festlegung von relevanten Bewertungskriterien

Folgende Bewertungskriterien werden nach dem Ausschluss in *Kapitel „3.3.1 Ausschluss von für dateibasierte DBSs unpassende Bewertungskriterien“* für die Kriterienanalyse in *Kapitel „4.1 Kriterienanalyse (Access & Vergleichs-DBS)“* festgelegt:

Tabelle 5: Übersicht über ausgewählte, für dateibasierte DBSs passende Bewertungskriterien

1. Themenbereich: SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen
Es muss einen Schutz gegen SQL-Injection-Angriffe geben.
Es muss die Möglichkeit geben Transaktionen zu nutzen (inklusive Rollback-Möglichkeit).
Es müssen Stored Procedures und Views definiert sowie Prepared Statements verwendet werden können.
Die Möglichkeit zur Ausführung von Mehrfachabfragen (Multi-Query-Statements) muss per Default deaktiviert sein.
2. Themenbereich: Konfiguration
Nicht benötigte Dienste, SQL-Befehle oder Funktionen müssen deaktiviert oder deinstalliert werden können.
Die Verwendung von Triggern muss deaktiviert werden können.
Ein definierter einheitlicher Konfigurationsstandard muss DBS beziehungsweise dateiübergreifend konfiguriert und überwacht werden können.
3. Themenbereich: Kryptographie
Als symmetrische Blockverschlüsselung muss der Advanced Encryption Standard (AES)-256 verwendet werden können.
Das Verschlüsselungsverfahren muss austauschbar sein.
Digitale Signaturen müssen verwendet werden können.
4. Themenbereich: Passwörter & Authentifizierung
Auswählbare Passwörter als Authentisierungsmerkmal müssen beliebig lang sein und Groß-, Kleinbuchstaben, Ziffern, Sonderzeichen akzeptieren.
Passwortwiederverwendungen bei Authentisierungsmerkmalen müssen erkannt werden.
Eingegebene Passwörter dürfen per Default nicht im Klartext angezeigt werden.
Es muss einen Schutz vor Brute-Force-Angriffen auf Passwörter, wie die Sperrung oder Dateilöschung bei Überschreiten einer maximalen Anzahl falscher Passworteingaben, geben.
Anstelle des Mixed-Modus beziehungsweise der SQL-Authentifizierung, in dem die Nutzerverwaltung von Username mit zugehörigem Passwort im jeweiligen DBS erfolgt und somit über unterschiedliche Systeme verteilt wird, muss der Windows-Modus, Azure AD oder eine vergleichbare Funktionalität verwendet werden können.
5. Themenbereich: Logging & Auswertungsmöglichkeiten
Alle Zugriffe auf das DBS, -dienste, DB-Procedures, DB-Inhalte, sonstige Befehlsausführungen oder Änderungen müssen automatisiert geloggt werden.

	Es müssen Transaktionslogs und Informationen über ausgeführte Abfragepläne geführt werden. *Optional: Im Idealfall können Transaktionslogs und DB-Dateien auf unterschiedlichen Festplatten gespeichert werden.*
	Log-Daten müssen an einen Log-Server weitergeleitet werden können.
	Es muss ein vollautomatisierbares Monitoring aller Betriebszustände und Modifikationen möglich sein, inklusive der Möglichkeit über einen konfigurierbaren Alarm benachrichtigt zu werden.
6. Themenbereich: Berechtigungen & Autorisierung	
	Es muss beim DBS-Zugriff die Möglichkeit geben das Need-to-Know-Prinzip und Least-Privilege-Prinzip einzuhalten. Die Zugriffsrechte des jeweiligen Nutzers müssen mit minimalen Privilegien konfigurierbar sein.
	Das DBS-Backend muss so isoliert wie möglich sein, um unerlaubten Zugriff zu verhindern.
	Der Zugriff auf Systemtabellen muss eingeschränkt werden können.
7. Themenbereich: Datenschutzkonformer Zugriff	
	Es muss die Möglichkeiten für einen datenschutzkonformen Zugriff auf die Daten geben.
	Für Log-Dateien muss eine automatische Löschroutine aktiviert werden können.
8. Themenbereich: Datensicherung	
	Es muss die Möglichkeit geben automatische Datensicherungen zu konfigurieren.
9. Themenbereich: Banking 4.0	
	Es müssen auch große Datenmengen verarbeitet werden können.
	Die Zentralisierung der Datenhaltung muss gefördert werden. Dabei muss ein Überblick über alle vorhandenen Daten möglich sein, um möglichst effizient eine KI mit Bankenwissen trainieren oder Data Mining betreiben zu können.
	Die Nutzung des DBS in der Cloud muss möglich sein.
	Es dürfen keine technischen Schulden entstehen.

3.4. Auswahl Vergleichs-DBS zur Kriterienanalyse (Azure SQL-DB)

Unter den Top 20 von 420 der populärsten DBMSs sind gemäß *DB-Engines* Stand August 2023 drei MS-Produkte vertreten. Neben dem SQL Server auf Platz 3 ist MS mit Access auf Platz 9 sowie mit der Azure SQL-DB, die Azure-Cloud-Variante des SQL Servers, auf Platz 16 vertreten. Im Ranking unter den relationalen DBMSs landet die Azure SQL-DB auf Platz 10 von 169 (*siehe Kapitel „1.1 Relevanz & wissenschaftlicher Mehrwert“*) [128]/[150].

In den Access-Einstellungen kann unter „*Datei → Optionen → Objekt-Designer*“ eine zum MS „SQL Server kompatible Syntax (ANSI 92)“ konfiguriert werden, um Punkte wie verwendete Zeichen für Platzhalter (Wildcards) auszutauschen [275, S. 237]:

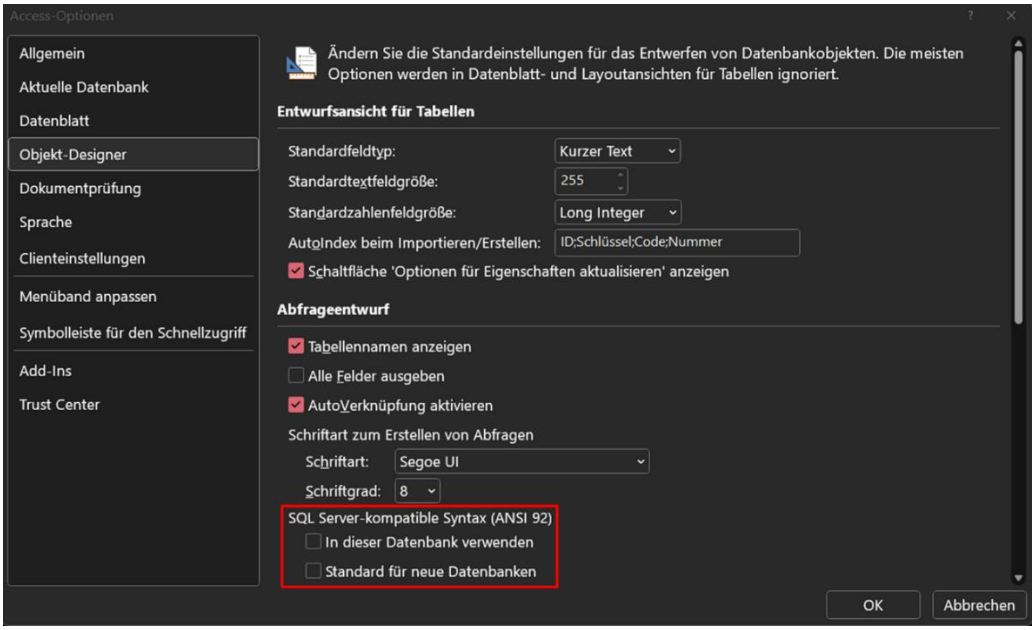


Bild 15: Access SQL-Dialekt-Konfiguration – SQL Server-kompatible Syntax (ANSI 92)

Auf diversen MS-Websites wird mit einer einfachen Migration und einer guten Kompatibilität mit dem MS SQL Server geworben. MS bietet ausführliche sowie umfassende Online-Dokumentationen für seine Produkte an [121]/[200]/[214].

Aber auch die Cloud-Technologie gewinnt gemäß *Gartner* „Hype Cycle for Emerging Technologies 2022“ sowie *Gartner* „2021-2023 Emerging Technology Roadmap for Large Enterprises“ zunehmend an Bedeutung:

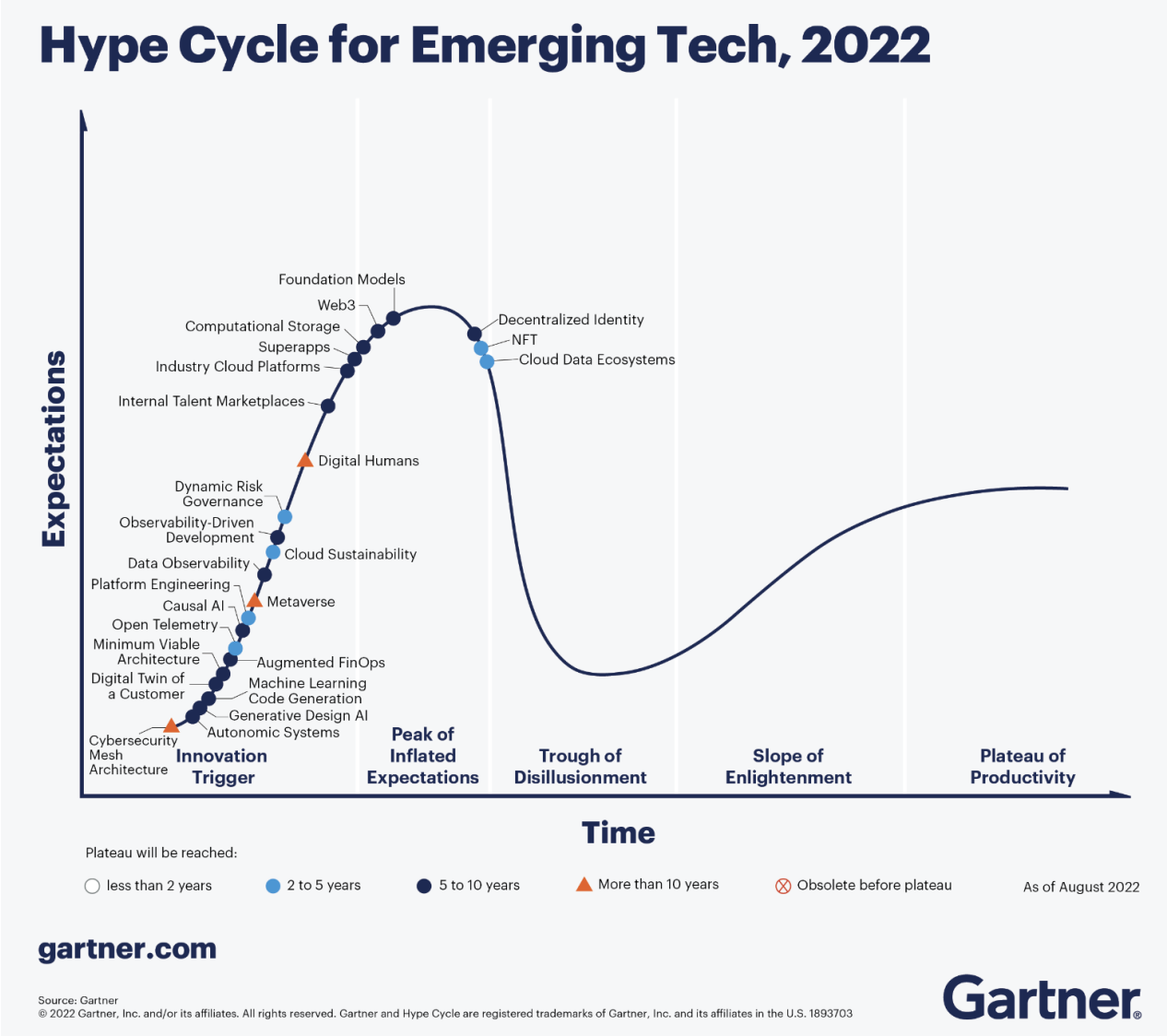


Bild 16: Gartner Hype Cycle for Emerging Tech, 2022 [44]

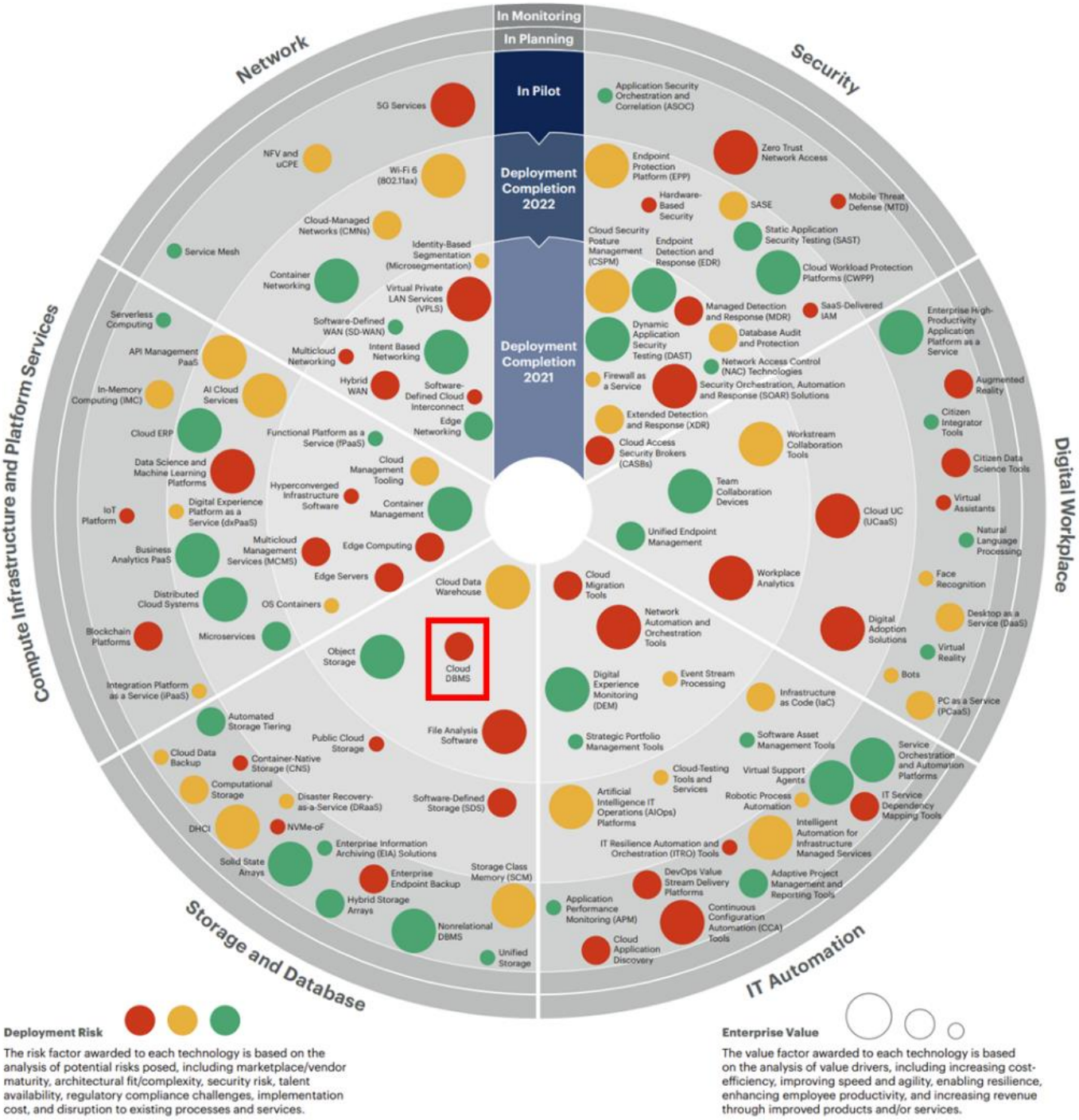


Bild 17: Gartner 2021-2023 Emerging Technology Roadmap for Large Enterprises [42]

Der *Gartner* Hype Cycle zeigt, dass Cloud-Technologie schon längst keine Vision mehr ist. Das Plateau der Produktivität wird in zwei bis fünf Jahren erwartet. Die Emerging Technology Roadmap zeigt, dass von Cloud-DBMSs aufgrund der hohen architektonischen Komplexität, der hohen Sicherheitsrisiken und Implementierungskosten, des Mangels an Fachexperten auf dem Arbeitsmarkt sowie dem Bruch zu bisherigen Prozessen ein hohes Risiko ausgehen kann. Dabei ermöglicht die Cloud-Technologie eine mittlere Steigerung des Unternehmenswerts in Bezug auf Kosteneffizienz,

Flexibilität, Resilienz (Widerstandsfähigkeit), Mitarbeitendenproduktivität und höhere Einnahmen durch verbesserte Produkte. Cloud-Plattformen für die Industrie stehen an vierter Stelle der zehn wichtigsten strategischen Technologietrends für 2023. Laut *Gartner* werden bis 2027 mehr als 50 % der Unternehmen Cloud-Plattformen nutzen, um ihre Geschäftsinitiativen zu beschleunigen [42]/[43]/[44]. Eine derartige Entwicklung wird auch bei Banking 4.0 erwartet (*siehe „2.7 Die Zukunft: Banking 4.0 inklusive Historie“*):

Bild 18: Die 10 wichtigsten strategischen Technologie-Trends von Gartner für 2023 [43]

Ein ähnliches Bild zeigt ein Ranking von Technologien nach erwarteter globaler Bedeutung im Jahr 2025 von *Statista*. Hier liegt die Cloud-Technologie auf Platz 10 von 16. Aber auch weitere Banking 4.0-Technologien wie KI oder Big Data sind hier vertreten [256]:

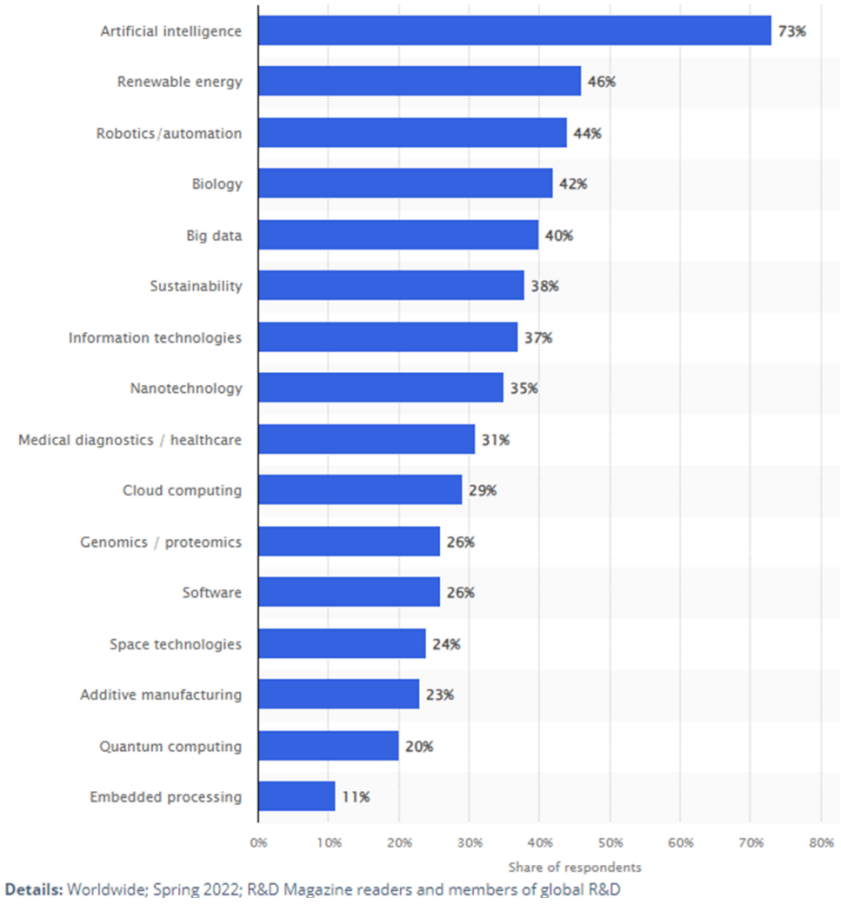


Bild 19: Ranking von Statista über die weltweit wichtigsten Technologien nach erwarteter Bedeutung aus Mai 2022 [256]

Für einige Anwendungsfälle scheint der von MS Access offiziell unterstütze Multi-User Support von maximal 255 gleichzeitigen Nutzern auszureichen, wobei MS eine maximale Nutzeranzahl zwischen 25 und 50 empfiehlt. Bei mehr als 50 Nutzern rät MS zur Migration auf eine Client-Server-Lösung [105]/[173]/[179]. Spätestens bei Überschreitung der von Access maximal unterstützen Nutzerzahl muss eine Alternative gefunden werden.

Das Gegenteil von Peer-to-Peer-Netzwerken (jeder Host kann Ressourcen seines Rechners für andere im Netzwerk freigeben und tritt somit gewissermaßen als Server in Erscheinung) sind Netzwerke basierend auf Client-Server-Architekturen, die neben einem erweiterten Funktionsumfang auch für eine große Nutzeranzahl geeignet sind (*siehe Kapitel „2.5 Unterschied dateibasierte & Client-Server DBS“*). Eine Kombination beider Vorgehensweisen ist in der Praxis die Regel [245, S. 198].

Aus den zuvor genannten Gründen wird die Wahl auf DBSs mit Client-Server-Architektur beschränkt. Um zusätzliche Komplexität zu vermeiden, werden nur relationale DBSs berücksichtigt.

Unter Berücksichtigung der sich aus Banking 4.0 ergebenden Anforderungen (*siehe Kapitel „2.7 Die Zukunft: Banking 4.0 inklusive Historie“*) und aktuellen technologischen Entwicklungen in Form der zuvor erwähnten Analysen von *Gartner*, wird die Auswahl auf

Cloud-DBSs eingegrenzt. Neben MS gehört nach dem magischen Quadranten von *Gartner* aus dem Jahr 2022 auch *Amazon* und *Google* mit ihren Cloud-Lösungen zu den „Leadern“ in diesem Bereich, wobei *Amazon* hier an der Spitze steht [45]:

Bild 20: Magischer Quadrant von Gartner mit den besten Anbietern von Cloud-Diensten [45]

Da Access aus dem Hause von MS stammt, MS ein etabliertes Unternehmen ist und gleich mit mehreren Produkten unter den Top DBSs vertreten ist, wird sich gegen *Amazon* sowie *Google* und für MS entschieden. Zudem wird die beste Kompatibilität mit Access bei anderen MS-Produkten erwartet. Daher fällt die Wahl des Vergleichs-DBS auf Azure-Cloud-Produkte von MS in Form von Azure SQL (*siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software“*). *Gartner* hat in ihrem magischen Quadranten aus dem Jahr 2022 vor Sicherheitsproblemen bei der Azure-Cloud gewarnt. Als Synergieeffekt wird durch die Auswahl gleichzeitig auch geprüft, ob ein Jahr später bereits eine Besserung der Umstände in Sicht ist.

Da der Administrationsaufwand für den Anwender möglichst gering sein soll, wird aus der Produktfamilie Azure SQL die Azure SQL-DB als Vergleichs-DBS ausgewählt, welches ebenfalls unter den Top 20 zu finden ist und auf dem weit verbreiteten SQL Server basiert.

Die Azure SQL-DB besitzt eine vier-schichtige Architektur:

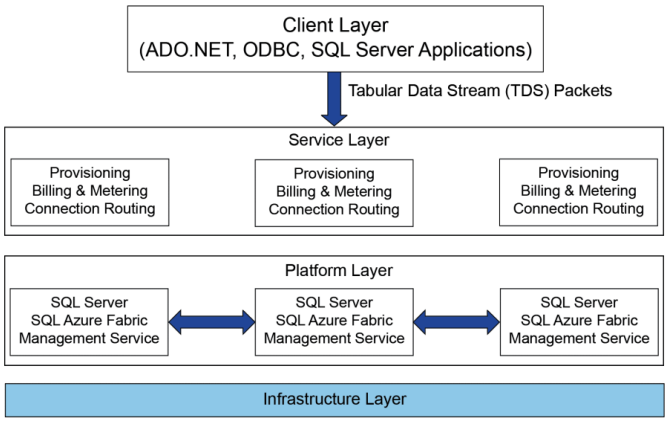


Bild 21: Vier-Schichten-Architektur von Azure SQL-DB. Vom Original-DBS werden zwei Kopien angelegt, die auf unterschiedlichen Servern laufen und somit die Verfügbarkeit der Dienste erhöhen [1, S. 4]

Die Client-Schicht ist die Schnittstelle zwischen den Applikationen und dem DBS. „Tabular Data Streams“ werden verwendet, damit .NET, ODBC, ADO.NET, Java oder Python-Anwendungen eine Verbindung zur Azure SQL-DB herstellen können und Daten wie Authentifizierungsmerkmale oder SQL-Abfragen austauschen können. Die Service-Schicht ist das Bindeglied („SQL Gateway Service“) zwischen dem Client und der Plattform-Schicht, da sie die Client-Anfragen an den Server in der Plattform-Schicht weiterleitet, auf dem das jeweilige DBS läuft. Sie ist verantwortlich für die Einrichtung neuer DBS-Instanzen und führt die Authentifizierung von Nutzenden durch. Auch Sicherheitsfeatures wie Firewall-Regeln oder Denial-of-Service-Schutzpläne sind hier verortet. Auf dieser Ebene findet auch die Berechnung der Gebühren statt. In der Plattform-Schicht sind die DBSs angesiedelt, die auf Servern laufen. Zur Gewährleistung einer hohen Verfügbarkeit, laufen zwei Kopien des DBS auf jeweils unterschiedlichen Servern. MS garantiert daher eine nahezu 100 % Verfügbarkeit (99,99 %-99,995 %). Die „Azure Service Fabric“ sorgt für eine gleichmäßige Lastverteilung auf den Servern, für eine automatische Ausfallsicherung und eine automatische Duplizierung der DBS-Instanzen. Der „Management Service“ kümmert sich um die Überwachung und Patches. Die Infrastruktur-Schicht beinhaltet die Administration der Hardware sowie des Betriebssystems, auf denen die physischen DBSs laufen [1, S. 5].

3.5. Auswahl Office-Vergleichsprogramm zur Schnittstellenanalyse (Access Excel)

Das Vergleichsprogramm für die Schnittstellenanalyse soll aufgrund der weiten Verbreitung von MS 365 ebenfalls aus dem MS Office-Umfeld stammen (*siehe Kapitel „1.1 Relevanz & wissenschaftlicher Mehrwert“*).

In der Office-Suite ist Excel zur Tabellenkalkulation das Mittel der Wahl. Darüber hinaus weist Excel einige Ähnlichkeiten mit Access auf. Mit beiden Anwendungen können Daten verwaltet sowie Abfragen und Datenanalysen inklusive Reporting-Funktionalitäten mit komplexen Berechnungen durchgeführt werden. Trotz der Gemeinsamkeiten basieren beide Anwendungen auf unterschiedlichen Implementierungsansätzen. MS ordnet daher beiden Tools konkrete Anwendungsfälle zu und empfiehlt den kombinierten Einsatz von

Excel als Frontend und Access als Backend zur Datenhaltung („Access Excel“). Als Vergleichsprogramm für die Schnittstellenanalyse fällt die Wahl daher auf Excel [139]/[184]/-[205]/[212]:

Tabelle 6: Vergleich und Anwendungsfälle von MS Access und MS Excel

Access wird von MS zur Datenverwaltung empfohlen
Access wird empfohlen, wenn Daten zu organisieren und zu durchsuchen sind. Dabei wird die Datenintegrität gemäß MS auch bei vielen Nutzern gewährleistet. Im Gegensatz zu Excel speichert Access kontinuierlich Änderungen, um Datenverlust bei unerwarteten Fehlern vorzubeugen, was bereits beim Öffnen zur Veränderung des letzten Änderungsdatums der jeweiligen Datei führt (Sicherungskopien nach einem Zeitplan werden als Ergänzung dennoch von MS empfohlen, für Transaktionen als potenzielle Quelle für Inkonsistenzen *siehe Kapitel „4.1.1 SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen“*). Außerdem bieten Access-Formulare eine komfortablere Oberfläche. Access unterstützt zwar keine Sicherheit auf Benutzerebene aber alle Benutzer-Sicherheitsmodelle des Client-Server-DBS, mit denen das Access-Frontend verbunden ist.
Excel wird von MS zur Datenanalyse empfohlen
Excel wird zur Durchführung komplexer Berechnungen, zur Analyse der Ergebnisse oder bei numerischen Daten empfohlen. Außerdem wird Excel zur grafischen Aufbereitung der Ergebnisse mittels Grafiken und Diagrammen ans Herz gelegt. Ist die „AutoWiederherstellen“-Funktionalität von Office aktiviert, speichert Excel zur Wiederherstellung im Fehlerfall in regelmäßigen Abständen AutoWiederherstellen-Informationen, anstelle wie bei Access automatisch die gesamte Datei. Im Gegensatz zu Access unterstützt Excel rudimentär den Datenschutz auf Benutzerebene durch Ausblenden von Zeilen und Spalten sowie anschließendem Aktivieren des Blattschutzes. Außerdem können benutzerbasierte Berechtigungen, wenn genutzt, über das Information Rights Management (IRM) für den Datenzugriff oder schreibgeschützte Rechte vergeben werden (*siehe Kapitel „6 Handlungsempfehlungen & Ausblick“*). So können Nutzende auch an Änderungen im Zugriff befindlicher Daten gehindert werden. Im Gegensatz zu Access kann für eine Excel-Datei ein Passwort zum Öffnen (Dateiverschlüsselung) und zum Manipulieren der Datei (Blattschutz) vergeben werden. Daneben kann auch die Arbeitsmappenstruktur durch ein Passwort geschützt werden. Dies verhindert, dass Nutzende ausgeblendete Arbeitsblätter (Worksheets) anzeigen, Worksheets hinzuzufügen, löschen, verschieben, ausblenden oder umbenennen können [206]. Das Kennzeichnen der Excel-Datei als „Final“ bietet keinen Schutz, da sie leicht deaktiviert werden kann und nicht durch ein Passwort geschützt ist. Digitale Signaturen und die Dateiverschlüsselung werden, wie bei Access unterstützt. Anders als Access bietet Excel mit Office-Skripten eine sicherere Alternative zu VBA [211].

3.6. Auswahl forensisches Tool zur forensischen Analyse (Autopsy)

Die Suche nach einer geeigneten digitalen forensischen Plattform für die forensische Analyse wird auf frei verfügbare Open Source-Anwendungen beschränkt, um einen Eindruck von der Zuverlässigkeit derartiger Programme zu erhalten. Damit scheiden Tools wie „X-Ways Forensics“ von *X-Ways Software Technology AG*, „Magnet AXIOM“ von *Magnet Forensics* oder „Access Forensics“ von *Thegrideon Software* aus der weiteren Betrachtung aus.

Auf einen Vergleich mit einem kostenpflichtigen Programm wird aus Zeitgründen und zur Fokussierung verzichtet, für einen Einblick sei auf die anonyme Projektarbeit „Forensik-Software – Test von Funktionalitäten auf Dateiidentifikations- & Dateirekonstruktionsebene in NTFS & APFS-Dateisystemen“ auf dem IT-Forensik Wiki verwiesen [68].

Die Wahl fällt auf „Autopsy“ von *Basistech*, da es im Gegensatz zur SIFT Workstation von *SysAdmin, Audit, Networking and Security* (SANS) neben Linux ohne Windows-Subsystem für Linux (WSL) auch unter einem Windows-Betriebssystem läuft.

Weitere Gründe für die Auswahl von Autopsy sind die kontinuierlichen Aktualisierungen und der breitaufgestellte Funktionsumfang im Vergleich zu anderen spezialisierten Open Source-Programmen wie das „Volatility Framework“ zur Arbeitsspeicher-Forensik oder „Xplico“ zur Extraktion von Anwendungsdaten aus aufgezeichnetem Internet-Verkehr. Autopsy stellt das offizielle GUI für „The Sleuth Kit®“ dar, was auch in die SIFT Workstation und andere forensische Plattformen integriert ist. Sleuth Kit ist eine Sammlung von Kommandozeilen-Tools zur Untersuchung von Datenträger-Images. Sowohl Autopsy als auch Sleuth Kit besitzen eine umfangreiche Online-Dokumentation und eine große Community. Die Funktionen reichen von Dateisystemanalysen wie der Schlüsselwortsuche, dem Suchen oder Wiederherstellen von (gelöschten) Dateien, der Auswertung der Windows-Registry oder von Web-Artefakten, der Erstellung von Timelines, in denen Aktivitäten auf einem Windows-Betriebssystem, wie das Öffnen von Archiven geordnet aufgelistet werden, bis hin zu Reporting-Funktionalitäten zur Darstellung der Analyseergebnisse. Autopsy bietet zusätzlich auch kostenpflichtige Schulungs- und erweiterte Supportmöglichkeiten an. Viele der über Sleuth Kit verfügbaren Funktionen werden durch Autopsy weiter automatisiert und die grafische Benutzeroberfläche verbessert die Nutzerfreundlichkeit. Aufgrund der Plugin-Architektur kann der von Autopsy standardmäßig angebotene Funktionsumfang durch selbstentwickelte Module zur Dateianalyse, für Reporting-Zwecke oder zur grafischen Auswertungen erweitert werden [2]/[59]/[244]/[261]/[263].

4. Durchführung der Sicherheitsanalyse

Nach der Vorbereitungsphase wird in diesem Kapitel die Sicherheitsanalyse durchgeführt. Die Sicherheitsanalyse besteht aus folgenden Teilaspekten, die jeweils in einzelnen Unterkapiteln behandelt werden: Kriterienanalyse, Schnittstellenanalyse, Dateiformatanalyse und forensische Analyse.

4.1. Kriterienanalyse (Access & Vergleichs-DBS)

In diesem Kapitel wird eine Kriterienanalyse gemäß den in *Kapitel „3.3.2 Festlegung von relevanten Bewertungskriterien“* ausgewählten Bewertungskriterien durchgeführt. Die Kriterienanalyse erfolgt neben Access auch für das ausgewählte Vergleichs-DBS (*siehe Kapitel „3.4 Auswahl Vergleichs-DBS zur Kriterienanalyse (Azure SQL-DB)“*).

Für weitere Details zur Azure SQL-DB im Zusammenhang mit SQL-Injection-Attacken sei an dieser Stelle auch auf die Projektarbeit zum Thema „SQL-Injection Angriffe ausführen und forensisch nachweisen“ (S. 62 ff.) von Stefan Augustin und Nils Majewski, dem Autor der vorliegenden Arbeit, im IT-Forensik Wiki verwiesen [69].

4.1.1. SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen

MS Access bietet mit „Access SQL“ einen eigenen SQL-Dialekt, mit eingeschränktem Befehlsumfang und Datentypen. Um nicht den Rahmen der vorliegenden Thesis zu sprengen, wird sich lediglich auf sicherheitsrelevante Aspekte konzentriert und nicht ausführlich auf die Unterschiede zur Azure SQL-DB beziehungsweise den dort eingesetzten, leicht abgewandelten Transact-SQL-Dialekt eingegangen [229].

Mehrfachabfragen (Multi-Query-Statements) werden von der ADE nicht unterstützt, was einen Schutz gegen SQL-Injection-Angriffe darstellt (*siehe Kapitel „2.4 Informationen zu Access, JET, ACE/ADE & .accdb-Dateiformat“*):

Bild 22: Links: Eingabe eines Multi-Query-Statements in Access; Rechts: Fehlermeldung beim Ausführen eines Multi-Query-Statements in Access

Anweisungen nach einem Semikolon führen zu einem Laufzeitfehler, allerdings können als Workaround `SELECT`-Abfragen mittels `UNION`-Statement verknüpft und so die ursprüngliche Abfrage beliebig erweitert werden:

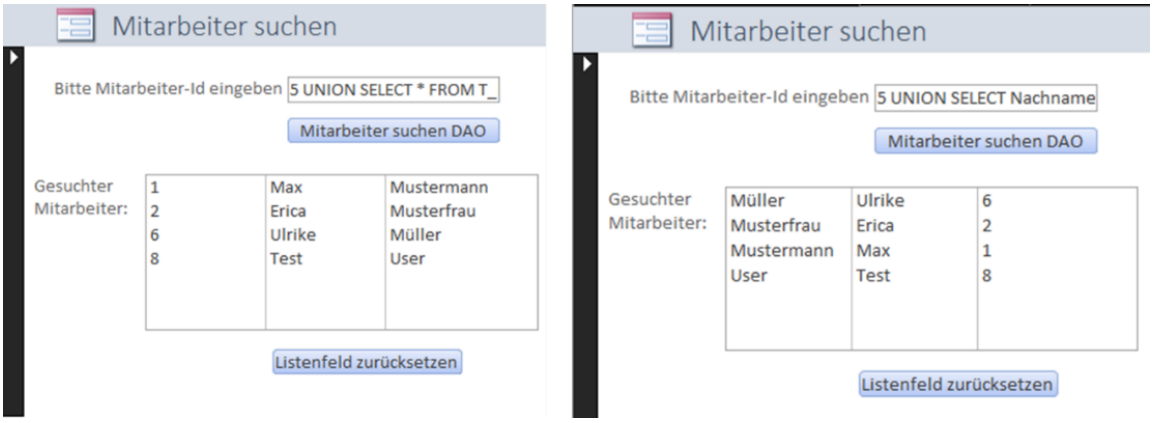


*Bild 23: Ohne die Nutzung von parametrisierten Abfragen können in Access SELECT-Statements über die Access-Oberfläche mittels UNION-Statement verknüpft werden, um die ursprüngliche Abfrage zu erweitern. Dabei scheint der dem Attribut zugrundeliegende Datentyp je nach verwendeter Zugriffs-Methode und Anzeige-Steuerelement irrelevant zu sein, wie in diesem Angriffsszenario mit „CurrentDb.OpenRecordset" zum Ausführen des SQL-Statements sowie dem Listenfeld-Steuerelement zur Anzeige des Ergebnisses demonstriert (Links wurde folgendes Statement ausgeführt: „5 UNION SELECT * FROM T_Mitarbeiter"; Rechts: „5 UNION SELECT Nachname, Vorname, ID FROM T_Mitarbeiter")*

Wichtig hierbei und Voraussetzung für derartige SQL-Injection-Angriffe ist, dass das anzugreifende Eingabe-Textfeld den Datentyp „*String*" besitzt [71, S. 87-89]. Werden SQL-Befehle über VBA oder über die Access-Oberfläche ausgeführt, können parametrisierte Abfragen als Schutz vor der Erweiterung des SQL-Statements durch Nutzende verwendet werden. Bei DAO kann die „*Parameters*"-Property des „*QueryDef*"-Objekt („*CreateQueryDef*"-Methode) verwendet werden, bei ADO die „*CreateParameter*"-Methode des „*Connection*"-Objekts und beim „*DoCmd*"-Objekt die „*SetParameter*"-Methode (*siehe Kapitel „2.2 Beispiele für Datenzugriffe in Access"*) [96]/[103]/[143]. Die Verwendung des *QueryDef*-Objekts ist das von MS empfohlene Vorgehen zum Ausführen von SQL-Pass-Through-Abfragen mit ODBC-DB [99]. Über Pass-Through-Abfragen können SQL-Befehle an ein Server-DBS durchgereicht werden [275, S. 559].

Ohne Nutzung von parametrisierten Abfragen kann über In-Band-SQL-Injection eine Parameterveränderung durchgeführt werden. Hier werden SQL-Befehle an die Eingabeparameter angefügt und somit die Ausgabe manipuliert beziehungsweise das ursprüngliche SQL-Statement erweitert. Als Beispiel hierfür dient die Erweiterung mit einem logischen Operator, wie „*OR 1 = 1*", der immer wahr ist.

Über VBA und das DAO können mittels Parameter „*Name*" der *CreateQueryDef*-Methode benannte und persistente *QueryDef*-Objekte erstellt werden, die im Navigationsbereich unter „Abfragen" angezeigt werden. Abfragen können in VBA über den vergebenen Bezeichner angesprochen werden. Wird für den Parameter „*Name*" ein Null-Argument wie „*vbNullString*" übergeben, wird ein temporäres *QueryDef*-Objekt angelegt, was nicht gespeichert wird und einem Prepared Statement ähnelt. Abfragen können auch benutzerfreundlich und ohne VBA über die Access-Oberfläche „*Erstellen → Abfrage-Assistent oder Abfrageentwurf*" erstellt werden. In der SQL-Ansicht kann die Abfrage direkt als SQL-Befehl erstellt werden, über die Entwurfsansicht ist die Erstellung auch ohne SQL-Erfahrungen möglich [99]:

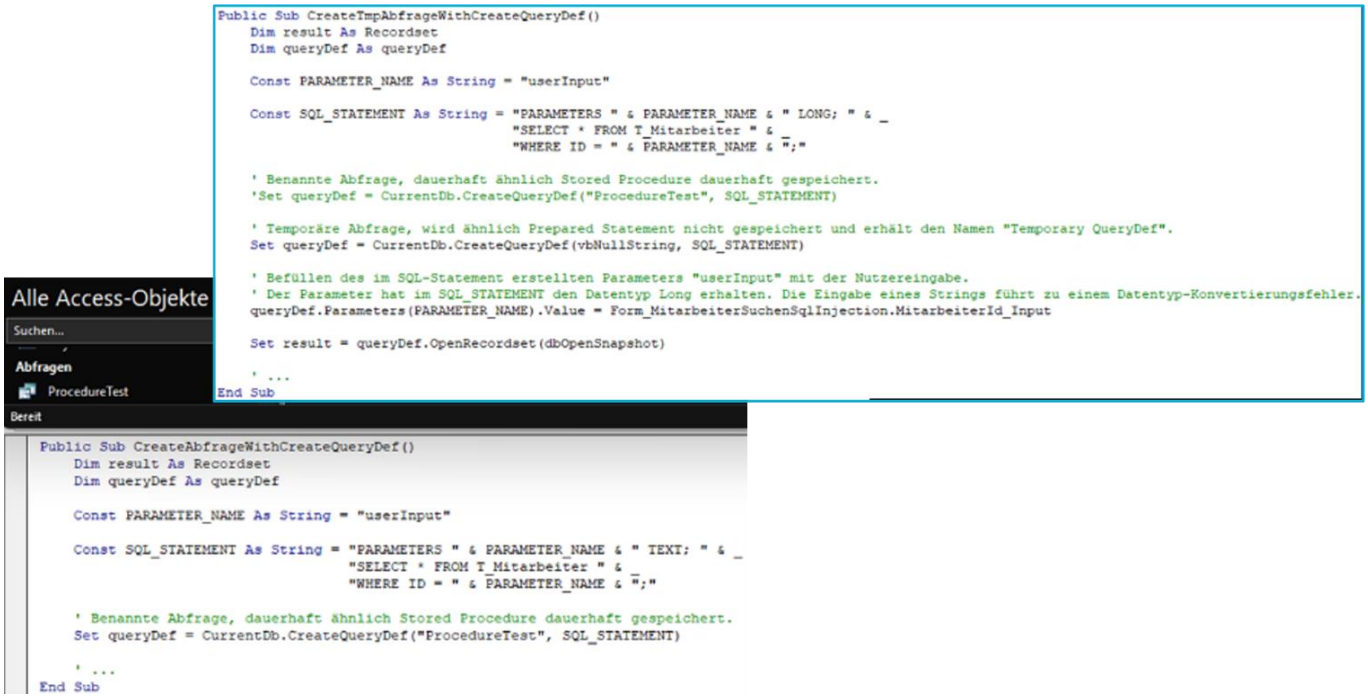


Bild 24: Links unten: Access-Navigationsbereich („Alle Access-Objekte“) mit einer persistenten Abfrage „ProcedureTest“ als Alternative für Stored Procedures. Unter dem Navigationsbereich ist der zugehörige VBA-Code zu finden, in dem die persistente, parametrisierte Abfrage „ProcedureTest“ mittels CreateQueryDef erstellt wird [143]; Rechts oben: Erstellen einer temporären parametrisierten Abfrage mittels CreateQueryDef als Alternative für Prepared Statements

Bild 25: Fehlermeldung beim Ausführen der zuvor erstellten Abfrage „ProcedureTest“ über die Access-Oberfläche, da ein Parameter vom Datentyp String und nicht Long übergeben wird

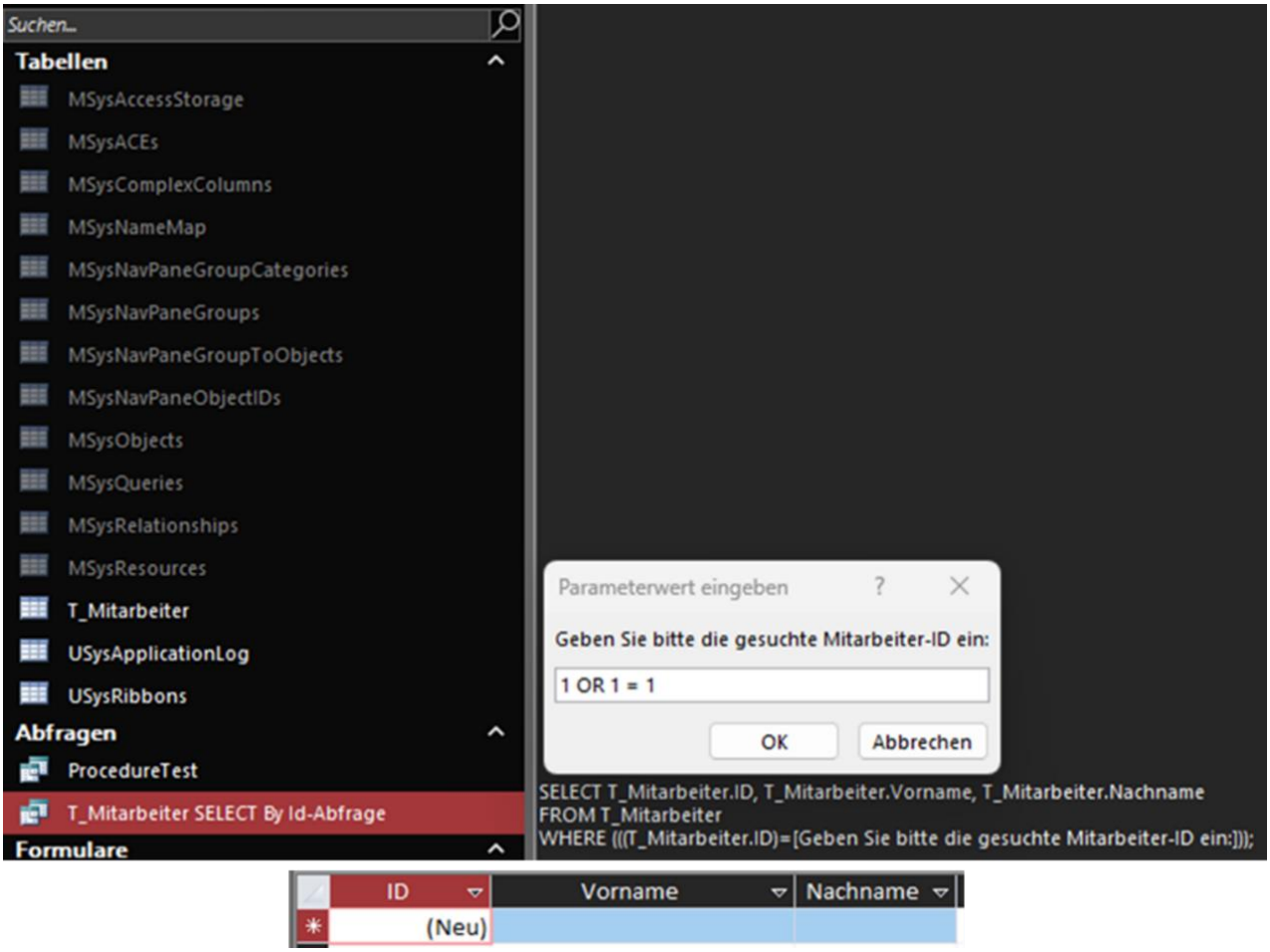


Bild 26: Oben: Weitere Abfrage, in der jedoch für den Parameter kein Datentyp angegeben wurde. Daher führt das Ausführen des Statements anders als im vorherigen Beispiel zu keinem Fehler; Unten: Der SQL-Injection-Angriff schlägt dank Nutzung von Abfragen dennoch fehl und liefert eine leere Ergebnismenge, da das für den Parameter übergebene Argument als zusammenhängende Zeichenkette ausgewertet wird. Es gibt keine ID „1 OR 1 = 1“

Wird über das in die Access-Datei integrierte GUI gearbeitet, können zusätzlich über die Entwurfsansicht Eingabetext-Steuerelemente mit Gültigkeitsregeln, einem Eingabeformat unter Verwendung einer vereinfachten Regular Expression-Syntax oder durch Aufrufen von Ereignissen geschützt werden. Ähnlich können auch die Attribute in einer Relation über Gültigkeitsregeln geschützt und somit das Hinzufügen von Datensätzen, die gegen die Gültigkeitsregel verstoßen, verhindert werden (*für eine ausführliche Anleitung siehe Kapitel „Anlage 3: Über das Entwurfsfenster konfigurierbare Eingabeprüfungen für Freitext-Steuerelemente zur Eingabe & Gültigkeitsregeln für Attribute in einer Relation (Access)“*). In DbVisualizer werden Abfragen unter dem Reiter „Views“ angezeigt [186]/[209]:

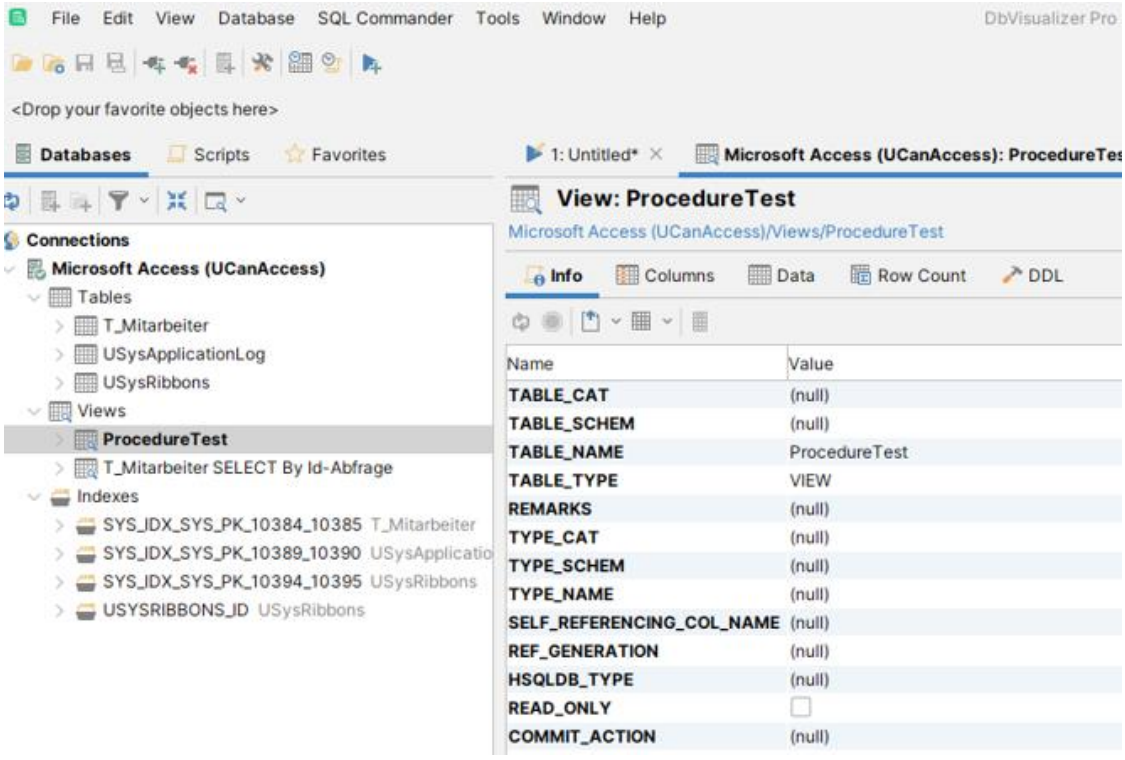


Bild 27: Anzeige der erstellten Abfrage unter „Views“ im DbVisualizer

Wird nicht über ein in die Access-Datei integriertes GUI in Form von Formularen oder über ein separates (Access-)Frontend gearbeitet, ist die Access-Datenquelle nicht verschlüsselt oder ist das Passwort zur Entschlüsselung der Access-Datei bekannt, sind alle zuvor genannten Schutzmechanismen hinfällig, da über den DbVisualizer oder die Access-Oberfläche (zur Deaktivierung eines benutzerdefinierten Menübands *siehe Kapitel „4.1.2 Konfiguration“*) aufgrund fehlender User-Level-Security sämtliche Relationen eingesehen und manipuliert werden können. In die Access-Datei integrierter VBA-Code oder integrierte Makros werden nur beim direkten Zugriff auf die Datei über die Access-Oberfläche und nicht beim Zugriff mittels Drittanbietersoftware ausgeführt.

Transkationen werden von der ADE über die ADO und DAO-Schnittstellen unterstützt. Dabei ist gemäß MS zu beachten, dass sich Transaktionen bei MS Access von anderen ODBC-Datenquellen wie dem SQL Server unterscheiden und einen eingeschränkten Funktionsumfang besitzen. Ist eine MS Access-DB mit einem Dateiserver verbunden und wird der Dateiserver während des Programmablaufs und vor dem Commit einer transaktionsbedingten Änderung angehalten, kann sich die Access-Datei danach in einem inkonsistenten Zustand befinden. MS empfiehlt für eine komplette und dauerhafte Transaktionsunterstützung die Verwendung einer Client-Server-Architektur [29]/[230]. MS Access protokolliert Transaktionen in einer Datei, die in einem Ordner gespeichert wird, auf den die „*TEMP*“-Systemumgebungsvariable des Windows-Betriebssystems referenziert. Ein Laufzeitfehler wird ausgelöst, wenn die Dateigröße den verfügbaren Speicherplatz überschreitet [171]. Die Auffindbarkeit der Datei konnte in einem Test nicht bestätigt werden.

Stored Procedures, Views und Prepared Statements werden von der ADE nicht direkt unterstützt, jedoch wird mit den zuvor vorgestellten Abfragen eine Alternative angeboten [178]/[182]:

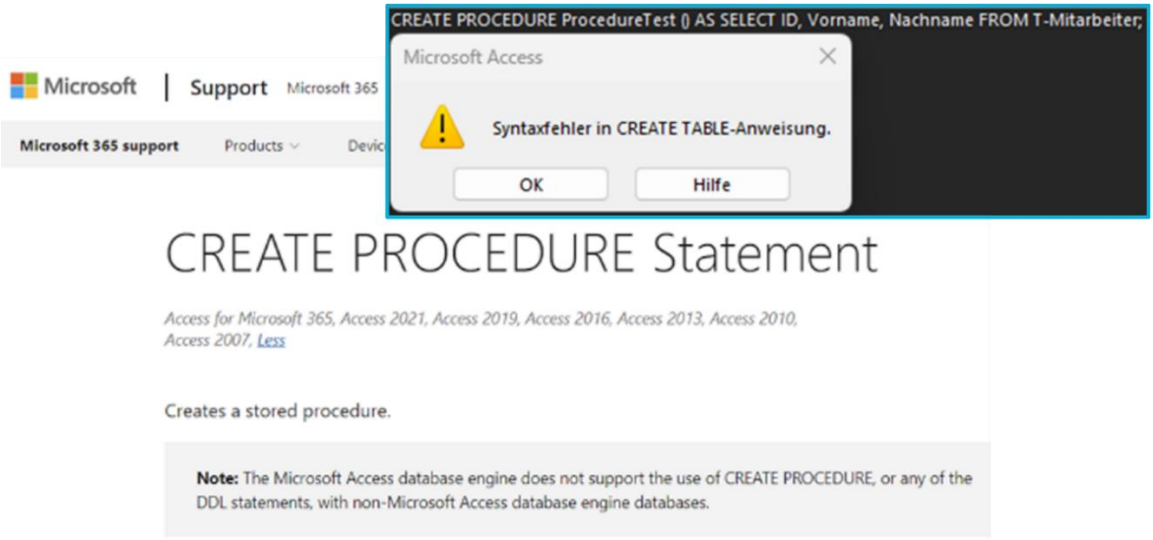


Bild 28: Links: MS-Support-Webseite, keine Unterstützung von CREATE PROCEDURE; Rechts: Fehlermeldung beim Ausführen eines Create Procedure-Statements in Access [178]

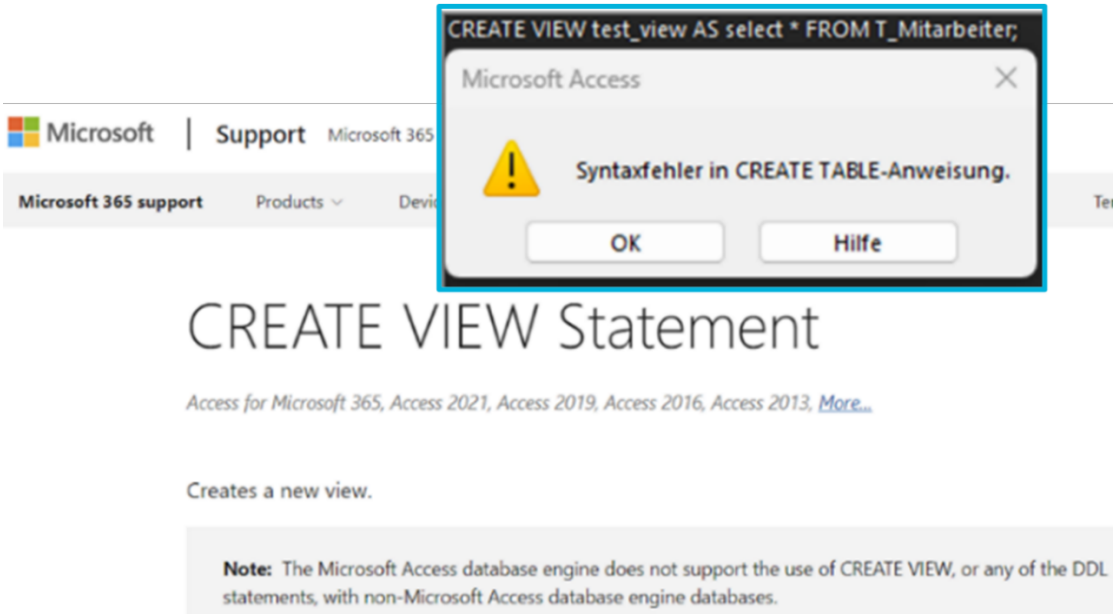


Bild 29: Links: MS-Support-Webseite, keine Unterstützung von CREATE VIEW; Rechts: Fehlermeldung beim Ausführen eines Create View-Statements in Access [182]

Da Azure SQL auf der MS SQL Server-DB-Engine basiert, wird hier auch der im Vergleich zu Access SQL viel mächtigere Transact-SQL Dialekt in leicht abgewandelter Form verwendet. Die Kernkomponenten wie Datentypen und Operatoren funktionieren bei der Azure SQL-DB identisch, es gibt allerdings Unterschiede in der DDL (darunter „*CREATE TABLE*“, „*ALTER TABLE*“, „*DROP TABLE*“) sowie der DML (darunter „*SELECT*“, „*INSERT*“, „*UPDATE*“, „*DELTE*“). Grund hierfür ist, dass Azure SQL so konzipiert ist, dass alle angebotenen Funktionalitäten keine Abhängigkeiten zum Betriebssystem und

zu System-DBs wie der Master-DB (enthält alle Informationen auf Systemebene wie Benutzer, verknüpfte Server, Systemkonfigurationen) besitzen. Daher sind auch keine Zugriffe auf das Betriebssystem oder Dateisystem möglich, auf dem die Azure SQL-DB läuft. Stored Procedures, parametrisierte Abfragen, Trigger und Transaktionen werden unterstützt [229].

Azure SQL sind reine DBSs, anders als in Access können keine Anwendungen inklusive einer GUI innerhalb des DBS implementiert werden (*siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software“*). Anstelle VBA stehen für die Anwendungsentwicklung diverse Programmiersprachen zur Verfügung. Voraussetzung ist, dass die von Azure SQL-DB angebotenen Schnittstellentechnologien wie ADO.NET, Java Database Connectivity (JDBC) oder ODBC unterstützt werden. Je nach verwendeter Programmiersprache stehen unterschiedliche Möglichkeiten zur Eingabeprüfung, Härtung oder auch Erkennung von SQL-Injection-Angriffen zur Verfügung. Eine in C# und mittels des Razor-Frameworks programmierte Demo-Web-Applikation bietet per Default einen automatischen Schutz vor Cross-Site-Scripting-Attacken, indem Razor alle in Variablen erhaltenen Zeichenketten kodiert (Encoding) und der Browser somit keine durch Angreifende eingeschleusten Skripte ausführt (aus „<“ wird „*&lt*“. Aus „>“ wird „*&gt*“ → aus *„<script>alert(1)</script>;“ wird „<script>alert(1)</script>;*“) [136][137]. Die Demo-Applikation wurde im Rahmen der in *Kapitel „4.1 Kriterienanalyse (Access & Vergleichs-DBS)“* erwähnten Projektarbeit initial erstellt.

Die innerhalb oder außerhalb der Azure-Cloud entwickelten Anwendungen können dann auch im Sinne des Platform-as-a-Service-Gedankens in der Azure-Cloud für Nutzende bereitgestellt und betrieben werden, was wiederum die Verwendung weiterer von MS Azure angebotener Sicherheitsfunktionalitäten ermöglicht. Mit *GitHub* können die eigenentwickelten Anwendungen auf Sicherheitsrisiken im Quellcode wie potenzielle Schwachstellen für SQL-Injection-Angriffe untersucht werden (beispielsweise bei jedem Push ins Git-Repository), was die Sicherheit und Robustheit des Codes erhöht.

Darüber hinaus bietet die Azure-Cloud auch eingebaute Überwachungssysteme an, welche potenziell sicherheitsrelevante Ereignisse wie SQL-Injection-Angriffe mittels KI erkennen und protokollieren. Bei einem erfolgreichen Angriff stellen die Überwachungssysteme eine Quelle für die Spurensicherung oder eine Datenwiederherstellung dar (*siehe Kapitel „4.1.5 Logging & Auswertungsmöglichkeiten“*) [51, S. 7, S. 39-40].

Bei *SELECT*-Mehrfachabfragen gibt Azure SQL-DB stets nur das Ergebnis des ersten *SELECT*-Statements zurück:

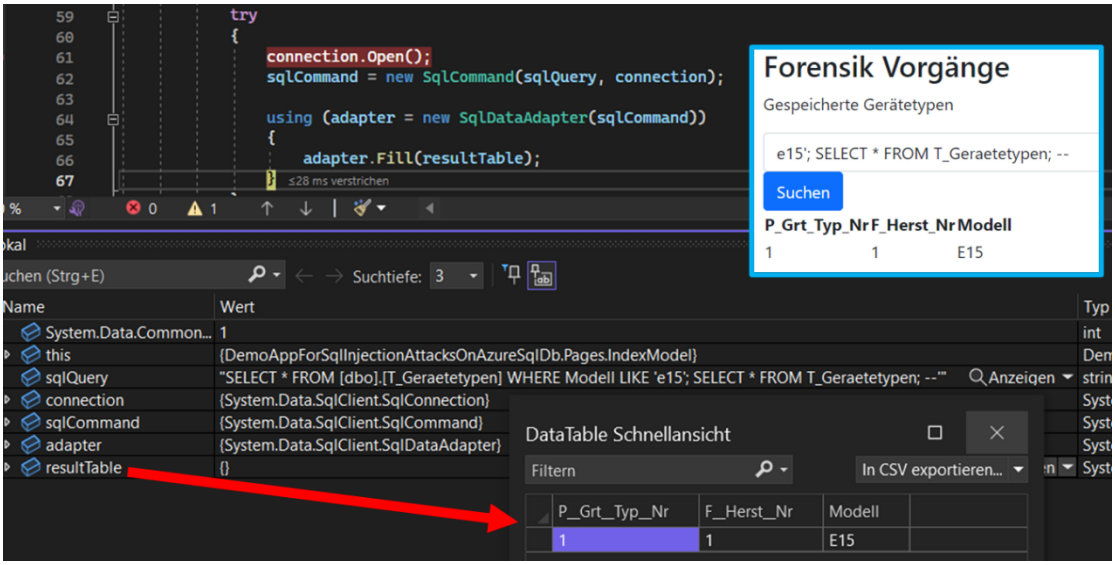


Bild 30: Zwar sind in Azure SQL-DB Multi-Query-Statements per Default aktiviert, allerdings wird beim Ergänzen eines zweiten SELECT-Statements in einem SQL-Injection-Angriff stets nur das Ergebnis des ersten Statements in der Demo-Web-Applikation (C# und Razor) angezeigt

Um mehrere `SELECT`-Abfragen miteinander zu verknüpfen, kann als Workaround, wie bei Access, das `UNION`-Statement verwendet werden. Hier ist anders als bei Access allerdings auf die korrekte Verwendung der jeweiligen Datentypen zu achten. Ansonsten sind in einer Azure SQL-DB im Gegensatz zu Access standardmäßig Mehrfachabfragen möglich, da in der SQL Server-DB-Engine vom Client übergebene Zeichenketten mit mehreren Anweisungen über denselben Verbindungshandle (Identifier für eine Verbindung, bestehend aus Treiber und Datenquelle) nacheinander ausgeführt werden. Abhängig von den Rechten des Nutzers (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung“*), über den das Statement ausgeführt wird, können Anweisungen wie das Anlegen neuer DB-Nutzer (inklusive Vergabe von Rechten), das Löschen und Aktualisieren von Tupel oder Relationen ausgeführt werden [71, S. 87-89]/[162]:

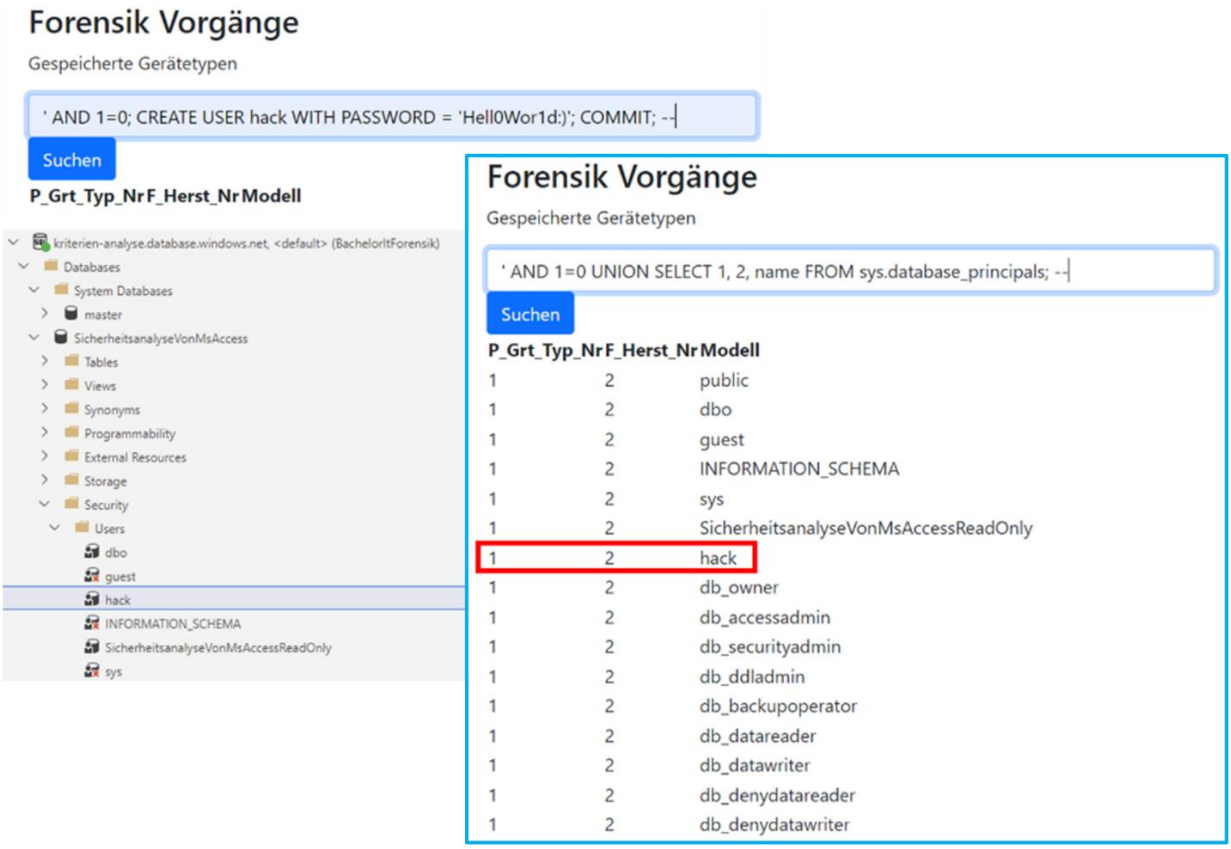


Bild 31: Links oben: Anders als in Access können in Azure SQL-DB per Default Multi-Query-Statements ausgeführt werden. In diesem Szenario wird ein neuer DB-Nutzer „hack“ angelegt; Links unten: Hier sind über das Azure Data Studio die in der DB angelegten Nutzer aufgelistet, darunter auch „hack“; Rechts: Hier wird die ursprüngliche SELECT-Abfrage der Demo-Web-Applikation zur Selektion der gespeicherten Gerätetypen über das Attribut „Modell“ mit einem über das GUI ergänztem und mittels UNION verknüpften SELECT-Statement zur Abfrage der sys.database_principals verknüpft. sys.database_principals enthält alle konfigurierten Security Principals wie Nutzer oder Rollen [152].

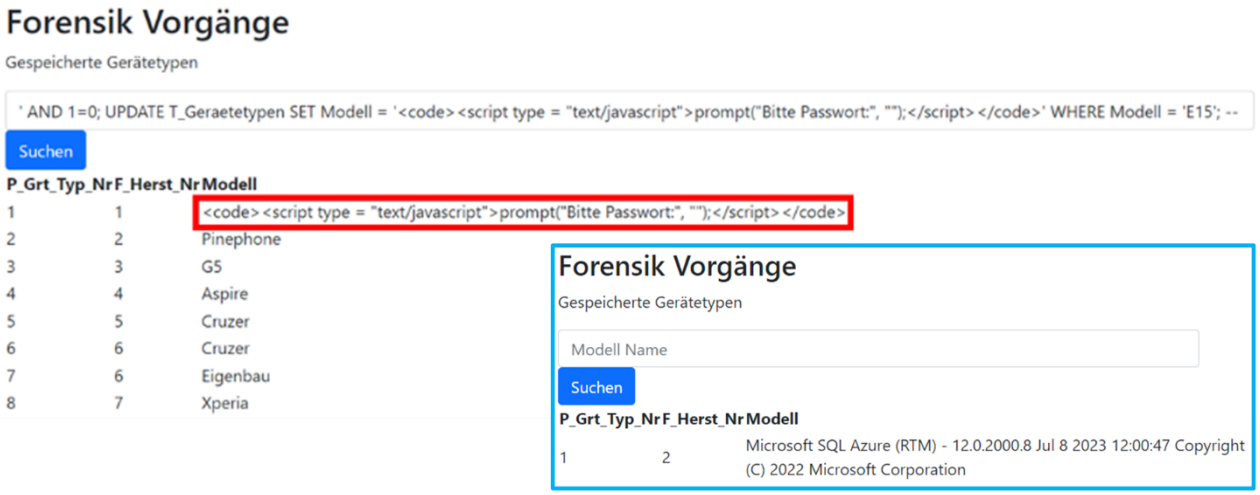


Bild 32: Links: In diesem Szenario können Angreifende erfolgreich eine Cross-Site Scripting-Attacke durchführen und ein JavaScript-Skript in die Azure SQL-DB einschleusen. Glücklicherweise besitzt das verwendete Razor-Framework der Demo-Wep-Applikation einen Schutzmechanismus gegen derartige Cross-Site Scripting-Angriffe und escaped Sonderzesichen wie „<“ oder „>“ (aus „<script>alert(1)</script>;“ wird „<script>alert(1)</script>“) [136][137]; Rechts: Hier

nutzten Angreifende die „@@Version"-Konfigurationsfunktion, um System- und Build-Informationen der Azure SQL-DB abzufragen und mittels UNION-Verknüpfung dem eigentlichen Abfrage-Ergebnis anzuhängen [76].

4.1.2. Konfiguration

Da Access im Dateiformat 2007-2016 keine User-Level-Security-Funktionalität bietet, können Konfigurationen nur global auf Dateiebene und nicht auf Nutzerebene vorgenommen werden. Nicht benötigte SQL-Befehle oder andere Dienste können nicht deaktiviert werden. Jedoch kann ein benutzerdefiniertes Menüband erstellt und Einstellungen wie das Deaktivieren des Navigationsbereichs, des Kontextmenüs und der Tastenkombinationen vorgenommen werden, um beispielsweise den Zugriff auf den „VBA-Editor" (Tastenkürzel `Alt + F11`) oder den Entwurfsmodus und damit Manipulationen zu verhindern. Leider ist es sowohl bei .accdb- sowie .accde-Dateien leicht möglich das benutzerdefinierte Menüband und auch in „*Datei → Optionen → Aktuelle Datenbank*" getätigte Konfigurationen über DbVisualizer zurückzusetzen, sodass diese Einstellungen nur für die Nutzerführung interessant sind und keinen wirklichen Schutz bieten (*für eine ausführliche Anleitung siehe Kapitel „Anlage 4: Benutzerdefinierte Menüband-Konfiguration & sonstige Einstellungen (Access)"*) [191]:

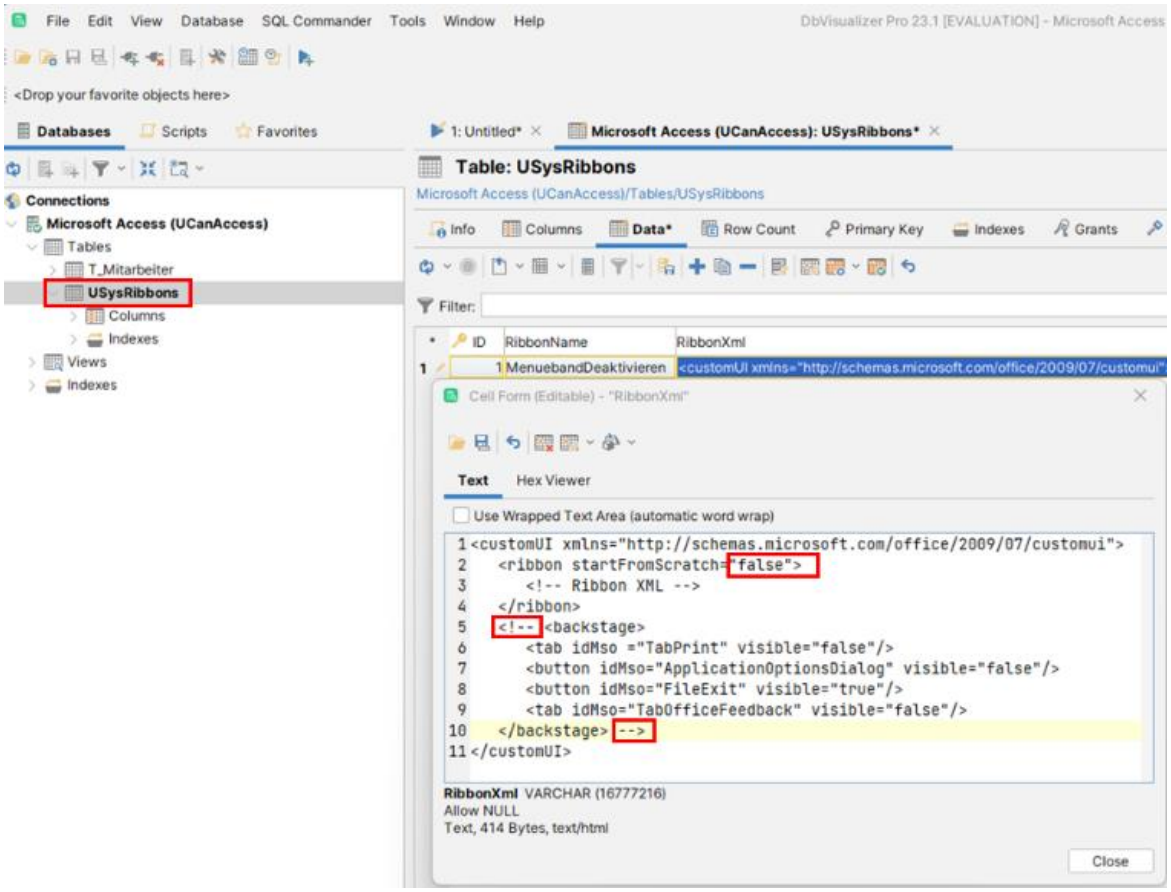


Bild 33: Deaktivieren des in der Relation „USysRibbons" konfigurierten benutzerdefinierten Menübands über DbVisualizer

In der Access-Datei sind die Optionen nach einem Neustart wieder unter „*Datei → Datenschutzoptionen*“ aufrufbar und getätigte Konfigurationen wie das Ausblenden des Navigationsbereichs und das Deaktivieren der Tastenkürzel der Tastatur können rückgängig gemacht werden:

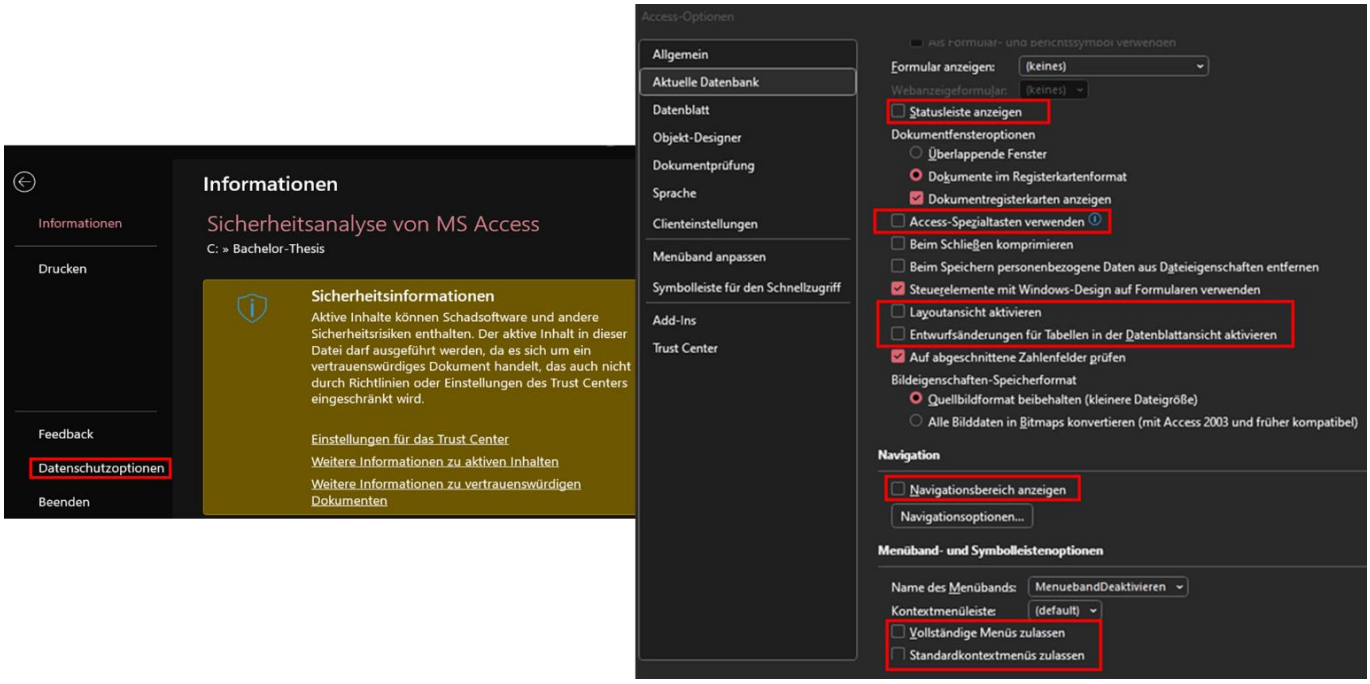


Bild 34: Links: Access-Optionen sind nach Deaktivieren des benutzerdefinierten Menübands über DbVisualizer in Access unter „Datei → Datenschutzoptionen“ aufrufbar; Rechts: Getätigte Einstellungen in den Access-Optionen zum Ausblenden des Navigationsbereichs und Deaktivieren der Tastenkürzel der Tastatur können rückgängig gemacht werden

Nach dem Neustart der Access-Datei ist der Vollzugriff wiederhergestellt:

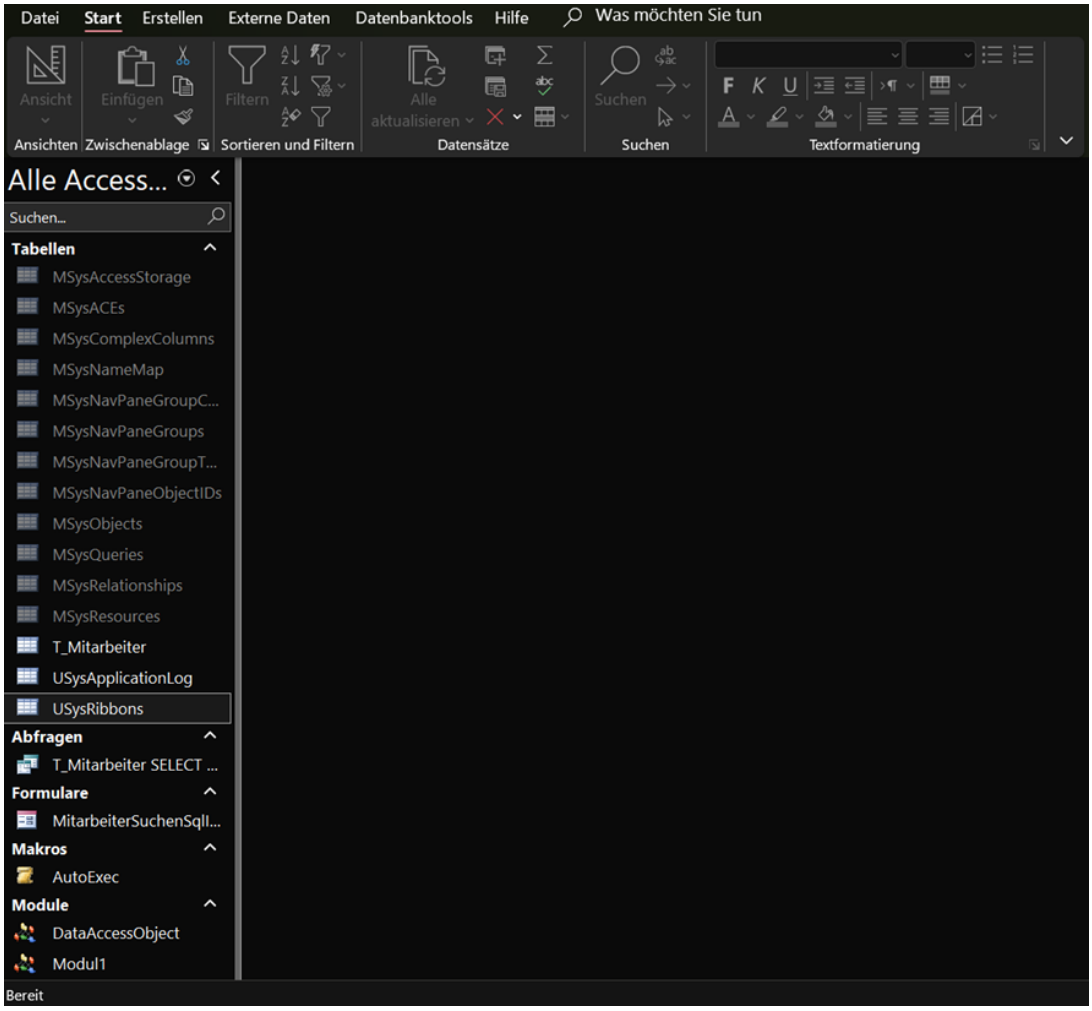


Bild 35: Benutzerdefinierte Menüband-Konfiguration – Menüband nach Wiederherstellung der Standard-konfiguration

Durch Kompilieren einer Access-Datei in das .accde-Dateiformat (Binärdatei) werden Entwurfs-, Formular- und Codeänderungen verhindert, die Dateigröße reduziert und die Leistung verbessert. Außerdem ist es nicht möglich den VBA-Quellcode einzusehen oder zu exportieren. Dabei ist zu beachten, dass die Entwurfsansicht bei Abfragen nicht deaktiviert ist und daher bestehende Abfragen manipuliert und neue Abfragen hinzugefügt werden können, was ein Einfallstor für böswillige Manipulationen ist. Es können auch neue Datenquellen hinzugefügt sowie bestehende Datenquellen über den „Tabellenverknüpfungs-Manager“ eingesehen und verändert werden. Zusätzlich können in einer .accde-Datei die weiter unten im Detail vorgestellten Datenmakroereignisse wie zum automatischen Ausführen von Quellcode oder AutoExec-Makros zum Starten einer windowsbasierten Anwendung (.exe) beim Öffnen der Datei hinzugefügt werden, sofern die Option im Office-Menüband nicht ausgeblendet ist. Anzumerken ist, dass auch bei .accde-Dateien das Menüband mittels des zuvor demonstrierten Angriffs wieder eingeblendet werden kann. Außerdem liegen Zeichenketten ohne Dateiverschlüsselung im Klartext vor (*siehe Kapitel „4.3 Dateiformatanalyse mittels Hexadezimal-Editor (Access)“*). .accde-Dateien können unter „*Datei → Speichern unter*“ nicht wieder in das .accdb-Format konvertiert werden. Eine weitere Möglichkeit ist die Bereitstellung der

Access-Datei gemeinsam mit der Access Runtime beziehungsweise das Starten der Access-Datei im Laufzeitmodus. Im Laufzeitmodus sind spezielle Tastenkombinationen zur Umgehung von Startoptionen, der Navigationsbereich, das Menüband oder die Entwurfs- sowie Layoutansicht nicht verfügbar. Der Laufzeitmodus ist jedoch kein primäres Mittel zum Schutz der Access-Datenbankanwendung, da eine Laufzeitdatenbankanwendung mit installierter Vollversion als normale und somit manipulierbare DB-Anwendung geöffnet werden kann. Der Schutz greift also nur, wenn keine Vollversion installiert ist [177]/[213].

Darüber hinaus können Windows-Administratoren Dienste und sicherheitsrelevante Funktionen datei- sowie instanzübergreifend zentral über das Trust Center, Registry-Konfigurationen oder Gruppenrichtlinien verwalten. Im Trust Center können beispielsweise unter „Add-Ins“ nur von vertrauenswürdigen Herausgebern signierte Add-Ins erlaubt werden oder unter „Makroeinstellungen“ alle Makros mit einer Benachrichtigung deaktiviert werden. Granulare Konfigurationen, um einzelne Makros oder Funktionen auszuschließen gibt es nicht. Die Access-Datei wird in einer geschützten Ansicht geöffnet, wenn in der Datei VBA-Code oder Makroaktionen enthalten sind, die von Nutzenden nicht ohne Erteilen eines expliziten vertrauenswürdigen Status genutzt werden können [187]. Liegt die Access-Datei nicht unter einem vertrauenswürdigen Pfad, müssen die Makros vor der Nutzung explizit durch die Nutzenden aktiviert werden, was einen menschlichen Schutzmechanismus gegen die automatische Ausführung von potenziell gefährlichem Quellcode und Makros darstellt:

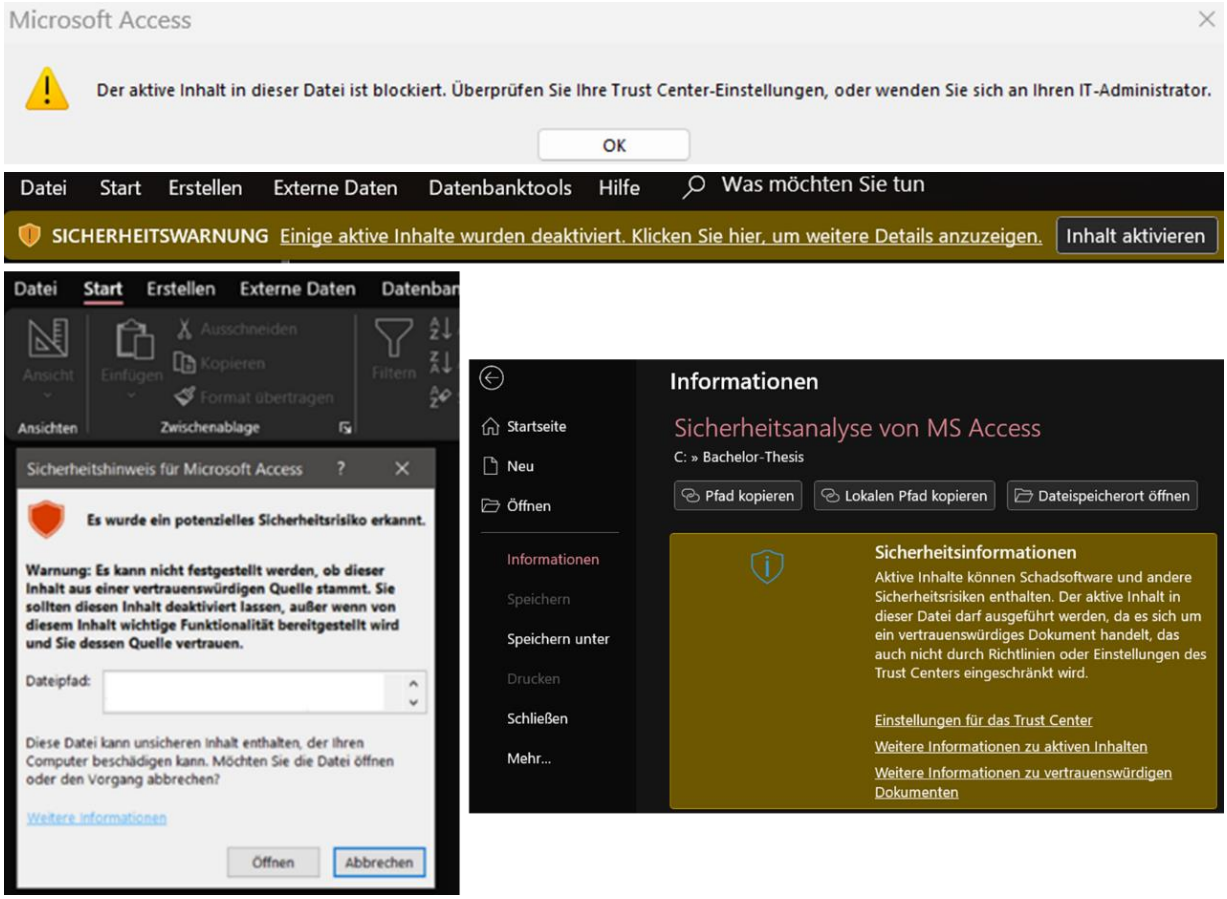


Bild 36: Geschützte Ansicht – Sicherheitshinweis aufgrund potenziellem Sicherheitsrisiko, wenn die Access-Datei nicht unter einem vertrauenswürdigen Pfad liegt und in der Datei VBA-Code oder Makroaktionen enthalten sind, die von Nutzenden nicht ohne Erteilen eines expliziten vertrauenswürdigen Status genutzt werden können [187]

Außerdem können vertrauenswürdige Herausgeber (*siehe Kapitel „4.1.3 Kryptographie“*), Dokumente und Pfade hinterlegt werden. Dateien unter vertrauenswürdigen Pfaden werden nicht in einer geschützten Ansicht und ohne Sicherheitshinweise zu VBA-Code, Makros oder ActiveX-Steuerelementen geöffnet. Vertrauenswürdige Pfade werden vom BSI nicht empfohlen und stellen somit keine Sicherheit, sondern vielmehr einen potenziellen Angriffspunkt dar. Die unter dem Kopfreiter „*Datei → Optionen → Clienteinstellungen*" vorgenommenen Einstellungen werden für alle mit der jeweiligen Access-Installation geöffneten Dateien übernommen.

Das BSI bietet Office- und konkrete Access-Konfigurationsempfehlungen an, die im IT-Grundschutz-Baustein „APP.1.1 Office-Produkte“ sowie in den Empfehlungen „Sichere Konfiguration von Microsoft Access 2013/2016/2019 für den Einsatz auf dem Betriebssystem Microsoft Windows“ zu finden sind (*siehe Kapitel „3.2 Identifikation von gängigen Sicherheitsstandards für relationale DBSs (Kriterienanalyse)“*) [5]/[8].

In den Mai-Updates 2021 ist Access eine über die Registry konfigurierbare Funktion hinzugefügt worden, mit der Referenzen auf externe Datenquellen in Abfragen deaktiviert werden können, was sich positiv auf die Sicherheit auswirkt. Seit den Oktober 2021-Updates können Windows-Events geloggt werden, wenn eine Anwendung externe DBSs in einer SQL-Anweisung anspricht [195]. Ein weiteres Beispiel für die Konfiguration über die Registry ist die maximale Anzahl von Sperren, die alle aktuellen Nutzenden (*für MS-Empfehlung zu Nutzerzahl siehe Kapitel „3.4 Auswahl Vergleichs-DBS zur Kriterienanalyse (Azure SQL-DB)“*) und die zugehörigen Transaktionen nicht überschreiten dürfen [75]/[84]. Access unterstützt keine Funktionalitäten für verbindlich zu nutzende dateiübergreifende Namenskonventionen oder sonstige Standardisierungen. Hier muss auf Arbeitsanweisungen zurückgegriffen werden, die von allen Mitarbeitenden verbindlich zu beachten sind. Über die Access-Oberfläche „*Datei → Optionen → Objekt-Designer → AutoIndex beim Importieren/Erstellen*“ können Attributnamen angegeben werden, für die automatisch ein Index angelegt wird.

In Access existieren keine Trigger, allerdings die bereits erwähnten und ähnlich funktionierenden Datenmakroereignisse, die auch in .accde-Dateien nachträglich hinzugefügt werden können [100]. Darüber hinaus können AutoExec-Makros mit variabler Logik erstellt werden, die beim Öffnen einer Datei automatisch ausgeführt werden. Ein Beispiel hierfür ist die „*AusführenAnwendung*“-Makroaktion, mit der windowsbasierte Anwendungen wie .exe-Dateien ausgeführt oder Konsolenfenster zum Ausführen von Befehlen geöffnet werden können. Dazu wird unter „*Erstellen → Makro“ ein neues Makro erstellt und anschließend unter „Makroentwurf*“ die Option „Alle Aktionen anzeigen“ aktiviert. Dadurch werden alle Aktionen im Aktionskatalog angezeigt und nicht nur solche, die nur in nicht vertrauenswürdigen Access-DBs ausgeführt werden können und daher nicht in einer geschützten Ansicht geöffnet werden. Unsichere Aktionen sind im Aktionskatalog mit einem Warnsymbol markiert [175]/[275, S. 823-824]:

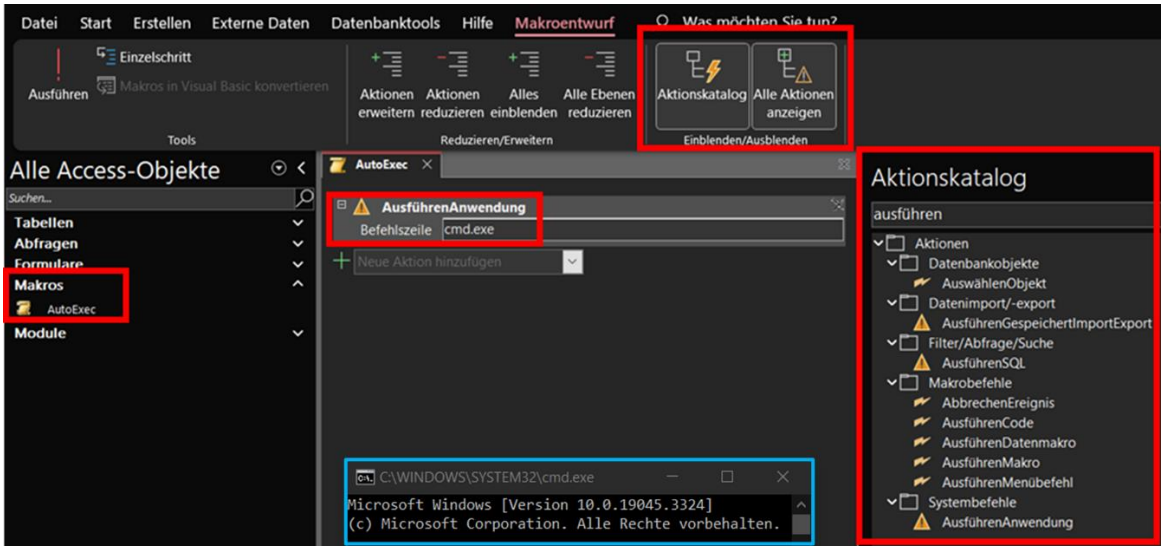


Bild 37: „AusführenAnwendung"-Makroaktion in Access zum automatischen Ausführen einer cmd.exe beim Öffnen der Datei mittels AutoExec-Makro 1/2

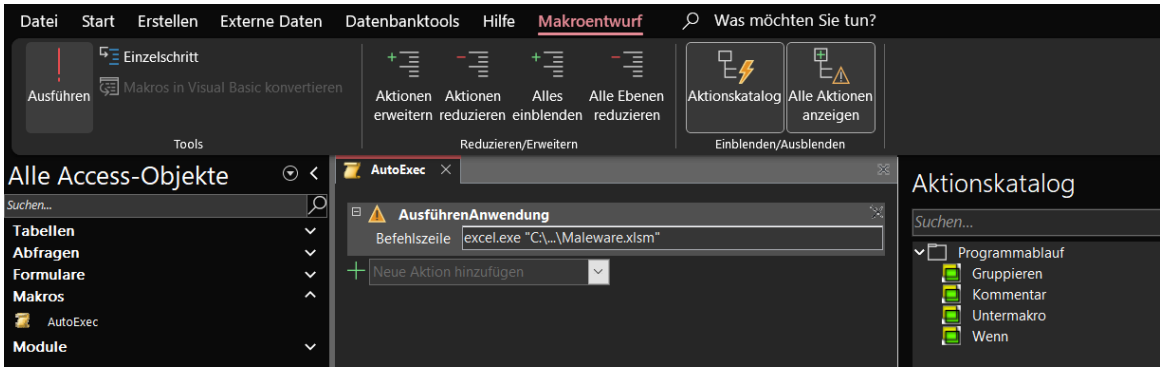


Bild 38: „AusführenAnwendung"-Makroaktion in Access zum automatischen Ausführen einer schadhaften excel.xlsm beim Öffnen der Datei mittels AutoExec-Makro 2/2

AutoExec-Makros können durch Drücken der „*SHIFT*"-Taste beim Starten der Access-Datei unterdrückt werden, was einerseits ein Sicherheitsfeature darstellt, andererseits aber auch selbsterstellte Funktionalitäten wie Logging-Routinen aushebelt. Die Unterdrückung mittels *SHIFT*-Taste funktioniert auch, wenn die Einstellung „Access-Spezialtasten verwenden" unter „*Datei → Optionen → Aktuelle Datenbank" deaktiviert ist. Eine Alternative sind Startup-Formulare, die über „Datei → Optionen → Aktuelle Datenbank → Formular anzeigen*" konfiguriert werden können. Hier kann das „*OnOpen*"-Event des Formulars zum automatischen Ausführen von Logik beim Start der Access-Datei zweckentfremdet werden, um in einer manipulierten Datei Schadcode auf den Endgeräten von Nutzenden auszuführen.

Die Programmausführung kann durch Drücken der „*STRG + Pause*"-Tasten vom Nutzenden angehalten werden. Wenn die Einstellung „Access-Spezialtasten verwenden" unter „*Datei → Optionen → Aktuelle Datenbank*" deaktiviert ist, kann das Anhalten der Programmausführung verhindert werden. Alternativ können Angreifende während der Ausführung von VBA-Routinen die Standard-Tastenbelegungen der Tastatur wie die *Pause*-Taste, temporär mit einer anderen Funktionalität überschreiben. Damit wird

sichergestellt, dass der Programmfluss nicht beeinträchtigt wird. Formulare können zwar in .accde-Dateien nachträglich ergänzt werden, da aber kein Zugriff auf die Entwurfsansicht sowie den VBA-Code besteht, kann hier kein Schadcode eingefügt werden [175]/[181].

Im Vergleich dazu stehen in der Azure-Cloud eine Vielzahl an Tools sowie Konfigurationsmöglichkeiten zur Verfügung, die teilweise in Verbindung mit der Azure SQL-DB genutzt werden können. Im Sinne des Platform-as-a-Service wird der beim Kunden anfallende Verwaltungsaufwand weiter reduziert. Kunden sind nur noch für das Datenmanagement und die Überwachung verantwortlich. Punkte wie die Wartung der Hardware oder Software-Updates werden von MS übernommen (*siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software“*). Nachfolgende Grafiken geben einen Überblick über einige in dieser Arbeit erwähnten Dienste (siehe rote Markierungen):

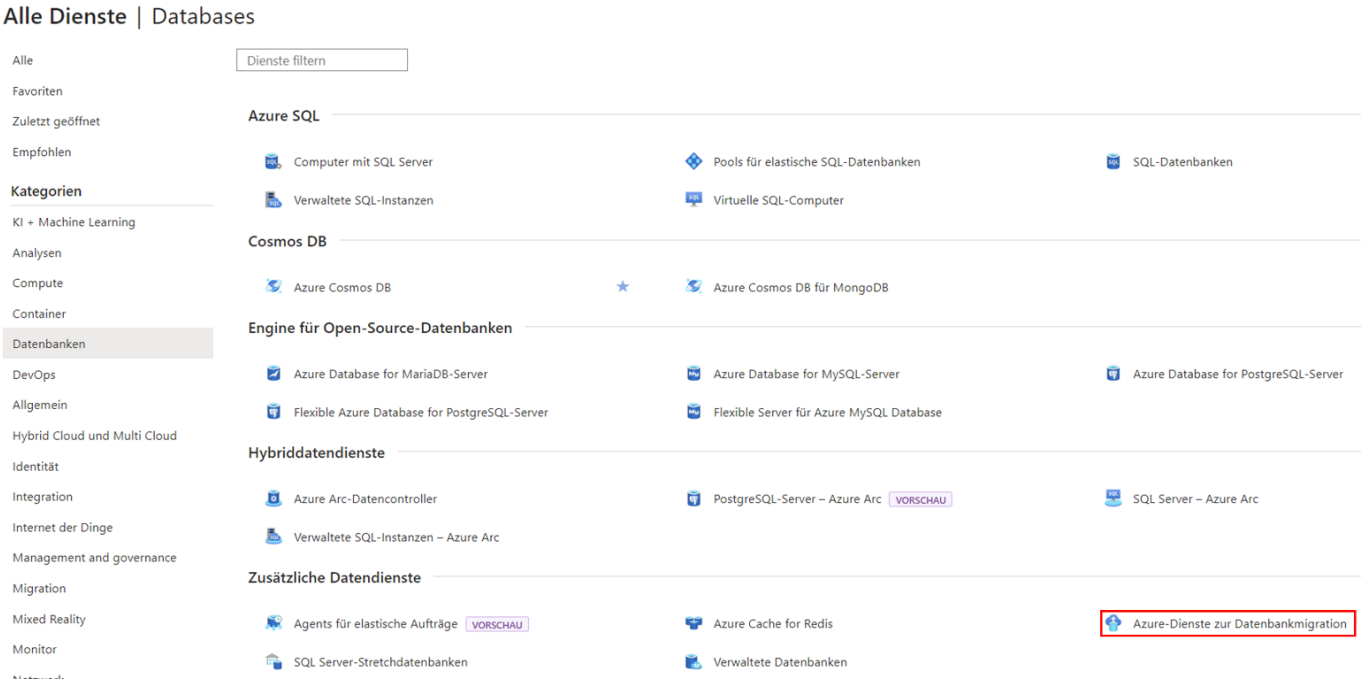


Bild 39: In der Azure-Cloud angebotene und teilweise gemeinsam mit Azure SQL-DB nutzbare Dienste in der Kategorie „Datenbanken“ (Auszug) 1/2

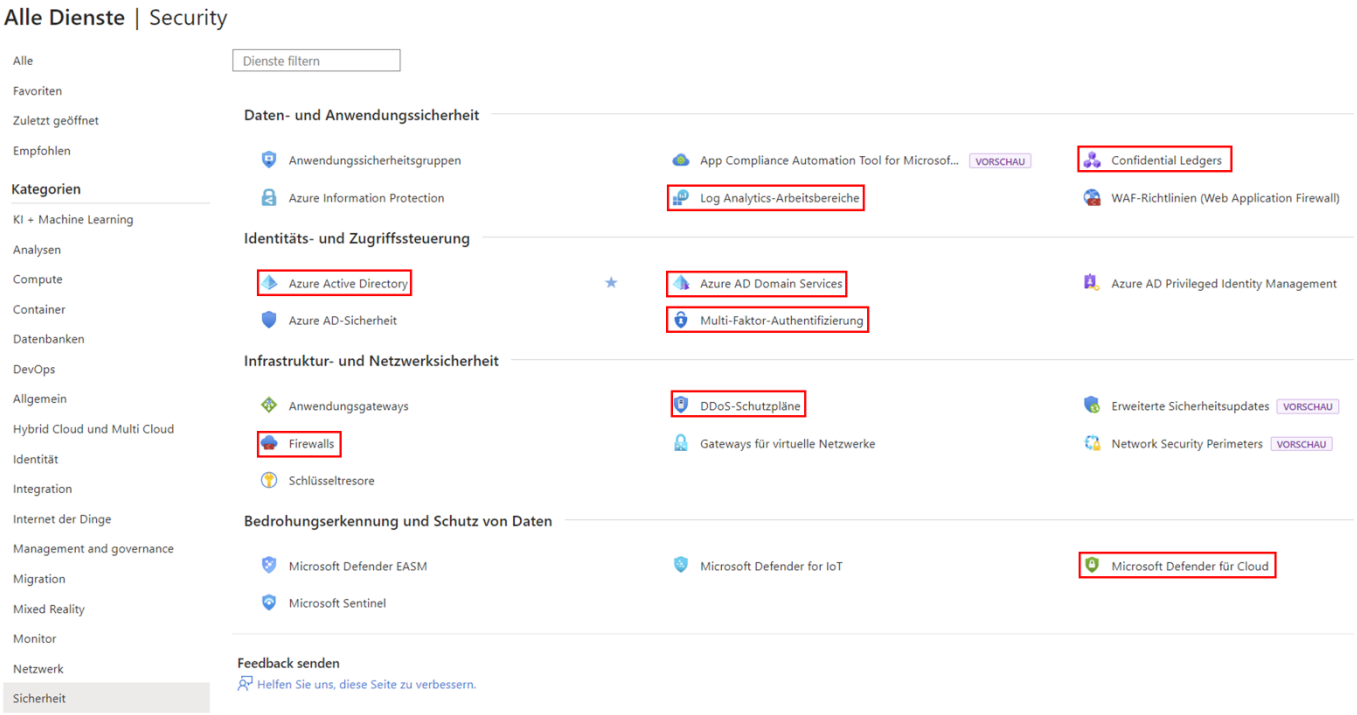


Bild 40: In der Azure-Cloud angebotene und teilweise gemeinsam mit Azure SQL-DB nutzbare Dienste in der Kategorie „Sicherheit“ (Auszug) 2/2

Die für die Azure SQL-DB angebotenen Dienste decken verschiedene Kategorien ab. Als Beispiele sind hier „Database Migration Services“ zur DB-Migration von einem Quell-DBS, wie dem SQL Server, PostgreSQL oder MySQL, in eine der drei Azure SQL-Bereitstellungsmethoden (Kategorie „Datenbanken“, *siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software“*) zu nennen.

Für die Datenmigration von einem SQL Server in die Azure-Cloud stehen jedoch auch externe Tools wie der „Data Migration Assistant“ oder die „SQLPackage.exe“ zur Verfügung. „Visual Studio“ kann mit dem „SQL Server Data Tools“ erweitert werden und auch das „SQL Server Management Studio“ unterstützt bei der Migration [1, S. 92].

Eine weitere Konfigurationsmöglichkeit zum Senken des Organisationsrisikos stellt die Multi-Faktor-Authentifizierung dar, mit der die Authentifizierung neben der Kontoanmeldung um einen zusätzlichen Faktor (Anruf, SMS, mobile App) erweitert wird. Darüber hinaus kann Azure AD als zentraler Authentifizierungs-Mechanismus für Benutzende (*siehe Kapitel „4.1.4 Passwörter & Authentifizierung“*) inklusive granularer Berechtigungsvergabe (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung“*) für den DB-Zugriff aber auch für den Zugriff auf das Azure-Portal sowie alle weiteren Azure-Dienste verwendet werden. Ein AD-Nutzer ist über mehrere DBs und Dienste gültig.

Außerdem können Distributed-Denial-of-Service-Schutzpläne konfiguriert werden, um den Datenverkehr bereits am Rande des Azure-Netzwerks auf der Service-Schicht zu bereinigen, bevor die Verfügbarkeit des Dienstes eingeschränkt wird.

Über den „MS Defender für Cloud“ (inklusive MS Defender für SQL) können als zentraler Anlaufpunkt eine Reihe von Sicherheitsmaßnahmen zum Schutz vor Cyber-

Bedrohungen und Sicherheitsrisiken wie Serverschutzmaßnahmen oder automatische Bedrohungserkennungsfunktionen zum Erkennen ungewöhnlicher und potenziell schädlicher Aktivitäten genutzt werden. Mittels Defender für SQL können potenzielle DB-Schwachstellen und ungewöhnliche, potenziell bedrohliche Aktivitäten ermittelt werden (Kategorie „Sicherheit“, *siehe Kapitel „4.1.5 Logging & Auswertungsmöglichkeiten“*).

Daneben werden auch eine Vielzahl an Diensten wie Gesichtserkennung, Spracherkennung oder Übersetzer (Kategorie „KI + Machine Learning“) angeboten, die nicht direkt im Zusammenhang mit der Azure SQL-DB stehen und daher nicht weiter betrachtet werden.

Getätigte Konfigurationen können in der Azure-Cloud zentral von Administratoren über eine Schnittstelle wie das Azure-Portal vorgenommen werden. An dieser Stelle seien auch die Firewall-Regeln als erste Sicherheitsebene im Rahmen der Authentifizierung erwähnt, um den Zugriff auf DBs oder DB-Server an zentraler Stelle nur für bekannte, öffentliche IP-Adressen zu erlauben oder den öffentlichen Netzwerkzugriff generell zu verbieten (*siehe Kapitel „4.1.4 Passwörter & Authentifizierung“*).

Neben der Möglichkeit nichtbenötigte SQL-Befehle granular innerhalb der jeweiligen DB über den `GRANT`-Befehl zu deaktivieren, können Trigger mittels `DISABLE TRIGGER`-Befehl deaktiviert werden (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung“*).

In der Azure SQL-DB können wie in Access keine einzuhaltenden Namenskonventionen konfiguriert werden. Dafür können mit „Azure Purview“ vorhandene Datenquellen in einem zentralen Datenkatalog gesammelt und klassifiziert werden (*siehe Kapitel „4.1.9 Banking 4.0“*). MS bietet zudem Best Practices für verschiedene Azure-Ressourcen sowie technische Unterstützung mit dem „Azure Naming Tool“. Durch die Zentralisierung der Datenhaltung in der Azure-Cloud wird auch die Konfiguration einschließlich der Umsetzung von Namenskonventionen an zentraler Stelle überwachbar und steuerbar.

Darüber hinaus können diverse Befehle wie das Anlegen von neuen DBs über PowerShell-Skripte automatisiert und standardisiert werden [1, S. 9, S. 46-49]/[108]. Im Gegensatz zu Access bietet die Azure SQL-DB über den MS Defender für Cloud (Sicherheits)Empfehlungen an, um die DB und den Server an zentraler Stelle mittels Best Practices weiter zu härten:

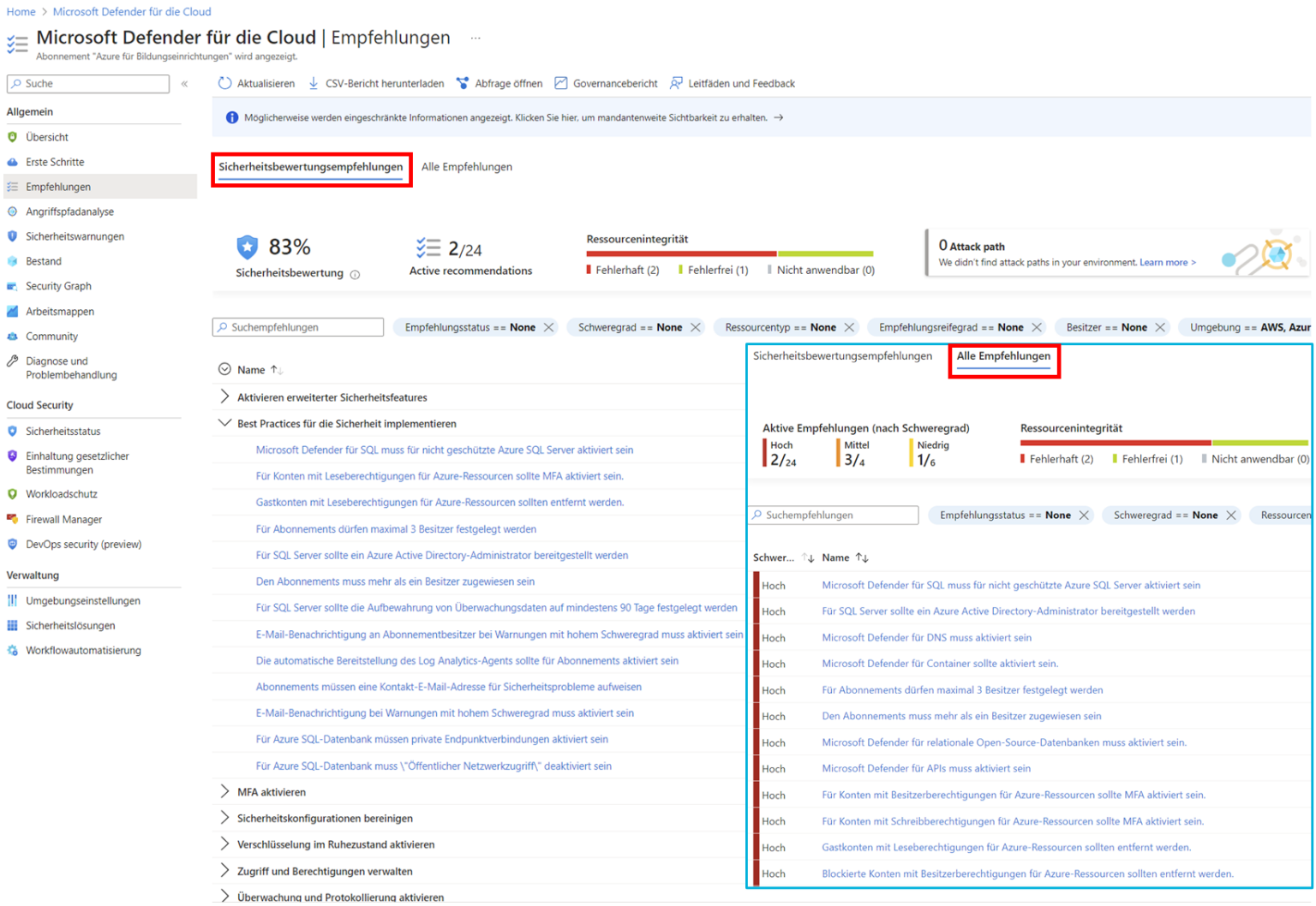


Bild 41: Im MS Defender für Cloud (Sicherheits)Empfehlungen zur Härtung des Systems

4.1.3. Kryptographie

MS Access bietet neben den Berechtigungen des Dateisystems als Schutzmechanismus nur die Dateiverschlüsselung an. Die Verschlüsselungsmethode kann unter „*Datei → Optionen → Clienteinstellungen*“ konfiguriert werden:

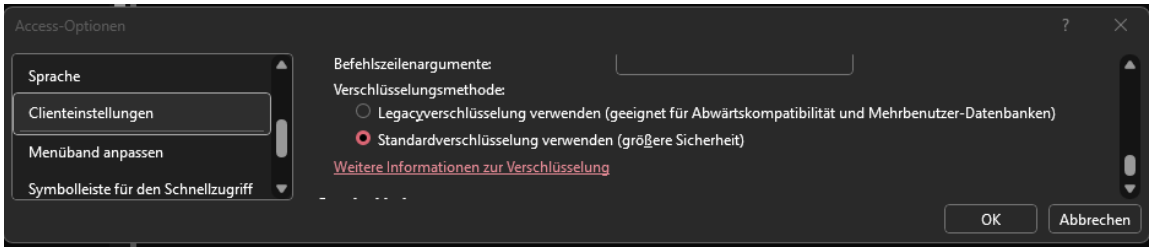


Bild 42: Konfiguration Verschlüsselungsmethode

Default Verschlüsselungsalgorithmus ist der stärkste Industriestandard-Algorithmus, der von der National Security Agency (NSA) in US-Behörden als zu nutzender Standard

festgelegt wurde: Der symmetrische AES-Algorithmus mit 256-Bit Schlüssellänge und Blockverschlüsselung. SHA-512 (SHA-2) wird als Default für die kryptographische Hashfunktion genutzt. Das Passwort ist jeweils mit einem Salt verknüpft als SHA-512-Hash sowie mitverschlüsselt jeweils im Datei-Header am Dateianfang eingebettet (*siehe Kapitel „4.3 Dateiformatanalyse mittels Hexadezimal-Editor (Access)“*) [97].

Mittels Verschlüsselung soll die unbefugte Verwendung der Access-Datei verhindert und die Vertraulichkeit gewährleistet werden. Dabei wird mittels ausgewähltem Verschlüsselungsalgorithmus die gesamte Datei inklusive Inhalt mit einem Passwort verschlüsselt. Zum Öffnen der Datei wird das Passwort benötigt. Da das Access-Dateiformat 2007-2016 keine User-Level-Security unterstützt, ist nach Eingabe des Passworts ein vollständiger Zugriff auf die Access-Datei inklusive aller Konfigurationsmöglichkeiten und Inhalte möglich. Neben der Deaktivierung der Verschlüsselung können so auch unbemerkt schadhafte AutoExec-Makros oder VBA-Schadcode hinzugefügt werden. Die Originaldatei kann bei passenden Berechtigungen unbemerkt mit beliebigem Schadcode manipuliert und ausgetauscht werden. Das Passwort sowie zugehörige Informationen wie der verwendete Salt, Verschlüsselungsalgorithmus oder die „Chaining Mode“ sind im Datei-Header gespeichert:

```
        00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F   Decoded Text
000002B0  3D 22 31 2E 30 22 20 65 6E 63 6F 64 69 6E 67 3D   = " 1 . 0 "   e n c o d i n g =
000002C0  22 55 54 46 2D 38 22 20 73 74 61 6E 64 61 6C 6F   " U T F - 8 "   s t a n d a l o
000002D0  6E 65 3D 22 79 65 73 22 3F 3E 0D 0A 3C 65 6E 63   n e = " y e s " ? > . . < e n c
000002E0  72 79 70 74 69 6F 6E 20 78 6D 6C 6E 73 3D 22 68   r y p t i o n   x m l n s = " h
000002F0  74 74 70 3A 2F 2F 73 63 68 65 6D 61 73 2E 6D 69   t t p : / / s c h e m a s . m i
00000300  63 72 6F 73 6F 66 74 2E 63 6F 6D 2F 6F 66 66 69   c r o s o f t . c o m / o f f i
00000310  63 65 2F 32 30 30 36 2F 65 6E 63 72 79 70 74 69   c e / 2 0 0 6 / e n c r y p t i
00000320  6F 6E 22 20 78 6D 6C 6E 73 3A 70 3D 22 68 74 74   o n "   x m l n s : p = " h t t
00000330  70 3A 2F 2F 73 63 68 65 6D 61 73 2E 6D 69 63 72   p : / / s c h e m a s . m i c r
00000340  6F 73 6F 66 74 2E 63 6F 6D 2F 6F 66 66 69 63 65   o s o f t . c o m / o f f i c e
00000350  2F 32 30 30 36 2F 6B 65 79 45 6E 63 72 79 70 74   / 2 0 0 6 / k e y E n c r y p t
00000360  6F 72 2F 70 61 73 73 77 6F 72 64 22 20 78 6D 6C   o r / p a s s w o r d "   x m l
00000370  6E 73 3A 63 3D 22 68 74 74 70 3A 2F 2F 73 63 68   n s : c = " h t t p : / / s c h
00000380  65 6D 61 73 2E 6D 69 63 72 6F 73 6F 66 74 2E 63   e m a s . m i c r o s o f t . c
00000390  6F 6D 2F 6F 66 66 69 63 65 2F 32 30 30 36 2F 6B   o m / o f f i c e / 2 0 0 6 / k
000003A0  65 79 45 6E 63 72 79 70 74 6F 72 2F 63 65 72 74   e y E n c r y p t o r / c e r t
000003B0  69 66 69 63 61 74 65 22 3E 3C 6B 65 79 44 61 74   i f i c a t e " > < k e y D a t
000003C0  61 20 73 61 6C 74 53 69 7A 65 3D 22 31 36 22 20   a   s a l t S i z e = " 1 6 "
000003D0  62 6C 6F 63 6B 53 69 7A 65 3D 22 31 36 22 20 6B   b l o c k S i z e = " 1 6 "   k
000003E0  65 79 42 69 74 73 3D 22 32 35 36 22 20 68 61 73   e y B i t s = " 2 5 6 "   h a s
000003F0  68 53 69 7A 65 3D 22 36 34 22 20 63 69 70 68 65   h S i z e = " 6 4 "   c i p h e
00000400  72 41 6C 67 6F 72 69 74 68 6D 3D 22 41 45 53 22   r A l g o r i t h m = " A E S "
00000410  20 63 69 70 68 65 72 43 68 61 69 6E 69 6E 67 3D     c i p h e r C h a i n i n g =
00000420  22 43 68 61 69 6E 69 6E 67 4D 6F 64 65 43 42 43   " C h a i n i n g M o d e C B C
00000430  22 20 68 61 73 68 41 6C 67 6F 72 69 74 68 6D 3D   "   h a s h A l g o r i t h m =
00000440  22 53 48 41 35 31 32 22 20 73 61 6C 74 56 61 6C   " S H A 5 1 2 "   s a l t V a l
00000450  75 65 3D 22 35 36 79 63 33 64 5A 32 6C 53 47 4B   u e = " 5 6 y c 3 d Z 2 l S G K
00000460  66 6D 71 38 74 51 6E 2B 4E 51 3D 3D 22 2F 3E 3C   f m q 8 t Q n + N Q = = " / > <
```

Bild 43: Dateiverschlüsselung – Abgelegte Passwort-Informationen und Angaben zum verwendeten Verschlüsselungsalgorithmus im Access-Datei-Header

MS wirbt zwar damit, dass bei Passwortverlust keine Wiederherstellung möglich ist, bietet jedoch das kostenlose „DocRecrypt-Tool“ zum Entfernen des Passwortschutzes für diverse Office-Programme, wenn im Vorfeld dem Datei-Header öffentliche Schlüsselinformationen eines Treuhandschlüssels beigefügt wurden. Der Treuhandschlüssel kann vom Administrator unbemerkt auf den Clients über den Registry-Key „*Computer\HKEY_CURRENT_USER\SOFTWARE\Microsoft\Office\16.0 \common\Security-\Crypto\EscrowCerts*“ zur Verfügung gestellt werden und wird dann beim Verschlüsseln in den Datei-Header geschrieben. Ist das Passwort zur Verschlüsselung nicht mehr verfügbar aber sind die öffentlichen Treuhandschlüsselinformationen im Datei-Header gespeichert, kann der Administrator die Datei mittels seinen privaten Schlüsselinformationen entschlüsseln und das Passwort entfernen oder neu setzen. Erhalten Angreifende Zugriff auf das private Zertifikat, können alle betroffenen Dateien entschlüsselt werden [107]/[216].

In Access kann die genutzte Verschlüsselungsmethode über „*Datei → Optionen → Clienteinstellungen → Verschlüsselungsmethode*“ ausgewählt werden. Zur Auswahl stehen hierbei die Legacyverschlüsselung, die zur Abwärtskompatibilität für alte Access-Dateiformate wie .mdb (*siehe Kapitel „1.2 Abgrenzung“*) empfohlen wird, sowie die Standardverschlüsselung mit höherer Sicherheit. In der folgenden Betrachtung liegt der Fokus auf der Standardverschlüsselung.

MS Office und somit auch Access unterstützen die beiden Kryptografie-APIs des Windows-Betriebssystems Cryptography API (CryptoAPI) sowie Crypto API: Next Generation (CNG) zur Konfiguration der genutzten Verschlüsselung. Über die beiden APIs und die Registry des Windows-Betriebssystems können auf dem Host über den Crypto Service Provider (CSP) des Betriebssystems hier verfügbare Algorithmen für die Verschlüsselung der Dateien, der Cipher Chaining Mode (Default: Cipher Block Chaining) oder der verwendete Zufallszahlengenerator (Default: Random Number Generator) angesprochen beziehungsweise ausgewählt und somit der Standard-Verschlüsselungsalgorithmus ausgetauscht werden. Es können auch Drittanbieterbibliotheken eingebunden werden. Unter folgendem Registry-Schlüssel können die (standardmäßig) installierten CSPs beziehungsweise kryptographischen Algorithmen im Registrierungs-Editor eingesehen werden („*HKEY_LOCAL_MACHINE/SOFTWARE/Microsoft/Cryptography/Defaults/Provider*“):

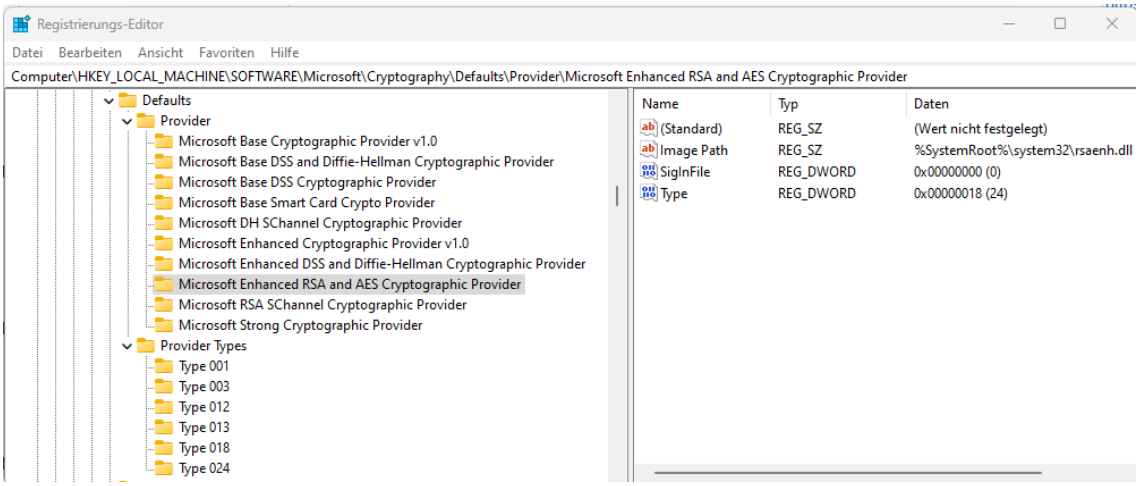


Bild 44: CryptoAPI und per Default installierte CSP des Windows-Betriebssystems [97]

MS empfiehlt den Standard nur zu überschreiben, wenn es das Sicherheitsmodell des jeweiligen Unternehmens erfordert. Außerdem empfiehlt MS Passwortrichtlinien festzulegen, die Vorgaben zu Komplexität und Länge machen. Für Windows-Administratoren gibt es keine Konfigurationsmöglichkeit, die das Verschlüsseln von Dateien erzwingt. Administratoren können nur global festlegen, dass Dokumente nicht verschlüsselt werden dürfen [97].

Auch digitale Signaturen werden vom MS Access Dateiformat 2007-2016 wie über „*Datei → Speichern unter → Packen und signieren*“ unterstützt:

Bild 45: Warnmeldung beim Öffnen einer von einem nicht vertrauenswürdigen Herausgeber signierten Access-Bereitstellungsdatei (.accdc) [275, S. 830]

Digitale Signaturen wahren die Integrität, Authentizität sowie Verbindlichkeit. Dabei verschiebt die „Packen und signieren“-Funktion die DB in eine Access-Bereitstellungsdatei (.accdc), erstellt ein Paket in Form einer Cabinet-Datei aus Bereitstellungsdatei und Signatur und führt eine Komprimierung des erstellten Pakets durch (für Details *siehe Kapitel „4.4 Forensische Analyse (Access)“*). Das signierte Paket bestätigt den Empfängern, dass die DB nicht manipuliert wurde und von einem vertrauenswürdigen Autor stammt. Nach dem Öffnen einer .accdc-Datei werden Nutzende aufgefordert die Datei im ursprünglichen Dateiformat wie .accdb oder .accde zu speichern und somit die DB aus dem Paket zu extrahieren. Es ist kein direktes Arbeiten mit einer .accdc-Datei

möglich. Nach dem Extrahieren der Access-Datenquelle aus dem Paket besteht keine Verbindung mehr zwischen dem signiertem .accdc-Paket und extrahierter DB. Manipulationen können nicht mehr erkannt werden und der Schutz der digitalen Signatur ist hinfällig. Ist ein selbst signiertes Zertifikat unter den vertrauenswürdigen Herausgebern gelistet, wird zugehörigen Paketen stets vertraut. Beim Extrahieren einer Access-Datei unter einem vertrauenswürdigen Pfad, wird der Inhalt automatisch aktiviert [177]/[275, S. 827-831].

Alternativ können über „*VBA-Editor → Extras → Digitale Signatur*“ auch Komponenten wie der VBA-Quellcode oder Abfragen mit einer digitalen Signatur geschützt werden. Dabei ist zu beachten, dass selbst ein Passwortschutz des VBA-Projekts nicht das Ersetzen der hinterlegten digitalen Signatur verhindert. Damit Administratoren genauer kontrollieren können, was auf den Endgeräten von Mitarbeitenden ausgeführt werden darf, können Vorlagen und Add-Ins zusätzlich signiert werden [185]. Ein Test mit einem selbstsignierten Test-Zertifikat unter Verwendung des PowerShell-Befehls „`New-SelfSignedCertificate`“ hat gezeigt, dass trotz nachträglicher Manipulation des VBA-Codes, Speichern, Schließen und neu Öffnen der Datei, die Signatur des VBA-Projekts aktiv bleibt und es keinen Hinweis auf die nachträgliche Manipulation der Logik gibt (*für weitere Details siehe Kapitel „Anlage 5: Digitale Signaturen (Access)“*). Auch mit aktiviertem VBA-Projekt-Passwortschutz kann die hinterlegte digitale Signatur über den „Entfernen“ oder „Wählen…“-Button entfernt oder verändert werden. Ist eine vertrauenswürdige digitale Signatur nicht vom VBA-Projekt einer mit Schadcode manipulierten Datei entfernt worden, kann das Nutzende fälschlicherweise in die Irre führen und zum unbedachten Ausführen der Datei verleiten oder die geschützte Ansicht umgehen. Selbstsignierte Zertifikate sollten nur zu Testzwecken genutzt werden, da sie oft veraltete kryptographische Algorithmen nutzen. Für eine höhere Sicherheit wird von MS der Kauf eines Zertifikats bei einer vertrauenswürdigen Certificate Authority empfohlen.

Auch das letzte Änderungsdatum der Datei ist nicht aussagekräftig, da der Zeitstempel (Timestamp) bereits beim Öffnen einer Datei aktualisiert wird. Access-Dateien speichern kontinuierlich Änderungen, um Datenverlust bei unerwarteten Fehlern vorzubeugen (*siehe Kapitel „3.5 Auswahl Office-Vergleichsprogramm zur Schnittstellenanalyse (Access Excel)“*) [95].

Im Unterschied zu Access bietet die Azure SQL-DB mit der Transparent Data Encryption (TDE) eine standardmäßig aktivierte Funktionalität an, mit der auf Speichermedien abgelegte Daten in der DB, dort abgelegte Transaction Logs sowie Backups in Echtzeit automatisch verschlüsselt werden und somit die Vertraulichkeit auch bei unautorisierten Offline-Zugriffen auf die Hardware gewahrt bleibt. Die TDE verschlüsselt die Daten, wenn sie vom Arbeitsspeicher auf einem Speichermedium abgelegt werden. Die Daten werden von der TDE entschlüsselt, wenn sie von einem Speichermedium in den Arbeitsspeicher geladen werden. Die TDE schützt jedoch keine Daten, die über Kommunikationskanäle übertragen werden. Die TDE führt die Ver- und Entschlüsselung in der Standardkonfiguration mittels symmetrischem Schlüssel („Database Encryption Key“) und AES-256-Algorithmus durch. Die Nutzung anderer Algorithmen ist jedoch auch möglich. Der Database Encryption Key wird durch ein in die Master-DB integriertes

Serverzertifikat verschlüsselt, beim Start des DBS entschlüsselt und in einem Prozess der DB-Engine für anstehende Ver- sowie Entschlüsselungen abgelegt. MS tauscht die Serverzertifikate gemäß ihrer internen Sicherheitsrichtlinien automatisch aus, der Stammschlüssel wird von einem „MS internen Geheimspeicher“ geschützt. Verwendeter Verschlüsselungsalgorithmus, die Schlüssellänge sowie weitere für die TDE relevante Informationen sind in System Dynamic Management Views (DMV) and Functions (DMF) wie „`sys.dm_database_encryption_keys`“ zu finden (*siehe Kapitel „4.1.5 Logging & Auswertungsmöglichkeiten“*). Wiederhergestellte DB-Kopien sind standardmäßig nicht durch die TDE verschlüsselt. Auch die Master-DB kann nicht verschlüsselt werden, da sie Objekte enthält, die zur Ausführung der TDE-Funktionalität benötigt werden. In System-DBs sollten daher keine vertraulichen Informationen abgelegt sein [1, S. 240, S. 302]/[51, S. 64]/[226]/[228]:

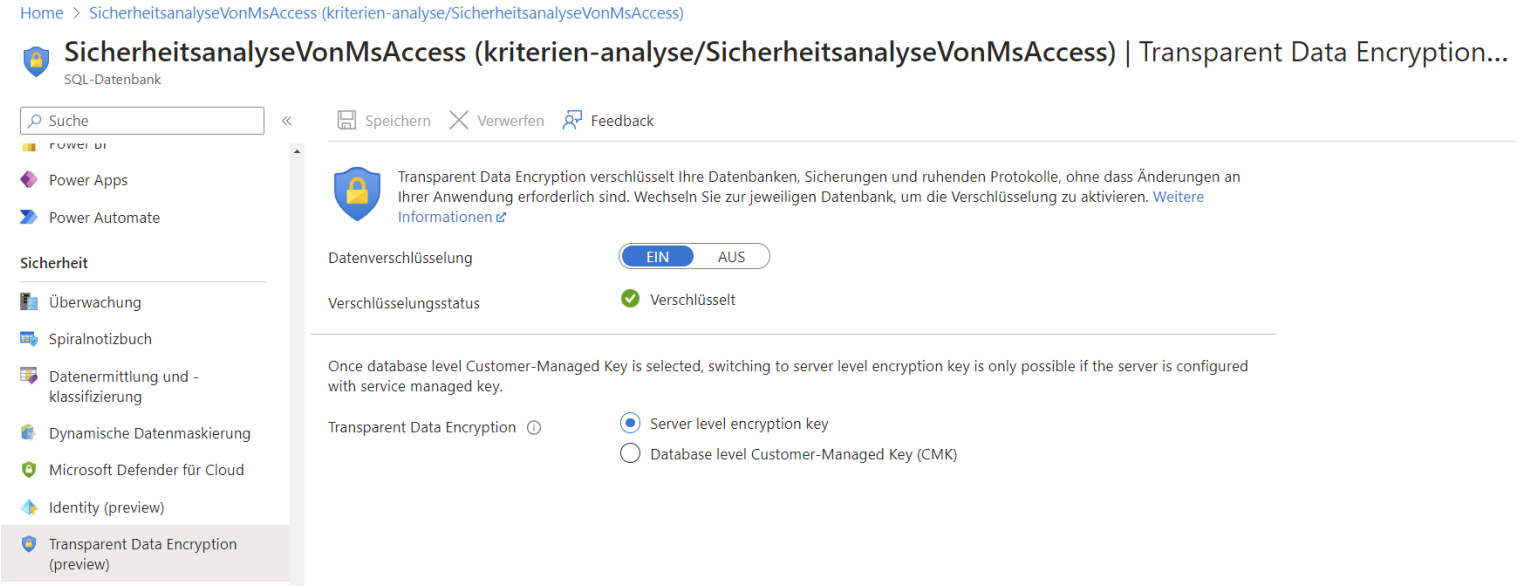


Bild 46: Überprüfung des Verschlüsselungsstatus auf DB-Ebene durch die in Azure SQL-DB standardmäßig aktivierte TDE-Funktionalität

Zur Verschlüsselung und damit zur Wahrung der Vertraulichkeit während der Übertragung sowie zur Gewährleistung der Authentizität und Integrität erfolgt die Kommunikation mit Azure SQL-DB verpflichtend über das Transport Layer Security (TLS)-Protokoll. Auf diese Weise ist die die gesamte Kommunikation zwischen Client und Azure SQL-DB (Server) verschlüsselt [81]/[155]. Im Azure-Portal kann auf dem Server unter „Netzwerk“ die TLS-Mindestversion konfiguriert werden, welche die sich verbindenden Clients für einen erfolgreichen Verbindungsaufbau verwenden müssen:

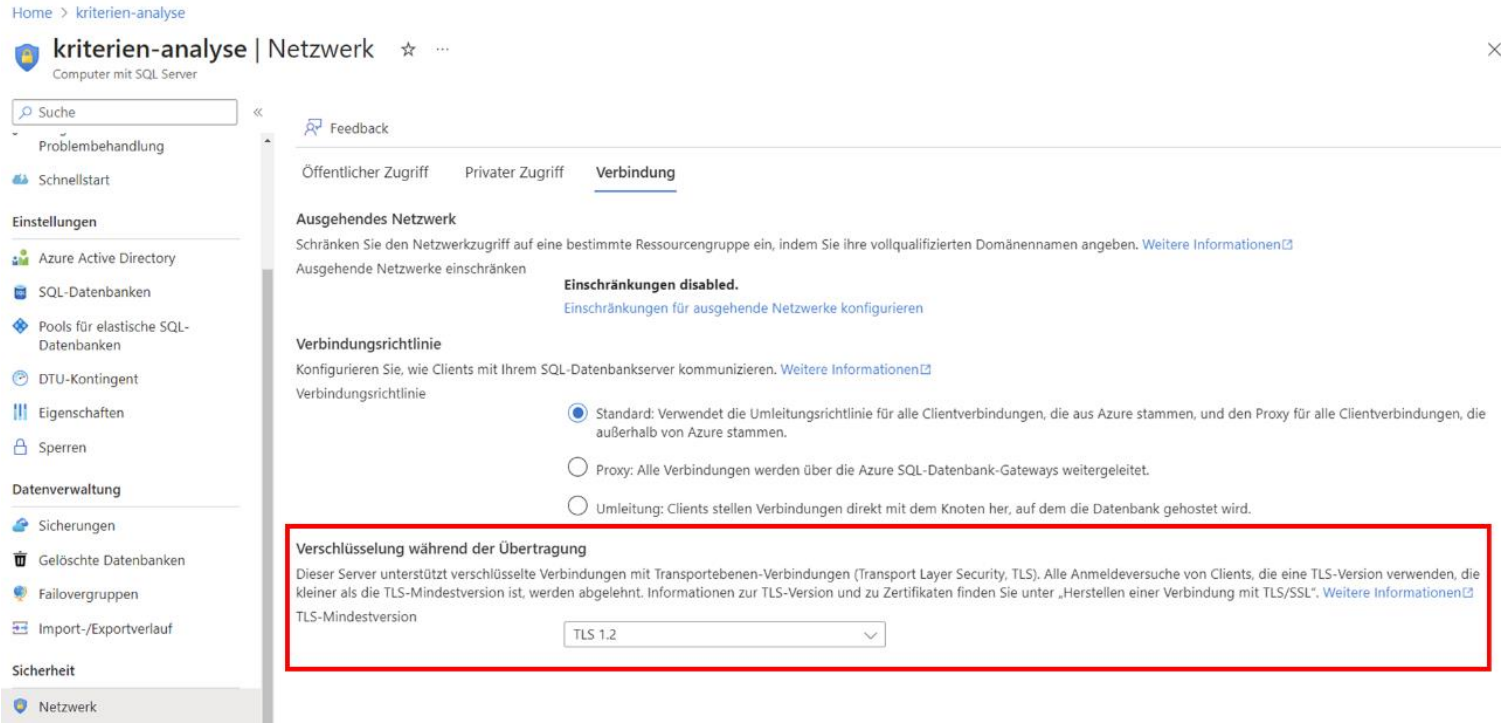


Bild 47: Konfiguration der TLS-Mindestversion des Azure SQL-DB-Servers über das Azure-Portal, die alle sich verbindenden Client nutzen müssen

Um zusätzlich zur TLS auch die über Kommunikationskanäle übertragenen Daten zu schützen und gleichzeitig im DBS den unautorisierten Zugriff im Klartext zu verhindern, bietet Azure SQL-DB die Funktionalität „Always Encrypted“ an. Dadurch werden die in der jeweiligen DB gespeicherten Daten durch Verschlüsselung vor unberechtigter Einsicht geschützt. Ziel ist es, dass nur Befugte Dateneinsicht bekommen. Unbefugte wie Administratoren für die Datenverwaltung oder Wartungstechniker haben keine Dateneinsicht. Mittels Always Encrypted-Funktionalität verschlüsseln die Clients die Daten innerhalb der jeweiligen Client-Anwendung, bevor sie an das DBS gesendet werden. Die Always Encrypted-Funktionalität ist dabei als Treiber umgesetzt, der auf dem jeweiligen Client installiert wird. Der Treiber verschlüsselt die vertraulichen Daten, bevor sie an das DBS weitergereicht werden. Der Treiber passt bei Bedarf auch SQL-Abfragen an. Im DBS selbst liegen die Daten in verschlüsselter Form vor. Vom DBS erhaltene, verschlüsselte Daten werden vom Treiber im Client entschlüsselt. Die zugehörigen Schlüssel zur Ver- und Entschlüsselung müssen der Azure SQL-DB nicht bekanntgemacht werden und verbleiben im Client [51, S. 64]/[80].

Mittels Transact-SQL-Befehl „*`Add Signature`*“ können Stored Procedures, Funktionen oder Trigger digital signiert werden. Dabei wird jedes einzelne Zeichen einschließlich Zeilenumbrüche wie Line Feeds und Carriage Returns bei der Signaturberechnung berücksichtigt. Wird das jeweilige Objekt verändert, wird die digitale Signatur verworfen. Informationen zu Signaturen sind in „*`sys.crypt_properties`*“ zu finden [78].

Abschließend sei noch die „Ledger“-Funktionalität als manipulationssicherer Beweis zur Wahrung der Datenintegrität mittels kryptographischen Hash-Funktionen erwähnt, die in *Kapitel „4.1.5 Logging & Auswertungsmöglichkeiten“* näher beschrieben wird.

4.1.4. Passwörter & Authentifizierung

Access besitzt, wie bereits an unterschiedlichen Stellen zuvor erwähnt, keine Sicherheitsfunktionalität auf Benutzerebene. Somit ist eine Analyse von Authentifizierungs-Funktionalitäten bei Access nicht möglich. In Access-Dateien kann ein Passwort für den VBA-Projektschutz sowie die Dateiverschlüsselung verwendet werden. Passwortwiederverwendungen werden nicht erkannt, allerdings werden Eingaben in den dazugehörigen Eingabe-Feldern für VBA-Projektschutz- und Passwörter zur Dateiverschlüsselung nicht im Klartext angezeigt. Maximale Passwortlänge für die Dateiverschlüsselung sind 20 Zeichen, maximale Passwortlänge zum Schutz des VBA-Projekts sind 32 Zeichen (Groß-, Kleinbuchstaben, Ziffern, Sonderzeichen einschließlich Umlaute). Mit der Dateiverschlüsselung soll die unbefugte Verwendung verhindert und die Vertraulichkeit geschützt werden. Mittels Verschlüsselung wird der zuvor im Klartext gespeicherte Inhalt sowie der Quellcode verschlüsselt und ein Passwort zum Öffnen der Datei festgelegt. Wird direkt und nicht über ein zugehöriges Frontend mit einer verschlüsselten Access-Datei gearbeitet, muss das Passwort zur Entschlüsselung den Nutzenden bekannt sein. Ist das Kennwort bekannt, besteht aufgrund fehlender Sicherheit auf Benutzerebene nach Eingabe Vollzugriff und das Passwort kann über die Access-Oberfläche entfernt werden [216].

Mit dem VBA-Projektschutz wird der VBA-Quellcode durch ein Passwort vor Manipulation und Anzeige geschützt, wobei dieser Schutz unter Umständen leicht umgangen werden kann (*siehe Kapitel „4.3 Dateiformatanalyse mittels Hexadezimal-Editor (Access)“*). Der Quellcode wird dabei nicht verschlüsselt, hierfür muss die Dateiverschlüsselung verwendet werden.

MS Access bietet keinen Schutz gegen Brute-Force-Angriffe auf den VBA-Projektschutz oder das Passwort zur Dateiverschlüsselung. Aufgrund der Verwendung gängiger und robuster Verschlüsselungsalgorithmen über die CryptoAPI des Windows-Betriebssystems bleibt einem Angreifenden ohne Hinzufügen eines öffentlichen Treuhandschlüssels in den Datei-Header (*siehe Kapitel „4.1.3 Kryptographie“*) nur die vollständige Suche nach dem Passwort. Der Erfolg des Angriffs und die hierfür benötigte Zeit ist somit von der Komplexität des verwendeten Passworts abhängig. Die Wahl eines guten sowie sicheren Passworts ist somit Grundvoraussetzung für die Datensicherheit. Im Internet sind eine Vielzahl an Tools zum Passwort-Cracking von MS Access-Dateien und anderen Office-Programmen verfügbar. Darunter das Tool „Accdb Password Recovery“ von „*TheGrideon Software*“ [262]. Derartige Tools arbeiten in der Regel nach dem Brute-Force-Ansatz oder mit Wörterbüchern. Das bedeutet je länger, kryptischer und mehr unterschiedliche Zeichen (Klein- sowie Großbuchstaben, Ziffern, Sonderzeichen) verwendet werden, desto länger dauert das Erraten des Passworts. Bei Beachtung dieses Grundsatzes kann das Passwort-Cracking Jahre dauern. Nachfolgendes Beispiel demonstriert den Aufwand, um alle möglichen Kombinationen einer Zeichenfolge bestehend aus 26 Klein- sowie 26 Großbuchstaben, 10 Ziffern und 32 Sonderzeichen durchzuprobieren. Es werden 346.350.000 Instructions per Seconds (Intel Core i5-11600K) angenommen:

Formel 4.1.4.1: Formel zur Berechnung aller Kombinationsmöglichkeiten eines Passworts

$$Gesamtanzahl\ Kombinationsmöglichkeiten = Summe\ Anzahl\ aller\ möglichen\ Zeichen^{Zeichenlänge\ Passwort} \quad (4.1.4.1)$$

Formel 4.1.4.2: Formel zur Berechnung der benötigten Sekunden, zum Durchgehen aller möglichen Passwörter

$$Benötigte\ Sekunden\ zum\ Durchgehen\ aller\ Kombinationsmöglichkeiten = \frac{Gesamtanzahl\ Kombinationsmöglichkeiten}{CPU\ Instructions\ per\ Second} \quad (4.1.4.2)$$

Formel 4.1.4.3: Beispiel mit 8 Zeichen langem Passwort bestehend aus Klein- sowie Großbuchstaben, Ziffern und Sonderzeichen. Inklusive Umrechnung benötigte Zeit zum Durchprobieren aller Möglichkeiten in Jahre

$$6.095.689.385.410.816\ Kombinationsmöglichkeiten = (26\ (Kleinbuchstaben) + 26\ (Großbuchstaben) + 10\ (Ziffern) + 32\ (Sonderzeichen))^{8\ Zeichen\ Passwort}$$

$$17.599.796\ Sekunden \approx \frac{6.095.689.385.410.816\ Gesamtanzahl\ Kombinationsmöglichkeiten}{346.350.000\ CPU\ Instructions\ per\ Second} \quad (4.1.4.3)$$

$$17.599.796\ Sekunden \triangleq 0{,}558\ Jahre$$

Formel 4.1.4.4: Beispiel mit 12 Zeichen langem Passwort bestehend aus Klein- sowie Großbuchstaben, Ziffern und Sonderzeichen. Inklusive Umrechnung benötigte Zeit zum Durchprobieren aller Möglichkeiten in Jahre

$$475.920.314.814.253.376.475.136 = 94^{12}$$

$$1.374.102.251.520.870\ Sekunden \approx \frac{475.920.314.814.253.376.475.136\ Gesamtanzahl\ Kombinationsmöglichkeiten}{346.350.000\ CPU\ Instructions\ per\ Second} \quad (4.1.4.4)$$

$$1.374.102.251.520.870\ Sekunden \triangleq 43.572.496{,}6\ Jahre$$

Selbst mit 2 Milliarden Instructions per Second würde das Durchprobieren aller Möglichkeiten eines 12 Zeichen langen Passworts nach den obigen Kriterien ungefähr 7,5 Millionen Jahre benötigen. Wird die Schlüssellänge weiter erhöht, können zumindest symmetrische Algorithmen auch Angriffen mit Quantencomputern standhalten. Das BSI empfiehlt hier Schlüssellängen ab 256 Bit. Anzumerken ist, dass MS, BSI und Geheimdienste ihre Strategie geändert haben und kein regelmäßiges Ändern der Passwörter mehr empfehlen. Regelmäßiges Ändern von Passwörtern führt in der Regel zu schwachen Passwörtern, da die Neuvergabe oft einem Passwortvergabe-Schema folgt. Ein starkes Passwort kann über Jahre sicher verwendet werden, da Brute-Force-Ansätze hier sehr zeitaufwändig sind [13]/[15]/[55]/[135]/[242]/[270].

Wurden öffentliche Treuhandschlüsselinformationen in den Datei-Header eingefügt, kann das in *Kapitel „4.1.3 Kryptographie“* vorgestellte DocRecrypt-Tool von MS zum Entfernen des Passwortschutzes für diverse Office-Programme verwendet werden.

Im Gegensatz zu Access verfügt die Azure SQL-DB über eine mehrschichtige Architektur, um den Zugriff auf das DBS und die darin enthaltenen Daten zu schützen. Die erste Sicherheitsebene stellen Firewall-Regeln auf der Service-Schicht dar, um Zugriffe auf eine DB („*sys.database_firewall_rules*“, Stored Procedure: „*sp_set_database_firewall_rule*“), den gesamten Server („*sys.firewall_rules*“, Stored Procedure: „*sp_set_firewall_rule*“) oder auch andere Azure-Dienste nur über explizit ausgewählte öffentliche IP-Adressen zuzulassen oder gesamthaft den öffentlichen Netzwerkzugriff zu untersagen (Netzwerk-Sicherheit). Damit bieten Firewall-Regeln auch gleichzeitig ein Schutz vor (Brute-Force-)Angriffen aus unbekannten Netzwerken. Statt über SQL können sie auch im Azure-Portal unter „Netzwerk“ hinterlegt werden.

Das Aktivieren der Funktion „Azure-Diensten und -Ressourcen den Zugriff auf diesen Server gestatten“ ermöglicht es allen Azure-Diensten im jeweiligen Abonnement auf den Server zuzugreifen, was eine potenzielle Schwachstelle darstellt.

Nach Erstellen einer neuen Azure SQL-DB ist der Zugriff in der Standardkonfiguration nur über private Endpunkte mit einer privaten IP-Adresse im selben „Azure Virtual Network“-Subnetz möglich. Zugriffe über öffentliche Endpunkte, wie ein Endgerät aus einem fremden Netzwerk, dass mittels öffentlicher IP-Adresse über das Internet auf die Azure SQL-DB zugreift, werden standardmäßig untersagt. Diese Konfiguration kann durch Hinzufügen einer virtuellen Firewall-Regel aufgelockert und der Zugriff für eine konkrete öffentliche IP-Adresse erlaubt werden:

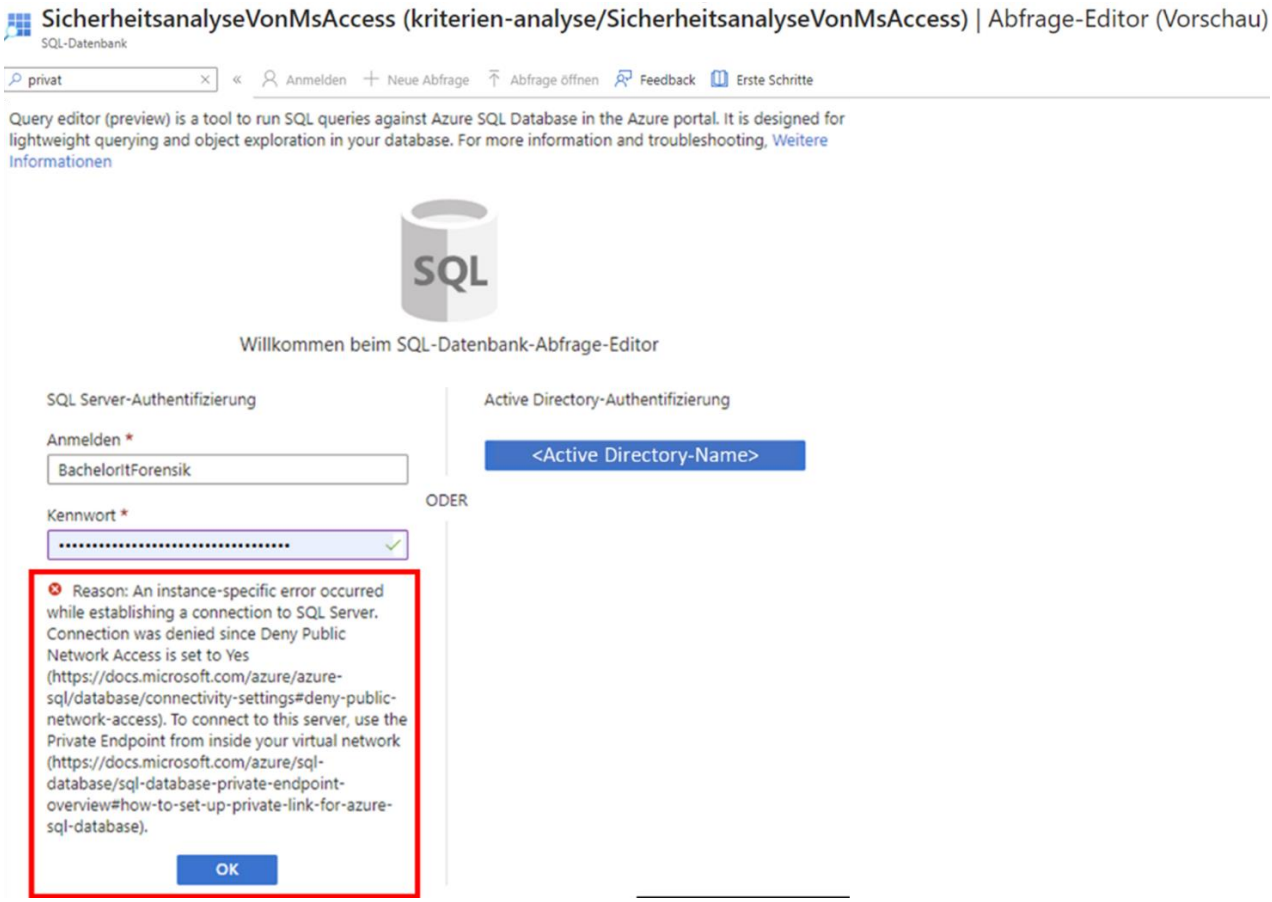


Bild 48: Ohne Konfiguration einer Firewall-Regel sind per Default keine Zugriffe auf die Azure SQL-DB über öffentliche Endpunkte, wie ein Endgerät aus einem fremden Netzwerk, das mittels öffentlicher IP-Adresse über das Internet auf die Azure SQL-DB zugreift, möglich

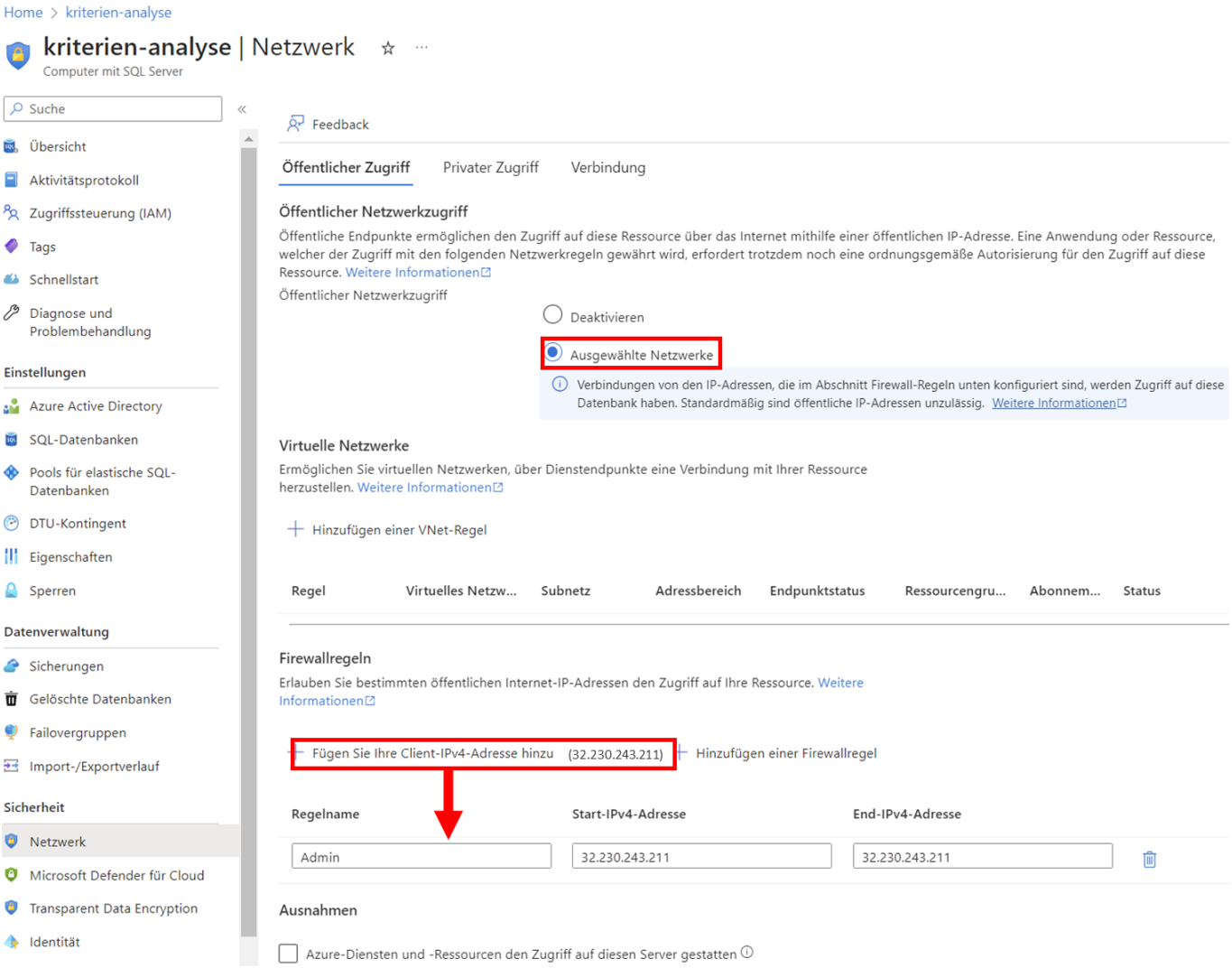
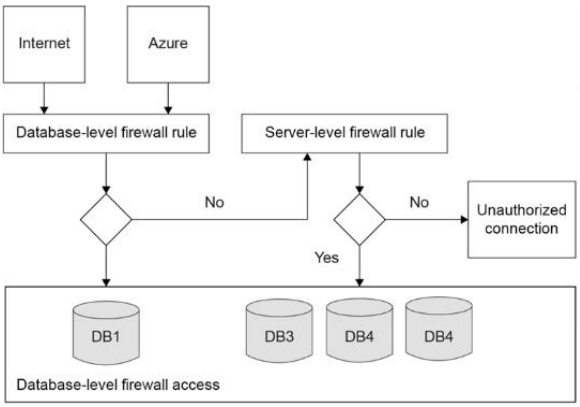


Bild 49: Durchzuführende Konfigurationen und Hinzufügen einer Firewall-Regel, damit der Zugriff über öffentliche Endpunkte möglich ist. Die öffentliche IP-Adresse wird automatisch erkannt und die Firewall-Regel kann mit einem Klick angelegt werden. Nach Speicher der Änderungen funktioniert der Zugriff problemlos

Dies ist auch ein Beispiel für die Zugriffsbeschränkungen durch Zulassungs- oder Verweigerungslisten, mit denen der Zugriff auf Apps sowie Dienste beschränkt werden kann, die vom „Azure App Service“ gehostet werden. Nachfolgende Grafik veranschaulicht die initiale Prüfung der Firewall-Regeln auf der Service-Schicht:



Server-level firewall rule	Database-level firewall rule
Allows clients to access all SQL databases in given logical SQL Server.	Allows clients to access particular SQL databases within the logical SQL Server.
Rules are stored in the master database.	Rules are stored within individual SQL databases.
Can be configured using the Azure portal, PowerShell, and T-SQL.	Can only be configured using T-SQL after configuring the first server-level firewall rule.

Bild 50: Links: Firewall-Regeln als erste Sicherheitsebene während der Authentifizierung [1, S. 241]; Rechts: Unterschied Server- und DB-Firewall-Regeln [1, S. 242]

Ist die Überprüfung der Firewall-Regeln erfolgreich, erfolgt die Authentifizierung mittels SQL-Authentifizierung (Username und Passwort) oder Azure AD-Authentifizierung. Für die Sicherheit auf Datenebene können Datenmaskierungen und Sicherheit auf Zeilen- sowie Spaltenebene konfiguriert werden (*siehe Kapitel „4.1.7 Datenschutzkonformer Zugriff“*). Bereits beim Anlegen eines neuen Servers für eine Azure SQL-DB wird eine der beiden Authentifizierungsmethoden festgelegt. Alle in den Dialogen eingegebenen Passwörter werden standardmäßig nicht im Klartext angezeigt. Für AD-Benutzer können Passwörter zwischen 8 und 256 Zeichen (Groß-, Kleinbuchstaben, Ziffern, Sonderzeichen exklusive Umlaute) vergeben werden. Darüber hinaus können AD-Nutzer granular neben der Azure SQL-DB auch für andere Azure-Dienste verwendet und berechtigt werden. Um mehrere Azure AD-Benutzer dieselben Rechte zu vergeben, werden sie in Azure AD-Gruppen zusammengefasst (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung“*) [1, S. 239-260]/[88]/[106]/[134]/[163]/[165]:

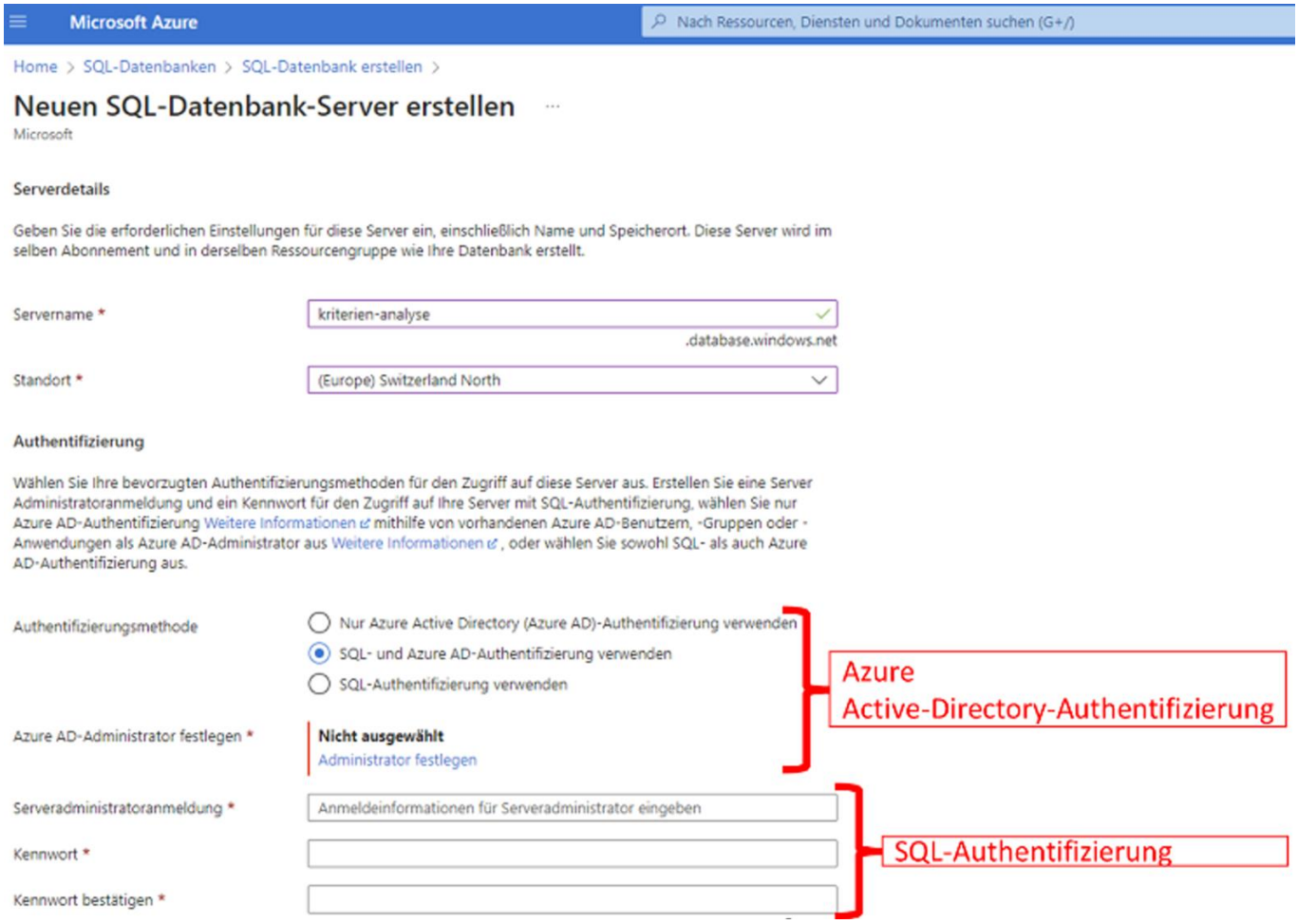


Bild 51: Authentifizierungsmethoden bei Anlage eines Servers für eine Azure SQL-DB

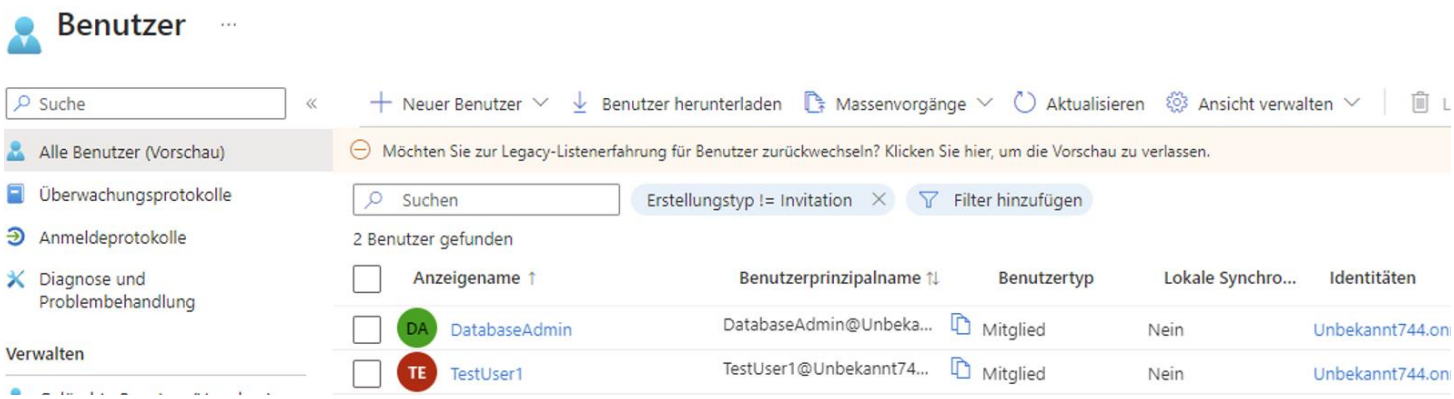


Bild 52: Anlage von zwei AD-Nutzern in Azure AD

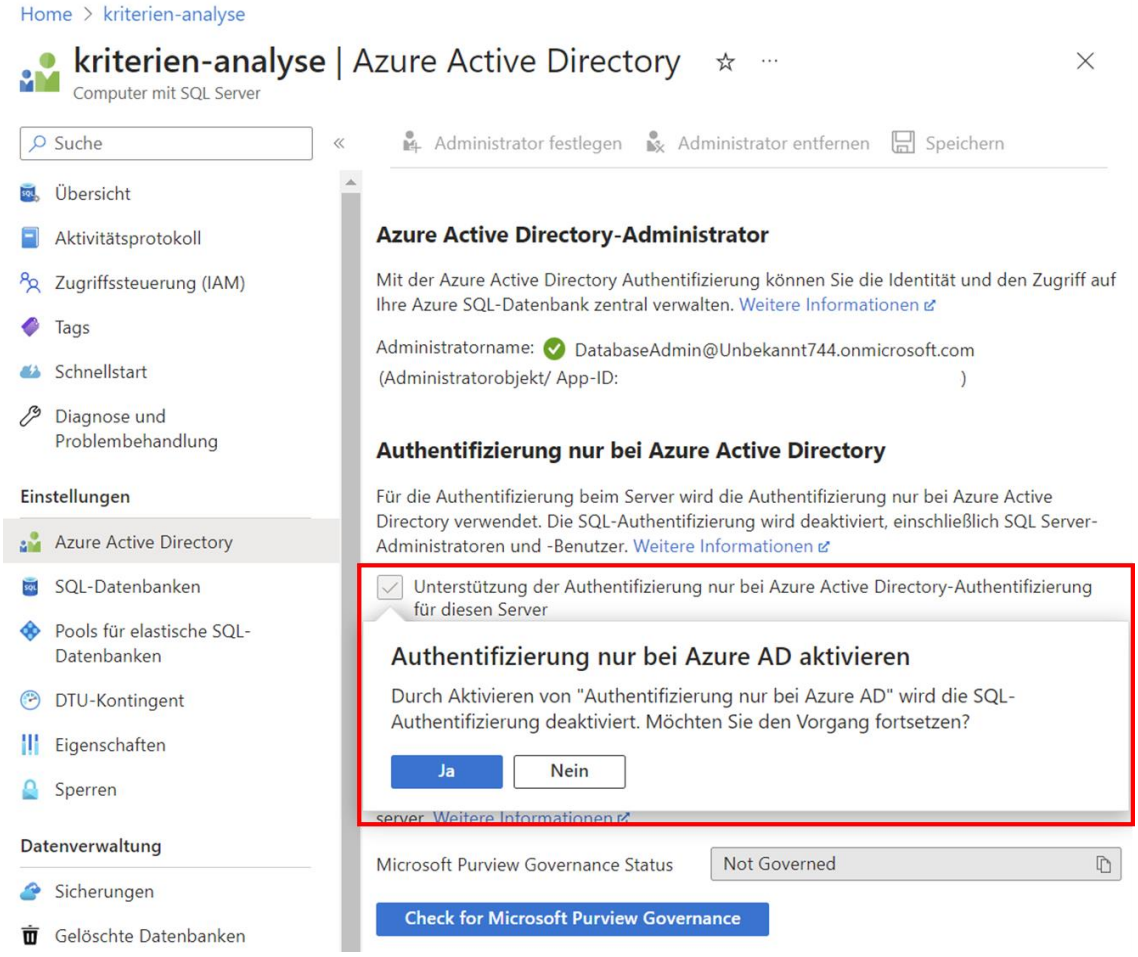


Bild 53: Azure AD-Einstellungen des SQL Servers, auf dem die DB läuft. Sollen AD-Nutzer und AD-Gruppen als DB-(Gast)Nutzer hinzugefügt werden, muss ein AD-Benutzer als Admin für den dbowner-Zugriff hinterlegt werden (hier DatabaseAdmin). Bei Aktivierung der AD-Authentifizierung, wird die SQL-Authentifizierung mittels Nutzername und Passwort deaktiviert [1, S. 268]

Mit Azure AD stehen drei Authentifizierungsmethoden zur Verfügung. In der einfachsten Variante wird der AD-Account als Username verwendet und das zugehörige Passwort angegeben. Im integrierten Modus wird der am jeweiligen Windows-Betriebssystem angemeldete Domänen-User zur Authentifizierung verwendet, es muss kein Passwort angegeben werden. Im universellen Modus mit Multi-Faktor-Authentifizierung, wird der jeweils am Windows-Betriebssystem angemeldete Domänen-User verwendet und es muss zusätzlich als weitere Sicherheitsebene ein zusätzlicher Faktor zur Authentifizierung (Anruf, SMS, mobile App, Zertifikate) angegeben werden [1, S. 261-263]:

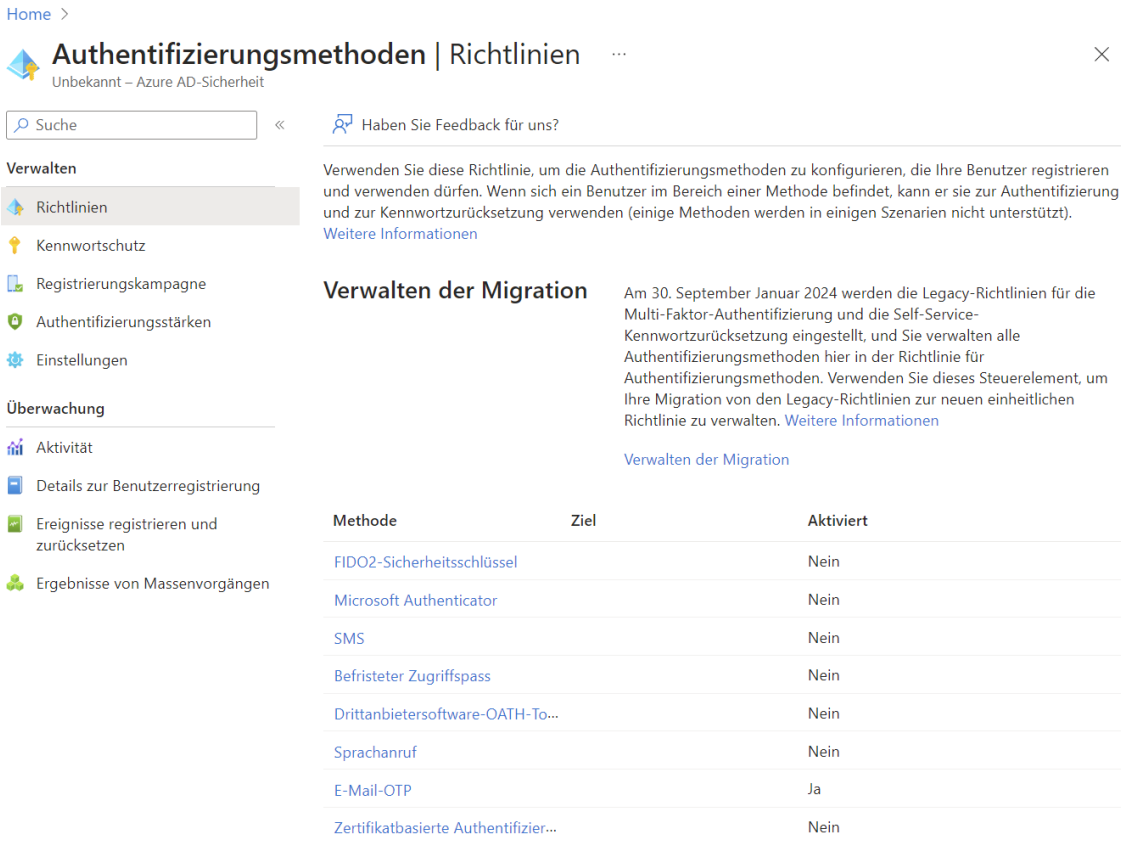


Bild 54: Konfiguration von unterschiedlichen Methoden zur Multi-Faktor-Authentifizierung über Azure AD-Authentifizierungsmethoden

Um Brute-Force-Angriffe auf Passwörter zu erschweren, können in den „Azure AD-Authentifizierungsmethoden“ benutzerdefinierte intelligente Sperren konfiguriert werden, die in der Testversion nicht zur Verfügung stehen. Standardmäßig werden AD-Accounts nach 10 erfolglosen Anmeldeversuchen gesperrt. Neben der Angabe nach wie vielen fehlerhaften Anmeldeversuchen die Sperre greift, kann auch die Mindestsperrdauer in Sekunden festgelegt werden. Wird nach einer Sperre erneut ein fehlerhafter Anmeldeversuch registriert, wird das Konto erneut gesperrt. Bei jeder aufeinanderfolgenden Sperrung verlängert sich die Sperrdauer. Neben Sperren können auch verbotene Zeichenketten angegeben werden, die Nutzende nicht als Passwort wählen dürfen. Durch die Option werden Wörterbuchangriffe erschwert und schwache Passwörter können verboten werden. Außerdem werden schwache Passwörter wie Ähnlichkeiten mit dem User-Namen erkannt [133]/[134]/[224]:

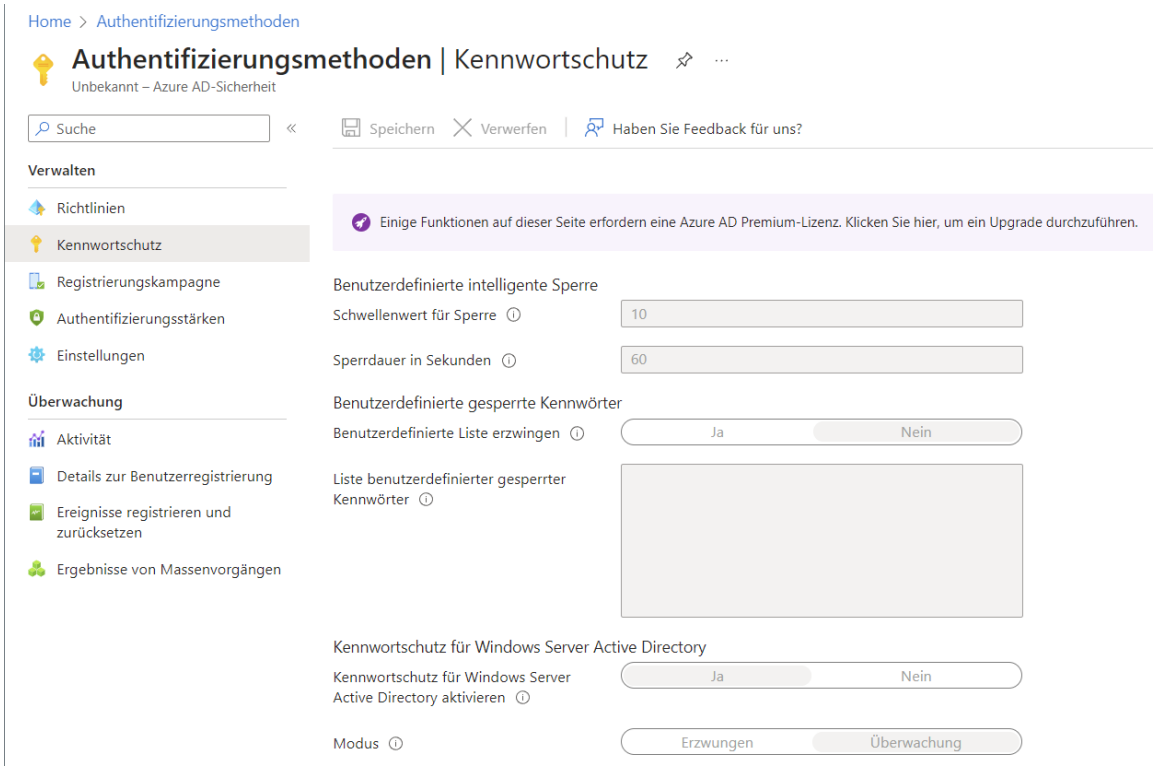


Bild 55: Konfiguration von intelligenten Sperren zum Kennwortschutz über Azure AD-Authentifizierungsmethoden

Beim Anlegen neuer AD-Nutzer kann entweder ein automatisch generiertes oder selbsterstelltes Initialpasswort vergeben werden. Das Initialpasswort hat folgende Mindestanforderungen:

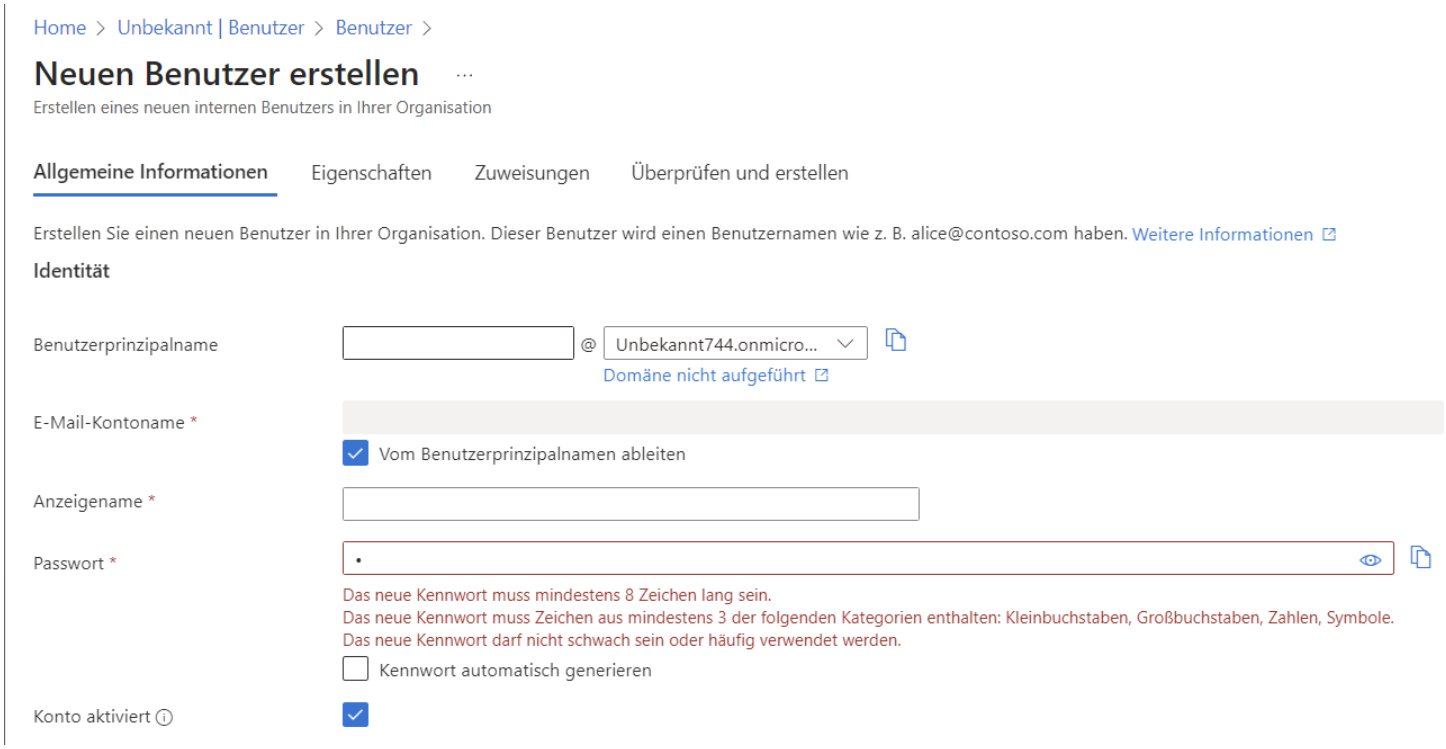


Bild 56: AD-Benutzer erstellen – Mindestanforderungen an das Initialpasswort

Der Nutzer kann sich nun mit dem neuen Azure AD-Benutzeraccount und dem Initialpasswort bei MS anmelden. Im nachfolgenden Szenario wird der Account dem „Azure Data Studio“ hinzugefügt. Es gelten die folgenden Mindestanforderungen:

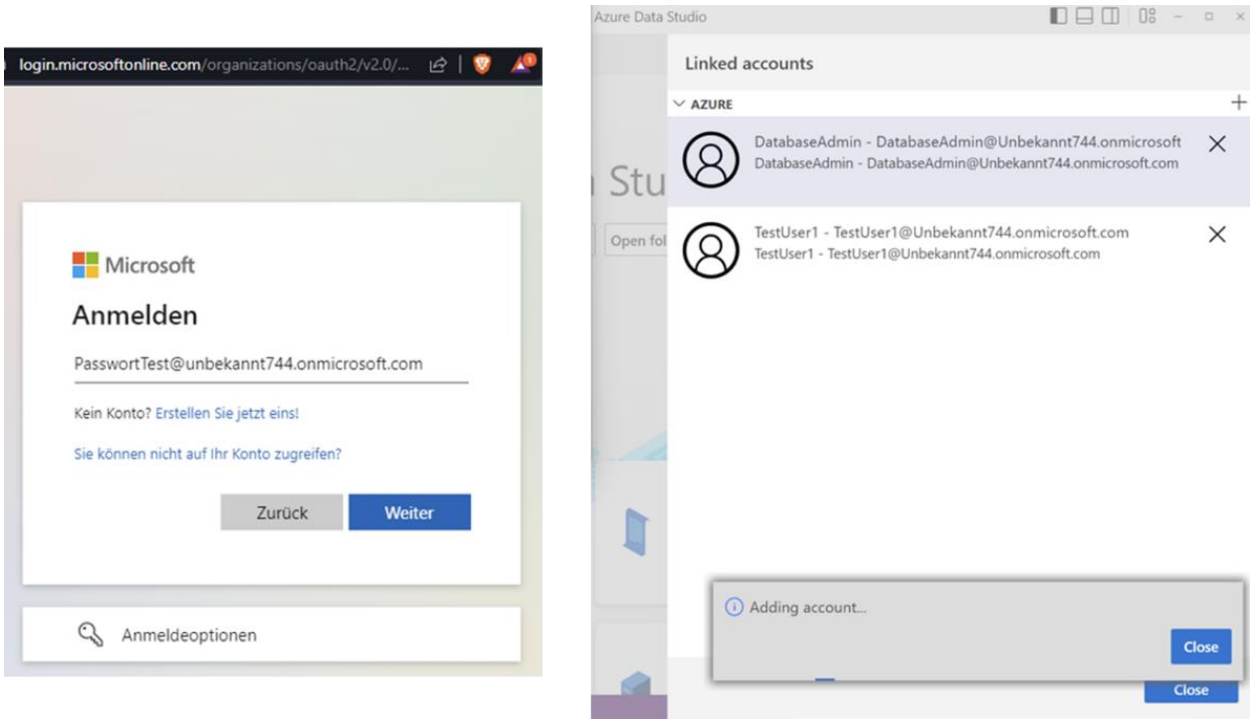


Bild 57: Hinzufügen des zuvor erstellten AD-Benutzers als User im Azure Data Studio

Im Rahmen der Erstanmeldung muss das Initialpasswort von Nutzenden geändert werden. Es gelten die folgenden Passwortanforderungen, wobei Umlaute nicht genutzt werden können:

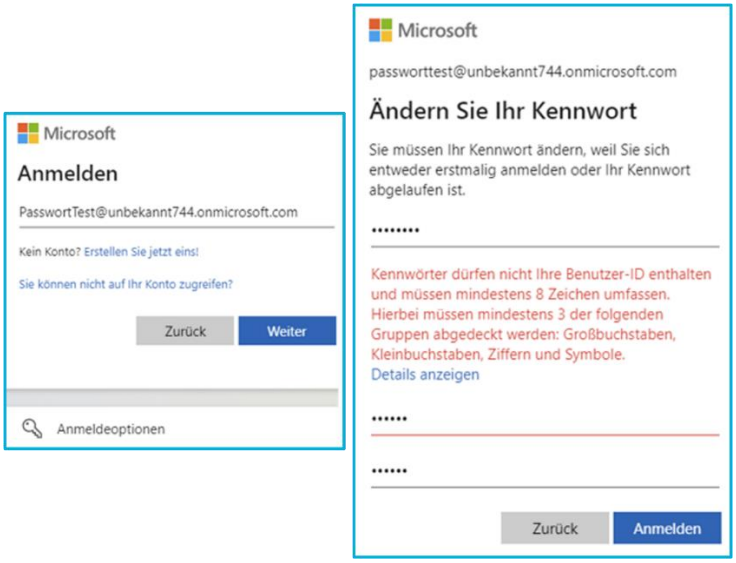


Bild 58: Vergabe eines Passworts für einen zuvor angelegten AD-Benutzer nach initialer Anmeldung mittels Initialpasswort

Passwortwiederverwendungen werden nicht erkannt. Wird das Kennwort über Azure AD zurückgesetzt, kann beliebig oft ein vorheriges Initialpasswort wiederverwendet werden:

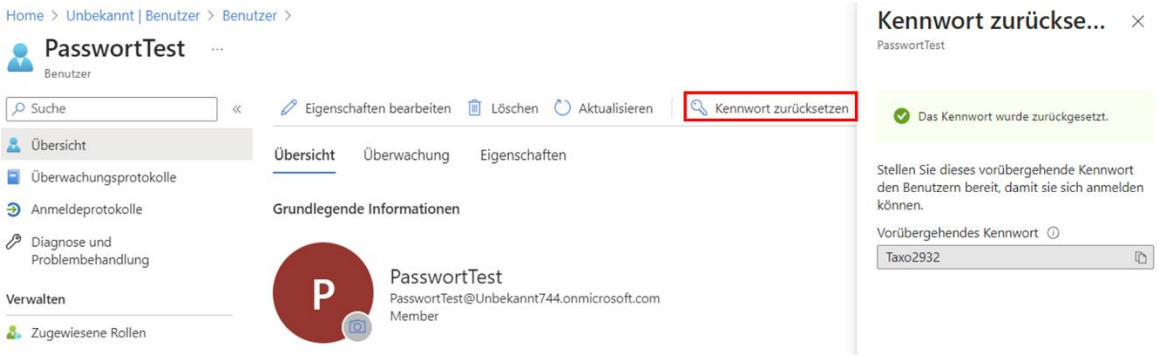


Bild 59: Passwort eines Azure AD-Benutzers über das Azure-Portal zurücksetzen und Vergabe eines neuen Initialpasswords, das bei der Anmeldung geändert werden muss

Doch neben den fortschrittlichen Ansätzen seien zum Abschluss dieses Unterkapitels noch zwei gravierende IT-Sicherheitsvorfälle bei der Azure-Cloud und Azure AD erwähnt:

Im März 2021 hat MS Sicherheitslücken im „MS Exchange Server“ geschlossen, über die Angreifer Zugriff auf Outlook E-Mail-Postfächer von Behörden und Unternehmen erhielten und sogar über eine Remote Code Execution Vulnerability Schadcode einschleusen konnten (CVE-2021-26855). Beim Angriff haben die Angreifenden einen von MS erbeuteten Signaturschlüssel verwendet, die Herkunft des Schlüssels ist unklar. Das BSI hat die aus der Schwachstelle entstehende IT-Bedrohungslage als geschäftskritisch mit massiven Beeinträchtigungen des Regelbetriebs eingestuft.

Gemäß IT-Sicherheitsdienstleister *Volexity*, *Wirtschaftswoche* und *Brian Krebs* hatte MS erstmals schon im Januar von der Zero Day-Schwachstelle erfahren. Teilweisen Schutz vor diesem Angriff gaben das Ablehnen von nichtvertrauenswürdigen Verbindungen auf Port 443, wie über das Internet, oder die ausschließliche Erreichbarkeit des Exchange Servers über das VPN. Ende Juli 2023 gab es einen weiteren Angriff auf MS, indem der zuvor bei Exchange eingesetzte Signaturschlüssel diesmal im Azure-Cloud-Umfeld verwendet wurde. Es hat sich herausgestellt, dass der von MS entwendete Signaturschlüssel nicht nur bei Exchange Servern funktionierte, sondern als eine Art Masterkey auch für alle MS-Cloud-Anwendungen einschließlich Outlook, Office, SharePoint und Teams eingesetzt werden konnte. Mittels des Signaturschlüssels konnten die Angreifenden Zugangstoken für beliebige Azure AD-Benutzer erstellen, einschließlich der von Kunden in der Azure-Cloud betriebenen Anwendungen und hatten somit Zugang zu nahezu allen in der Azure-Cloud gespeicherten Kundendaten. Sobald ein „Login with Microsoft“ möglich war, konnte der mit dem gestohlenen Schlüssel signierte Zugangstoken bei der Anmeldung verwendet werden. Anwendungen, die ihre vertrauenswürdigen Zertifikate in einem lokalen Zertifikatspeicher oder Cache halten, sind weiterhin von dem Angriff bedroht, bis sie den kompromittierten Signaturschlüssel aus ihren Zwischenspeichern entfernen. Über den vom Azure-Portal angebotenen Funktionsumfang hinausgehende Log-Dateien hat MS anfänglich nur gegen zusätzliches Entgelt herausgegebenen, damit Kunden ihre Azure-Konten auf auffällige Aktivitäten überprüfen konnten [12]/[32]/[53]/[54]/[73]/[267]/[272]/[273]/[274]:

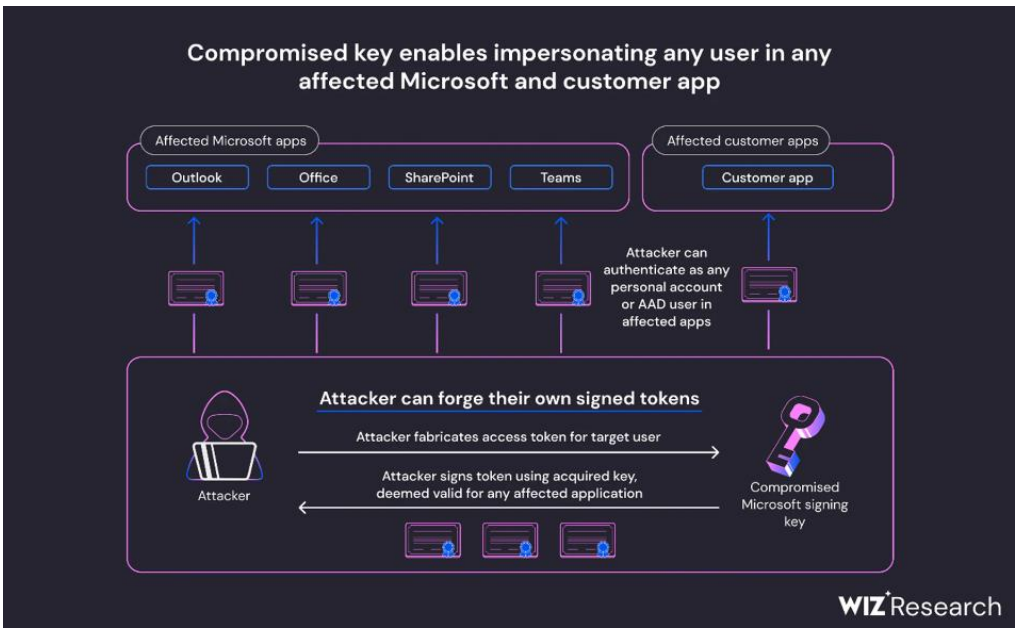


Bild 60: Angriff auf Azure AD – Ablauf [274]

4.1.5. Logging & Auswertungsmöglichkeiten

Access bietet von Haus aus keine automatisierten, manipulationssicheren Logging- oder Überwachungsfunktionalitäten, Abfragepläne oder Transaktionslogs, um Zugriffe auf das DBS, -dienste, Relationen oder ähnliche Objekte sowie Nutzer-Aktivitäten detailliert

zu loggen. Die unter dem Pfad in der „*TEMP*“-Systemumgebungsvariable des Windows-Betriebssystems abgelegte Datei mit protokollierten Transaktionen konnte nicht gefunden werden (*siehe Kapitel „4.1.1 SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen“*).

Aufgrund fehlender Sicherheit auf Benutzerebene können Aktivitäten nicht eindeutig konkreten Nutzenden zugeordnet werden.

Ähnlich wie die in *Kapitel „4.1.6 Berechtigungen & Autorisierung“* beschriebenen Möglichkeiten den Zugriff über lesende und schreibende Rechte des Dateisystems zu steuern, kann zur Überwachung auch über „*Rechtsklick → Eigenschaften → Sicherheit → Erweitert → Überwachung*“ auf Funktionalitäten des Datei- und Betriebssystems zurückgegriffen werden. Zur Analyse des Funktionsumfangs sowie der Möglichkeiten, ob und wie die Zugriffe und Zugänge über das Dateisystem auf die Datei jederzeit zweifelsfrei einer handelnden Person idealerweise automatisiert zuordbar sind, bedarf es einer separaten Analyse:

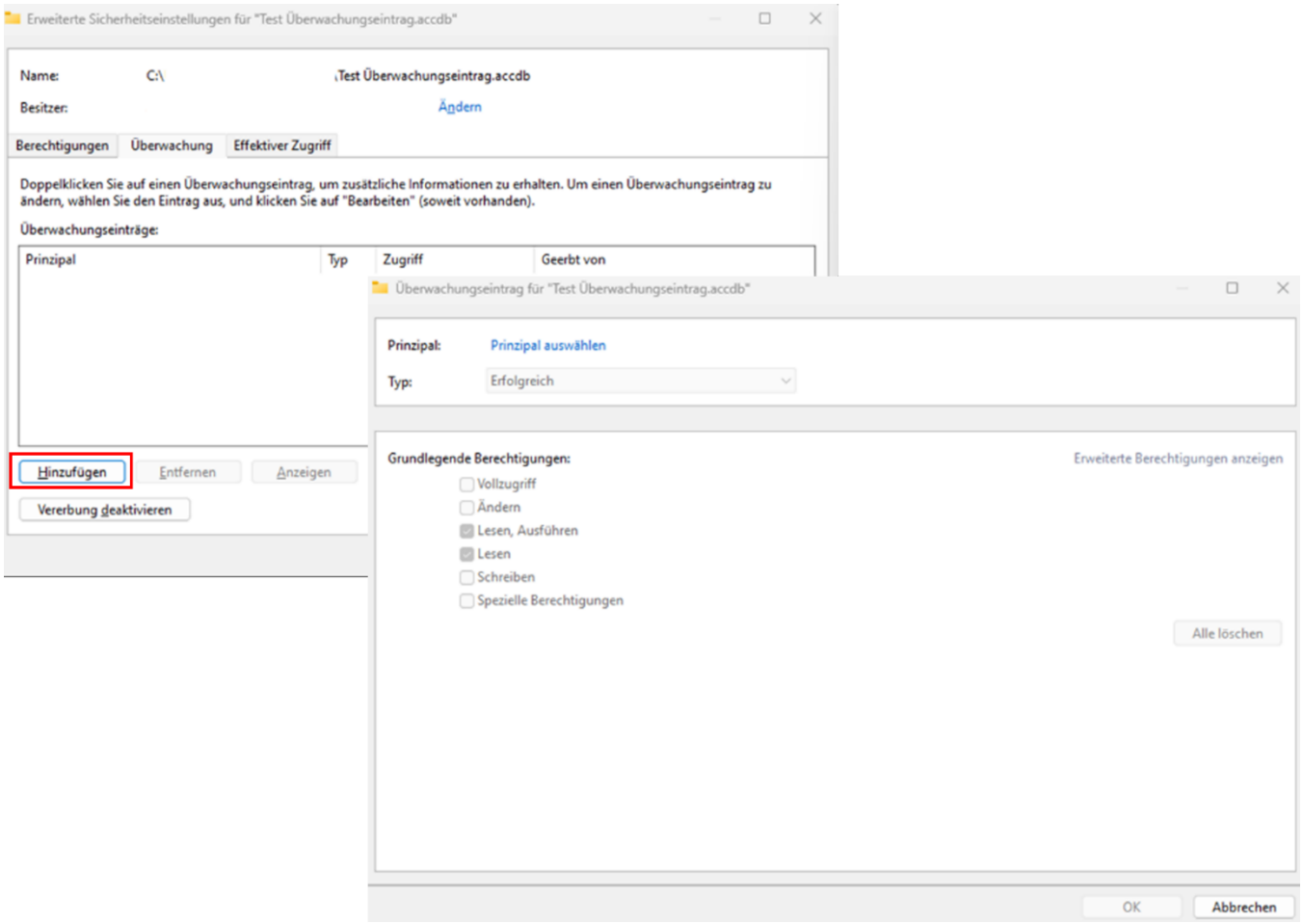


Bild 61: Überwachungsfunktionalität auf Datei- und Ordnerebene im NTFS-Dateisystem – Hinzufügen eines Überwachungseintrags

Zur Funktionserweiterung müsste in Access auf (VBA-)Eigenimplementierungen zurückgegriffen werden. Hierfür kann nach dem Öffnen einer Tabelle unter dem Kopfreiter „Tabelle“ auf der Access-Oberfläche ein Vorab- oder Nachfolgeereignis in Form der

bereits beschriebenen Makros angelegt werden, die bei unterschiedlichen Ereignissen, wie dem Einfügen von Tupeln in die Relation, automatisch ausgeführt werden. Die Auswahl der Ereignisse erfolgt über den Aktionskatalog. Durch Auswahl der Aktion „*ProtokollierenEreignis*“ wird nach auslösen des Ereignisses die Systemrelation „*USysApplicationLog*“ erstellt, wenn sie nicht schon existiert, und mit einem zugehörigen Datensatz befüllt. Die Relation wird auch automatisch angelegt, wenn in der Applikation ein Fehler auftritt. In *USysApplicationLog* werden die Fehler der Datenmakros und alle Aufrufe der Aktion „*AuslösenFehler*“ protokolliert. Dem Datensatz kann auch ein frei wählbarer Kommentar hinzugefügt werden, um ein benutzerdefiniertes Log zu erstellen [120]/[144]:

ID	SourceObject	Data Macro Instance ID	Error Number	Category	Object Type	Description
1	T_Mitarbeiter.AfterInsert	{D7466EF2-5AA2-4FC7-93C9-F061E631086D}	1	User	Macro	Datensatz wurde eingefügt

Bild 62: „USysApplicationLog“ nach „NachEinfügung“-Ereignis

Mit der „*SendeEmail*“-Makroaktion können in Datenmakros E-Mails mit beliebigem Inhalt versendet werden [147]. Zusätzlich kann über die Aktion „*DatensatzErstellen*“ und dem Ereignis „*NachEinfügen*“ eine Kopie des eingefügten Datensatzes als Änderungshistorie in eine Access-Relation geschrieben und somit eine Backup-Funktionalität implementiert werden [275, S. 849-850]:

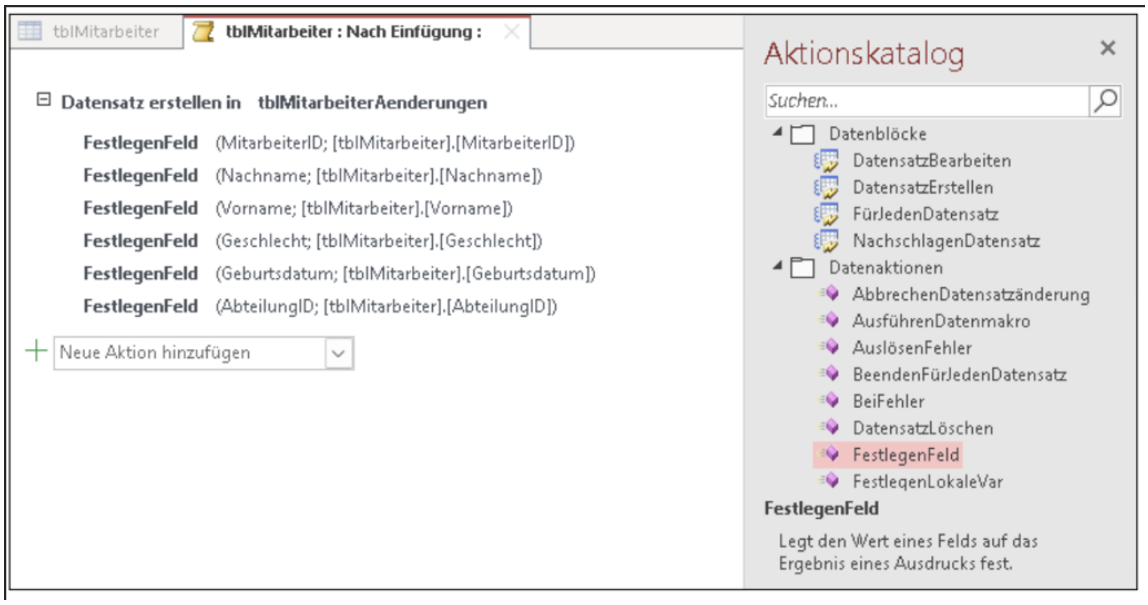


Bild 63: Beispiel für „NachEinfügen“-Ereignis zum Kopieren eines Datensatzes mittels DatensatzErstellen-Aktion [275, 849-850]

Beim Datentyp „*Langer Text*“ kann außerdem über die Feldeigenschaften des jeweiligen Attributs über die Option „Nur anfügen“ eine Änderungsprotokollierung aktiviert werden. In diesem Fall kann über „*Rechtsklick auf den jeweiligen Attributwert eines Tupels → Spaltenverlauf anzeigen...*“ eine Protokollierung aktiviert werden [275, S. 63-66]:

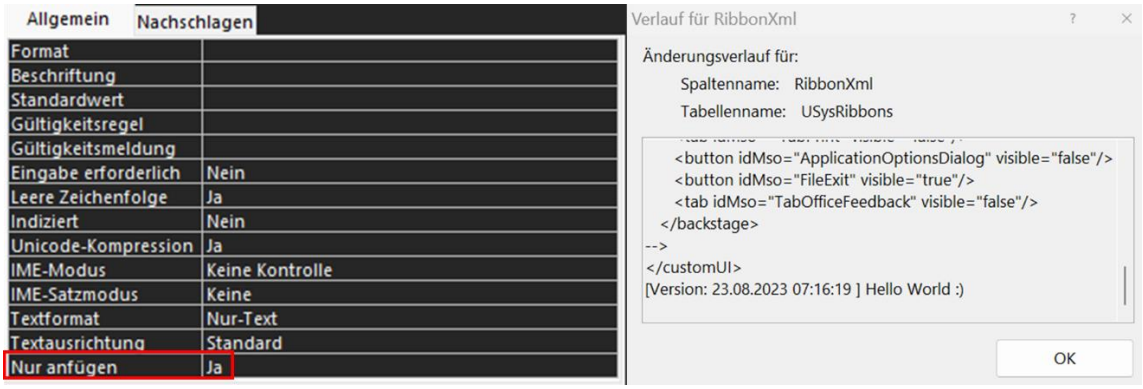


Bild 64: Feldeigenschaften für Attribut mit Datentyp „Langer Text" – Aktivierung der Protokollierungsfunktion „Nur anfügen" & Anzeige des Änderungsverlaufs für einen Attributwert („Rechtsklick auf Datensatz → Spaltenverlauf anzeigen…") [275, S. 63-66]

Bei allen zuvor genannten Access-Protokollierungsfunktionalitäten ist zu beachten, dass sie genauso wie VBA-Eigenimplementierungen oder Makros nicht ausgeführt werden, wenn auf die jeweilige Relation über ein Drittanbietertool wie DbVisualizer und nicht über die Access-Oberfläche zugegriffen wird. Derartige (Protokollierungs)Funktionalitäten sind also nicht belastbar und funktionieren nur unzuverlässig. Außerdem ist zu beachten, dass eine durch VBA-Eigenentwicklungen erstellte Log-Datei an zentraler Stelle abgelegt werden muss, auf die alle Nutzenden Schreibzugriff haben, da Access-Anwendungen unter dem jeweils angemeldeten Windows-User ausgeführt werden. Dieser Umstand macht eine Log-Datei zusätzlich anfällig für Manipulationen, wenn die Log-Datei nicht in einem Client-Server-DBS mit Sicherheit auf Benutzerebene ausgelagert oder gespiegelt wird. Ist das Passwort zur Entschlüsselung der Access-Datenquelle bekannt, besteht danach Vollzugriff auf den Inhalt im Klartext, daher sind auch in der Access-Datei gespeicherte Logs nicht zwangsläufig vor Manipulationen geschützt. Die Access-Systemtabellen enthalten kaum wertvolle Informationen für eine Auswertung, denn sie werden von Access für interne Verwaltungszwecke verwendet.

Beim Öffnen einer Access-Datei (.accdc, .accde) wird eine Sperrdatei (bei einer .accdb-Datei .laccdb) angelegt oder bei mehreren Nutzenden jeweils erweitert. Da die Sperrdatei wieder gelöscht wird, wenn die letzte Nutzende die Datei schließt, werden im Rahmen der ordnungsgemäßen Nutzung von Access Lese-, Schreib-, Erstellungs- und Löschberechtigungen zumindest auf den Ordner benötigt, in dem die Datei liegt (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung"*). In der Sperrdatei sind für alle Nutzenden jeweils der zugehörige Computername, der Sicherheitsname (bei .accdb immer „`Admin`") sowie die jeweils gesperrten Datensätze in Form von Bytesperren hinterlegt (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung"*). Mit Bytesperren werden Teile von Dateien gesperrt, die von mehreren Anwendungen genutzt werden. Den Prozessen wird exklusiver Zugriff auf den angegebenen Bytebereich in der Datei gegeben. Andere Prozesse können nicht auf einen exklusiv geöffneten Bytebereich zugreifen. Die Sperrdatei wird nicht gelöscht, wenn Nutzende über keine Löschrechte verfügen oder die DB als beschädigt gekennzeichnet ist. Aus der Sperrdatei kann dann entnommen werden, wer die DB zum Zeitpunkt der Beschädigung verwendet hat [105]/[148].

Darüber hinaus kann auch das „EventLog“ vom Windows-Betriebssystem oder die „Computerverwaltung“ hilfreich sein. Unter „*Computerverwaltung → Freigegebene Ordner → Geöffnete Dateien*“ kann auf Endgeräten mit Windows-Betriebssystem, wie einem Windows Server, eine Liste mit aktuell geöffneten und freigegebenen Dateien eingesehen werden, einschließlich durch wen der Zugriff erfolgt und Informationen zum Öffnungs-Modus. Um nicht den Rahmen dieser Ausarbeitung zu überschreiten, werden diese Auswertungsmöglichkeiten nicht weiter berücksichtigt und anstelle dessen auf eine separate Analyse verwiesen [4]/[170].

Im Unterschied zu Access bietet die Azure SQL-DB proaktive, automatisierte Sicherheits- und Überwachungsfunktionalitäten in Form von „Threat Protection“ (Bedrohungserkennung) und Auditing-Funktionalität wie der „Azure SQL-Überwachung“. Mit der Azure SQL-Überwachung kann der Server oder einzelne DBs überwacht werden. Dabei werden eintretende Ereignisse (Events) in einem Überwachungsprotokoll wie dem Aktivitätsprotokoll in „Log Analytics“ gespeichert, das dann weiter ausgewertet werden kann. Es können auch Support-Vorgänge von MS geloggt werden. Mit den automatisch erstellten Audit-Logs können regulatorische und gesetzliche Vorschriften eingehalten, Aktivitäten im DBS oder auf dem Server nachverfolgt und Änderungen in der Konfiguration, Sicherheitsverletzungen oder Anomalien erkannt werden [1, S. 239, S. 285]/[51, S. 64, S. 104]/[86]:

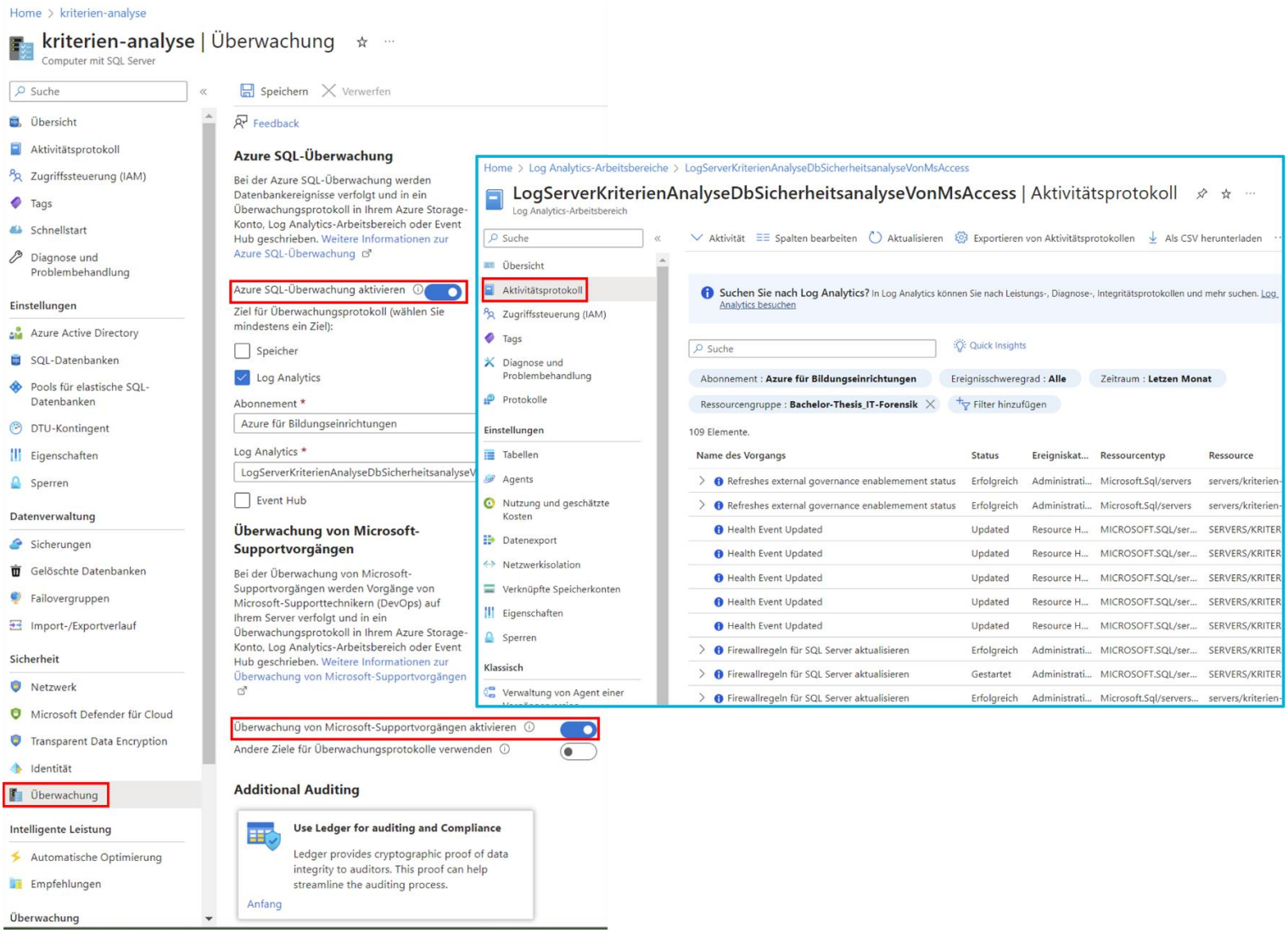


Bild 65: Links: Aktivierung der automatischen Auditing-Funktionalität „Azure SQL-Überwachung"; Rechts: Log Analytics-Arbeitsbereich, Anzeige des Aktivitätsprotokolls mit allen Aktivitäten wie Aktivierungen von Firewall-Regeln, Alarm zu potenziellen SQL-Injection-Angriffen oder von MS ausgelösten Ereignissen

Die „erweiterte Bedrohungserkennung" (Advanced Threat Protection) sowie konfigurierbare „Sicherheitsrisikobewertungen" (Vulnerability Assessment) setzen auf der Azure SQL-Überwachung auf und sind in den „MS Defender für die Cloud" integriert [122]/[149]/[196]. Die Azure SQL-Überwachung muss aktiviert sein, damit die Ereignisse in das Audit-Log geschrieben werden. Die Advanced Threat Protection bietet zusätzlichen Schutz für die Daten sowie für die gehosteten Ressourcen wie Applikationen und DBSs. Advanced Threat Protection erkennt mittels KI potenziell schädliche Ereignisse und Angriffe, mit denen Angreifende versuchen Zugriff zum DBS zu erhalten oder die DB zu beschädigen. Administratoren können mittels konfigurierbarem Alarm über derartige Anomalien, wie einen SQL-Injection-Angriff oder andere benutzerdefinierte Ereignisse, benachrichtigt werden (*vergleiche auch weiter unten „Azure Monitor Alerts"*). Die im *Kapitel „4.1.1 SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen"* über die in C# und dem Razor-Framework programmierte Demo-Web-

Applikation durchgeführten SQL-Injection-Angriffe werden im MS Defender für die Cloud als Sicherheitswarnung mit relevanten Details aufgelistet. Es werden weitere Aktionen, wie Best Practices zur Schließung der Schwachstelle und Erhöhung der Sicherheit oder das Einsehen der zugehörigen Log Analytics-Protokolle, angeboten:

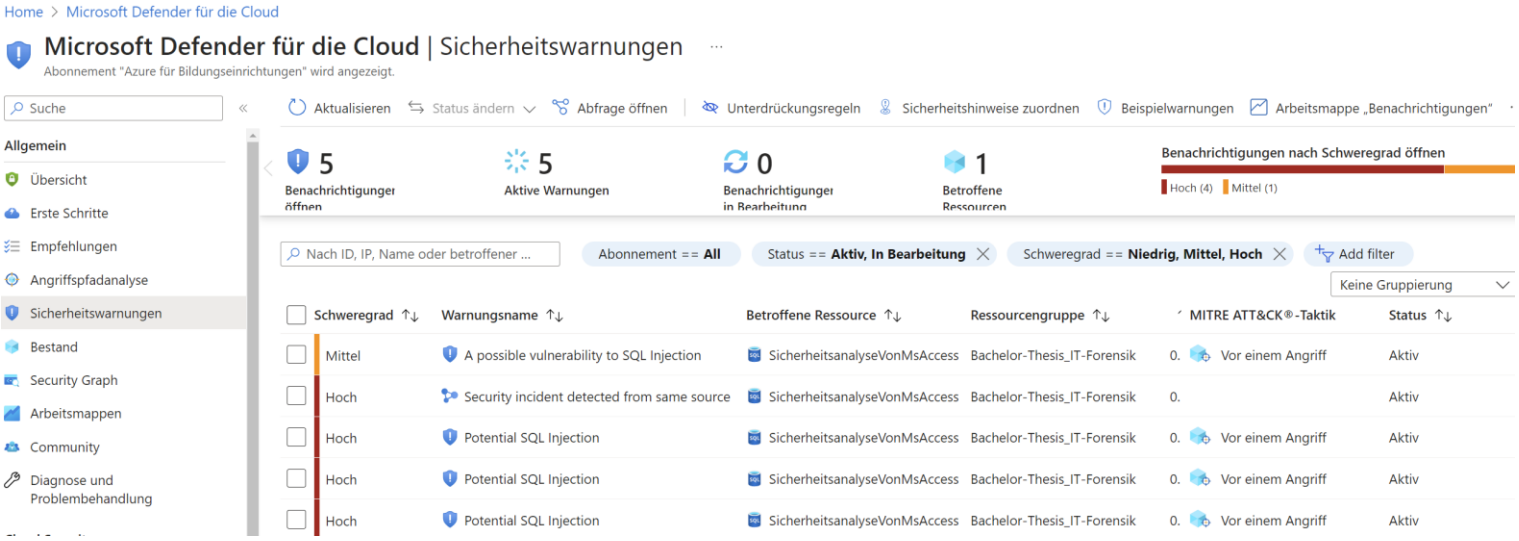


Bild 66: MS Defender für die Cloud – Sicherheitswarnungen für erkannte SQL-Injection-Angriffe (Übersicht aller Ereignisse) 1/4

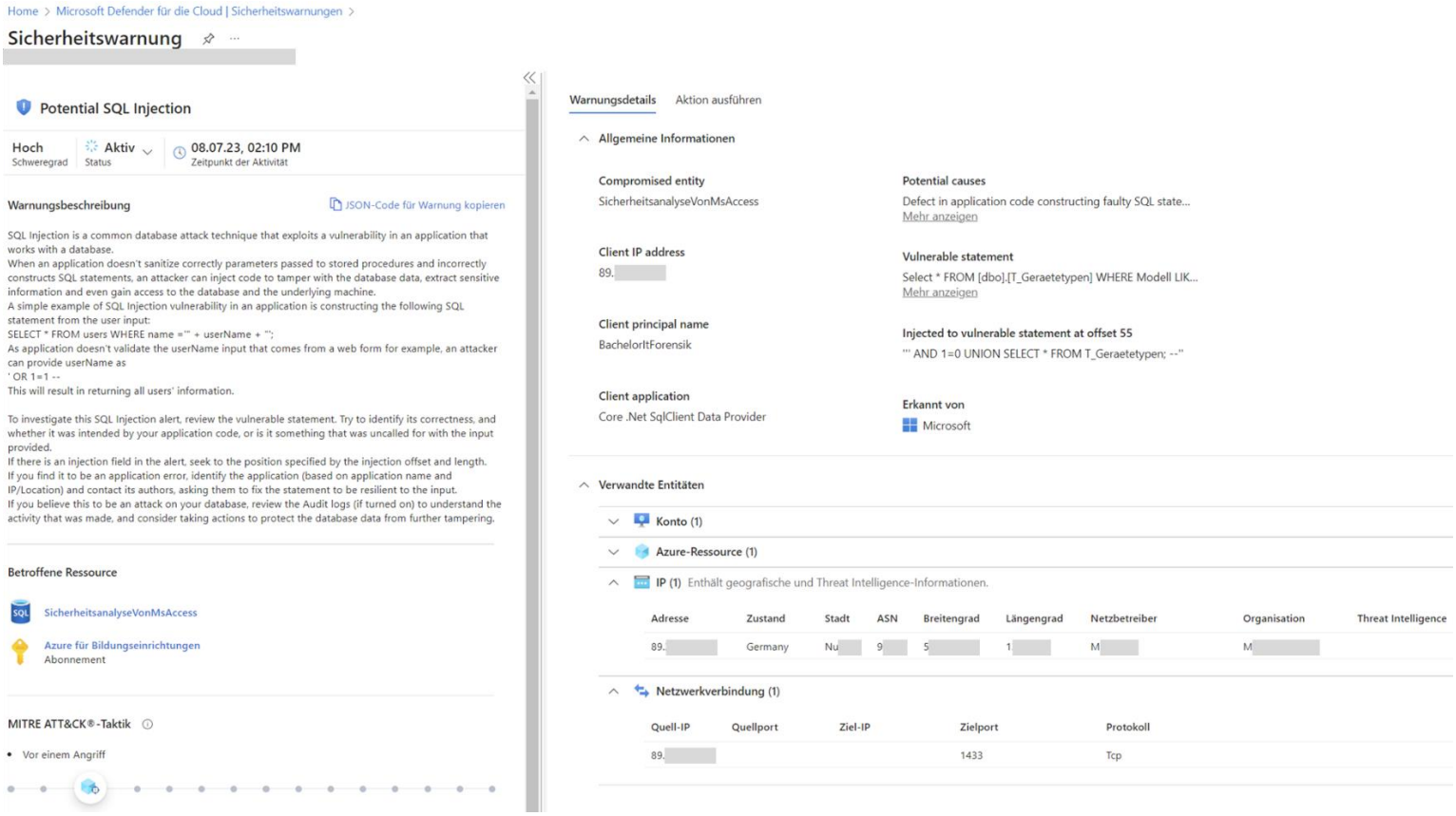


Bild 67: MS Defender für die Cloud – Sicherheitswarnungen für erkannte SQL-Injection-Angriffe (Detailseite eines ausgewählten Ereignisses mit diversen Angaben wie SQL-Statement oder IP-Adresse) 2/4

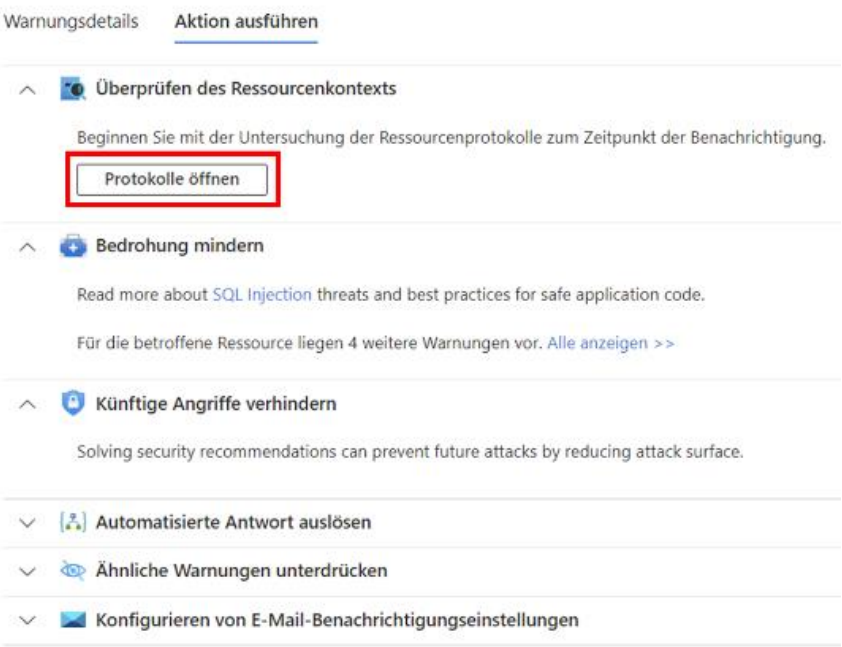


Bild 68: MS Defender für die Cloud – Sicherheitswarnungen für erkannte SQL-Injection-Angriffe (Ausführbare Aktionen für das zuvor ausgewählte Ereignis inklusive Einsicht zugehöriger Protokolle) 3/4

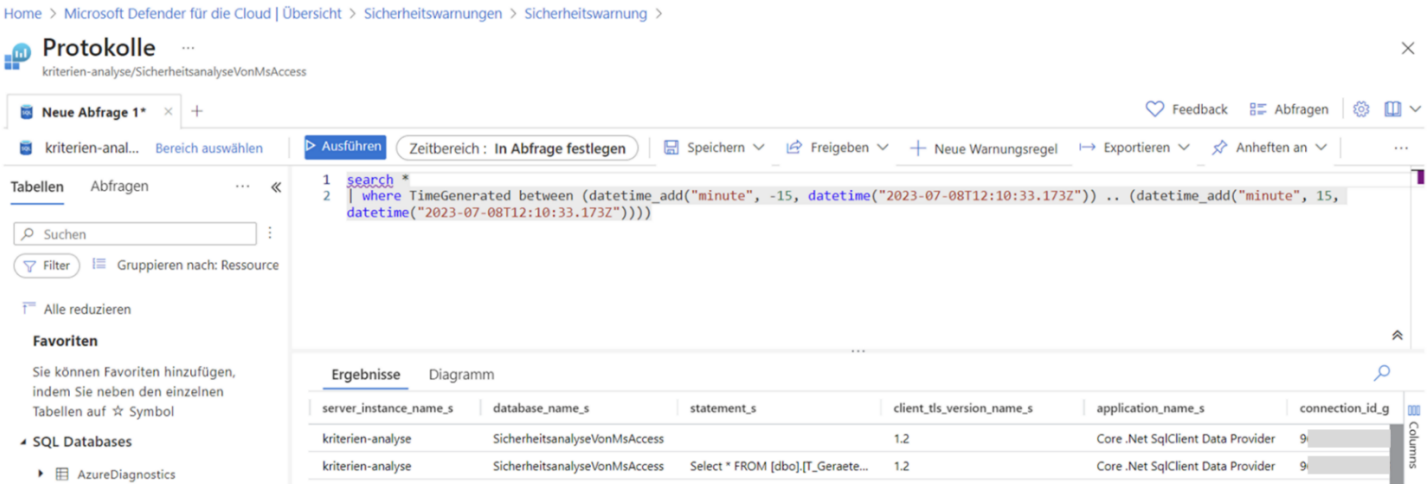


Bild 69: MS Defender für die Cloud – Sicherheitswarnungen für erkannte SQL-Injection-Angriffe (Einsicht des gefilterten Protokolls im zugehörigem „Log Analytics-Arbeitsbereich" mit vielen weiteren Attributen wie zugehöriger Host-Name, Applikation-Name und zugehörige Client-IP-Adresse) 4/4

Alle in einem Log Analytics-Arbeitsbereich gespeicherten Protokolle werden standardmäßig und ohne zusätzliche Kosten 30 Tage von MS aufbewahrt und dann gelöscht. Gegen einen Aufpreis kann die Aufbewahrungsfrist auf 730 Tage erhöht werden. Wird eine längere Speicherfrist benötigt, können die Protokolle archiviert und hiermit die Gesamtaufbewahrungszeit auf 7 Jahre erhöht werden [115]/[116]:

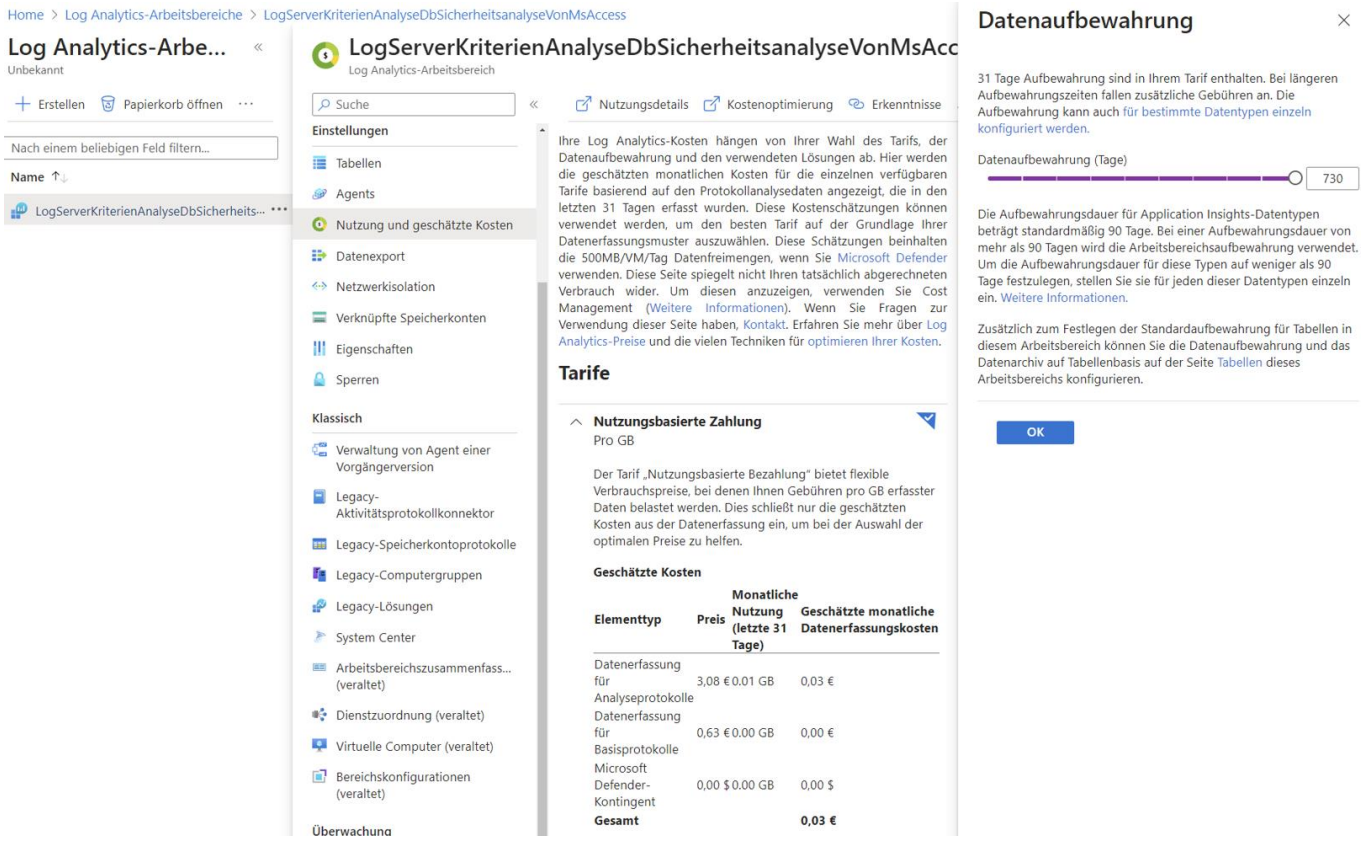


Bild 70: Konfiguration der Datenaufbewahrung von in Log Analytics-Arbeitsbereichen gespeicherten Logs

Darüber hinaus ist es auch möglich mittels „Azure Monitor" Cloud- und On-Premises-Dienste wie DBs oder Applikationen zu überwachen. Die gesammelten Daten können zur weiteren Auswertung mit diversen Werkzeugen wie Log Analytics analysiert und ausgewertet werden. Es können auch zusätzliche Tools von Drittanbietern eingebunden und die gesammelten Daten exportiert werden, um die Auswertung außerhalb des Azure Monitors vorzunehmen [129]/[176]:

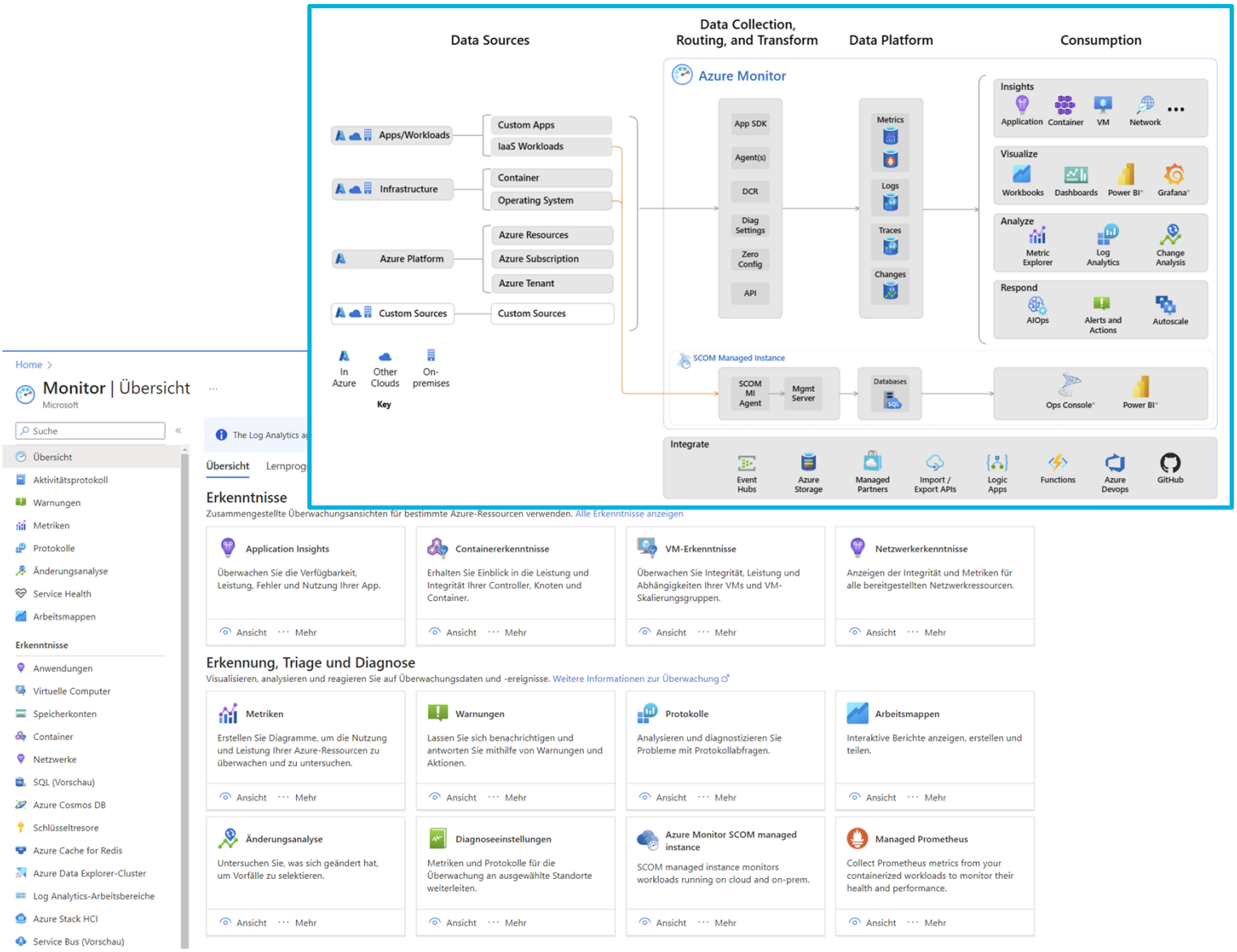


Bild 71: Oben: Architektur des Azure Monitors [176]; Unten: Azure Monitor-Startseite

Ein Alarm (Alert) kann für alle Log-Daten und Metriken im Azure Monitor konfiguriert werden. Sobald etwas in der überwachten Ressource wie einer Azure SQL-DB passiert, wird von der „Alert Rule“ geprüft, ob das Signal die konfigurierte Bedingung erfüllt. Falls ja, wird ein Alarm ausgelöst, der die durchzuführende Aktion anstößt [167]:

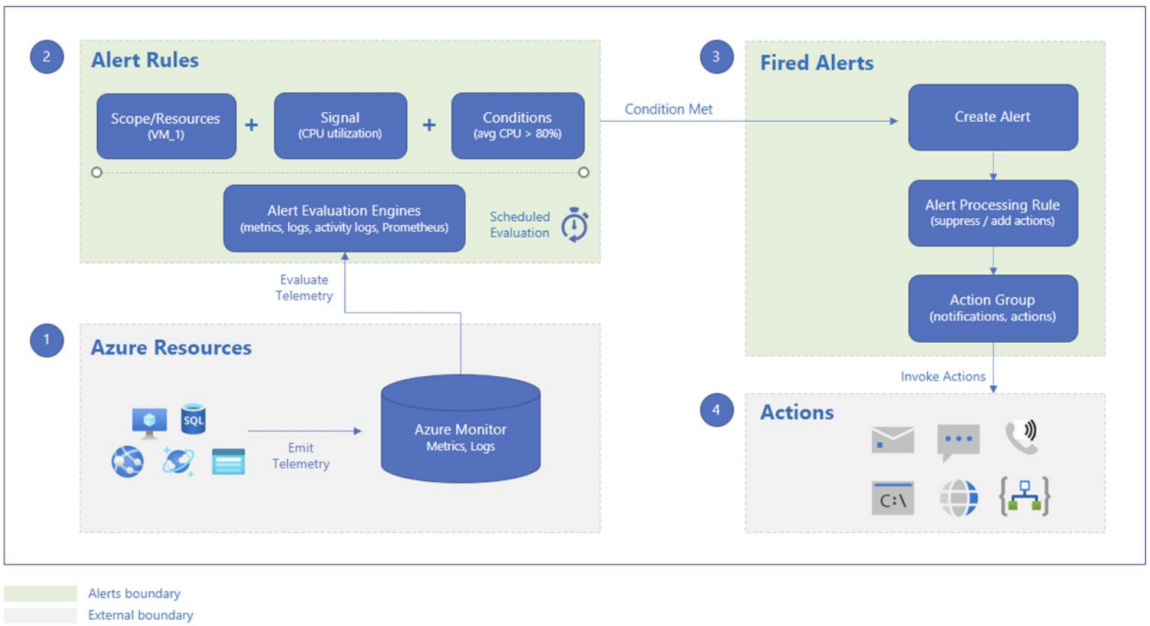


Bild 72: Funktionsweise eines Azure Monitor Alerts [167]

Es stehen außerdem Transaction Logs zur Verfügung. Hier sind alle durchgeführten Transaktionen und Modifikationen enthalten. Daher wird das Transaction Log verwendet, um den vorherigen konsistenten Zustand wiederherzustellen, wenn ein unerwartetes Ereignis wie ein „*ROLLBACK*“ eintritt. Der Zugriff erfolgt dabei über teilweise undokumentierte Funktionen wie „*sys.dbLog()*“ zur Anzeige des aktiven Teils der Tranaction Log-Datei der jeweiligen DB. Über die dokumentierte Relation „*sys.dm_db_log_info*“ können Informationen über die „Virual Log Files“ in die das Transaction Log aufgesplittet ist, eingesehen werden. Da keine Zugriffe auf das Betriebssystem möglich sind (*siehe Kapitel „4.1.1 SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen“*) und MS Interna geheim hält, kann abschließend nicht beantwortet werden, ob die Transaction Logs und DB-Dateien auf unterschiedlichen Festplatten gespeichert werden [72]/[151]/[227]:

Run Cancel Disconnect Change Database: SicherheitsanalyseVonM... Estimated Plan Enable Actual Plan Parse

```
1  SELECT * FROM sys.databases;
2  SELECT * FROM sys.dm_db_log_info (2);
3
4  SELECT * FROM sys.fn_dblog(NULL, NULL);
```

Results Messages

	database_id	file_id	vlf_begin_offset	vlf_size_mb	vlf_sequence_number	vlf_active	vlf_status	vlf_parity
1	2	2	8192	3,93	51	0	0	128
2	2	2	4136960	3,93	52	0	0	128
3	2	2	8265728	3,93	53	1	2	128
4	2	2	12394496	4,17	50	0	0	64

	Current LSN	Operation	Context	Transaction ID	LogBlockGeneration	Tag Bits	Log Record Fixed Lengt
1	00000043:00000398:0003	LOP_INSERT_ROWS	LCX_CLUSTERED	0000:00001f83	0	0x0000	62
2	00000043:00000398:0004	LOP_SET_BITS	LCX_PFS	0000:00000000	0	0x0000	54
3	00000043:00000398:0005	LOP_INSERT_ROWS	LCX_INDEX_LEAF	0000:00001f83	0	0x0000	62
4	00000043:00000398:0006	LOP_PREP_XACT	LCX_NULL	0000:00001f83	0	0x0000	64
5	00000043:000003A0:0001	LOP_COMMIT_XACT	LCX_NULL	0000:00001f83	0	0x0000	80
6	00000043:000003A8:0001	LOP_BEGIN_XACT	LCX_NULL	0000:00001f84	0	0x0000	76
7	00000043:000003A8:0002	LOP_MODIFY_COLUMNS	LCX_CLUSTERED	0000:00001f84	0	0x0000	62
8	00000043:000003A8:0003	LOP_MODIFY_COLUMNS	LCX_CLUSTERED	0000:00001f84	0	0x0000	62

Bild 73: Anzeigen von Informationen zu den Virtual Log Files (sys.dm_db_log_info) des Transaction Logs sowie des aktiven Transaction Log-Inhalts (sys.dblog())

Der „*DBCC Loginfo*“-Befehl für weitere Details zu den Virtual Log Files des Transaction Logs wird nicht unterstützt [250]:

Run Cancel Disconnect Change Database: SicherheitsanalyseVonM... Estimated Pla

```
1  DBCC LOGINFO(logspace);
```

Messages

```
15:15:17    Started executing query at Line 1
            Msg 40518, Level 16, State 1, Line 1
            DBCC command 'LOGINFO' is not supported in this version of SQL Server.
            Total execution time: 00:00:00.025
```

Bild 74: Keine Unterstützung des DBCC LOGINFO (logspace)-Befehls in Azure SQL-DB

Für die Analyse stehen drei unterschiedliche Arten von Ausführungsplänen zur Verfügung. Ausführungspläne werden vom „Abfrageoptimierer“ der DB-Engine während der Ausführung von Anweisungen im Rahmen der Analyse erstellt. Ausführungspläne stellen die interne Vorgehensweise dar, wie gemäß übergebenem SQL-Statement am effizientesten auf die Daten zugegriffen und die zugehörige Verarbeitung erfolgen kann. Ein Ausführungsplan beinhaltet Punkte wie die Reihenfolge der Zugriffe auf die Quellrelationen, die verwendeten Methoden, um die Daten aus den einzelnen Relationen zu extrahieren (je nach Abfrage und DB-Größe kann die Suche nach einem Tabellenindex oder eine indexbasierte Suche effizienter sein) oder die verwendeten Methoden zum Filtern, Aggregieren und Sortieren der Daten aus den einzelnen Relationen. Damit der Abfrageoptimierer den Ausführungsplan erstellen kann, benötigt er als Input das jeweilige DB-Schema (Relations- und Indexdefinitionen) sowie diverse DB-Statistiken. Wird eine bisher unbekannte Abfrage dem Abfrageoptimierer übergeben, erstellt er einen oder mehrere Abfragepläne. Dabei wählt der Abfrageoptimierer nach Kriterien, wie Kompilierungszeit oder Planoptimalität, den wirtschaftlichsten Abfrageplan aus. Die erstellten Ausführungspläne werden zur Wiederverwendung im Arbeitsspeicher abgelegt

(Plancache) und während der Ausführung des SQL-Befehls von den beteiligten Komponenten verwendet.

Nach einem erfolgreichen Angriff können dem Ausführungsplan konkrete Details zu den beim Angriff ausgeführten Befehlen, einschließlich auskommentierte Teile, und betroffene Objekte, entnommen werden. Ein Indiz für blind SQL-Injection-Angriffe können viele kurz hintereinander erstellte und im Plancache gespeicherte Abfragepläne sein. Bei einem blinden SQL-Injection-Angriff wird aufgrund einer allgemeingehaltenen Fehlerseite und fehlenden Details versucht weitere Informationen zu sammeln. Ein mögliches Einfallstor für SQL-Injection-Angriffe kann zum Beispiel angenommen werden, wenn eine ungültige SQL-Abfrage, wie die Eingabe eines einzelnen Anführungszeichens in das Eingabe-Steuerelement der jeweiligen Webseite, zur Anzeige einer allgemein gehaltenen Fehlerseite führt. Das Vorhandensein einer SQL-Injection-Schwachstelle könnte im Rahmen eines blinden SQL-Injection-Angriffs auch durch eingeschleuste Verzögerungen wie „*WAITFOR DELAY*“ erfolgen.

Auch die unlogische Verwendung von bedingten Anweisungen wie „*1 = 1*“ kann als potenzieller Indikator für einen SQL-Injection-Angriff aus Abfrageplänen abgelesen werden [71, S. 247, S. 468-474]/[85]/[118]/[156]:

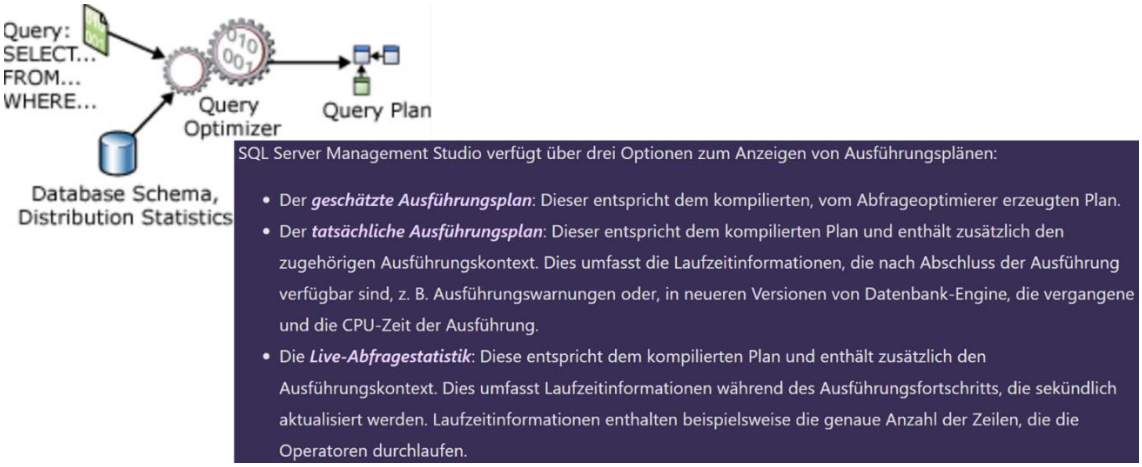


Bild 75: Links: Ein- und Ausgaben des Abfrageoptimierers [118]; Rechts: Einsehbare Arten an Ausführungsplänen [118]

Der Zugriff auf die Abfragepläne und zugehörige Informationen kann über den „Query Plan Viewer“ im Azure Data Studio erfolgen, der die geschätzten und tatsächlichen Abfragepläne in visualisierter Form darstellt. Auch eine Extensible Markup Language (XML)-Ansicht ist über das Kontextmenü „Show Query Plan XML“ genauso wie ein Vergleich mit anderen Abfrageplänen vorhanden [142]:

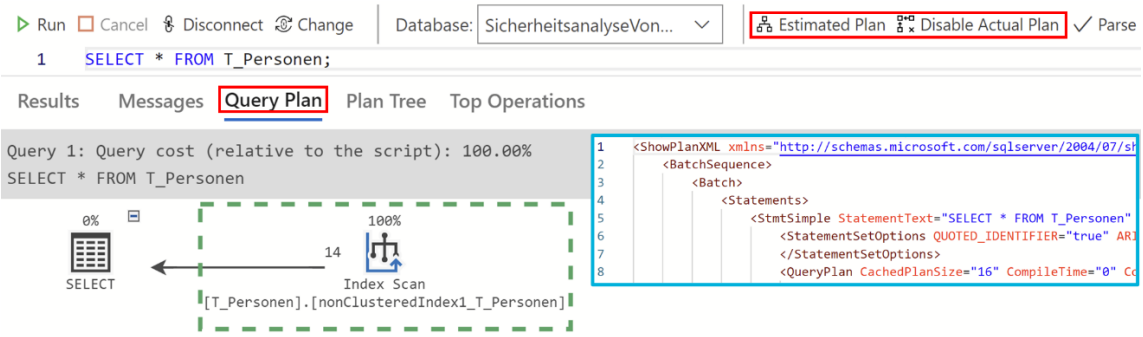


Bild 76: „Query Plan Viewer“ im Azure Data Studio [142]

Alternativ kann die Einsicht der Abfragepläne und zugehöriger Informationen über diverse DMVs und DMFs erfolgen. DMVs und DMFs enthalten verschiedene Statusinformationen, die zur Überwachung des Servers und der DB-Instanz, zur Problemdiagnose und zu Optimierungszwecken verwendet werden können. Sie sind im „*sys*“-Schema zu finden und beginnen mit „*dm_*“. So finden sich in „*sys.dm_exec_sessions*“ alle authentifizierten Sitzungen mit zugehörigen Informationen wie Client-Programmname, Login-Zeitstempel, Login-User und mit der Session zusammenhängende interne Tasks der Azure SQL-DB. MS empfiehlt auf die Verwendung dieser Views und Funktionen im Quellcode zu verzichten, da sich der Aufbau dieser Views stets in nachfolgenden Versionen ändern kann.

Über die „Execution Related DMVs and DMFs“ „*sys.dm_exec_query_stats*“ und „*sys.dm_exec_sql_text*“ können weitere Informationen zu den Ausführungsplänen eingeholt werden. *sys.dm_exec_query_stats* enthält Daten wie aggregierte Performance-Indikatoren für alle Einzelanweisungen von allen im Plancache gespeicherten Ausführungsplänen sowie die ID des SQL-Handle, über die mit *sys.dm_exec_sql_text* das zugehörige SQL-Statement ermittelt werden kann und, ob das SQL-Statement verschlüsselt ist. Aber auch in „*sys.dm_exec_query_plan*“ (enthält je gespeicherten Abfrageplan eine Zeile mit diversen Informationen wie den Text oder die Anzahl der Wiederverwendungen) und „*sys.dm_exec_query_plan*“ sind Informationen zu gespeicherten Abfrageplänen zu finden. So können mit folgendem SQL-Statement alle aktuell im Plancache gespeicherten Abfragepläne im XML-Format mit weiteren Details eingesehen werden:

```
SELECT *
FROM sys.dm_exec_cached_plans AS cp
CROSS APPLY sys.dm_exec_query_plan(cp.plan_handle);
```

Sicherheitsrelevante DMVs und DMFs stellen Objekte wie „*sys.dm_database_encryption_keys*“ dar. Über *sys.dm_database_encryption_keys* können Informationen zur verwendeten DB-Verschlüsselung im Rahmen der TDE (*siehe Kapitel „4.1.3 Kryptographie“*) ermittelt werden. Dazu zählen die ID der DB, Status der DB-Verschlüsselung wie „unverschlüsselt“ oder „verschlüsselt“, der für die DB-Verschlüsselung verwendete Algorithmus inklusive der zugehörigen Schlüssellänge [71, S. 474]/[153]/[154]/[225]:

```
SELECT * FROM sys.dm_exec_query_stats;

SELECT * FROM sys.dm_exec_sql_text(0x06000500A8F9CD0570D2A3174B020000010000000000000000000000000000000000000000000000000000);
```

Results | Messages

	sql_handle	statement_start_offset	statement_end_offset	plan_gene
1	0x0300FF7FFE084FC103A9CC0039B000000100000000000000000000…	438	586	1
2	0x0300FF7F3A6E05E803A9CC0039B000000100000000000000000000…	420		
3	0x0300FF7F99F756F08528CC0039B000000100000000000000000000…	1242		
4	0x0300FF7F99F756F08528CC0039B000000100000000000000000000…	5234		
5	0x0300FF7F99F756F08528CC0039B000000100000000000000000000…	1242		
6	0x0300FF7F99F756F08528CC0039B000000100000000000000000000…	5234		
7	0x0300FF7F417A40D005A9CC0039B000000100000000000000000000…	860		
8	0x0300FF7F417A40D005A9CC0039B000000100000000000000000000…	1122		
9	0x0300FF7F417A40D005A9CC0039B000000100000000000000000000…	1478		
10	0x0300FF7F417A40D005A9CC0039B000000100000000000000000000…	1864		

	dbid	objectid	number	encrypted	text
1	5	NULL	NULL	0	SELECT * FROM sys.dm_exec_sql_text(0x0500FF7FFE084FC1809D…

```
SELECT database_id,
       encryption_state,
       key_algorithm,
       key_length
FROM sys.dm_database_encryption_keys;
```

Results | Messages

	database_id	encryption_state	key_algorithm	key_length
1	2	3	AES	256
2	5	3	AES	256

Bild 77: Links: Über „sys.dm_exec_query_stats“ und „sys.dm_exec_sql_text“ ermittelte Informationen wie Ausführungshäufigkeit aller SQL-Anweisungen, die jeweils in allen im Plancache gespeicherten Ausführungsplänen enthalten sind; Rechts: Über „sys.dm_database_encryption_keys“ ermittelte Informationen zur verwendeten DB-Verschlüsselung

Über den Dienst „Query Performance Insight“ können Abfragen über das Azure-Portal weiter analysiert werden, um hier beispielsweise besonders rechenintensive Statements zu ermitteln und Empfehlungen zur Optimierung zu erhalten [141]:

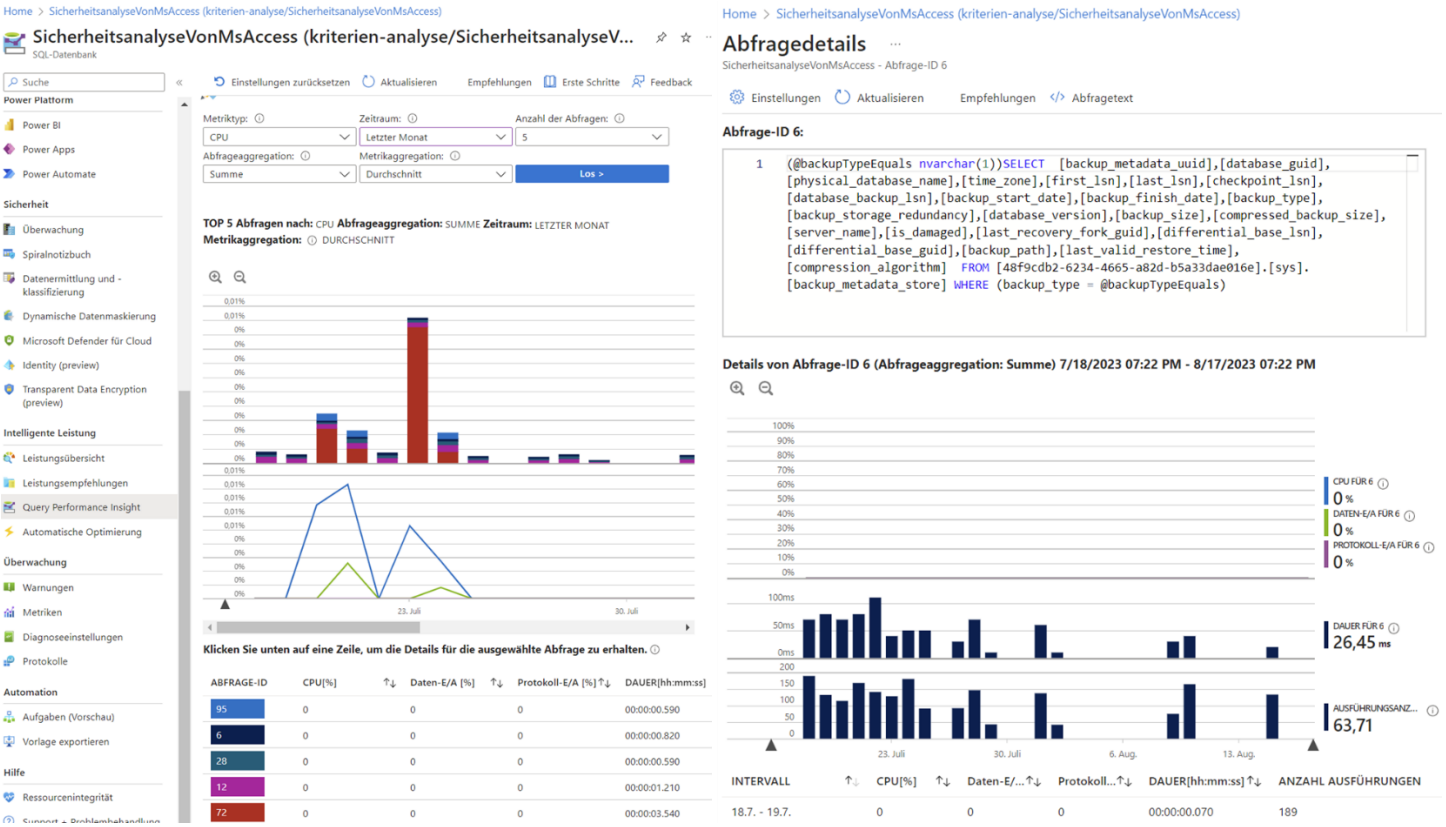


Bild 78: Links: Query Performance Insight-Funktionalität im Azure-Portal; Rechts: Abfragedetails zur Abfrage-ID 6

Darüber hinaus können alle Änderungen (`INSERT`, `UPDATE`, `DELETE`) an Datensätzen und Relationen über die „Change Data Capture“-Funktionalität von Azure SQL-DB

nachverfolgt werden. Quelle der Change Data Capture-Funktionalität ist das Transaction Log. Die Change Data Capture-Funktion liest neue Datensätze aus dem Transaction Log aus und ergänzt sie mit Zusatzinformationen über die konkreten Änderungen sowie betroffene Relationen. Über angebotene Funktionen können Nutzende in gefilterter Form auf die in den Änderungstabellen gespeicherten Einträge zugreifen. Die Change Data-Capture-Funktionalität muss mit der Stored Procedure „`sys.sp_cdc_enable_db`“ explizit aktiviert werden.

Log Shipping zum Weiterleiten von Log-Daten an einen Log-Server werden von der Azure SQL-DB nicht unterstützt, da in der Azure-Cloud andere Funktionalitäten und Dienste zur Aufrechterhaltung des Systembetriebs und der Wiederherstellung zur Verfügung stehen [1, S. 30-33]/[168]:

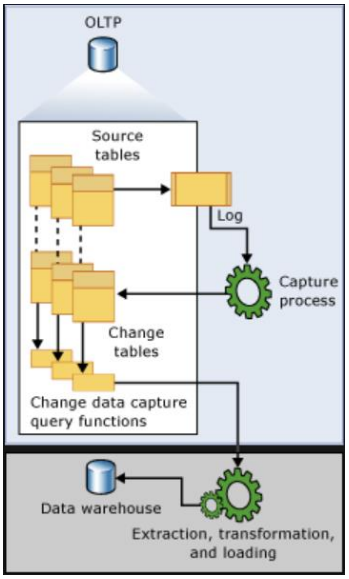


Bild 79: Ablauf zur Erstellung einer Änderungshistorie durch die Change Data Capture-Funktionalität in der Azure SQL-DB

Eine Alternative zur Change Data Capture-Funktionalität ist die erweiterte Auditing-Funktionalität „Ledger“ der Azure SQL-DB, mit der im Unterschied dazu die Datenintegrität durch manipulationssichere Beweise überprüfbar sichergestellt wird und somit auch gegenüber Dritten nachgewiesen werden kann. Zu diesem Zweck erstellt Ledger eine Historie aller an einer DB vorgenommenen Änderungen. Ledger erkennt alle Änderungen an den Tupeln einer Relation, speichert den Stand vor der Änderung als Verlauf in eine geschützte Relation (Historienrelation) ab und erstellt so eine Chronologie der Änderungen. So können selbst Manipulationen durch Nutzende mit umfangreichen Rechten wie Administratoren nachvollziehbar dokumentiert werden. Die geschützte Historientabelle mit dem Änderungsverlauf können zur Bestätigung der Datenintegrität, für Auditing-Zwecke oder auch zur Spurensicherung nach einem erfolgreichen Angriff genutzt werden. Anders als die Data Capture-Funktionalität analysiert Ledger nicht das Transaction Log, sondern nutzt die eigenen Objekte „Updatable Ledger Tables“ (Tupel in den Relationen können gelöscht und geändert werden) und „Append-Only Ledger Tables“ (in diese Relationen können nur neue Tupel eingefügt werden) sowie einen Blockchain-Ansatz.

Jedes während einer Transaktion veränderte Tupel in einer Ledger-Relation wird bevor die Änderungen übernommen wird in ein SHA-256-Hash konvertiert und in einer Baumstruktur gespeichert. Der Wurzelknoten dieser Baumstruktur ist ein Root-Hash, der aus allen in der jeweiligen Transaktion geänderten Tupeln berechnet wird. Für jede während einer Transaktion geänderte Relation wird ein eigenständiger Baum angelegt. Zu jedem geänderten Tupel werden zusätzlich Metadaten wie der zugehörige Datentyp oder die Anzahl an Spalten gespeichert, um die Werte im Nachgang interpretieren zu können:

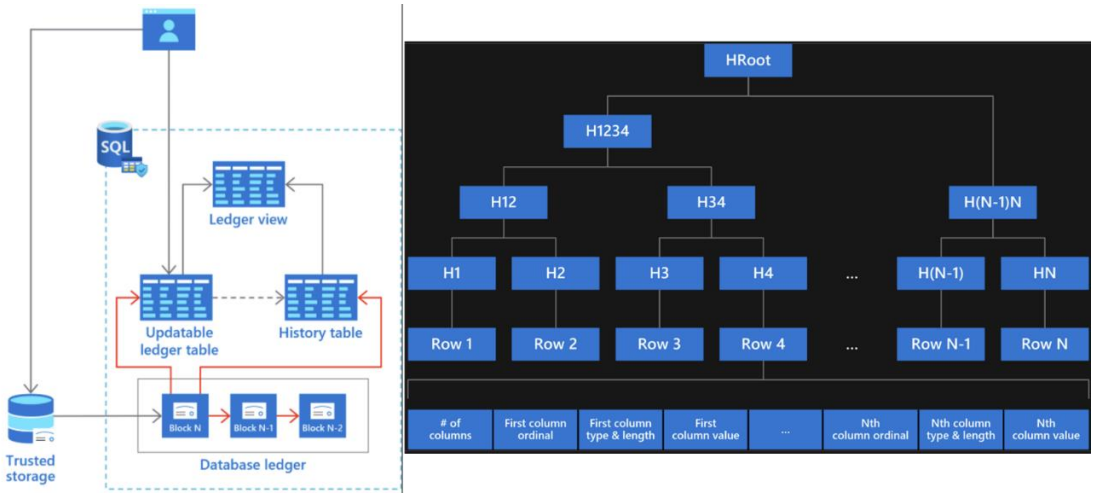


Bild 80: Links: Funktionsweise von Ledger in Azure SQL-DB [117]; Rechts: Jedes während einer Transaktion veränderte Tupel in einer Ledger-Relation wird in einen SHA-256-Hash konvertiert und in einer Baumstruktur gespeichert. Der Wurzelknoten wird aus allen in der Transaktion durchgeführten Änderungen in einer Relation berechnet [169]

Auch die vom DBMS ausgeführten Transaktionen werden in eine Baumstruktur übertragen. Für jede Transaktion erstellt Ledger ebenfalls einen SHA-256-Hash. Wurzelknoten ist auch hier ein aus allen Transaktionen generierter Root-Hash („Database Digests“). Die so zusammengefassten Transaktionen bilden einen Block. Um den DB-Zustand als Historie festzuhalten, ergänzt Ledger für jede Transaktion die zugehörigen Metadaten wie Commit-Zeitstempel, Name des Ausführenden sowie die Wurzelknoten je betroffener Relation aus der zuvor erstellten Baumstruktur. Der Root-Hash eines Blocks, der aus mehreren Transaktionen besteht, wird dann mit dem Root-Hash des Vorgängerblocks verknüpft und wieder in eine Hash-Funktion gegeben, um eine manipulationssichere Blockchain zu erstellen:

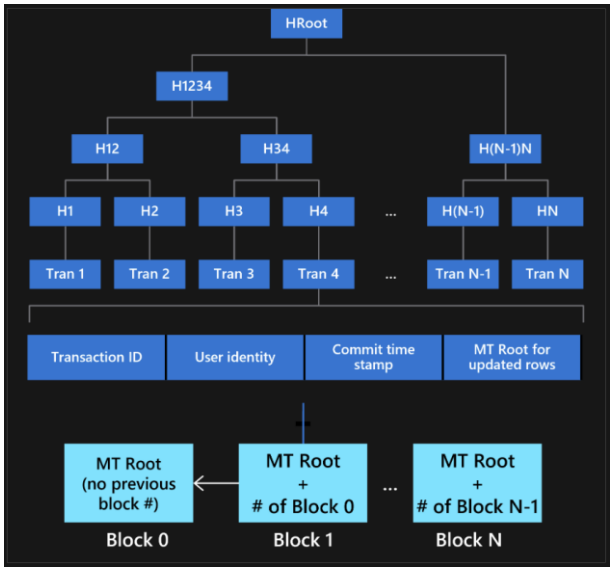


Bild 81: Ausschnitt der Ledger-Blockchain als manipulationssicherer Beweis zur Gewährleistung der Datenintegrität [169]

Im Zuge des Ledger-Überprüfungsprozesses werden die Hash-Werte entsprechend dem aktuellen Stand der Ledger-Relationen neu berechnet und mit den übergebenen Hash-Werten der zu prüfenden Daten verglichen. Unterscheiden sich die berechneten Hash-Werte von den Eingabedaten, liegt eine Datenmanipulation vor. Durch dieses Vorgehen können bei einem Vergleich Angriffe erkannt werden, bei denen alle Systemüberprüfungen umgangen und die Daten direkt innerhalb des Speichers manipulieren wurden [117]/[169].

4.1.6. Berechtigungen & Autorisierung

Beim Öffnen einer Access-Datei wird eine Sperrdatei unter demselben Pfad angelegt, unter dem die Datei liegt (*siehe Kapitel „4.1.5 Logging & Auswertungsmöglichkeiten“*). Die Sperrdatei wird wieder gelöscht, wenn der letzte Nutzende die Datei schließt. MS empfiehlt daher im Rahmen der ordnungsgemäßen Nutzung von Access zumindest auf den Ordner Lese-, Schreib-, Erstellungs- und Löschberechtigungen (nicht „Unterordner und Dateien löschen“-Berechtigungen) einzurichten, indem die Datei liegt.

Da Access Dateiformate 2007-2016 im Gegensatz zur Vorgängerversion keine Sicherheit auf Benutzerebene mehr unterstützen, kann der Zugriff auf die Access-Datei selbst zur Wahrung des Need-to-Know-Prinzips sowie des Least-Privilege-Prinzips nur mittels Vererbungsbruch über separate Lese- und Schreibberechtigungen auf die jeweiligen Dateien eingeschränkt werden. Dateisystemrechte stellen das erste Sicherheitslevel dar und sind die einzige Möglichkeit das Access-Backend zu isolieren. Benötigen Nutzende lediglich Lesezugriff auf eine Access-Datei, ist es erforderlich Leseberechtigung auf die Datei sowie Lese-, Schreib-, Erstellungs- und Löschberechtigungen auf den jeweiligen Ordner zu vergeben, in dem die Access-Datei liegt [105].

In einem lokalen Test mit zwei Windows-Nutzern, ohne Nutzung von Freigaben auf Netzlaufwerken und mit deaktivierter Vererbung konnten Access-Dateien ordnungsgemäß geöffnet, allerdings nicht umbenannt oder gelöscht werden [233]:

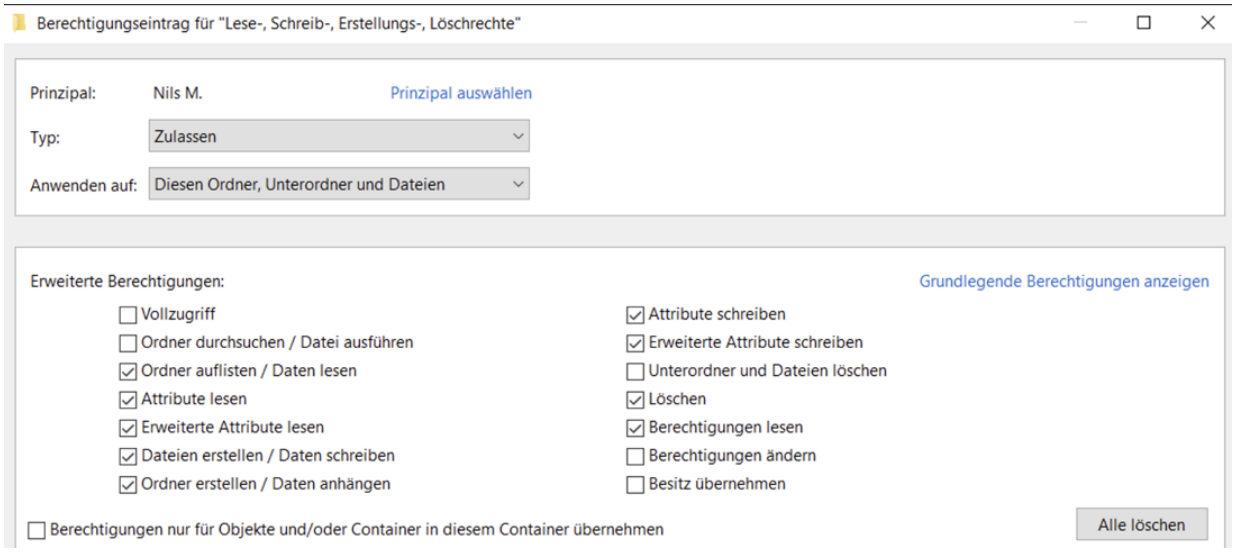


Bild 82: An Nutzende vergebene Lese-, Schreib-, Erstellungs- und Löschberechtigungen auf den Elternordner („erweiterte Berechtigungen"-Ansicht) im NTFS-Dateisystem

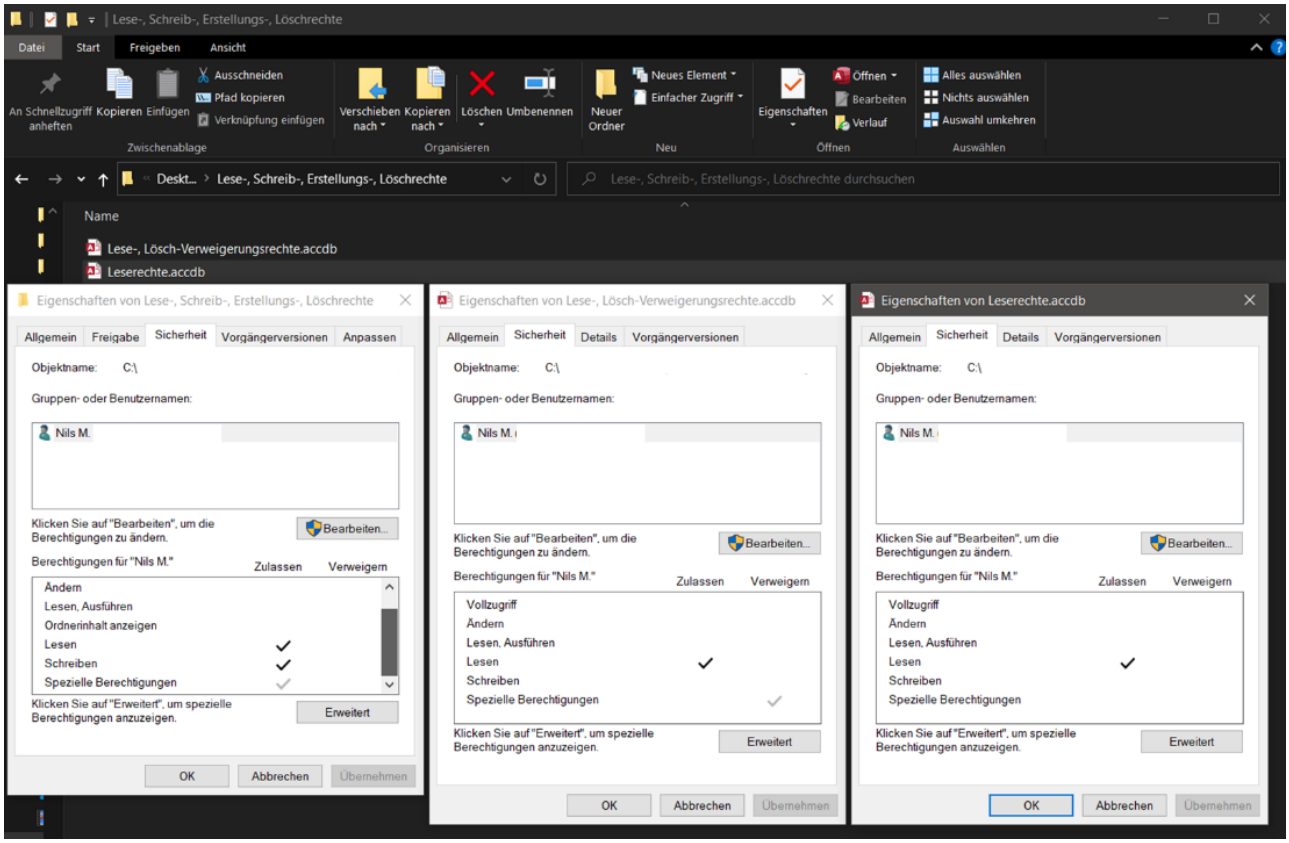


Bild 83: Übersicht über vergebene Rechte auf den Elternordner sowie seine Kind-Elemente. Unter „Spezielle Berechtigungen" ist lediglich „Löschen" ausgewählt

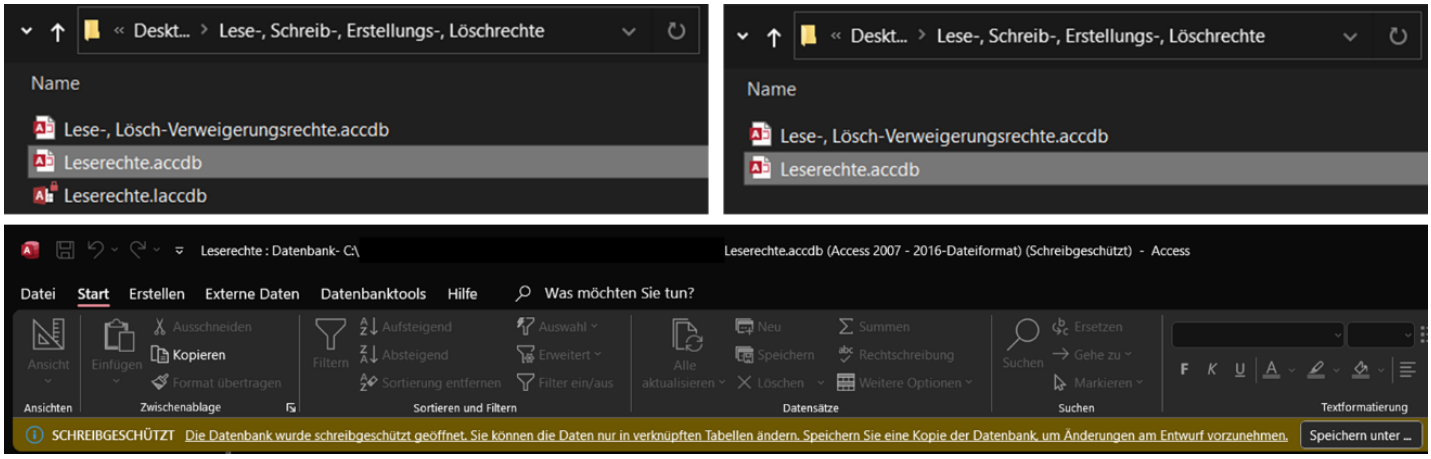


Bild 84: Links oben: Die Access-Datei kann ordnungsgemäß geöffnet werden. Der Nutzer ist der Besitzer der erstellten Sperrdatei und besitzt daher übergreifende Rechte; Rechts oben: Beim Schließen der Access-Datei wird die Sperrdatei ordnungsgemäß gelöscht, wenn es keine anderen Nutzenden mehr gibt; Unten: Da lediglich Leserechte auf die Datei vergeben wurden, wird die Datei im Schreibzugriff geöffnet. Anders als mit Schreibrechten ändert sich das letzte Änderungsdatum der Access-Datei beim reinen Lesezugriff nicht

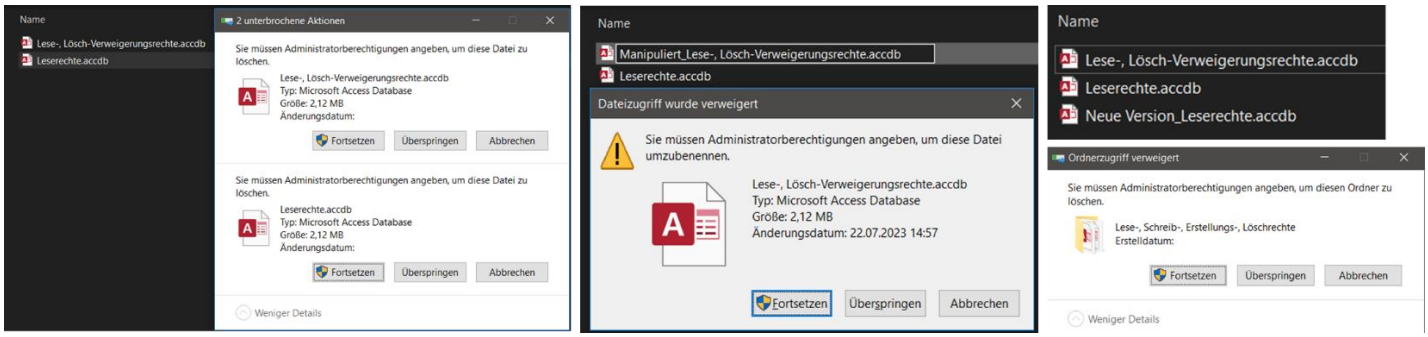


Bild 85: Links: Beide Dateien können nicht gelöscht werden; Mitte: Keine Umbenennung beider Dateien möglich; Rechts oben: Neue Dateien können im Ordner erstellt und je nach Berechtigungen gelöscht oder manipuliert werden; Rechts unten: Der Ordner kann nicht gelöscht werden

Einzige Möglichkeit für einen Angreifenden mit den von MS empfohlenen Berechtigungen ist das Ablegen einer neuen Datei mit ähnlichem Namen, in der Hoffnung, dass unvorsichtige Nutzende auf den Trick hereinfallen. Besitzen Nutzende „Unterordner und Dateien löschen“-Berechtigungen auf den Ordner, können sie auch Dateien und Ordner mit verweigerten Löschberechtigungen löschen. Besitzen Nutzende „Vollzugriff“-Dateisystemrechte auf den Ordner, können untergeordnete Dateien auch mit Lese- und verweigertem Löschrecht gelöscht werden.

Der Passwortschutz zum Öffnen/Entschlüsseln der Datei stellt das zweites Sicherheitslevel dar. Eigenimplementierungen in VBA zur Nutzerverwaltung dienen lediglich zur Nutzerführung und bieten keine Sicherheit. Aus Gründen der Abwärtskompatibilität liefert die VBA-Funktion „`CurrentUser`“ im Access-Dateiformat 2007-2016 immer „`Admin`“ zurück und wurde bisher nicht entfernt. Da es sich bei Access um ein dateibasiertes DBS handelt, benötigt der jeweilige Windows-User zum Zugriff mindestens Leserechte auf die Datei. Mit dem Leserecht kann eine Datei problemlos kopiert und ein Angriff ungestört lokal fortgesetzt werden (*siehe Kapitel „2.5 Unterschied dateibasierte*

& Client-Server DBS"). Sobald Schreibzugriff auf den Ablageort und die Access-Datei besteht, kann sie unbemerkt gegen eine mit Schadcode manipulierte Datei ausgetauscht werden, was eine gravierende Sicherheitslücke darstellt. Wird eine Datei exklusiv geöffnet, sind Leserechte auf den Ordner ausreichend, da keine Sperrdatei angelegt wird (*siehe Kapitel 4.1.5 Logging & Auswertungsmöglichkeiten"*) [105]. Durch das exklusive Öffnen einer Datei mit Schreibrechten entstehen daher weniger Spuren für eine potenzielle Auswertung. Ist das Passwort zum Entschlüsseln bekannt, besteht Vollzugriff auf die Access-Datei, inklusive aller Systemrelationen, die Access hauptsächlich für interne Verwaltungszwecke nutzt, und somit kein Schutz (*siehe Kapitel „2.5 Unterschied dateibasierte & Client-Server DBS"*). Über die Access-Oberfläche ausgeblendete Relationen sind weiterhin über DbVisualizer einsehbar, der Zugriff auf die Systemrelationen funktioniert jedoch nicht über den „UCanAccess"-Treiber (*für weitere Details siehe Kapitel „Anlage 6: Tabelle ausblenden (Access)"*). Einschränkungen in der Navigationsansicht (Menüband) können leicht rückgängig gemacht werden (*siehe Kapitel „4.1.2 Konfiguration"*). Eine weitere Möglichkeit das Need-to-Know-Prinzip einzuhalten, kann die Aufteilung einer Access-Anwendung in Datenhaltungs-, Logik- sowie Präsentationsschicht sein. Als Synergieeffekt werden Leistung, Verfügbarkeit, Sicherheit, Zuverlässigkeit positiv beeinflusst und die Flexibilität durch Trennung der Anwendungskomponenten wird gesteigert. Das Passwort zum Entschlüsseln des Access-Backends wird vor Nutzenden verborgen, da es im Frontend abgelegt ist (Achtung: siehe gefundene Schwachstelle in *Kapitel „4.3 Dateiformatanalyse mittels Hexadezimal-Editor (Access)" sowie Kapitel „4.2 Schnittstellenanalyse (Access Excel)"*). Das Frontend beinhaltet die Logik sowie das GUI. Ohne Passwort kann auf die Daten im Backend nicht zugegriffen werden. MS empfiehlt darüber hinaus diverse Freigabemethoden für konkrete Anwendungsfälle:

Freigabemethode	Front-End	Back-End	Datenspeicherort	Szenario	Benutzer	Workload
Einzelne Datenbank	---	---	Heimnetzwerk	Home/Kleinunternehmen	Ein paar	Light
SharePoint	Access	SharePoint-Listen	SharePoint-Website	Enterprise-Team	Dutzende	Light
Aufteilen der Datenbank	Access	Access	Netzwerkordner	Enterprise-Team	Dutzende	Mittel
Client/Server	Access	SQL Server	Datenbankserver	Unternehmensweit	Viele	Heavy
Hybrid Client/Server	Access	Azure SQL	Azure Cloud	Unternehmensweit	Viele	Heavy

Bild 86: Übersicht über die Möglichkeiten zum Freigeben von Access-Dateien [201]. WICHTIG: Die SharePoint-Variante (Web-Apps) wird von MS nicht mehr empfohlen und die Unterstützung in den nächsten Versionen entfernt [219]

Bei einer großen Nutzeranzahl und hoher Auslastung empfiehlt MS Client-Server-Systeme in der Cloud oder als On-Premises-Variante. Bei geringer bis mittlerer Arbeitslast und mittlerer Benutzerzahl (bis 50 Nutzende) kann ein einzelnes Access-DBS oder eine aufgeteilte Access-Datenquelle verwendet werden. Die Vorteile einer Freigabemethode mit einem Client-Server-DBS wie der Azure SQL-DB hängen vom verwendeten DBS ab. Zu den verfügbaren Standardfunktionen gehören eine Benutzerauthentifizierung, ein selektiver Datenzugriff (*siehe Kapitel „4.1.7 Datenschutzkonformer Zugriff"*), Abfragepläne, Transaktionsprotokolle, eine bessere Datenverfügbarkeit und integrierte Tools für die Datenverwaltung oder automatisiertes Monitoring. Der SQL Server in der Azure-Cloud wird je nach gewählter Bereitstellungsmethode (*siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software"*) mit Aspekten wie höherer

Performance, Verfügbarkeit, Sicherheit, automatisches Recovery, serverbasierter Verarbeitung sowie dynamischer Skalierbarkeit ohne Ausfallzeiten beworben [201]/[0.236].

Im Unterschied zu Access bietet die Azure SQL-DB umfangreiche Authentifizierungs-Möglichkeiten (*siehe Kapitel „4.1.4 Passwörter & Authentifizierung“*). Wird die Azure SQL-DB als Backend für ein Access-Frontend verwendet, kann neben den Dateisystemberechtigungen für den Zugriff auf die Access-Datei und der Dateiverschlüsselung als dritte Sicherheitsebene die Nutzerauthentifizierung über einen lokalen DB-Benutzer oder über Azure AD (Azure AD-Authentifizierung) erfolgen. Die Verwendung der Azure AD-Authentifizierung steht gleichzeitig im Konsens mit Stimmen, die sich gegen die Verwendung von Verfahrensusern aussprechen (*siehe Kapitel „3.3.1 Ausschluss von für dateibasierte DBSs unpassende Bewertungskriterien“*) [201]. Um mehrere Azure AD-Benutzer thematisch zu gruppieren, können diese in Azure AD-Gruppen zusammengefasst werden. DB-Rollen werden in Azure SQL-DB nicht unterstützt. Einzelne AD-Benutzer und -Gruppen werden dann auf Server oder DB-Ebene bekanntgemacht. Die so importierten AD-Benutzer und -Gruppen können über (SQL-)Befehle wie `GRANT` oder `REVOKE` granular gemäß dem Need-to-Know-Prinzip berechtigt werden. Nachfolgendes Statement teilt einer existierenden AD-Gruppe „`SicherheitsanalyseVonMsAccessReadOnly`“ Leserechte auf die Relation „`T_Personen`“ zu:

```
GRANT SELECT ON T_Personen
TO SicherheitsanalyseVonMsAccessReadOnly
```

Dabei können die Zugriffe auf (System)Relationen, Views, Funktionen wie Stored Procedures oder konkrete DDL oder DML-Befehle sowie Attribute innerhalb einer Relation (*siehe Kapitel „4.1.7 Datenschutzkonformer Zugriff“*) eingeschränkt werden. Nicht-Admin-Benutzer haben keinen Zugriff auf die Master-DB und standardmäßig nur eingeschränkten Zugriff auf Systemtabellen [1, S. 270-277]/[163]:

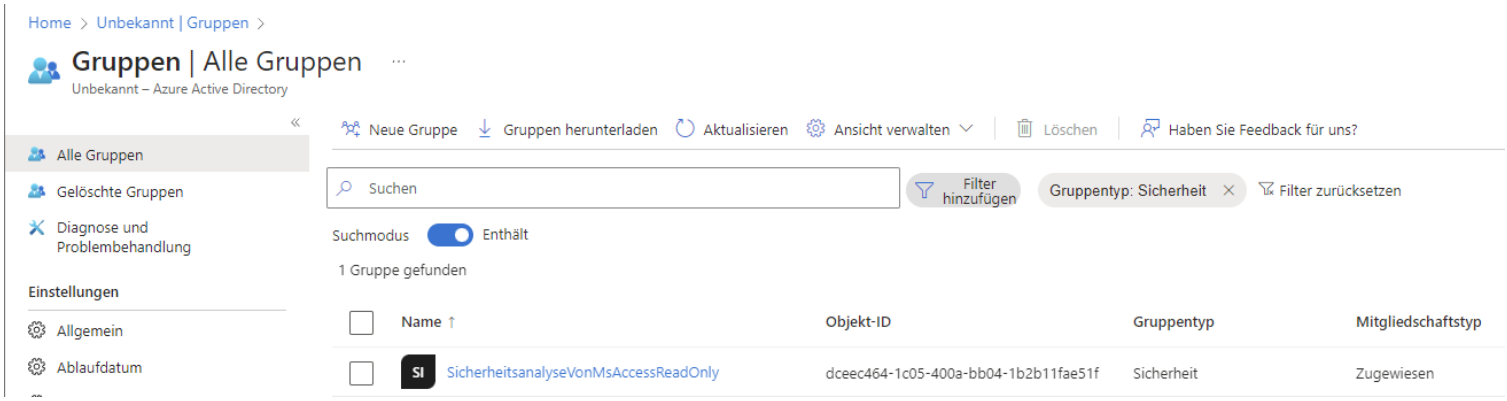


Bild 87: Anlage einer Azure AD-Gruppe, der mehrere AD-Benutzer hinzugefügt werden können. Alle Mitglieder erhalten dieselben Rechte

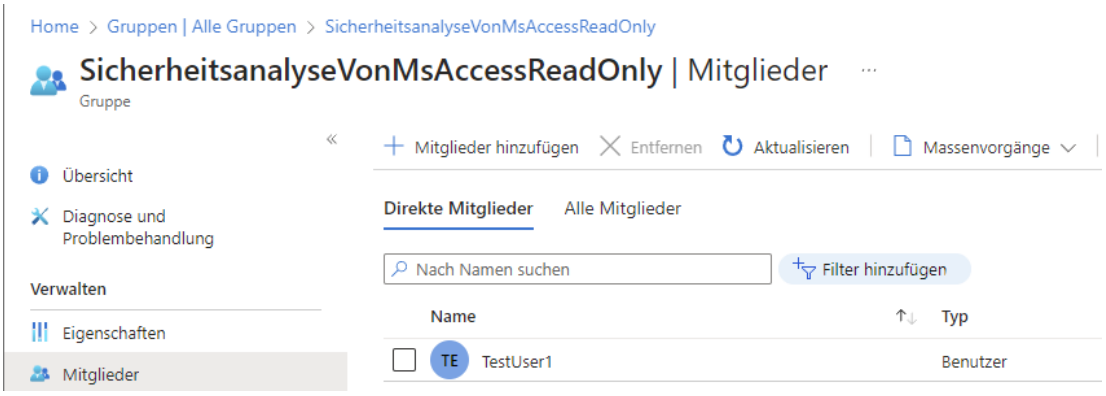


Bild 88: Hinzufügen des Test-Users in die zuvor erstellte Azure AD-Gruppe

Mit folgendem Statement kann die mit dem Test-User befüllte Azure AD-Gruppe der Azure SQL-DB als Gastnutzer bekanntgemacht werden **(1)**, die Gruppe erhält über die vorkonfigurierte DB-Rolle „*db_datareader*“ ausschließlich Leserechte auf die gesamte DB **(2)** [1, S. 270]/[94]:

(1)
```
CREATE USER [SicherheitsanalyseVonMsAccessReadOnly]
FROM external provider
```

(2)
```
ALTER ROLE db_datareader ADD MEMBER [SicherheitsanalyseVonMsAccess-
ReadOnly];
```

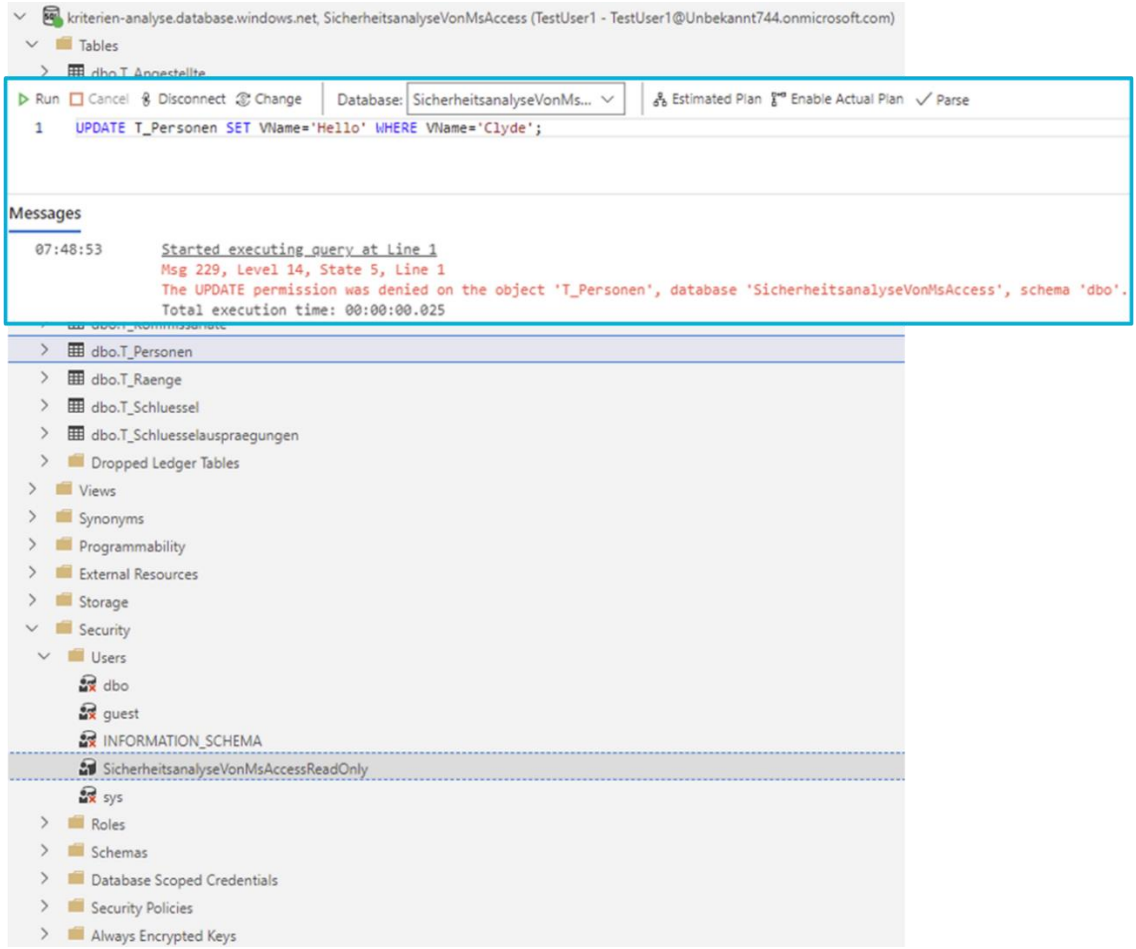


Bild 89: Ein Schreibzugriff des Test-Users schlägt aufgrund der zuvor für die Gruppe vergebenen Leserechte wie erwarte fehl. Im Azure Data Studio wird die AD-Gruppe unter „Users" angezeigt

4.1.7. Datenschutzkonformer Zugriff

Access bietet anders als der SQL Server oder die Azure SQL-DB keine Column-Level- oder Row-level Permissions [91]/[110]/[111]. Als Alternative und Ersatz für gefilterte Views oder Formulare können vorkonfigurierte Abfragen angeboten werden. Da nach dem Öffnen einer Access-Datei Vollzugriff besteht und Formulare leicht durch DB-Tools wie DbVisualizer umgangen werden können, bietet dieser Mechanismus allerdings nur Sicherheit für einen datenschutzkonformen Zugriff, wenn der Nutzenden-Zugriff über ein separates Frontend erfolgt, das MS Access-Backend verschlüsselt und das Passwort den Nutzenden unbekannt ist. Nachteil dieses Ansatzes ist, dass im Frontend die Benutzerverwaltung, Column-Level- und Row-level Permissions-Funktionalität nachimplementiert werden muss, was die Komplexität der Programmlogik und die Fehleranfälligkeit deutlich erhöht. Ansonsten bietet Access lediglich unter dem Kopfreiter „*Datei → Optionen → Aktuelle Datenbank*" die Funktion „Beim Speichern personenbezogene Daten aus Dateieigenschaften entfernen", die nur gemeinsam mit der Funktion „Beim Schließen komprimieren" aktiviert werden kann. Da Access keine Log-Dateien erstellt, können hierfür auch keine automatischen Löschroutinen angewendet werden

(*siehe Kapitel „4.1.5 Logging & Auswertungsmöglichkeiten“*). Zur Verbesserung des datenschutzkonformen Zugriffs können die Daten alternativ in ein Client-Server-DBS wie den MS SQL Server oder Azure SQL-DB ausgelagert werden (*siehe Kapitel „2.5 Unterschied dateibasierte & Client-Server DBS“*). Damit steht gleichzeitig auch der gesamte Funktionsumfang des jeweiligen Client-Server-DBS zur Verfügung, inklusive der in *Kapitel „4.1.6 Berechtigungen & Autorisierung“* beispielhaft skizzierten Sicherheitsfunktionalitäten beziehungsweise der für dieses Kapitel relevanten Funktionalitäten für den datenschutzkonformen Zugriff wie Row- und Column-Level-Permissions. Im SQL Server kann zudem die Datenhaltung zentralisiert werden (*siehe Kapitel „4.1.9 Banking 4.0“*), was neben der Einhaltung des Need-to-Know-Prinzips (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung“*) gleichzeitig auch die Einhaltung der europäischen DS-GVO erleichtert. Datenänderungen und Zugriffe können überwacht sowie auf Benutzerebene individuell konfiguriert werden, da eine Access-Datei über den jeweils angemeldeten Benutzer ausgeführt wird. Dies erleichtert die datenschutzkonforme Zugriffssteuerung in Abhängigkeit von den Zuständigkeiten der Mitarbeitenden. Eine Historisierung der Daten ist ebenfalls möglich und kann durch eigene Zugriffsrechte zusätzlich geschützt werden. Die Daten in der Access-Datei können regelmäßig gelöscht werden und müssen nicht auf dem Dateisystem archiviert werden, was die von der DS-GVO geforderte Löschpflicht (vergleiche Art. 17 Abs. 1 DS-GVO, Recht auf Löschung) nach Erlöschen des Zwecks (vergleiche Art. 5 Abs. 1 lit. e DS-GVO, Speicherbegrenzung) aufgrund der zentralen Datenhaltung im Client-Server-DBS vereinfacht. Es können auch Views im SQL Server erstellt werden, um nach dem Need-to-Know-Prinzip nur ausgewählte Ansichten nach außen anzubieten. In diesem Fall wird Access nur noch als Frontend-Tool genutzt, was die Daten vom SQL Server bezieht, verarbeitet und bei Bedarf ausgibt [196].

MS bietet mit dem „SQL Server Migration Assistant“ ein Programm zur einfachen Migration von Access zum SQL Server oder der Azure SQL-DB an, um den Umstieg zu erleichtern und gleichzeitig die Nutzerfreundlichkeit zu verbessern [121].

In diesem Zusammenhang ist auch darauf hinzuweisen, dass laut *Heise Online* auf der *104. Datenschutzkonferenz* im Jahr 2022 MS 365 und somit auch Access erneut für datenschutzwidrig erklärt wurde. Die Erstbewertung erfolgte im Jahr 2020. Unternehmen, Behörden und Schulen können das gesamte Office 365-Paket ohne zusätzliche technische Maßnahmen nicht rechtskonform nutzen. Hintergrund ist, dass MS im September 2022 eine Neufassung seines Auftragsverarbeitungsvertrages („Microsoft Products and Services Data Protection Addendum“) publizieren musste, da der Europäische Gerichtshof mit dem „Schrems-II-Urteil“ den transatlantischen „Privacy Shield“ und damit eine der wichtigsten Grundlagen für den Transfer von Kundendaten in die USA für ungültig erklärt hatte. Bei der Überarbeitung des Vertrages hat MS auch festgelegt welche Daten für eigene Zwecke (Telemetrie) verwendet werden, um damit unter anderem eigene Applikationen weiterzuentwickeln oder Trainingsdaten für KI zu sammeln. Auf der Access-Oberfläche „*Trust Center → Datenschutzoptionen*“ sind weitere Informationen über von MS gesammelte erforderliche und optionale Diagnosedaten zu finden:

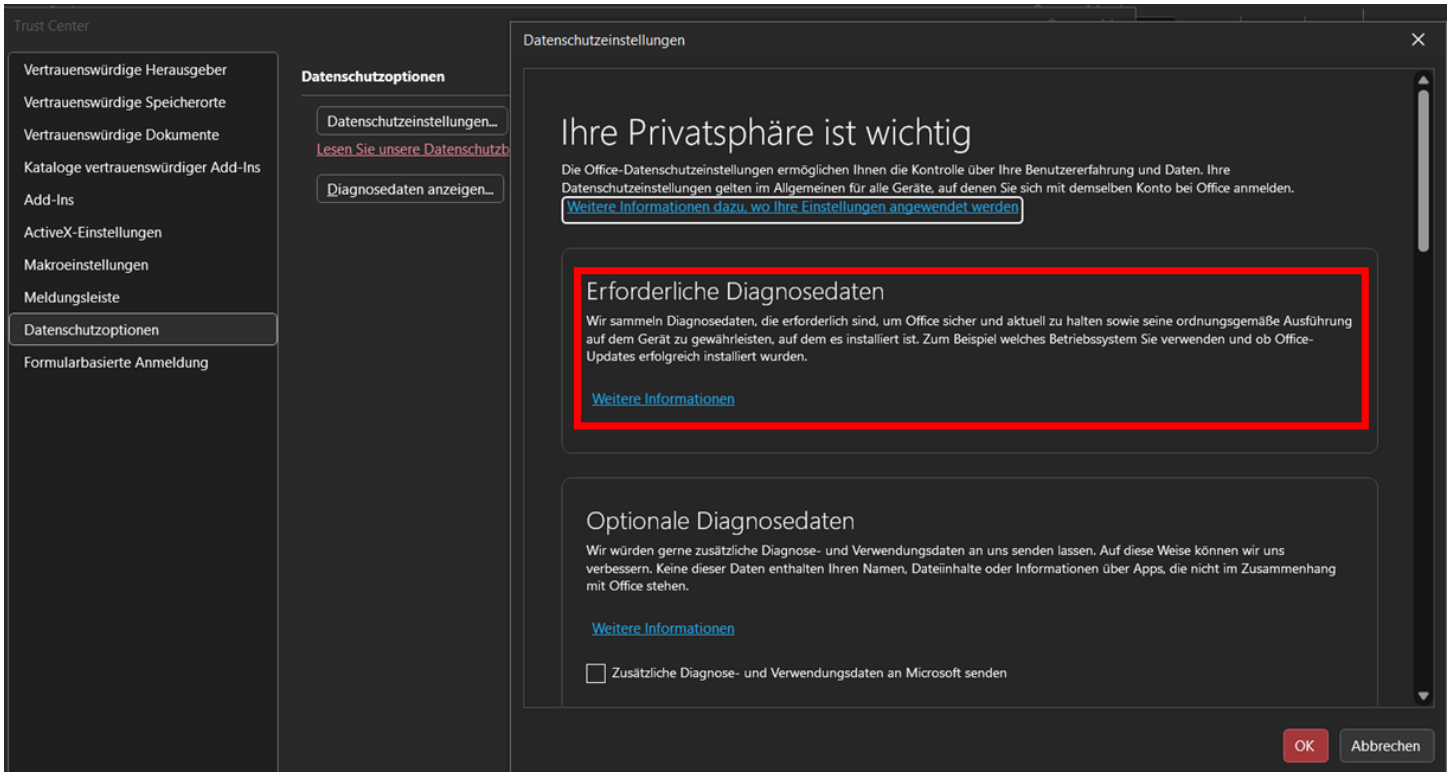


Bild 90: „Trust Center → Datenschutzoptionen“ Informationsmeldung über erforderliche Diagnosedaten [56]

Zur Erstellung von Statistiken werden statistische, nicht personenbezogene Daten aus pseudonymisierten und aggregierten Daten erzeugt. Auf den Inhalt der Kundendaten soll dabei nicht zugegriffen werden. Verwendet ein Auftragsverarbeiter Daten für eigene Zwecke, übernimmt er die Rolle des Verantwortlichen. Bisher genannte Zwecke wie die Bekämpfung von Betrug oder Cyberkriminalität werden nicht mehr erwähnt. Die Erläuterungen von MS wurden auf der *Datenschutzkonferenz* als unzureichend angesehen, da die Dokumente nicht die notwendige Transparenz liefern, welche konkreten Daten MS für eigene Zwecke verwendet. Es ist nicht bekannt, welche konkreten Informationen und Diagnosewerte an MS übermittelt werden, was eine Prüfung der Rechtmäßigkeit der Verarbeitung unmöglich macht. Zum Schutz vor Telemetrie kann ein zwischengeschalteter Filter-Proxy verwendet werden [56].

Auch das BSI erachtet das Sammeln von Daten in MS 365 als kritisch und ergänzt als weiteres Sicherheitsrisiko „verbundene Erfahrungen“. Verbundene Erfahrungen sind Funktionen von Office-Anwendungen, die während des Betriebs mit dem MS-Backend kommunizieren und Daten austauschen. Im Gegensatz zur Telemetrie ist das Ziel verbundener Erfahrungen keine Diagnostik. Mit Hilfe verbundener Erfahrungen werden dem Anwender konkrete Funktionen, wie die Verwendung der Diktierfunktion oder das Übersetzen von Texten, angeboten. Sowohl Office-Anwendungen als auch verbundene Erfahrungen führen zu Diagnoseereignissen, die im Rahmen von Telemetrie möglicherweise an MS gesendet werden [7]/[183].

Anzumerken bleibt, dass MS im Jahr 2018 mit dem „Diagnosedaten-Viewer“ ein Programm zur Überprüfung von Windows- und Office-Rohdiagnosedaten veröffentlicht hat [218]:

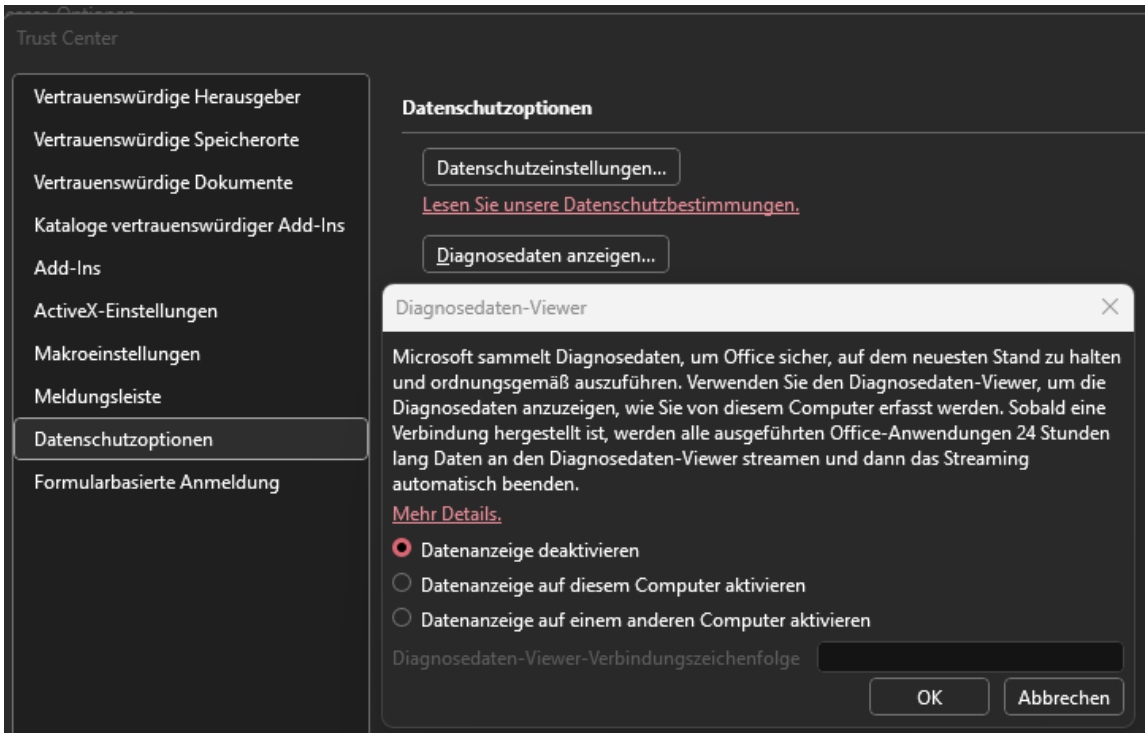


Bild 91: Datenanzeige mittels Diagnosedaten-Viewer für Access über „Trust Center → Datenschutzoptionen“ aktivieren

Im Unterschied zu Access bietet die Azure SQL-DB neben einer Sicherheit auf Benutzerebene (*siehe Kapitel „4.1.4 Passwörter & Authentifizierung“*) und einer granularen Berechtigungsvergabe (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung“*) zusätzlich die Möglichkeit den datenschutzkonformen Zugriff auf Zeilen- und Spaltenebene zu implementieren. Die Sicherheit auf Spaltenebene wird über SQL-Befehle wie …

```
GRANT SELECT ON <Schema-Name>.<Table-Name>(<Column-Name 1>,<Column-
Name 2>, <…>)
TO <User-Name>
```

… geregelt. Werden hier konkrete Spaltennamen angegeben, so ist der lesende (`SELECT`) oder schreibende (`INSERT`, `UPDATE`) Zugriff ausschließlich auf die genannten Spalten beschränkt:

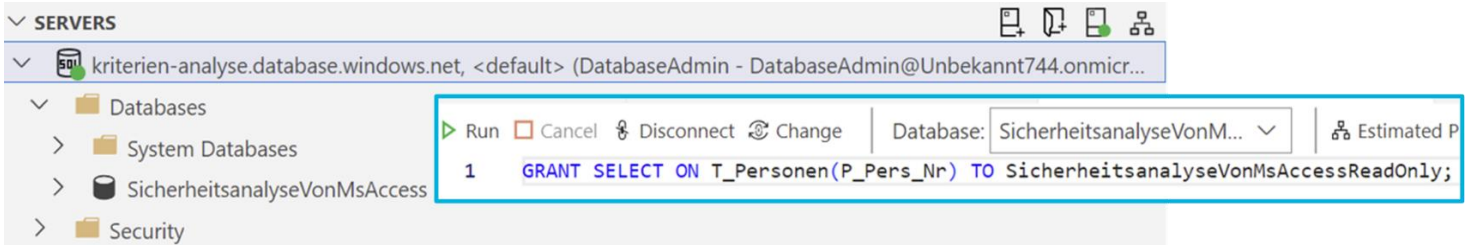


Bild 92: Hinzufügen einer SELECT-Berechtigung durch den DB-Admin, um der AD-Gruppe „SicherheitsanalyseVonMsAccessReadOnly“ ausschließlich Lesezugriff auf das Attribut „P_Pers_Nr“ der Relation „T_Personen“ zu geben (Azure Data Studio)

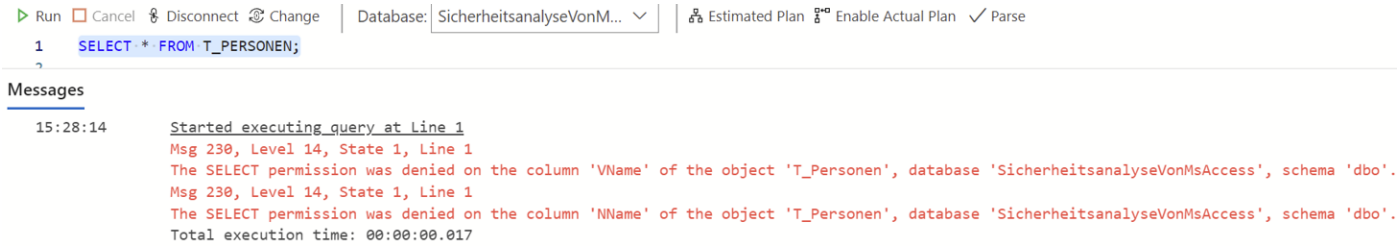


Bild 93: Die Anzeige der gesamten Relation „T_Personen“ durch den AD-Benutzer „TestUser1“ (AD-Gruppenmitglied von „SicherheitsanalyseVonMsAccess“) führt zu einem Fehler

Bild 94: Erfolgreiche Anzeige des Attributs „P_Pers_Nr“ der Relation „T_Personen“ durch den AD-Benutzer „TestUser1“ (AD-Gruppenmitglied von „SicherheitsanalyseVonMsAccess“) aufgrund der durch die AD-Gruppe zugeordneten Rechte

Im Gegensatz dazu ist die Filterlogik für die Sicherheit auf Zeilenebene in „Inline Table-Valued Functions“ eingebettet. Die Funktion wird automatisch auf der Ergebnismenge aufgerufen, um sie auf Zeilenebene erneut zu filtern, bevor sie an den Aufrufer zurückgegeben wird. Die Filterlogik der Funktion gibt für den jeweils betrachteten Datensatz „*1*“ (True) zurück, wenn ihre Bedingung erfüllt ist. Die Funktionen werden im zweiten Schritt einer erstellten „Security Policy“ als „Predicate“ (Filter) hinzugefügt und somit aktiviert. Predicates werden in „Filter Predicates“ und „Block Predicates“ unterschieden. Filter Predicates werden bei allen Lesezugriffen und nahezu allen Statements wie *SELECT*, *UPDATE*, *DELTE* ausgeführt. Filter Predicates greifen jedoch nicht bei SQL-Statements wie *AFTER INSERT*, *AFTER UPDATE* oder *BEFORE UPDATE* welche die Daten erst nach der Manipulation filtern. Hierfür werden Block Predicates verwendet. Darüber hinaus kann der zeilenbasierte Zugriff mittels dynamischer Datenmaskierung weiter eingeschränkt werden, indem Attribute mit vertraulichen oder personenbezogenen Daten, wie Kreditkarteninformationen, für nichtautorisierte Benutzende unkenntlich gemacht werden. Die Konfiguration der Datenmaskierung erfolgt bei Anlage einer Relation

mittels *CREATE TABLE* oder im Nachgang mittels *ALTER COLUMN*-Statement. Dabei können entweder vorkonfigurierte Maskierungsfunktionen wie zur Maskierung von E-Mail-Adressen oder für Zufallszahlen in einem vorgegebenen Wertebereich verwendet werden, oder es können benutzerdefinierte Maskierungsmuster erstellt werden. Die von der Maskierung betroffenen Daten liegen in unmaskierter Form in der DB und werden nur in der Ergebnismenge für Benutzende maskiert. Um Nutzende für den unmaskierten Zugriff freizuschalten, müssen sie explizit mittels *GRANT* berechtigt oder im Azure-Portal ausgeschlossen werden [1, S. 277-280]/[51, S. 64]/[104]/[145]:

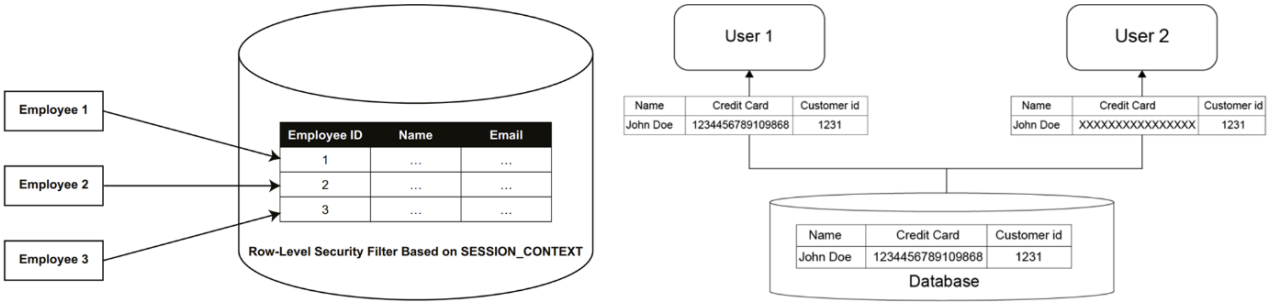


Bild 95: Links: Sicherheit auf Zeilenebene [1, S. 278]; Rechts: Dynamische Datenmaskierung [1, S. 279]

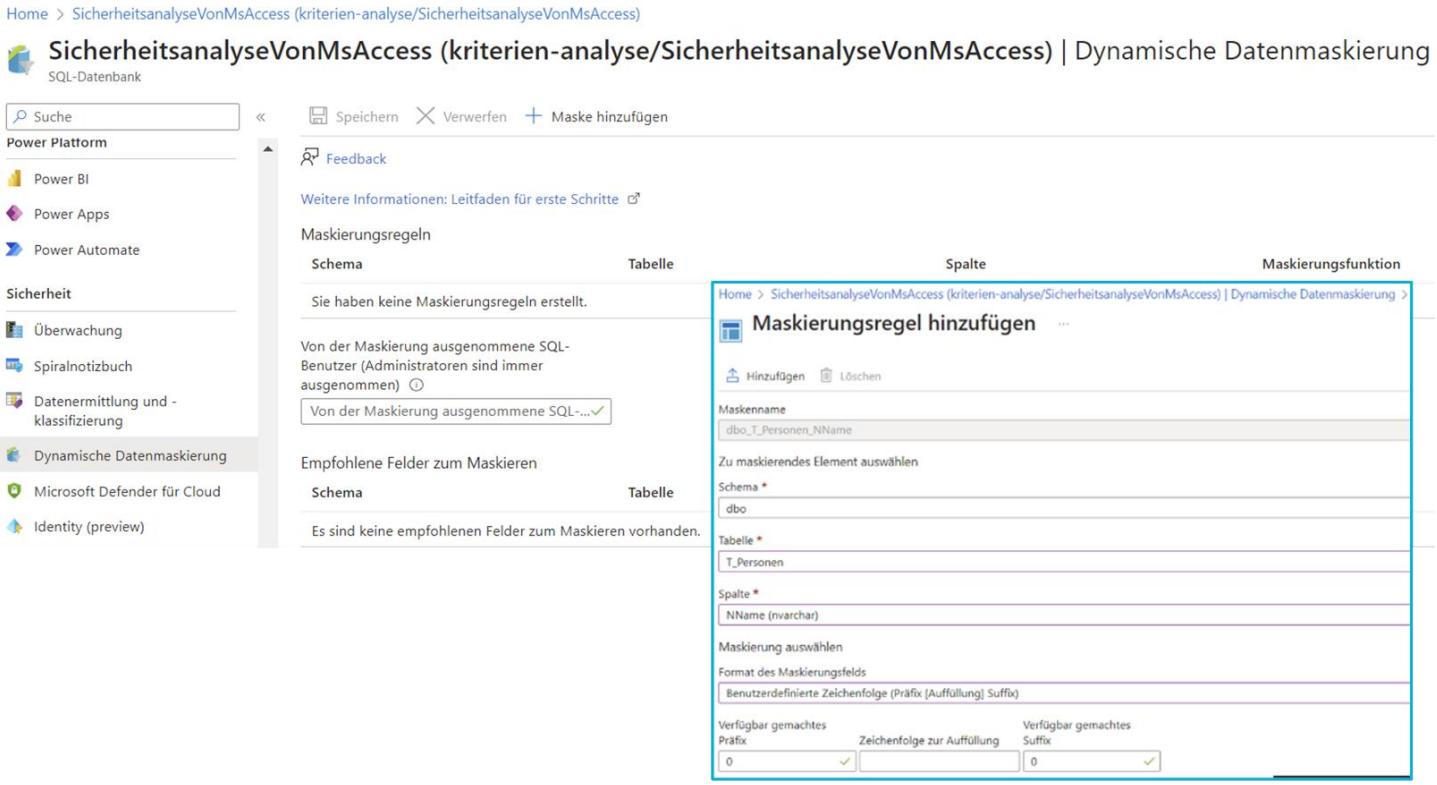


Bild 96: Aktivierung der dynamischen Datenmaskierung für eine DB im Azure-Portal

Darüber hinaus ermöglicht der Defender für die Cloud die Überprüfung der Einhaltung gesetzlicher Bestimmungen anhand durchgeführter Compliance-Kontrollen. Die Kontrollen sind für unterschiedliche Bereiche, wie den Datenschutz, verfügbar. Dabei sind den Kontrollen ein Satz von Bewertungen zugeordnet, die durch den Defender für die Cloud durchgeführt werden. Ein grüner Haken bedeutet, dass die Bewertung oder Kontrolle

bestanden wurde. Zu den einzelnen Bewertungen stehen Detailseiten mit weiteren Informationen und Hinweisen zur Behebung zur Verfügung. Über „Bericht herunterladen“ kann ein „Microsoft Cloud Security Benchmark“-Report zur Gesamtübersicht erstellt werden:

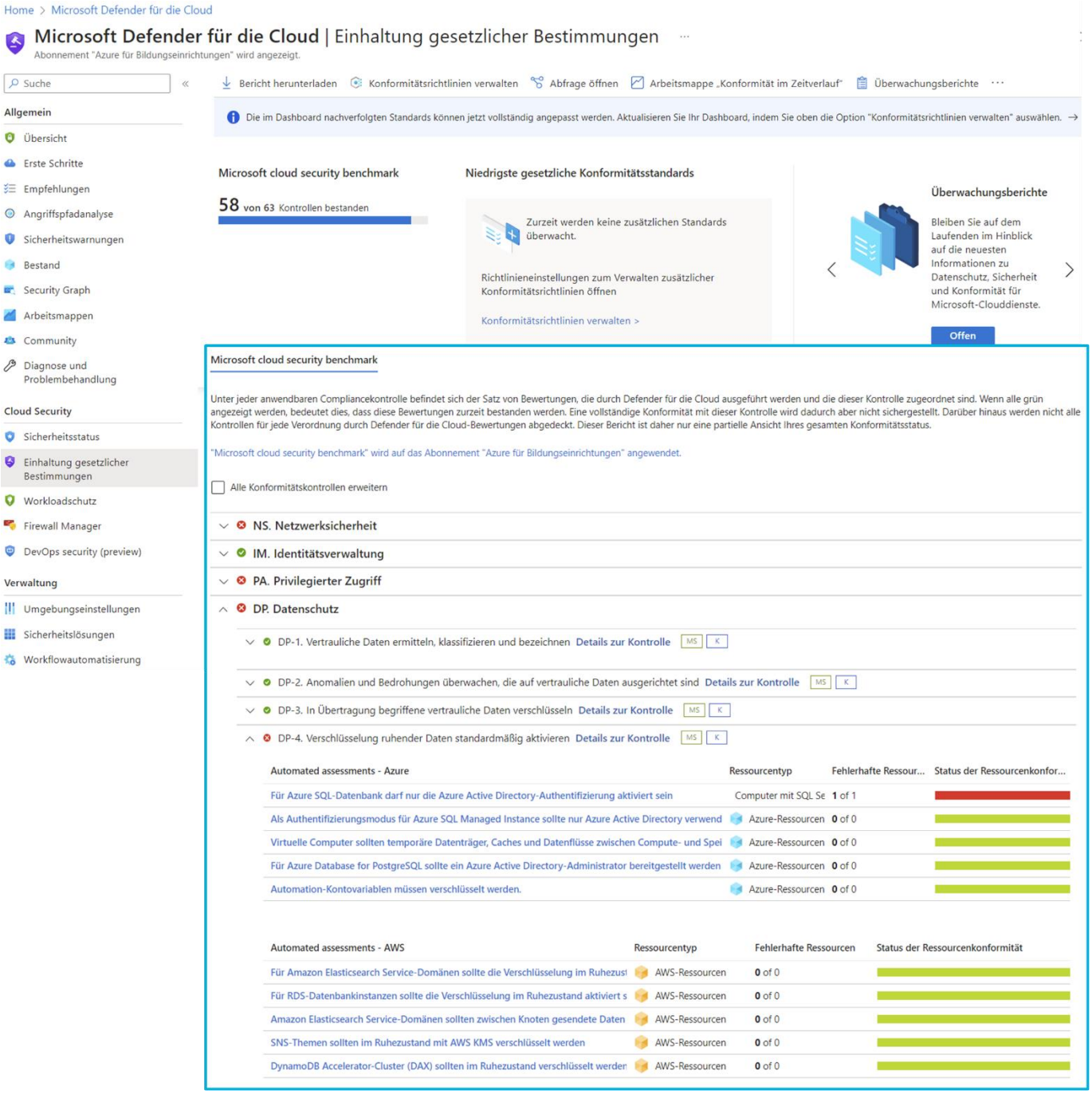


Bild 97: MS Defender für die Cloud – Einhaltung gesetzlicher Bestimmungen am Beispiel Datenschutz (Übersicht über Kontrollen) 1/2

Home > Microsoft Defender für die Cloud | Einhaltung gesetzlicher Bestimmungen >

Für Azure SQL-Datenbank darf nur die Azure Active Directory-Authentifizierung aktiviert sein

Microsoft cloud security benchmark

Ausnahme | Verweigern | Richtliniendefinition anzeigen | Abfrage öffnen

Severity	Freshness interval	Tactics and techniques
Medium	30 Min	Erster Zugriff

Description

Durch das Deaktivieren lokaler Authentifizierungsmethoden und das ausschließliche Zulassen der Azure Active Directory-Authentifizierung wird die Sicherheit verbessert, indem sichergestellt wird, dass nur über Azure Active Directory-Identitäten auf Azure SQL-Datenbanken zugegriffen werden kann. Weitere Informationen finden Sie unter aka.ms/adonlycreate.

Remediation steps

Manual remediation:

So deaktivieren Sie lokale Authentifizierungsmethoden und erlauben nur die Azure Active Directory-Authentifizierung:

1. Suchen Sie Ihren Azure SQL-Server im Portal.
2. Navigieren Sie zu Azure Active Directory im linken Navigationsbereich.
3. Wählen Sie "Administrator einstellen", wenn der Azure Active Directory-Administrator nicht bereits eingestellt ist.
4. Markieren Sie das Kästchen "Nur Azure Active Directory-Authentifizierung für diesen Server unterstützen" und klicken Sie auf "Speichern". Weitere Einzelheiten finden Sie unter https://learn.microsoft.com/en-us/azure/azure-sql/database/authentication-azure-ad-only-authentication-tutorial?WT.mc_id=Portal-Microsoft_Azure_Security.

Affected resources

Unhealthy resources (1) | Healthy resources (0) | Not applicable resources (0)

Search Azure-Ressourcen

Name	Subscription	Owner	Due date	Status	Last change date
kriterien-analyse	Azure für Bildungseinrichtungen				4.7.2023, 20:07:06

Bild 98: MS Defender für die Cloud – Einhaltung gesetzlicher Bestimmungen am Beispiel Datenschutz (Vorgehen zur Schließung einer fehlgeschlagenen Bewertung, da die SQL-Authentifizierung nicht deaktiviert ist) 2/2

Executive summary

Introduction

Microsoft Defender for Cloud executes a set of automated assessments on your Cloud environment which can help provide evidence relevant to specific controls in a compliance framework or standard This report summarizes the current status of those assessments on your environment, as they map to the associated controls. This report does not represent a complete compliance report for the standard, nor does it ensure compliance.

Compliance with Microsoft cloud security benchmark controls

Your environment is compliant with 58 of 63 supported Microsoft cloud security benchmark controls.

Coverage

subscriptions: 1 resources: 3

Area	Failed controls	Passed controls
AM. Asset Management	0	5
BR. Backup and recovery	0	2
DP. Data Protection	1	7
DS. DevOps Security	0	2
ES. Endpoint security	0	3
IM. Identity Management	0	9
IR. Incident Response	1	4
LT. Logging and threat detection	0	6
NS. Network Security	1	8
PA. Privileged Access	1	7
PV. Posture and Vulnerability Management	1	5

Bild 99: Ausschnitt aus einem „Microsoft Cloud Security Benchmark“-Report

4.1.8. Datensicherung

Access bietet von Haus aus keine automatischen Backup-Funktionalitäten an. MS verweist auf Drittanbieterprodukte zur automatisierten Sicherung des Dateisystems wie Redundant Array of Independent Disks (RAID)-Systeme sowie auf Eigenimplementierungen wie über die VBA-Funktion „*DoCmd.CopyDatabaseFile*“. Mit der VBA-Funktion kann die mit dem aktuellen Projekt verbundene DB zum Upload in eine SQL Server DB-Datei (.mdf) exportiert werden [82]/[102]/[207]. Darüber hinaus speichert Access kontinuierlich Änderungen, um Datenverlust bei unerwarteten Fehlern vorzubeugen. Daher wird auch ohne Änderungen der Zeitstempel des letzten Änderungsdatums bereits beim Öffnen der Datei aktualisiert, was den Zeitstempel ungeeignet für die Ermittlung tatsächlicher Änderungen an der Datei macht. Im Gegensatz zu Excel oder Word wird der Name des letzten Bearbeiters („zuletzt gespeichert von“) nicht in den Access-Dateieigenschaften gespeichert. Trotz dem kontinuierlichen Speichern empfiehlt MS weiterhin das Anlegen von regelmäßigen Sicherungskopien (*siehe Kapitel „3.5 Auswahl Office-Vergleichsprogramm zur Schnittstellenanalyse (Access Excel)“*). Manuelle Sicherungskopien können entweder durch Kopieren der Datei oder über die Access-Oberfläche unter „*Datei → Speichern unter → Datenbank sichern*“ erstellt werden. Bei Erstellung einer Sicherungskopie über die Access-Oberfläche ist ein exklusiver Zugriff erforderlich, Nutzende können in der Zwischenzeit nicht mit der Datei arbeiten. Die Wiederherstellung einer beschädigten Access-Datei erfolgt durch einfaches Ersetzen mit der entsprechenden Sicherungskopie. Wiederherzustellende Objekte können aus einer Sicherungskopie über die Access-Oberfläche „*Externe Daten → Neue Datenquelle → Aus Datenbank → Access → Importieren Sie Tabellen, Abfragen, Formulare, Berichte, Markos, und Module in die aktuelle Datenbank*“ ausgewählt und in die beschädigte Access-Datei importiert werden [207].

Im Gegensatz zu Access wird die Datensicherung einer Azure SQL-DB automatisch von MS und je nach gewähltem Kaufmodell und Konfiguration ohne Zusatzkosten übernommen. Transact SQL-Befehle zur Datensicherung im SQL Server werden von der Azure SQL-DB nicht unterstützt. Durch dieses Vorgehen ist gewährleistet, dass frühere Zustände des DBS zu jedem verfügbaren Zeitpunkt innerhalb der Aufbewahrungsfrist wiederhergestellt werden können („Point-in-Time-Wiederherstellung“). Mittels Point-in-Time-Wiederherstellung können unerwartete Datenmanipulationen oder die Korrumpierung einer DB rückgängig gemacht, aber auch alte Stände wiederhergestellt oder der Test-Prozess unterstützt werden. Dabei wird eine wiederhergestellte Version stets als neue DB hinzugefügt, die alte DB muss manuell gelöscht werden. Per Default liegt die Aufbewahrungsfrist für Point-in-Time-Wiederherstellungen bei 7 Tagen, sie kann aber auf Werte zwischen 1 und 35 Tagen angepasst werden. Die Aufbewahrungsfrist der Sicherungskopien kann mittels Langzeitaufbewahrungsfunktionalität auf bis maximal 10 Jahre verlängert werden. Im Rahmen der Langzeitaufbewahrung können separate Aufbewahrungsfristen für wöchentliche, monatliche (erste Sicherung jedes Monats) und jährliche Sicherungen konfiguriert werden. Nach Neuanlage einer DB erfolgt das erste vollständige Backup, welches standardmäßig in wöchentlicher Frequenz wiederholt wird. Differenzielle Backups, in denen nur die Datenunterschiede zur letzten Vollsicherung gesichert werden, erfolgen je nach Konfiguration alle 12 oder 24 Stunden. Transaktionslogs werden alle 5-10 Minuten gesichert:

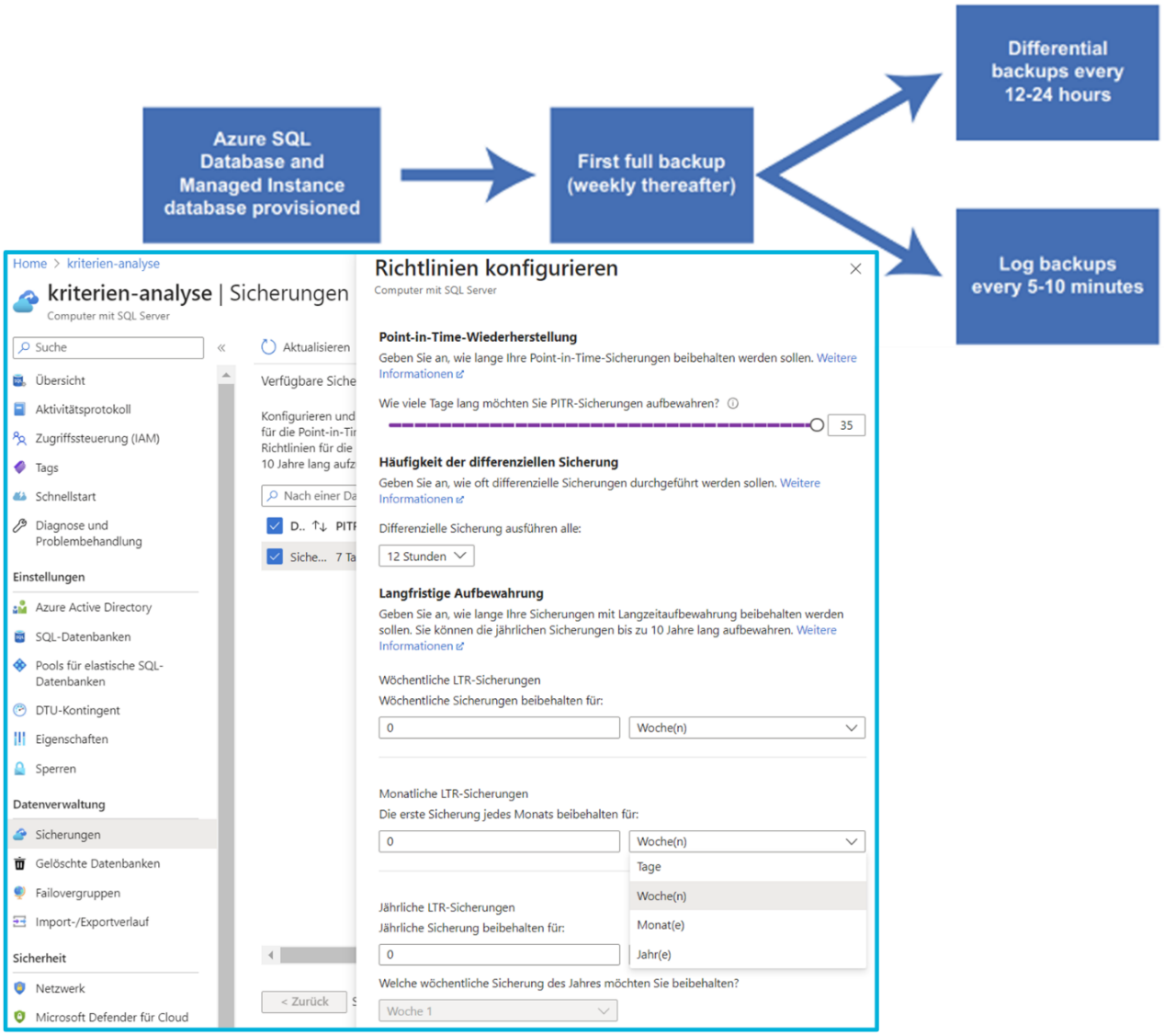


Bild 100:Oben: Automatische Datensicherung in der Azure SQL-DB [1, S. 156]; Unten: Konfigurationsmöglichkeiten der Aufbewahrungsfristen einer Azure SQL-DB über das Azure-Portal

Vorhandene Sicherungen können im Azure-Portal unter dem jeweiligen SQL Server unter „Sicherungen“ wiederhergestellt werden. Datensicherungen können aber auch beim Erstellen einer neuen Azure SQL-DB verwendet werden. Gelöschte DBs, bei denen die Aufbewahrungsfrist noch nicht abgelaufen ist, können unter „Gelöschte Datenbanken“ wiederhergestellt werden. Zusätzlich zu den automatisierten Sicherungen kann eine DB manuell als .bacpac- oder .dacpac-Datei für zusätzliche Datensicherungszwecke exportiert und in andere DBSs importiert werden. Ist die Geo-Replikationsfunktionalität aktiviert, kann eine kontinuierlich synchronisierte, lesbare sekundäre DB (Geo-Replikat) auf Grundlage einer primären DB erstellt werden. Dabei kann sich das Geo-Replikat in derselben aber auch in einer anderen Region wie die primäre DB befinden (Schutz vor großflächigen Ausfällen in einer Region) [1, S. 29, S. 156-157, S. 194, S. 214]/[79].

ACHTUNG: Auch in Azure SQL setzt MS auf Telemetrie:

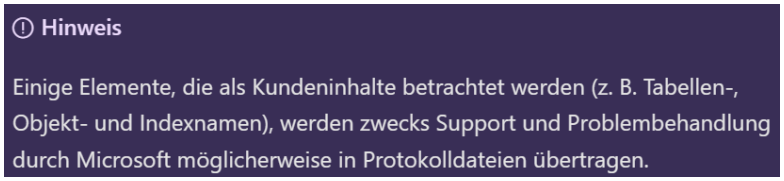


Bild 101:Hinweis von MS zu Telemetrie in Azure SQL [226]

4.1.9. Banking 4.0

MS Access-Dateien im Dateiformat 2007-2016 besitzen eine maximale Dateigröße von 2 Gigabyte und es können theoretisch maximal 255 gleichzeitige Nutzende verwaltet werden, wobei MS maximal 25-50 Nutzende empfiehlt (*siehe Kapitel „3.4 Auswahl Vergleichs-DBS zur Kriterienanalyse (Azure SQL-DB)“*). Auch die Gesamtanzahl von Objekten, wie Abfragen, ist in einer Access-DB auf 32.768 beschränkt. Als Workaround zur Umgehung der Limitierungen können die Daten auf mehrere miteinander verknüpfte Access-Dateien verteilt werden, was allerdings die Komplexität und den Verwaltungsaufwand erhöht und die Transparenz negativ beeinflusst, da die Daten nun in mehreren Dateien verortet sind. Durch die genannten Umstände ist Access ungeeignet für große Datenmengen und eine große Nutzerzahl [173].

Dazu kommt, dass Access nicht in der Cloud betrieben werden kann, von MS angebotene Alternativen hierfür sind (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung“ für eine vollständige Liste aller Bereitstellungs-Möglichkeiten*) [201]:

- Migration der Daten in eine Cloud-DB wie Azure SQL (*siehe Kapitel „6 Handlungsempfehlungen & Ausblick“*). Access wird dann nur noch als Frontend genutzt, das die Daten abruft, verarbeitet und bei Bedarf ausgibt. Auf der Access-Oberfläche können in der jeweiligen Access-Datei gespeicherte Daten unter *„Externe Daten → Weitere Optionen → ODBC-Datenbank“* in eine ODBC-Datenquelle wie den SQL Server migriert werden.
- Migration der Daten in das „Dataverse“ (*siehe Kapitel „6 Handlungsempfehlungen & Ausblick“*). Beim Dataverse handelt es sich um eine Clouddatenbank, auf der „Power Plattform-Apps“, „automatisierte Workflows“ oder „virtuelle Agenten“ erstellt werden können. Auch hier wird Access dann nur noch als Frontend genutzt [190]/[219]. Auf der Access-Oberfläche können in der jeweiligen Access-Datei gespeicherte Daten unter *„Externe Daten → Dataverse“* ins Dataverse exportiert werden:

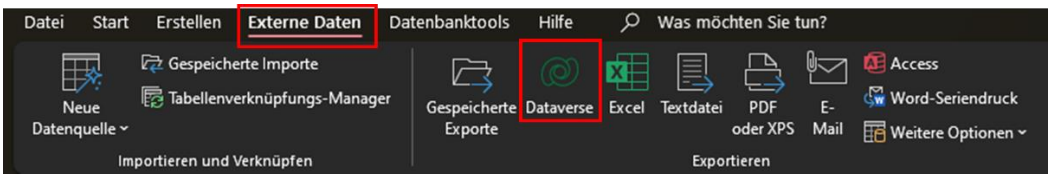


Bild 102: Über die Access-Oberfläche erreichbare Dataverse-Funktionalität

- Die Nutzung von Access als Web-App in SharePoint, um über den Web-Browser auf die Access-Datei zuzugreifen zu können, wird nicht weiter betrachtet, da die

Nutzung von MS nicht mehr empfohlen und die Integrationsunterstützung in den nächsten Versionen entfernt wird [219]).

- MS Teams besitzt ein Dataverse for Teams Built-in sowie einen Organisations-App-Store über den Eigenentwicklungen verteilt werden können (*siehe Kapitel „6 Handlungsempfehlungen & Ausblick“*) [77]/[140]

Da Access ein dateibasiertes DBS ist, kann die Zentralisierung der Daten nur durch die verpflichtende Ablage aller Access-Dateien in einem durch Unternehmensrichtlinien vorgegebenen Verzeichnis erfolgen. Die Dateien können dann in regelmäßigen Abständen in ein Client-Server DBS migriert werden, um die Zentralisierung und Sicherheit der Daten zu erhöhen (*siehe Kapitel „2.5 Unterschied dateibasierte & Client-Server DBS“*). Ansonsten bietet Access keine Möglichkeit die Zentralisierung der Datenspeicherung zu unterstützen. Jede Access-Datei enthält eine DB:

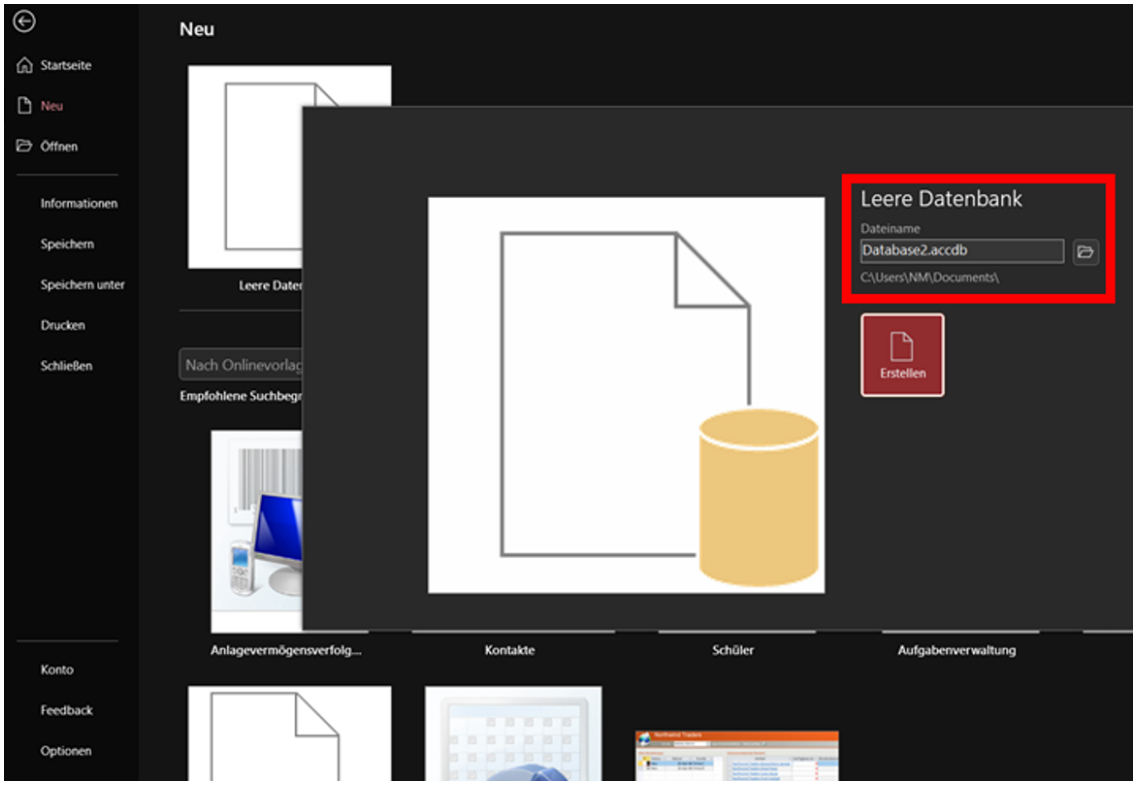


Bild 103: Neue Access-DB erstellen, eine Datei enthält genau eine DB

Je mehr Dateien verwaltet werden müssen, desto größer ist der Verwaltungsaufwand. Insbesondere, wenn eine DB auf mehrere Access-Dateien verteilt ist. Dies erschwert die Umsetzung der MaRisk-Anforderung, dass Risikodaten zweifelsfrei identifiziert, zusammengeführt und ausgewertet werden müssen sowie zeitnah zur Verfügung stehen. Aber auch für Nicht-Risikodaten ist diese Vorgabe mit zunehmender Kritikalität und Bedeutung durchaus relevant. Die dezentrale Datenablage erschwert auch die konsequente Einhaltung sowie Prüfung von einheitlichen Namenskonventionen, was sich negativ auf Data Mining-Tätigkeiten sowie die Erstellung von Trainingsdaten für eine Banken-KI auswirkt. Außerdem entsteht das Risiko von Redundanzen, Dateninkonsistenzen und somit einer widersprüchliche Datenlage. Die genannten Umstände wirken

sich obendrein negativ auf die Einhaltung der MaRisk-Vorgabe aus, die Verfahren zur Sicherstellung der Qualität von Modelldaten fordert.

Aufgrund der in *Kapitel „4 Durchführung der Sicherheitsanalyse“* festgestellten Schwachstellen und fehlenden Funktionalitäten, wie Sicherheit auf Benutzerebene oder umfangreicher Multi-User-Support, ist MS Access nur in Ausnahmefällen geeignet für Web-Anwendungen, Zugriffe über das Internet und somit API-Lösungen (*siehe zusätzlich Kapitel „1.2 Abgrenzung“*).

Die in *Kapitel „2.7 Die Zukunft: Banking 4.0 inklusive Historie“* genannten Grundpfeiler der Industrialisierung (darunter Standardisierung, Konsistenz, Wiederbenutzung, Flexibilität, Robustheit/Ausfallsicherheit, Selbstkorrektur, Selbstversorgung, Echtzeit sowie Massenverarbeitungsfähigkeit) können von Access nur beschränkt oder nicht erfüllt werden, was im Widerspruch zur Vision Banking 4.0 steht und neue technische Schulden erzeugt. Eine Access-Datei kann durch diverse Ereignisse schnell in einen inkonsistenten Zustand kommen und die Ausfallsicherheit negativ beeinflussen (für Transaktionen als potenzielle Quelle für Inkonsistenzen *siehe Kapitel „4.1.1 SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen“*).

Möchten Nutzende in die DB schreiben, werden zwingend Schreibrechte auf die Datei benötigt. So kann die Datei absichtlich manipuliert, aber auch versehentlich verschoben oder gelöscht werden. Dieser Umstand wirkt sich negativ auf die Robustheit und Ausfallsicherheit aus. Die in VBA implementierten Schutzmechanismen werden nur dann ausgeführt, wenn direkt über die entsprechende Access-Datei oder das zugehörige Frontend gearbeitet wird. Beim Zugriff über Drittanbietertools wie DbVisualizer besteht Vollzugriff und Ereignisse oder VBA-Routinen, wie selbsterstellte Logging-Funktionalitäten, werden nicht ausgeführt. Dateibasierte DBSs können bereits mit Leserechten kopiert und der Angriff an einem ungestörten Ort fortgesetzt werden. Der Vollständigkeit halber sei auch auf die restlichen, während der Sicherheitsanalyse gefundenen Schwachstellen verwiesen, die sich negativ auf die Robustheit auswirken.

Da jede Access-Datei eine DB enthält, ist eine Wiederverwendung nur eingeschränkt möglich. Dazu muss die Datei kopiert werden. Beim Kopieren können Redundanzen entstehen. Die Dateien werden im ungünstigsten Szenario an unterschiedlichen Speicherorten abgelegt und es können voneinander abweichende oder sogar widersprüchliche Datenbestände entstehen.

Um Trainingsdaten für eine KI zu erstellen, müssen die Daten mühsam aus mehreren Access-Dateien zusammengesucht und -geführt werden, wobei alle Ablageorte bekannt sein und Inkonsistenzen oder Widersprüche behoben sein müssen.

Aufgrund der fehlenden Eignung für Web-Anwendungen, des obligatorischen Zugriffs über Netzlaufwerke und des eingeschränkten Multi-User-Supports ist Access nur bedingt flexibel einsetzbar.

Da ein dateibasiertes DBS im Gegensatz zu Client-Server-Lösungen nicht als eigenständiger Prozess permanent im Hintergrund läuft und darauf wartet eingehende Anfragen zu beantworten (im Unix-Umfeld auch Daemon, unter Windows Dienst/Service genannt), ist in Access nur eine halbautomatische Selbstkorrektur oder Selbstversorgung möglich,

wenn die Datei geöffnet ist. Da Access anders als die Azure SQL-DB keine automatisierten Überwachungsfunktionen bietet, sind Selbstkorrekturen auf Contraints, Primär- und Fremdschlüsselbezüge sowie Eingabeprüfungen und VBA-Eigenentwicklungen beschränkt.

Aufgrund der zuvor genannten Restriktionen, insbesondere bezüglich maximaler Nutzerzahl und Dateigröße ist Access höchstens im kleinen Umfang für eine Echtzeitverarbeitung geeignet und ungeeignet für Massenverarbeitungen.

Im Rahmen von Platform-as-a-Service steht in der Azure-Cloud auch eine vollständige Entwicklungs- und Deployment-Umgebung von eigenen Anwendungen zur Verfügung (*siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software“*). So kann die für Banking 4.0 (*siehe Kapitel „2.7 Die Zukunft: Banking 4.0 inklusive Historie“*) präferierte Microservice-Architektur durch „Azure Spring Cloud“ umgesetzt werden. Die zur Umsetzung des Banking-as-a-Service-Gedankens relevanten APIs können durch ein eigenes API-Management verwaltet und auch vor potenziellen Gefahrenquellen, wie Denial-of-Service-Angriffen, geschützt werden.

Echtzeitverarbeitung sowie Big Data-Verarbeitung, wie in Form von „Azure Data Lake Storage“, werden ebenfalls unterstützt. Die von der Azure-Cloud angebotenen KI-Dienste können in die selbstentwickelten Anwendungen integriert werden und Aufgaben in Prozessen wie Analysen, Entscheidungsfindungen oder die Absicherung von Applikationen und Daten übernehmen (*siehe Kapitel „4.1.1 SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen“*). Azure SQL-DB bietet mit „Azure Purview“ die Möglichkeit, vorhandene Datenquellen in einem zentralen Datenkatalog zu sammeln und zu klassifizieren. Die Nutzenden können die Daten mit verschiedenen Werkzeugen, wie einer Suchmaske oder einem Glossar, durchsuchen. Darüber hinaus können den Daten Labels (Angaben zur Kritikalität und zusätzliche Details zum Attribut) zugeordnet werden. Mit Labels können sensible Daten markiert und zusätzlich geschützt werden. Das erleichtert die Implementierung von Verfahren zur Qualitätssicherung von in Modellen verwendeten Daten. Die Daten sind bekannt, auffindbar und jederzeit abrufbar. Es können auch Überwachungsfunktionen implementiert werden, die Zugriffe manipulationssicher in Audit-Logs protokollieren. Weitere Ziele der „Data Discovery & Classification“-Funktionalität sind die Erfüllung datenschutzrechtlicher oder regulatorischer Anforderungen wie, dass Risikodaten zweifelsfrei identifiziert, zusammengeführt und ausgewertet werden müssen [51, S. 7, S. 20, S. 58, S. 62, S. 71, S. 79, S. 86-92, S. 104]/0.164, S. 280-285]:

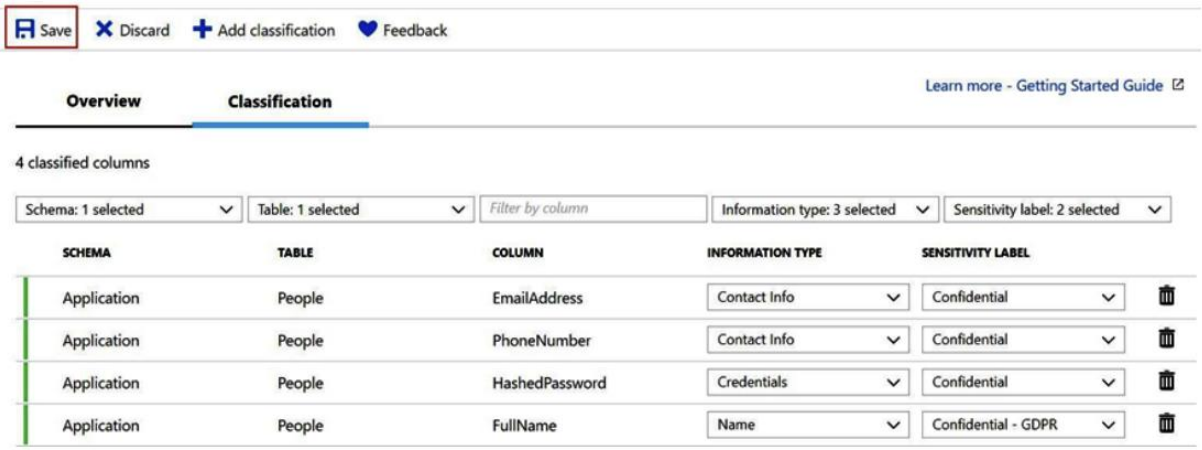


Bild 104: Azure Purview – Discovery & Classification-Funktionalität [1, S. 284]

Die maximale Speicherkapazität hängt vom gewählten Kaufmodell ab und liegt im günstigsten Modell („Basic“, circa 5,04 €/Monat) bei 2 Gigabyte. Im Standardmodell für Workloads mit typischen Leistungsanforderungen beträgt die maximale Datengröße bis 50 Datenbanktransaktionseinheiten (Kombination aus CPU, Arbeitsspeicher, Lese- sowie Schreibvorgängen) bei 250 Gigabyte. Ab 100 Datenbanktransaktionseinheiten können maximal 1.024 Gigabyte gebucht werden. Bei anderen Kaufmodellen, wie Premium oder Abrechnung nach virtuellen Kernen, können maximal 4.096 Gigabyte gewählt werden. Der zugehörige Protokollspeicherplatz hängt von der ausgewählten Datengröße ab und liegt im Kaufmodell „Allgemein (Skalierbare Computer- und Speicheroptionen)“ zwischen 614,4 Megabyte bei 2 Gigabyte Datengröße und 1 Terrabyte bei 4.096 Gigabyte Datengröße. Mit der Hyperscale-Dienstebene (hier sind mehr Computer- und Speicherressourcen als auf den Dienstebenen „Universell“ oder „Unternehmenskritisch“ verfügbar) wird eine flexible Speicherarchitektur angeboten, in der die Datengröße auf bis zu 100 Terrabyte erhöht werden kann. Hyperscale-DBs werden ohne Definition einer maximalen Größe erstellt. Es erfolgt ein automatisches Skalieren des Speichers mit Unterstützung von Pools für elastische DBs. Im Gegensatz zu herkömmlichen DBSs sind Daten und Logs bei Nutzung der Hyperscale-Dienstebene nicht auf dem DB-Server abgelegt [1, S. 68]/[112].

4.2. Schnittstellenanalyse (Access Excel)

Moderne kryptographische Verfahren sind ohne die Verwendung von Quantencomputern sicher und robust. In vielen Fällen stellen kryptographische Verfahren daher nicht das primäre Angriffsziel dar. Die Sicherheit und Robustheit von modernen kryptographischen Verfahren wird auch durch das Kerckhoffs’sche Prinzip gewährleistet. Es besagt, dass die Sicherheit eines Verschlüsselungsverfahrens von der Geheimhaltung des Schlüssels und nicht vom verwendeten Algorithmus abhängen darf. Daher wird in diesem Kapitel anstelle des Verschlüsselungsverfahrens die Schnittstelle angegriffen.

Recherchen zum ausgewählten Office-Programm Excel für die Schnittstellenanalyse *(siehe Kapitel „3.5 Auswahl Office-Vergleichsprogramm zur Schnittstellenanalyse (Access Excel)“)* haben Schwachstellen aufgedeckt, die auf verschiedenen Websites, wie

der Internetplattform für Softwareentwickler *Stack Overflow*, publiziert und teilweise schon seit dem Jahr 2011 bekannt sind. Zwar sind die auf den Webseiten beschriebenen Schwachstellen nicht mehr mit dem dort skizzierten Vorgehen ausnutzbar, können aber mit abgewandelter Vorgehensweise derzeit noch für einen Angriff genutzt werden. Sie hebeln ohne großen Aufwand viele wichtige Schutzmechanismen aus, mit denen auch Informationen für einen weiteren Angriff auf ein Access-Backend gesammelt werden können.

Excel, Word und PowerPoint basieren seit 2007 auf dem Open XML-Format (erkennbar an dem „x“ in den zugehörigen Dateierweiterungen wie .xlsx, .docx und .pptx). Das Open XML-Format ist ein Dateninteroperabilitätsframework, die Daten werden im XML-Format gespeichert und können so ohne Einschränkungen kostenlos verwendet werden. Auch die Entwicklung wird erleichtert, da in einem Office-Dokument gespeicherte Informationen aufgrund des XML-Formats leichter von anderen Anwendungen verwendet werden können. Zum Öffnen ist ein Zipper (ZIP)-Hilfsprogramm wie „WinRar“ sowie ein Text-Editor ausreichend. Im Open XML-Format sind die Dateien modular aufgebaut, die verschiedenen Komponenten wie ein Diagramm, eine Tabelle oder Bilder sind voneinander getrennt. Außerdem erfolgt mittels ZIP-Komprimierungstechnologie eine Komprimierung der Datei. Als Schutzmechanismus können nur Open XML-Formate mit einem „m“ als Suffix in der Dateierweiterung VBA-Code enthalten [202].

Im ersten Beispiel wird das Passwort aus dem VBA-Projekt einer Excel-.xlsm-Frontend-Datei entfernt, sodass das Projekt danach im Vollzugriff vorliegt. Im Gegensatz zu Access besitzt Excel unter „*Datei → Speichern unter*“ keine Möglichkeit, um Dateien zu kompilieren und das VBA-Projekt so vor Zugriffen zu schützen. Eine Alternative kann das .xlsb-Dateiformat sein, das die Datei im Binärformat „BIFF12“ speichert. Hier bedarf es einer separaten Analyse, da sich in diesem Kapitel auf die gängigen Dateitypen .xlsx und .xlsm beschränkt wird (*siehe Kapitel „1.2 Abgrenzung“*) [63].

Als Ausgangspunkt dient die Empfehlung von MS, Access Excel einzusetzen (*siehe Kapitel „3.5 Auswahl Office-Vergleichsprogramm zur Schnittstellenanalyse (Access Excel)“*). Das .xlsm-Frontend enthält die Logik in Form von VBA-Code inklusive eines Passworts für das verschlüsselte .accdb oder .accde Access-Backend. Durch Ändern der Excel-Dateierweiterung im Dateinamen von .xlsm auf .zip kann die Datei als Archiv geöffnet werden.

Wichtig für das erfolgreiche Entfernen des Passworts für den Projektschutz ist, dass das .zip-Archiv nicht manuell über Windows 10 und 11-Bordmittel gemäß dem folgenden Algorithmus entpackt und manipuliert wird:

a) *Rechtsklick auf .zip-Archiv → Alle extrahieren → Dialog bestätigen.*
b) *Öffnen und Manipulieren der gesuchten Datei in dem in Schritt a) entpackten Verzeichnis.*
c) *Rechtsklick auf das in Schritt b) manipulierte Verzeichnis → In ZIP-Datei komprimieren.*
d) *Umbenennen der Dateierweiterung im Dateinamen des in Schritt c) erstellten Archivs von .zip auf .xlsm.*

In diesem Fall hat MS den Bug behoben und die Manipulation wird erkannt. Die Excel-Datei kann aufgrund folgender Fehlermeldung nicht mehr geöffnet werden [251]:

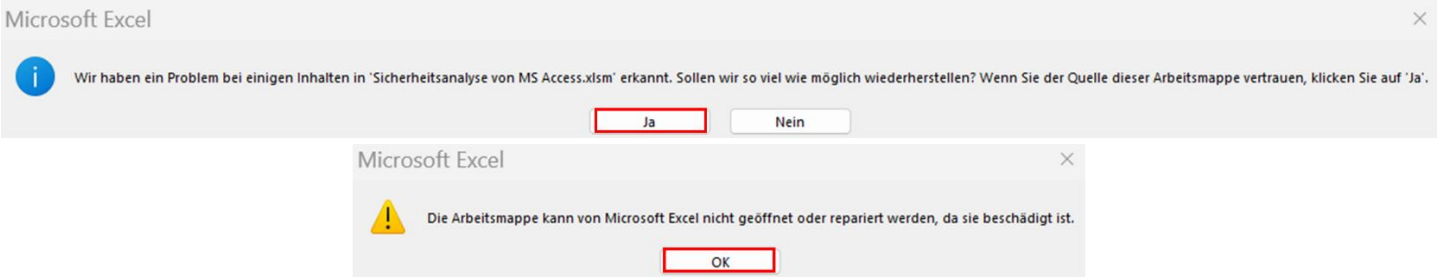


Bild 105: Excel – Erkennung von Manipulationen durch falsches Vorgehen

Damit die Manipulation von Excel nicht erkannt wird, ist folgender Algorithmus anzuwenden (*für eine ausführliche Anleitung siehe Kapitel „Anlage 7: Entfernen von VBA-Projekt-Passwortschutz in Excel (Access Excel)“*):

1) *Öffnen des .zip-Archivs in WinRAR (siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software“).*
2) *Öffnen der gesuchten Datei über die WinRAR-Oberfläche mittels Standard-Editor (bei .bin Hexadezimal-Editor, bei .xml Text-Editor), in diesem Fall „Visual Studio Code“ (siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software“). Manipulieren der Datei, ohne das Archiv vorher zu entpacken.*
3) *Nach Speichern der Änderungen muss die Datei im Archiv erneuert werden. Hierfür wird der automatisch in WinRAR erscheinende Dialog mit „Ja“ bestätigt.*

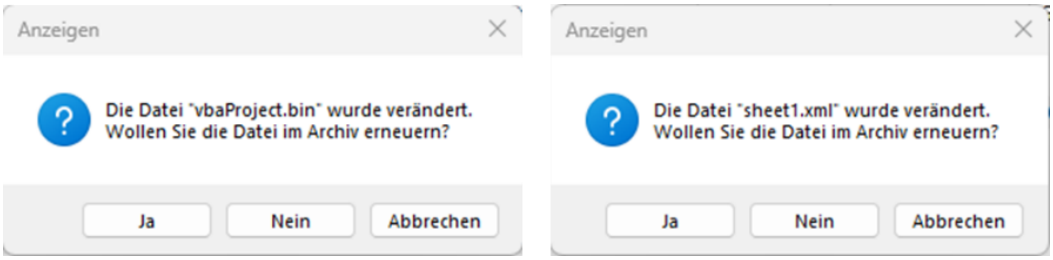


Bild 106: Excel – Erkennung von Manipulationen durch richtiges Vorgehen verhindern

4) *Die Dateierweiterung im Dateinamen des über WinRAR in Schritt 2) manipulierten .zip-Archivs wird wieder in die ursprüngliche Dateierweiterung .xlsm geändert und die Datei geöffnet. Die Datei kann problemlos geöffnet werden, es erscheint keine Fehler- oder Warnmeldung.*

Zum Entfernen des VBA-Projekt-Passwortschutzes wird in Schritt 2) die Datei unter „`xl\vbaProject.bin`“ mit einem Hexadezimal-Editor geöffnet, nach „*DPB*“ gesucht und der letzte Buchstabe des Treffers durch einen beliebigen anderen Buchstaben ersetzt:

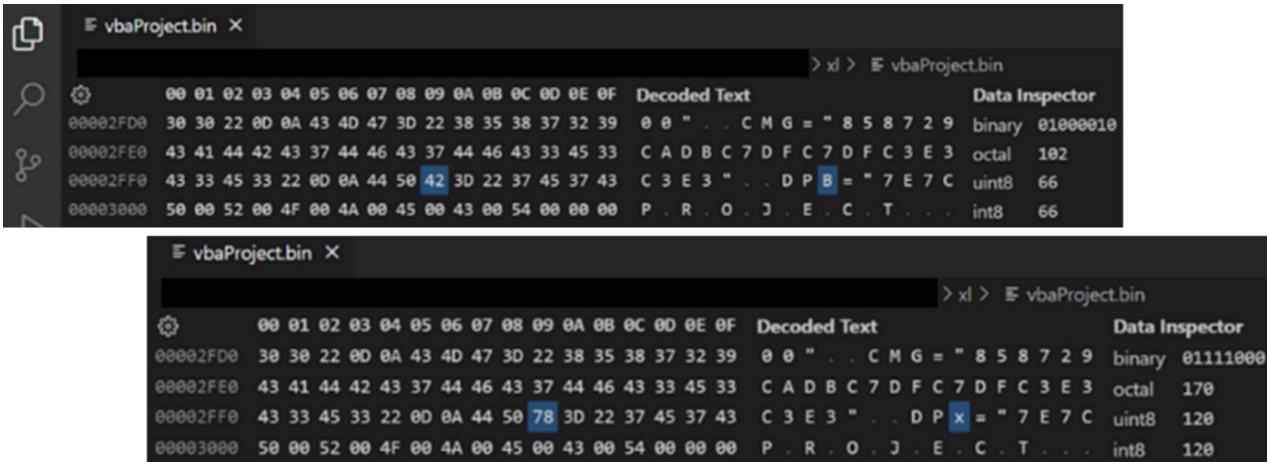


Bild 107: Excel – Änderungen an der Datei „xl\vbaProject.bin“ im Hexadezimal-Editor, um den VBA-Projekt-Passwortschutz zu entfernen (Oben: Vor Anpassung; Unten: Nach Anpassung)

Nach der Anpassung wird die Dateierweiterung im Dateinamen des .zip-Archivs wieder zu .xlsm geändert. Nach dem Öffnen der Datei und Klick auf den Kopfreiter „*Entwicklertools → Visual Basic*“ erscheint folgender Dialog:

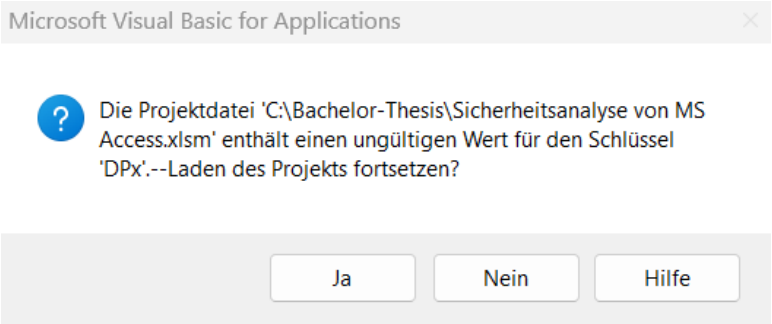


Bild 108: Excel – Fehlermeldung nach den Änderungen an der Datei „xl\vbaProject.bin“, um den VBA-Projekt-Passwortschutz zu entfernen

Nach Bestätigen und mehrmaligem Akzeptieren einer darauffolgenden Warnmeldung öffnet sich der VBA-Editor. Unter „*Extras → Eigenschaften von VBAProject… → Kopfreiter Schutz*“ wird die Option „Projekt für die Anzeige sperren“ deaktiviert. Zu beachten ist, dass das zuvor unter „Kennwort“ vergebene Passwort nicht mehr vorhanden ist:

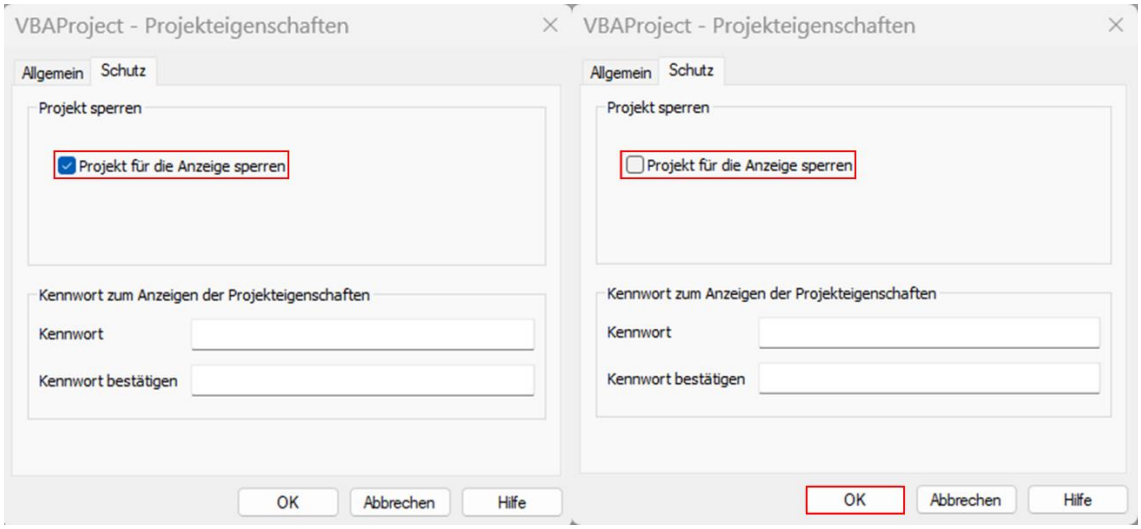


Bild 109: Excel – Ergebnis der erfolgreichen Änderung an der Datei „xl\vbaProject.bin“, um den VBA-Projekt-Passwortschutz zu entfernen (Links: Vor Anpassung; Rechts: Nach Anpassung)

Eine Kopie der Datei wird gespeichert und geöffnet. Der Passwortschutz des VBA-Projekts ist nun entfernt, der Zugriff auf den Quellcode möglich. Als Angriffsszenario kann der Quelltext um Schadcode erweitert und die Datei unbemerkt ausgetauscht werden. In diesem Fall würde der Schadcode auf den Endgeräten aller Nutzenden ausgeführt. Sind im VBA-Code vertrauliche Verbindungsinformationen zu Datenquellen, wie das Passwort für ein verschlüsseltes Access-Backend oder Anmeldedaten an einen SQL Server enthalten, können sie durch diesen Angriff eingesehen werden [251].

Auch der von MS Excel unterstützte Datenschutz auf Benutzerebene durch Ausblenden von Zeilen und Spalten sowie das anschließende Aktivieren des Blattschutzes bieten keine Sicherheit, denn der Blattschutz kann ebenso leicht wie das Passwort des VBA-Projekts entfernt werden (*für eine ausführliche Anleitung siehe Kapitel „Anlage 8: Entfernen von Blattschutz in Excel (Access Excel)“*). Dazu wird die Dateierweiterung im Excel-Dateinamen von .xlsm (funktioniert auch mit .xlsx) auf .zip geändert. Die Datei „`xl\worksheets\sheet1.xml`“ wird wieder über WinRAR und Visual Studio Code als Text-Editor geöffnet und das gesamte Element „`<sheetProtection>`“ mit Attributen gelöscht:

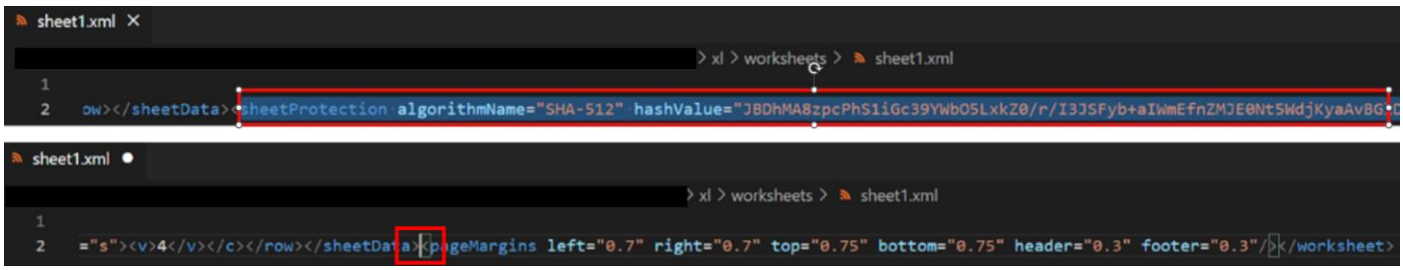


Bild 110: Excel – Änderungen an der Datei „xl\worksheets\sheet1.xml“ im Hexadezimal-Editor, um den Blattschutz zu entfernen (Oben: Vor Anpassung; Unten: Nach Anpassung)

Nach den Änderungen wird die Dateierweiterung im Dateinamen des über WinRAR manipulierten Archivs wieder von .zip auf .xlsm geändert. Die manipulierte Excel-Datei

wird wie gewohnt geöffnet, der Blattschutz ist entfernt und zuvor ausgeblendete Zellen mit vertraulichen oder personenbezogenen Daten oder Verbindungsinformationen zum DB-Backend können problemlos eingeblendet und der Inhalt eingesehen oder verändert werden. Unter Ausnutzung beider Schwachstellen können Angreifende vollständigen Zugriff auf den Inhalt und die Logik der Excel-Datei erlangen, wenn die Datei nicht mit einem starken Passwort verschlüsselt ist.

4.3. Dateiformatanalyse mittels Hexadezimal-Editor (Access)

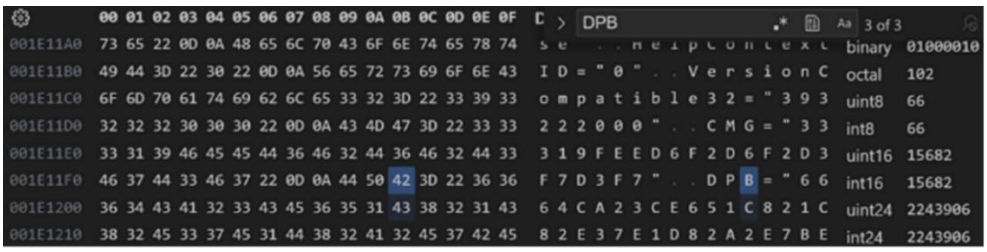
Ziel dieses Kapitels ist es, Schwachstellen im Access-Dateiformat 2007-2016 zu identifizieren (*siehe Kapitel „2.4 Informationen zu Access, JET, ACE/ADE & .accdb-Dateiformat“*).

Access-Dateien basieren anders als Excel, Word und PowerPoint nicht auf dem Open XML-Format und lassen sich nicht in ein .zip-Archiv konvertieren (*siehe Kapitel „4.2 Schnittstellenanalyse (Access Excel)“*). Access-Dateien liegen im Binärformat vor und können daher ohne Umwege direkt im Hexadezimal-Editor geöffnet und manipuliert werden (*siehe Kapitel „2.5 Unterschied dateibasierte & Client-Server DBS“*).

Die im *Kapitel „4.2 Schnittstellenanalyse (Access Excel)“* durch das Excel-Frontend gewonnenen Informationen, können in diesem Kapitel für einen Angriff auf das Access-Backend weiterverarbeitet werden.

Dabei ist zu beachten, dass das Aufheben des Passwortschutzes für das VBA-Projekt nur bei .accdb-Dateien funktioniert. Bei .accde-Dateien ist generell kein Zugriff auf das VBA-Projekt möglich (*für eine ausführliche Anleitung siehe Kapitel „Anlage 9: Entfernen von VBA-Projekt-Passwortschutz in Access (nur .accdb)“*).

Zum Entfernen des VBA-Projekt-Passwortschutzes wird analog der Anleitung im *Kapitel „4.2 Schnittstellenanalyse (Access Excel)“* die .accdb-Datei mit einem Hexadezimal-Editor geöffnet, nach „*DPB*“ gesucht und der letzte Buchstabe der Treffer wie bei der Schnittstellenanalyse durch einen beliebigen anderen Buchstaben ersetzt. Im Gegensatz zu Excel resultiert die Suche in drei Treffern, die einzeln nacheinander ausprobiert werden:



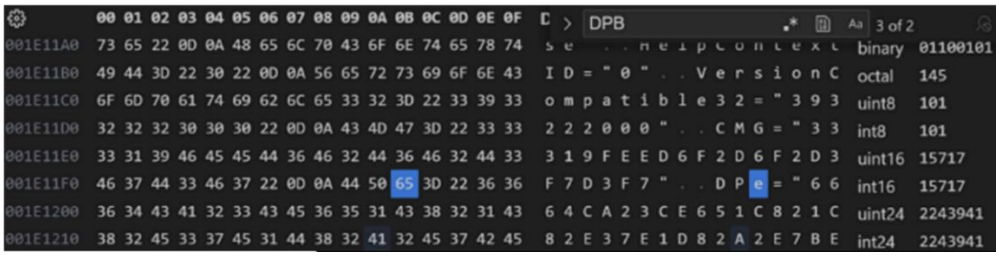


Bild 111: Änderungen an der .accdb-Datei im Hexadezimal-Editor, um den VBA-Projekt-Passwortschutz zu entfernen (Oben: Vor Anpassung; Unten: Nach Anpassung)

Die drei Treffer sind unabhängig voneinander, da erst beim dritten Suchergebnis beim Öffnen des VBA-Editors eine Fehlermeldung ähnlich zu *Kapitel „4.2 Schnittstellen-analyse (Access Excel“* erscheint:

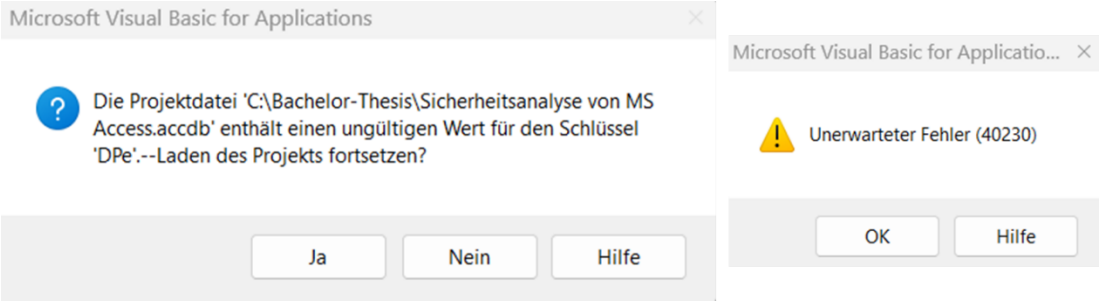


Bild 112: Links: Fehlermeldung nach den Änderungen an der .accdb-Datei, um den VBA-Projekt-Passwortschutz zu entfernen; Rechts: Warnmeldung nach Bestätigung der Fehlermeldung

Nach Bestätigung und mehrmaligem Akzeptieren einer darauffolgenden Warnmeldung öffnet sich der VBA-Editor. Unter „*Extras → Eigenschaften von … → Kopfreiter Schutz*“ wird die Option „Projekt für die Anzeige sperren“ deaktiviert. Zu beachten ist, dass das zuvor unter „Kennwort“ vergebene Passwort nicht mehr existiert:

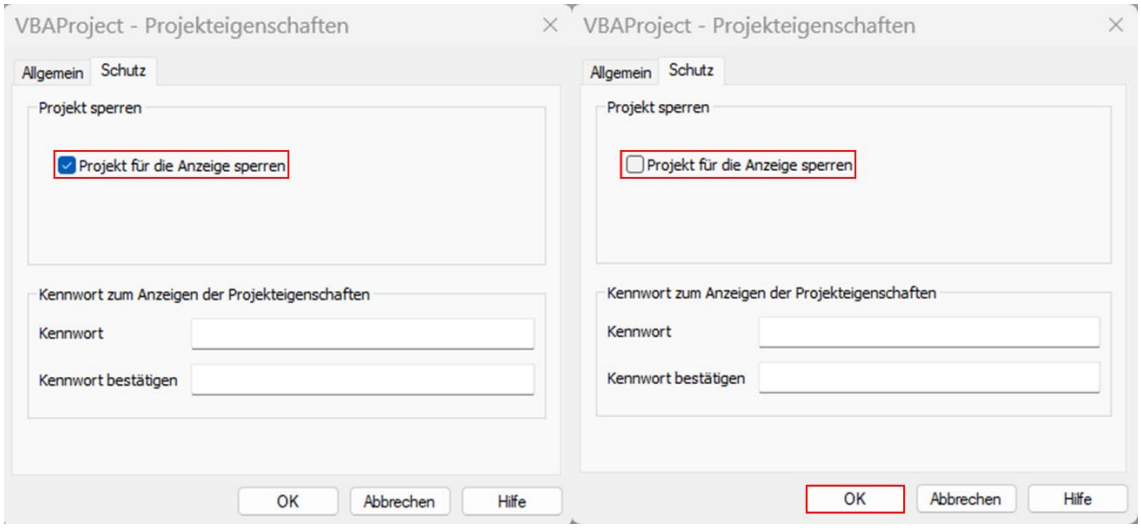


Bild 113: Erfolgreich entferntes Passwort nach den Änderungen an der .accdb-Datei, um den VBA-Projekt-Passwortschutz zu entfernen (Links: Vor Anpassung; Rechts: Nach Anpassung)

Eine Kopie der Datei wird gespeichert und geöffnet. Anders als bei Excel erscheinen bei Access mehrfach Fehlermeldungen, nach mehrmaligem Bestätigen öffnet sich der VBA-Editor und der Passwortschutz des VBA-Projekts ist entfernt. Im Gegensatz zu Excel kann der VBA-Code nicht anzeigt werden. Um den Inhalt der angezeigten Module und Klassenobjekte zu sehen, müssen sie über „*über Rechtsklick → Datei exportieren …*“ exportiert werden. Wird die Access-Datei, wie von MS empfohlen, als Frontend für ein Client-Server-DBS Backend wie die Azure SQL-DB (Datenhaltungsschicht) genutzt und wird nicht der Windows-Authentication-Modus oder Azure AD verwendet, können im Quellcode Verbindungsinformationen eingesehen werden [56]. Zusätzlich kann eine

Kopie der Datei erstellt werden, welche die zuvor exportierten und um Schadcode erweiterten VBA-Module hinzugefügt werden. Die Originaldatei wird unbemerkt mit der manipulierten Datei ausgetauscht, sodass alle Nutzenden den Schadcode bei Verwendung der Datei ausführen. Liegt die Datei unter einem vertrauenswürdigen Pfad (*siehe Kapitel „4.1.2 Konfiguration“*), wird die manipulierte Datei nicht in einer geschützten Ansicht geöffnet.

Bei näherer Betrachtung des Dateiformats einer unverschlüsselten Access-Datenquelle im Hexadezimal-Editor fällt auf, dass alle in .accdb- und .accde-Dateien hinterlegten Zeichenketten, wie konkrete Attributwerte von in Relationen hinterlegten Daten oder im VBA-Code hinterlegte Verbindungsinformationen, im Klartext gespeichert werden und über einen Hexadezimal-Editor ausgelesen werden können (*für eine ausführliche Anleitung siehe Kapitel „Anlage 10: Betrachtung des Dateiformats im Hexadezimal-Editor (Access)“*) [64]. Über den Hexadezimal-Editor sind auch über die Oberfläche gelöschte Datensätze einsehbar (vergleiche nachfolgender Screenshot „`Gelöschter Datensatz`“):

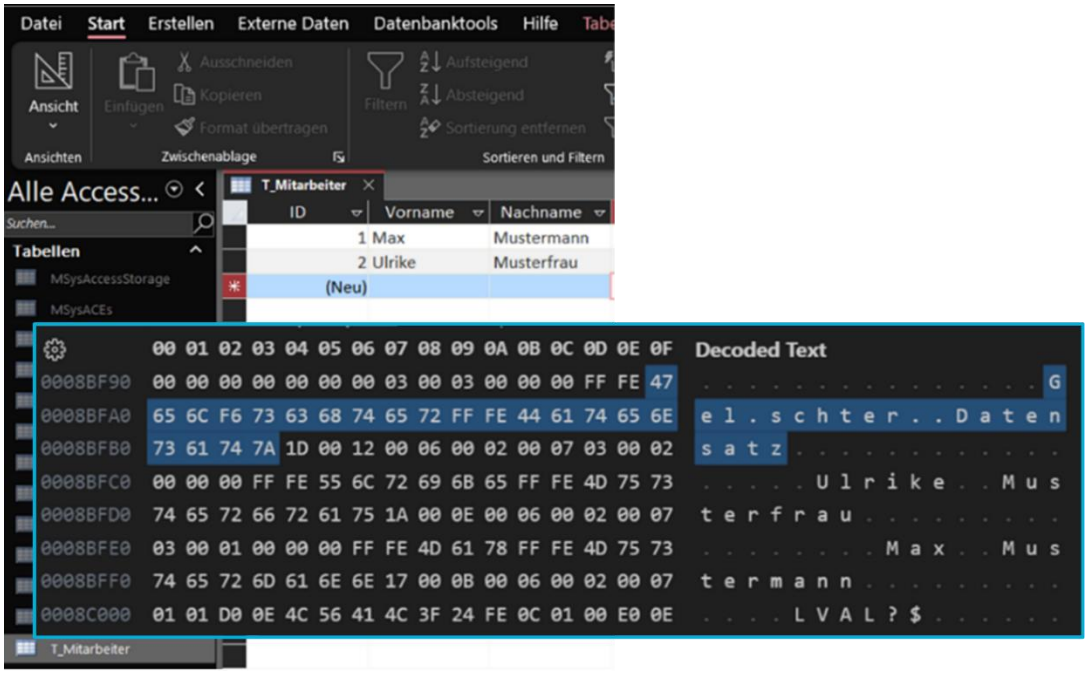


Bild 114: Hexadezimal-Editor-Analyse – Speicherung von Datensätzen im Klartext und Verfügbarkeit von gelöschten Datensätzen („Gelöschter Datensatz“)

Die gelöschten Daten werden erst mit der Funktion „Datenbank komprimieren und reparieren unter „*Datei → Informationen*“ entfernt. Dabei werden nicht die Daten in der DB komprimiert, sondern der ungenutzte Speicher wird bereinigt, in dem sich auch die gelöschten Daten befinden. Gleichzeitig kann durch die Bereinigung die Performance von großen Access-Dateien verbessert werden. Unter „*Datei → Optionen → Aktuelle Datenbank → Beim Schließen komprimieren*“, kann das automatische Bereinigen bei jedem Schließen der Datei aktiviert werden. Die Gefahr dabei ist, dass Access unter Umständen einige Daten aus den Tabellen entfernt, die möglicherweise als beschädigt markiert sind und somit unbemerkt Informationen verlorengehen. MS empfiehlt daher, vor jeder Ausführung der Funktion eine Sicherungskopie zu erstellen [215]. Während der

Bereinigung wird auch die Indexdatei neu erstellt (*siehe Kapitel „2.4 Informationen zu Access, JET, ACE/ADE & .accdb-Dateiformat“*). Eine weitere Maßnahme gegen unbefugten Zugriff auf gelöschte Daten ist die Dateiverschlüsselung oder das Einfügen von bereits verschlüsselten Daten. Für die Dateiverschlüsselung wird der Verschlüsselungsalgorithmus verwendet, der über die bereits zuvor erwähnte CryptoAPI konfiguriert wurde (*siehe Kapitel „4.1.3 Kryptographie“*). Das Passwort wird mit einem Salt verknüpft, ein Hash gebildet und der Hash mitverschlüsselt im Datei-Header am Dateianfang eingebettet. AES-256 sowie SHA-512 stellen die Standardkonfiguration dar:

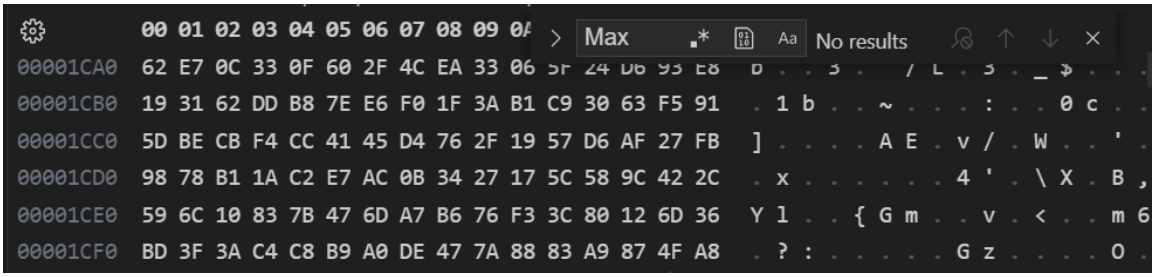


Bild 115: Hexadezimal-Editor-Analyse – Verschlüsselter Dateiinhalt und VBA-Code nach Aktivierung der Funktion „Mit Kennwort verschlüsseln“

4.4. Forensische Analyse (Access)

In diesem Kapitel wird das Verhalten des in *Kapitel „3.6 Auswahl forensisches Tool zur forensischen Analyse (Autopsy)“* ausgewählten forensischen Tools bei der Analyse von MS Access-Dateien untersucht. Der Fokus liegt darauf, ob Autopsy in der Standardkonfiguration alle Access-Testdateien findet und damit bestätigt wird, dass MS Access bei der forensischen Auswertung und dem Auffinden bisher unbekannter Dateien hinreichend unterstützt wird.

Als Dummy-Image dient ein 512 Megabyte großer Universal Serial Bus (USB)-Stick mit zufällig ausgewählten Ordnern und Dateien aus „`C:\Windows`“ und dem NTFS-Dateisystem.

Anhand des Dateinamens ist ersichtlich, ob die Dateien Eigenschaften wie signierten Code besitzen oder, ob sie gelöscht wurden. In alle Access-Testdateien ist in einer Tabelle eine Datei eingebettet (in Relationen eingebettete Dateien werden in Access als „Anhang“ bezeichnet). Aus Gründen der Übersichtlichkeit ist dieser Zusatz nicht in den Dateinamen hinzugefügt worden (*für eine ausführliche Anleitung, Details zur Vorbereitung der Testumgebung und Testdateien sowie zur Konfiguration von Autopsy siehe Kapitel „Anlage 11: Vorbereitung der Testumgebung für die forensische Analyse einer Access-Datei“*):

```
C:\>format Z: /fs:NTFS /p:1
Legen Sie eine neue Diskette in Laufwerk Z: ein,
und drücken Sie die EINGABETASTE.
Der Typ des Dateisystems ist NTFS.
Überprüfung von 496,9 MB
Volumebezeichnung (32 Zeichen, EINGABETASTE für keine)? Z
Struktur des Dateisystems wird erstellt.
Formatieren beendet.
     496,9 MB Speicherplatz auf dem Datenträger insgesamt.
     492,6 MB sind verfügbar.
```

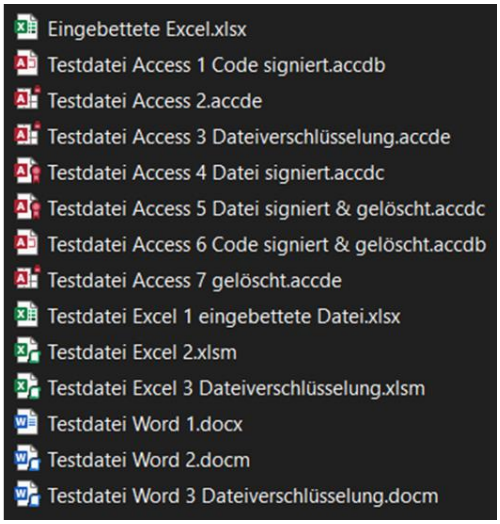


Bild 116: Links: Wipe des Dummy-USB-Sticks für die forensische Analyse und zugehörige Informationen; Rechts: Im Dummy-USB-Stick versteckte Testdateien

In der Testumgebung wird das NTFS genutzt, da es das primäre Dateisystem für alle aktuellen Versionen von Windows und Windows Server ist. Das NTFS unterstützt Sicherheitsfeatures wie die Sicherheit basierend auf Zugriffsteuerungslisten, die den Zugriff auf Datei- und Ordnerebene für Nutzende und Gruppen regeln. Außerdem wird die Verschlüsselung von Laufwerken unterstützt. Bei der Formatierung eines Volums werden mehrere NTFS-Systemdateien, wie die „$MFT“, „$Bitmap“ oder „$LogFile“, angelegt. Systemdateien enthalten Informationen über die gesamten Dateien und Ordner im NTFS-Dateisystem. Die $MFT enthält für jede Datei auf dem Dateisystem einen Eintrag mit Informationen wie Zeitstempelangaben zum Erstellungs- oder letzten Änderungsdatum, Datei- oder Ordnernamen, den Besitzer der Datei, wer auf die Datei zugreifen kann, die Art (darunter gelöschte Datei, Datei, gelöschtes Verzeichnis, Verzeichnis), miteinander kombinierbare Eigenschaften (darunter schreibgeschützt, versteckt, System, kompromittiert, verschlüsselt) oder den Speicherort. Ist die Datei kleiner oder gleich 512 Bytes, wird die Datei in die $MFT integriert. Für mehr Details zum NTFS-Dateisystem sei auch auf die anonyme Projektarbeit „Forensik-Software – Test von Funktionalitäten auf Dateiidentifikations- & Dateirekonstruktionsebene in NTFS & APFS-Dateisystemen“ auf dem IT-Forensik Wiki verwiesen [68]/[158]/[234]/[235]/[236].

Autopsy hat alle versteckten Testdateien gefunden und es konnten alle Dateien erfolgreich extrahiert und geöffnet werden. Im Unterschied zu Excel können in einer Access-Datei eingebettete Dateien, die als Anhang in einer Relation gespeichert sind, nicht durch Autopsy beziehungsweise dem verwendeten Modul „Embedded File Extractor“ gefunden werden, was eine potenzielle Verschleierungsmöglichkeit darstellt. Im Folgenden werden einige Details der Auswertung näher betrachtet.

Die Dateiansicht „*File Views → File Types → By Extension → Documents → Database*“ ist leer, Autopsy führt MS Access nicht als DBS:

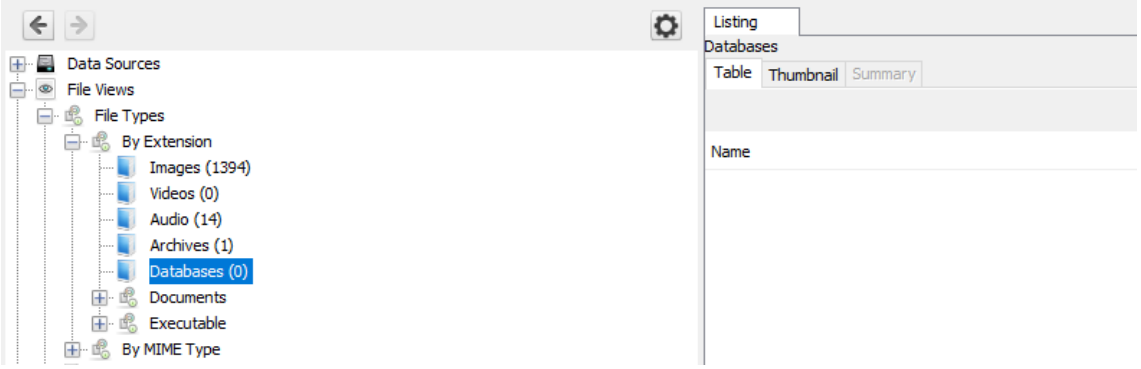


Bild 117: Autopsy – Auswertungsergebnis von DBSs („File Views → File Types → By Extension → Documents → Database“)

Unter der Dateiansicht „*File Views → File Types → By Extension → Documents → Office*“ sind lediglich die Standard Word- und Excel-Dateiformate .xlsx sowie .docx zu finden. Anders als den Anhang in Access-Dateien hat Autopsy die in die Datei „*Testdatei Excel 1 eingebettete Datei.xlsx*“ eingebettete Datei „*Microsoft_Excel_Worksheet.xlsx*“ gefunden. Allerdings kann die eingebettete Datei nach dem Extrahieren nicht in Excel geöffnet werden, was eine weitere Auswertung verhindert. Da der Fokus auf Access liegt, wird dieser Umstand nicht weiter analysiert. Word und Excel-Dateien mit VBA (.docm, .xlsm) sowie Access-Dateien (.accdb, .accde, .accdc) sind in dieser Sicht nicht enthalten, was die weitere Analyse erschwert. Da es keine Ansicht gibt, die zumindest alle Access-Datenquellen enthält, müssen sie aus mehreren Ansichten zusammengesucht werden:

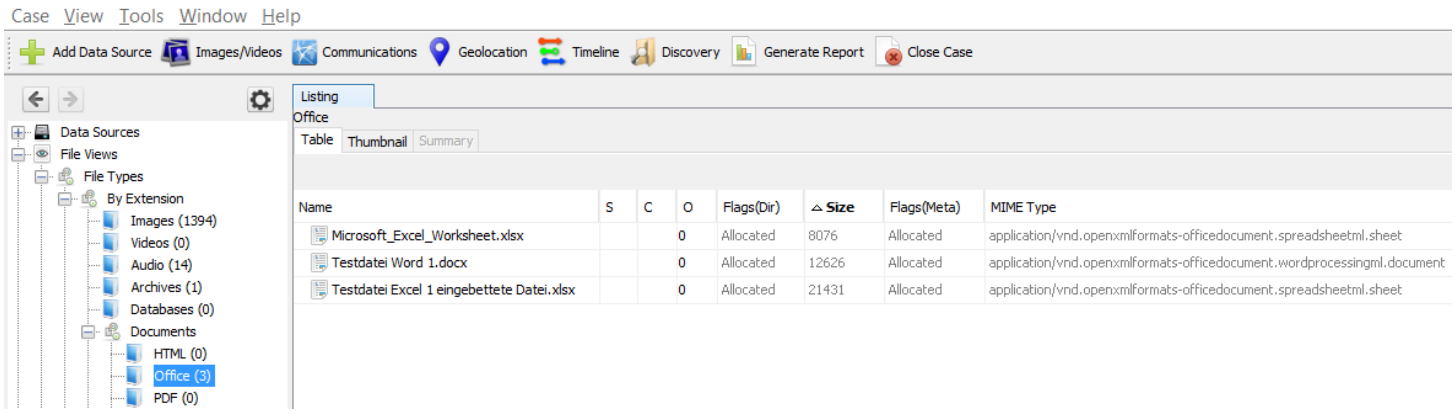


Bild 118: Autopsy – Auswertungsergebnis von Office-Dateien („File Views → File Types → By Extension → Documents → Office“)

Darüber hinaus ist unter „*File Views → File Types → By Extension → Archives*“ eine .cab-Datei (Cabinet File) aus dem Unallocated Space (nicht allokierter Speicherbereich, Speicherort für gelöschte Dateien) zu finden („*f0000000.cab*“):

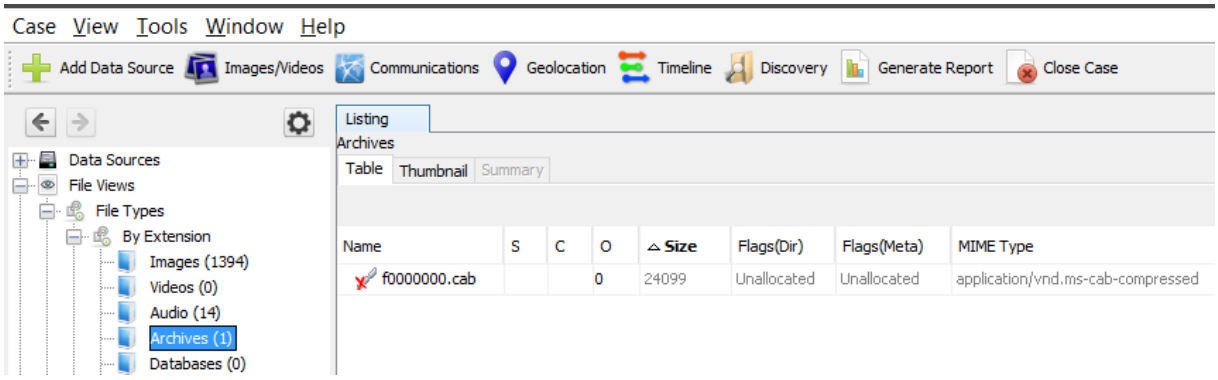


Bild 119: Autopsy – Auswertungsergebnis über File Carving gefundene .accdc-Datei („File Views → File Types → By Extension → Archives“)

Ein Cabinet ist eine von MS entwickelte Dateikomprimierung, die ähnlich einer .zip-Datei mehrere Dateien im komprimierten Zustand enthält. Die Vorteile von Cabinets sind, dass die Komprimierung über Dateigrenzen hinweg erfolgt und große Dateien über mehrere Cabinets verteilt werden können [89]. Bei der gefundenen .cab-Datei handelt es sich um die Datei „*Testdatei Access 5 Datei signiert & gelöscht.accdc*“, die mittels „File Carving“ rekonstruiert werden kann:

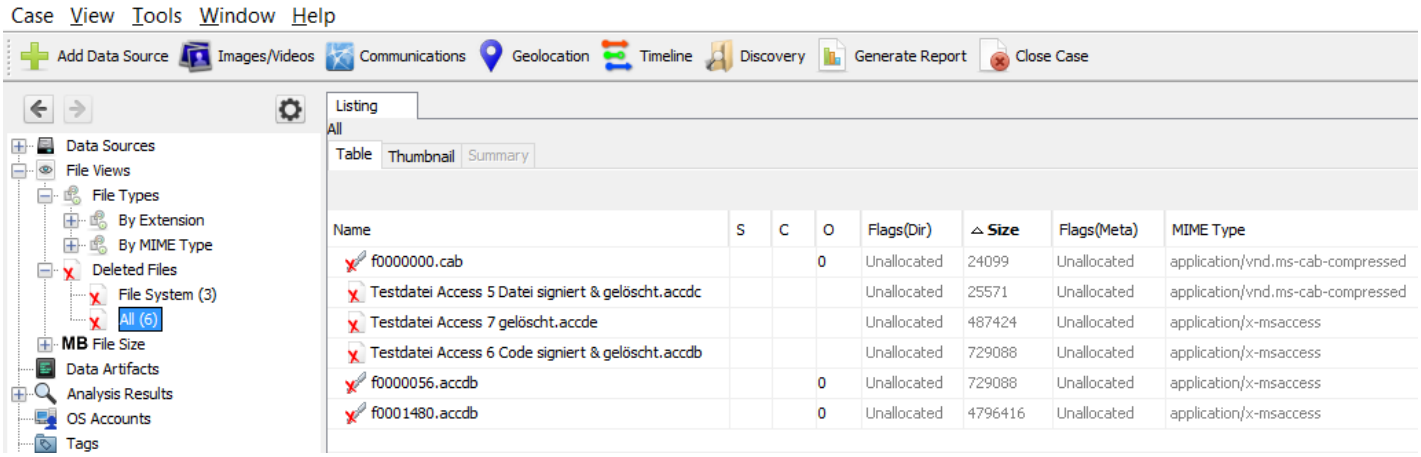


Bild 120: Autopsy – Auswertungsergebnis über File Carving (ohne Metadaten) und $MFT (mit Metadaten) gefundene, gelöschte Dateien („File Views → Deleted Files → File System → All“)

Wird der zu einer gelöschten Datei gehörende $MFT-Eintrag mit einem neuen Eintrag überschrieben, kann Autopsy gelöschte Dateien nur noch über den nicht allokierten Speicherbereich mittels File Carving wiederherstellen. Dabei darf der betroffene Speicherbereich, in dem sich die gelöschte Datei befindet, noch nicht wieder allokiert und mit neuen Daten überschrieben worden sein. Beim File Carving wird nicht adressierter Speicherbereich byteweise durchsucht und versucht Datei-Header, -Footer sowie Dateisignaturen zu ermitteln. Die Dateisignatur (magische Zahl) befindet sich im Header der Datei. Es handelt sich um eine Folge von Bytes, die für jeden Dateityp eindeutig ist (bei JPEG „*0xFFD8FF*“):

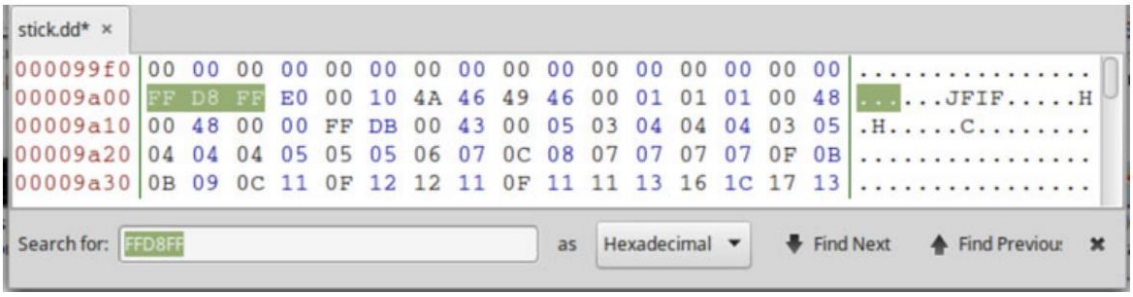


Bild 121: Dateisignatur (magische Zahl) einer JPEG-Datei [39, S. 138]

Je nach Dateityp kann auch das Dateiende standardisiert sein (bei JPEG „*0xFFD900*“). Wenn keine Endsequenz vorhanden ist, wie bei Word-Dokumenten, muss die Dateigröße anders bestimmt werden. Steht die Dateigröße im Header, kann sie dort ausgelesen werden. Sind Header und Footer ermittelt, werden die Daten-Bytes zwischen Header und Footer ausgelesen, um die Datei wiederherzustellen. Im Idealfall sind die Daten nicht fragmentiert und die zur Datei gehörenden Daten liegen somit auf dem Datenträger in aufeinanderfolgenden Datenblöcken.

Kann die Datei nur über den Unallocated Space und nicht über den zugehörigen $MFT-Eintrag wiederhergestellt werden, gehen beim File Carving die Metadaten der Datei, wie der Dateiname, verloren [39, S. 137].

Beim Signieren von Access-Dateien, wird die zu signierende Datei in eine .cab-Datei (Cabinet) eingebettet und dabei komprimiert (.accdc-Dateiformat). Auch die gelöschte .accdc-Datei in Form der mittels File Carving gefundenen .cab-Datei kann problemlos extrahiert und die hier eingebettete .accdb-Datei im Anschluss geöffnet werden. Da die Datei jedoch im .cab-Format vorliegt, gehen bei der Wiederherstellung Informationen über die beim Packen der .accdc-Datei verwendete Signatur verloren. Wie unter „*File Views → Deleted Files → File System → All*“ zu sehen ist, kann die Datei auch über die $MFT inklusive der zugehörigen Metadaten, wie dem Dateinamen wiederhergestellt werden, da der Eintrag noch nicht überschrieben wurde. Beim Löschen von Dateien im NTFS-Dateisystem wird die Attributart der Datei in der $MFT von „*1*“ (Identifier für Datei) auf „*0*“ (Identifier für gelöschte Datei) geändert. Wird der Eintrag danach nicht durch einen neuen Eintrag überschrieben, bleibt er erhalten und die Datei kann trotz Löschung mit allen Metadaten wiederhergestellt werden. Aus diesem Grund werden die gelöschten Dateien doppelt gefunden [266].

Unter „*File Views → File Types → By Extension*“ ist nur die gelöschte und über File Carving rekonstruierte .accdc-Datei („*f0000000.cab*“) zu finden. Die übrigen Dateien sind teilweise über den Inhaltstyp (Content-Type) oder auch Multipurpose Internet Mail Extensions (MIME-Type) unter „*File Views → File Types → By MIME Type → application*“ aufgelistet. Beim MIME handelt es sich um ein Format für den Versand beliebiger Text- und Binärdaten. Der an dieser Stelle relevante MIME-Type ist im MIME-Header verortet und gibt den Datentyp, also den Inhalt der Nachricht an. Der MIME-Type besteht aus einer Haupttyp-Angabe und einer genaueren Untertyp-Bezeichnung. Beide Angaben sind durch einen Slash („/“) voneinander getrennt. Beispiele für Haupttypen

sind „image“ (Bilddaten), „text“ (ASCII-Text), „video“ (Digitalvideo) oder „application“ (proprietäres Dateiformat eines konkreten Anwendungsprogramms). Beispiele für konkreti-sierende Untertypbezeichnungen des Haupttypts „text“ sind „plain“, „html“ oder „xml“. Beispiele für „image“ sind „gif“ oder „jpeg“ und Beispiele für „application“ sind „x-msaccess“ oder „vnd.ms-cab-compressed“. Eine vollständige Liste aller bei der *Internet Assigned Numbers Authority* (IANA) registrierten MIME-Types kann unter folgender Uniform Resource Locator (URL) eingesehen werden:

https://www.iana.org/assignments/media-types/media-types.xhtml. Aspekte von MIME werden in den Request for Comments (RFC) 2045-2049 dargelegt [245, S. 268-270].

Unter „x-msaccess“ ist die Datei „`Testdatei Access 3 Dateiverschlüsselung.accde`“ als beachtenswert markiert, da Autopsy eine Verschlüsselung mittels Passworts erkannt hat:

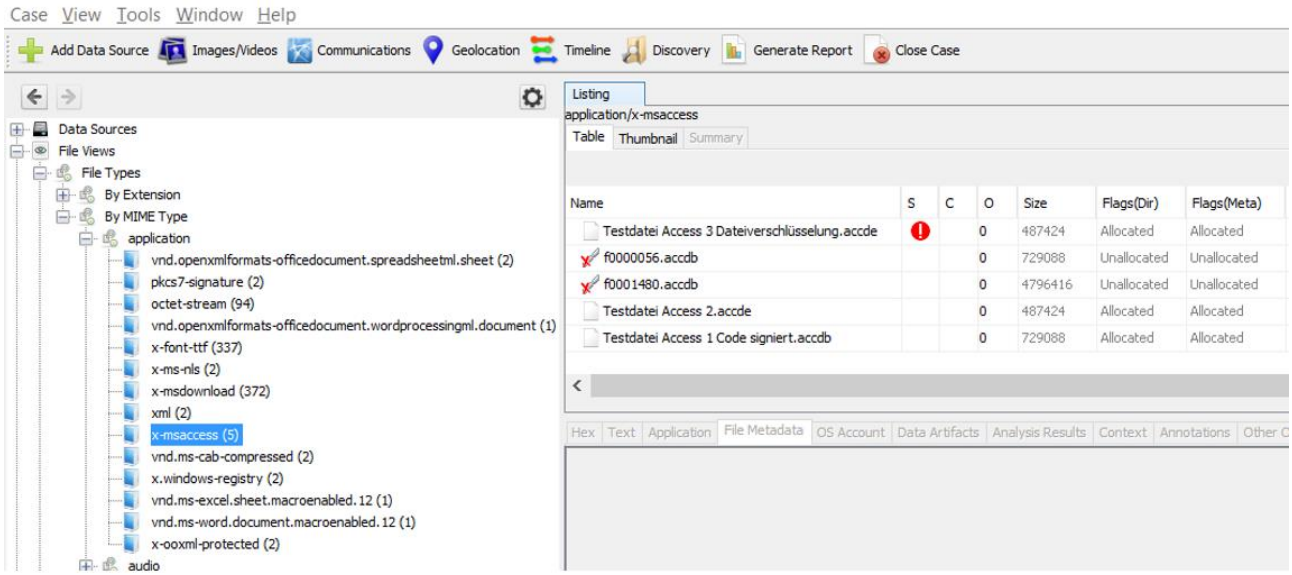


Bild 122: Autopsy – Auswertungsergebnis über $MFT und File Carving gefundene, gelöschte Dateien („File Views → File Types → By MIME Type → application → x-msaccess“)

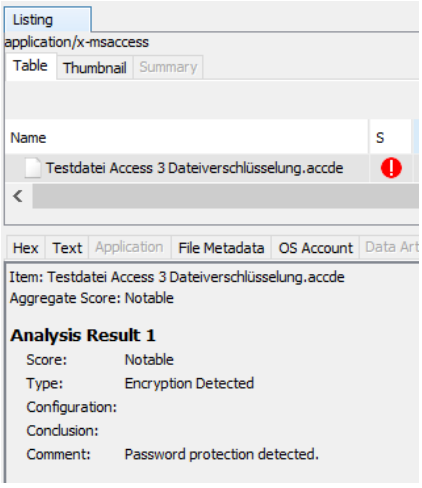


Bild 123: Autopsy – Auswertungsergebnis der von Autopsy als interessant markierten Datei „Testdatei Access 3 Dateiverschlüsselung.accde“

Unter den MIME-Type „vnd.ms-cab-compressed“ sind die signierten .accdc-Dateien zu finden:

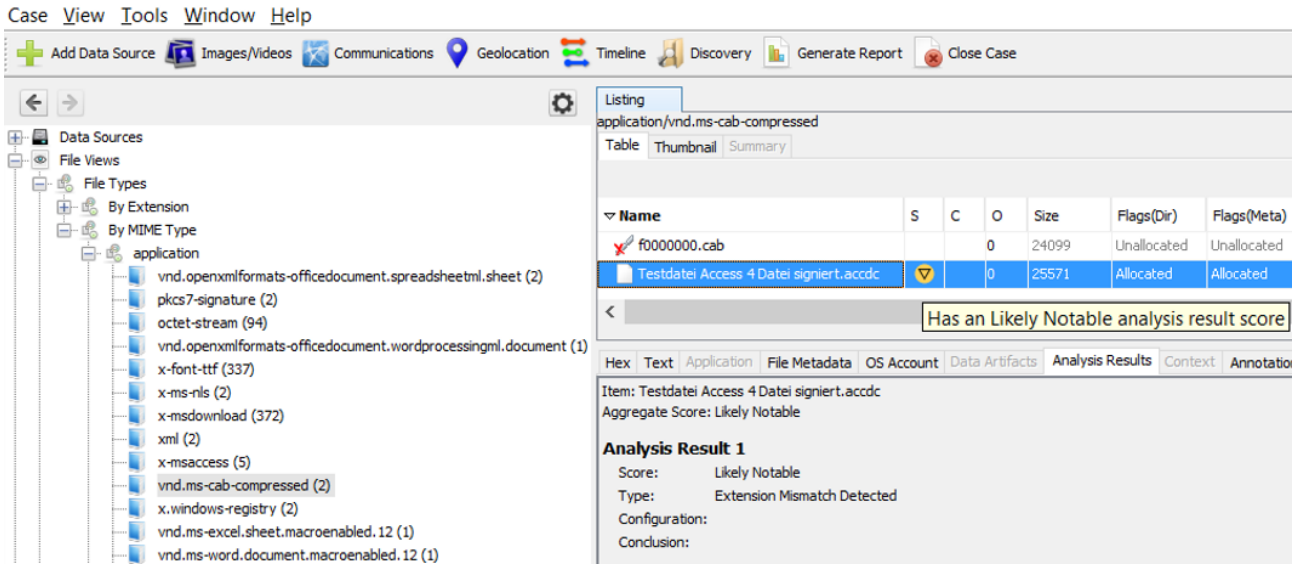


Bild 124: Autopsy – Auswertungsergebnis signierte Access-Dateien („File Views → File Types → By MIME Type → application → vnd.ms-cab-compressed“)

Im Gegensatz zu der über die $MFT wiederhergestellten Datei „*Testdatei Access 5 Datei signiert & gelöscht.accdc*“, ist die Datei „*Testdatei Access 4 Datei signiert.accdc*“ von Autopsy als beachtenswert markiert, da die gefundene Dateisignatur von der Dateierweiterung im Dateinamen abweicht. Dieser Umstand ist in diesem Fall jedoch uninteressant, da Access-Datenquellen beim Signieren in eine .cab-Datei (Cabine) eingebettet, komprimiert und im Ergebnis als .accdc-Datei angezeigt werden. Die signierte Testdatei ist auch unter „*Analysis Results → Extension Mismatch Detected*“ auffindbar.

Auch die verschlüsselten Testdateien wurden unter „*Analysis Results → Encryption Detected*“ gefunden:

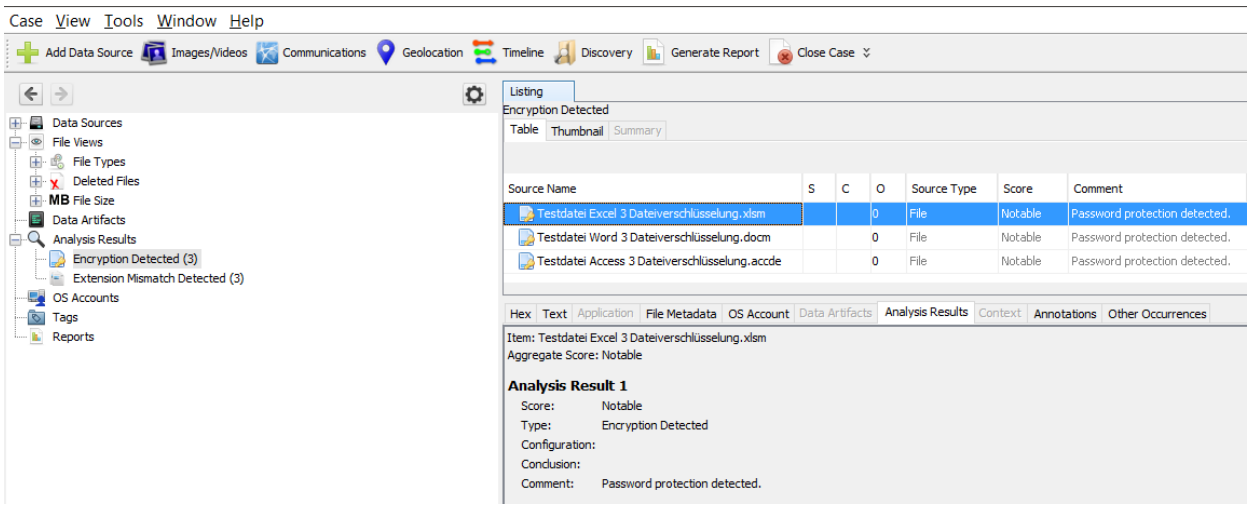


Bild 125: Autopsy – Auswertungsergebnis verschlüsselte Access-Dateien („Analysis Results → Encryption Detected“)

5. Auswertung der Sicherheitsanalyse

Die vorherigen Analysen (*siehe Kapitel „4 Durchführung der Sicherheitsanalyse“*) haben sowohl in MS Access und Access Excel (*siehe Kapitel „4.2 Schnittstellenanalyse (Access Excel)“*) als auch bei der Azure-Cloud beziehungsweise Azure AD (*siehe Kapitel „4.1.4 Passwörter & Authentifizierung“*) erhebliche Sicherheitslücken aufgezeigt, die für Unternehmen mit kritischen und sensiblen Prozessen sowie Daten einen potenziellen „Showstopper“ darstellen können:

Tabelle 7: Zusammenfassung der Sicherheitsanalyse je Teilaspekt

Zusammenfassung der Sicherheitsanalyse je Teilaspekt	***Access***	***Azure SQL-DB***
Kriterienanalyse (Access & Vergleichs-DBS) 1. Themenbereich: SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen		
Es muss einen Schutz gegen SQL-Injection-Angriffe geben. ***Zu Access: Gültigkeitsregeln, Eingabeprüfungen, (Daten)Makros und Co. greifen nur, wenn der Zugriff über die Access-Oberfläche erfolgt. Schutz ist durch Zugriff über Drittanbietertools wie DbVisualizer leicht aushebelbar. Sobald das Passwort zur Entschlüsselung bekannt ist, besteht Vollzugriff auf die Access-Datenquelle.***	😐	😊
Es muss die Möglichkeit geben Transaktionen zu nutzen (inklusive Rollback-Möglichkeit). ***Zu Access: Abgeschwächter Funktionsumfang.***	😐	😊
Es müssen Stored Procedures und Views definiert sowie Prepared Statements verwendet werden können. ***Zu Access: Abfragen sind die Alternative zu Views und Stored Procedures.***	😊	😊
Die Möglichkeit zur Ausführung von Mehrfachabfragen (Multi-Query-Statements) muss per Default deaktiviert sein. ***Zu Access: Mehrfachabfragen werden nicht unterstützt.***	😊	☹
Kriterienanalyse (Access & Vergleichs-DBS) 2. Themenbereich: Konfiguration		
Nicht benötigte Dienste, SQL-Befehle oder Funktionen müssen deaktiviert oder deinstalliert werden können. ***Zu Access: Menüband-Konfiguration und deaktivierte Tastenkürzel sind über DbVisualizer aushebelbar. AutoExec-Makros sind durch Drücken der SHIFT-Taste umgehbar (nicht deaktivierbar). .accde-Dateiformat verhindert die Einsicht und Manipulation des VBA-Projekts und deaktiviert den Entwurfsmodus beispielsweise bei Formularen. Jedoch Vollzugriff auf Abfragen, (Daten)Makros und Datenquellen.***	☹	😊
Die Verwendung von Triggern muss deaktiviert werden können. ***Zu Access: Datenmakros als Alternative zu Triggern. Die Ausführung erfolgt nur beim Zugriff über die Access-Oberfläche. Daher ist dieser Umstand gemäß Anforderung positiv zu bewerten, wenn der reguläre Zugriff auf die DB über Drittanbietertools wie DbVisualizer oder Eigenentwicklungen erfolgt. Da Trigger so aber auch leicht umgangen werden können, erfolgt eine neutrale Bewertung.***	😐	😊
Ein definierter einheitlicher Konfigurationsstandard muss DBS beziehungsweise dateiübergreifend konfiguriert und überwacht werden können. ***Zu Access: Positiv ist die zentrale Konfigurationsmöglichkeit über das Trust Center, die Windows Registry und Gruppenrichtlinien. Jedoch teilweise keine granularen Konfigurationen möglich. Makros können nur gesamthaft mit oder ohne Benachrichtigung deaktiviert werden. Bei VBA-Code und unsicheren Makroaktionen geschützte Ansicht. Negative Bewertung, wenn nur Access-Funktionalität in die Bewertung miteinfließt.***	😐	😊
Kriterienanalyse (Access & Vergleichs-DBS) 3. Themenbereich: Kryptographie		
Als symmetrische Blockverschlüsselung muss der Advanced Encryption Standard (AES)-256 verwendet werden können.	😊	😊
Das Verschlüsselungsverfahren muss austauschbar sein.	😊	😊
Digitale Signaturen müssen verwendet werden können. ***Zu Access: Digitale Signaturen sind zwar vorhanden, werden bei Änderungen an der Logik jedoch nicht verworfen und gehen beim Entpacken einer signierten .accdc-Datei verloren. Somit nur geringer Nutzen und eher eine Gefahr, wenn eine manipulierte Datei von einem vertrauenswürdigen Herausgeber signiert ist und die Datei nicht in der geschützten Ansicht geöffnet wird.***	☹	😊

Zusammenfassung der Sicherheitsanalyse je Teilaspekt	***Access***	***Azure SQL-DB***
Kriterienanalyse (Access & Vergleichs-DBS) 4. Themenbereich: Passwörter & Authentifizierung		
Auswählbare Passwörter als Authentisierungsmerkmal müssen beliebig lang sein und Groß-, Kleinbuchstaben, Ziffern, Sonderzeichen akzeptieren. ***Zu Access: Maximale Passwortlänge (Groß-, Kleinbuchstaben, Ziffern, Sonderzeichen) bei Dateiverschlüsselung sind 20 Zeichen, bei VBA-Projekt-Passwortschutz 32 Zeichen. Maximum zur Dateiverschlüsselung entspricht dem von MS geforderten Minimum von 18-20 Zeichen für eine Widerstandsfähigkeit gegen Brute-Force-Angriffe auf das Passwort [224]. Treuhandschlüssel als Masterkey zur Dateientschlüsselung.*** ***Zu Azure SQL-DB: Maximale Passwortlänge zur Anmeldung an einem AD-Konto sind 256 Zeichen, wobei Umlaute nicht verwendet werden können.***	😐	😊
Passwortwiederverwendungen bei Authentisierungsmerkmalen müssen erkannt werden.	☹	☹
Eingegebene Passwörter dürfen per Default nicht im Klartext angezeigt werden.	😊	😊
Es muss einen Schutz vor Brute-Force-Angriffen auf Passwörter, wie die Sperrung oder Dateilöschung bei Überschreiten einer maximalen Anzahl falscher Passworteingaben, geben. ***Zu Azure SQL-DB: Mehrschichtige Sicherheitsarchitektur mit vielen Funktionalitäten wie Firewall-Regeln, intelligente Sperren, erweiterte Bedrohungserkennung und Multi-Faktor-Authentifizierung.***	☹	😊
Anstelle des Mixed-Modus beziehungsweise der SQL-Authentifizierung muss der Windows-Modus, Azure AD oder eine vergleichbare Funktionalität verwendet werden können. ***Zu Access: Keine Sicherheit auf Benutzerebene, nur Dateiverschlüsselung und Dateisystemberechtigungen.*** ***Zu Azure SQL-DB: Links: Ohne Berücksichtigung der Sicherheitslücke in Azure AD.*** ***Rechts: Mit Berücksichtigung der Sicherheitslücke.***	☹	😊 / ☹
Kriterienanalyse (Access & Vergleichs-DBS) 5. Themenbereich: Logging & Auswertungsmöglichkeiten		
Alle Zugriffe auf das DBS, -dienste, DB-Procedures, DB-Inhalte, sonstige Befehlsausführungen oder Änderungen müssen automatisiert geloggt werden. ***Zu Access: Access bietet von Haus aus keine derartigen Funktionen. VBA-Routinen und Makros werden nur beim Zugriff über die Access-Oberfläche ausgeführt, daher keine zuverlässige Methode.***	☹	😊
Es müssen Transaktionslogs und Informationen über ausgeführte Abfragepläne geführt werden. *Optional: Im Idealfall können Transaktionslogs und DB-Dateien auf unterschiedlichen Festplatten gespeichert werden.* ***Zu Access: Nur Funktionen des Datei-, Betriebssystems oder von Drittanbietern zum Monitoring oder zur Protokollierung nutzbar, die es in einer separaten Analyse zu prüfen gilt. Die Datei in der Access Transaktionen protokolliert, ist unter dem Pfad in der `TEMP`-Systemumgebungsvariable nicht auffindbar.***	☹	😊
Log-Daten müssen an einen Log-Server weitergeleitet werden können.	☹	☹
Es muss ein vollautomatisierbares Monitoring aller Betriebszustände und Modifikationen möglich sein, inklusive der Möglichkeit über einen konfigurierbaren Alarm benachrichtigt zu werden.	☹	😊
Kriterienanalyse (Access & Vergleichs-DBS) 6. Themenbereich: Berechtigungen & Autorisierung		
Es muss beim DBS-Zugriff die Möglichkeit geben das Need-to-Know-Prinzip und Least-Privilege-Prinzip einzuhalten.	☹	😊

Zusammenfassung der Sicherheitsanalyse je Teilaspekt	***Access***	***Azure SQL-DB***
Die Zugriffsrechte des jeweiligen Nutzers müssen mit minimalen Privilegien konfigurierbar sein. ***Zu Access: Nur Lese- und Schreibzugriff über Dateisystemberechtigungen steuerbar. Keine Möglichkeiten für granulare Berechtigungsvergabe. Ist das Passwort zur Dateientschlüsselung bekannt, besteht Vollzugriff und alle in die Datei eingebetteten Schutzmechanismen können über DbVisualizer umgangen werden.***		
Das DBS-Backend muss so isoliert wie möglich sein, um unerlaubten Zugriff zu verhindern. ***Zu Access: Isolierung nur über Dateisystemrechte möglich. Für ordnungsgemäßen Betrieb werden Lese-, Schreib-, Erstellungs- und Löschberechtigungen auf das Verzeichnis benötigt. Mit zusätzlichen Leserechten auf die Datei ist kein Umbenennen oder Löschen möglich, jedoch Ablage neuer und Kopieren von Dateien.*** ***Zu Azure SQL-DB: Isolierung vom Betriebssystem, DBS ist an MS ausgelagert, standardisierte Schutzmechanismen (mehrschichtige Sicherheitsarchitektur): Zugriff nur nach Freischaltung durch Firewall-Regeln → Azure AD oder SQL-Authentifizierung → granulare Berechtigungsvergabe, Datenmaskierung, Sicherheit auf Spalten- und Zeilenebene (siehe auch Anforderung zur Einhaltung des Need-to-Know-Prinzips).***	🙁	🙂
Der Zugriff auf Systemtabellen muss eingeschränkt werden können. ***Zu Access: Systemtabellen nur für interne Verwaltungszwecke benötigt.***	🙁	🙂
Kriterienanalyse (Access & Vergleichs-DBS) 7. Themenbereich: Datenschutzkonformer Zugriff		
Es muss die Möglichkeiten für einen datenschutzkonformen Zugriff auf die Daten geben. ***Zu Access: Nur über Frontend steuerbar. Passwort zur Dateientschlüsselung darf Nutzenden nicht bekannt sein. MS 365 auf Datenschutzkonferenz im Jahr 2022 wiederholt für datenschutzwidrig erklärt.***	🙁	🙂
Für Log-Dateien muss eine automatische Löschroutine aktiviert werden können. ***Zu Access: Nicht relevant, da von Haus aus keine automatischen Protokollierungsfunktionalitäten.***	*Nicht relevant*	🙂
Kriterienanalyse (Access & Vergleichs-DBS) 8. Themenbereich: Datensicherung		
Es muss die Möglichkeit geben automatische Datensicherungen zu konfigurieren. ***Zu Access: Nur manuell, als SQL Server DB-Datei (.mdf) oder über Sicherungen des Dateisystems möglich.***	😐	🙂
Kriterienanalyse (Access & Vergleichs-DBS) Themenbereich: 9. Banking 4.0		
Es müssen auch große Datenmengen verarbeitet werden können.	🙁	🙂
Die Zentralisierung der Datenhaltung muss gefördert werden. Dabei muss ein Überblick über alle vorhandenen Daten möglich sein, um möglichst effizient eine KI mit Bankenwissen trainieren oder Data Mining betreiben zu können.	🙁	🙂
Die Nutzung des DBS in der Cloud muss möglich sein.	🙁	🙂
Es dürfen keine technischen Schulden entstehen. ***Zu Azure SQL-DB: Die Azure-Cloud einschließlich Azure SQL-DB und weitere angebotene Dienste, wie zum API-Management, Big Data oder zur Unterstützung der Microservice-Architektur, werden als Banking 4.0-Konform angesehen. Technische Schulden können auch durch Umstände, wie eine falsche Technologiewahl des Frontends, entstehen, die nicht durch die Azure-Cloud beeinflusst werden können.***	🙁	🙂
Schnittstellenanalyse (Access Excel) ***Mittels Hexadezimal-Editor: Ohne Dateiverschlüsselung (starkes Passwort empfohlen) sind sämtliche Zeichenketten einsehbar, Blattschutz- und VBA-Projekt-Passwortschutz sind leicht aushebelbar.***	🙁	*Nicht relevant*
Dateiformatanalyse mittels Hexadezimal-Editor (Access) ***Mittels Hexadezimal-Editor: Ohne Dateiverschlüsselung (starkes Passwort empfohlen) sind sämtliche Zeichenketten einsehbar (betrifft .accdb, .accde), der VBA-Projekt-Passwortschutz ist bei .accdb-Dateien leicht aushebelbar. Gelöschte Daten erst nach Komprimierung wirklich gelöscht, kann auch gültige Daten löschen.***	🙁	*Nicht relevant*
Forensische Analyse (Access) ***Alle Dateien wurden gefunden, es wird jedoch kein Anhang erkannt. Verbesserungsvorschläge identifiziert.***	🙂	*Nicht relevant*
Gesamtergebnis ***Zu Azure SQL-DB: Links: Unter Berücksichtigung des Funktionsumfangs und Konzepts.*** ***Rechts: Unter Berücksichtigung der Sicherheitsbedenken.***	😰	😎 / 😰

Ohne eine detaillierte Sicherheitsanalyse der Azure-Cloud sind weitere Einfallstore zu befürchten. Da der Fokus dieser Arbeit auf Access liegt, beschränkt sich die Auswertung der Sicherheitsanalyse auf den Funktionsumfang der Azure SQL-DB (*siehe Kapitel „1.2 Abgrenzung“*). Der in der Aufgabenstellung geforderte Vergleich zwischen Access und dem Vergleichs-DBS ist bereits in *Kapitel „4.1 Kriterienanalyse (Access & Vergleichs-DBS)“* erfolgt, wird aber an dieser Stelle erneut aufgegriffen und ergänzt. Für weitere Details zur Azure SQL-DB sei auf die oben genannten Unterkapitel von *Kapitel „4.1 Kriterienanalyse (Access & Vergleichs-DBS)“* verwiesen.

Im Vergleich zu Access ist der Funktionsumfang sowie der Ansatz der Azure SQL-DB durchweg überzeugend und zukunftsweisend zu bewerten (ohne Berücksichtigung der Sicherheitsbedenken und detaillierter Sicherheitsanalyse der einzelnen Dienste). In die Bewertung fließen Punkte mit ein wie: Die Banking 4.0-Konformität und die damit verbundene Reduzierung von technischen Schulden sowie die Standardisierung, der Platform-as-a-Service-Gedanke, die einfache und schnelle Bereitstellung neuer Azure SQL-DBs, die Datenzentralisierung und die zentrale Verwaltungsmöglichkeit durch Administratoren, die steile Lernkurve im Umgang mit der Azure-Cloud, die umfangreichen Dokumentationen und die unzähligen in der Azure-Cloud verfügbaren (sicherheitsrelevanten) Services (darunter Azure AD, die Data Discovery & Classification-Funktionalität oder Dienste zum API-Management).

Azure SQL basiert anders als Access auf einem mehrschichtigen „Defense-in-Depth-Schichtenmodell“, dass von außen nach innen durchlaufen wird. In der äußersten Schicht verhindern Firewalls den Netzwerkzugriff auf die Server in der Azure-Cloud, bis der Zugriff explizit über die IP-Adresse oder das virtuelle Azure-Netzwerk gewährt wird (*siehe Kapitel „4.1.4 Passwörter & Authentifizierung“*). Die Authentifizierung kann über SQL-, Azure AD-Authentifizierung aber auch über ein On-Premises-AD erfolgen. Optional kann zusätzlich eine Multi-Faktor-Authentifizierung aktiviert werden. Aufgrund der Authentifizierungs-Funktionalität können sich Nutzende mit ihrem (Windows)-Account und den hierüber zugewiesenen Rechten über verschiedene Tools an einer Azure SQL-DB anmelden (*siehe Kapitel „4.1.4 Passwörter & Authentifizierung“*). Die granular steuerbare Autorisierung erfolgt idealerweise über Gruppen- und Rollenzuweisungen, aber auch die Vergabe von individuellen Einzelrechten ist möglich. Neben der Sicherheit auf Spaltenebene wird auch die Sicherheit auf Zeilenebene unterstützt (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung“ sowie Kapitel „4.1.7 Datenschutzkonformer Zugriff“*). Mit Hilfe der Threat Protection, wie der SQL-Überwachung, werden DB-Aktivitäten automatisch protokolliert. Die Ereignisse werden in einem Überwachungsprotokoll, wie Log Analytics, zur weiteren Analyse und Identifikation von potenziellen Bedrohungen sowie Sicherheitsverletzungen gespeichert. Mittels Advanced Threat Protection werden die erzeugten Logs automatisiert und mittels KI auf ungewöhnliches Verhalten analysiert. Für gefundene Anomalien, wie potenzielle Brute-Force- oder SQL-Injection-Angriffe, werden Warnmeldungen sowie Empfehlungen von MS zur Schließung der Sicherheitslücke gegeben. Aber auch Ledger als manipulationssicherer Nachweis, um Dritten gegenüber die Datenintegrität zu belegen, ist als erweiterte Funktionalität zur Bedrohungserkennung zu nennen (*siehe Kapitel „4.1.5 Logging & Auswertungsmöglichkeiten“*). Zur Wahrung der Vertraulichkeit, Integrität und Authentizität im Rahmen des Datenaustauschs erfordert die Azure SQL-DB die

verpflichtende Kommunikation über TLS. Mittels TDE werden ruhende Daten durch Verschlüsseln der gesamten DB mittels AES-256-Algorithmus (Standardkonfiguration) vor unautorisiertem Zugriff und Handwarediebstahl geschützt. Durch die Always Encryped-Funktionalität werden die in der Azure SQL-DB zu speichernden Daten im Vorfeld der Datenübertragung auf dem Client verschlüsselt. Nur autorisierte Nutzende können die Daten mit einem Schlüssel entschlüsseln. Dabei wird der Schlüssel der Azure SQL-DB nicht bekannt gemacht und extern verwaltet (*siehe Kapitel „4.1.3 Kryptographie“*). Mit der Sicherheitsrisikobewertung kann die allgemeine DB-Sicherheit verbessert werden (siehe *Kapitel „4.1.2 Konfiguration“*). „Azure Purview“ bietet Funktionen zur Datenerkennung und -klassifizierung. Dies erleichtert die Einhaltung des Datenschutzes sowie die Verwaltung und den Zugriff auf vertrauliche und personenbezogene Daten (*siehe Kapitel „4.1.9 Banking 4.0“*) [81]:

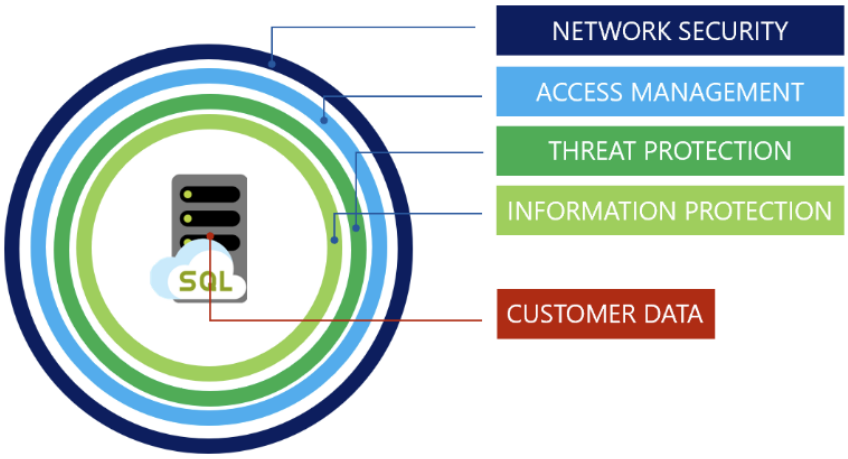


Bild 126: Azure SQL Defense-in-Depth-Schichtenmodell (wird von außen nach innen durchlaufen) [81]

Abschließend sei erwähnt, dass nach *Heise Online*-Recherchen im Jahr 2024 eine eigene Azure-Cloud für deutsche Behörden online gehen soll [56].

Anders als Azure SQL basiert Access nicht auf dem mehrschichtigen Defense-in-Depth-Schichtenmodell, wobei einige Funktionalitäten teilweise über Drittsysteme und Eigenimplementierungen mit reduziertem Funktionsumfang nachgebildet werden können. Als Beispiel für eine Nutzerauthentifizierung können hier Zugriffsbeschränkungen auf Basis von Lese- und Schreibrechten des Dateisystems genannt werden, wobei mit diesem Ansatz die Vertraulichkeit von Teilen der Daten keineswegs granular nach dem Need-to-Know-Prinzip und dem Least-Privilege-Prinzip sichergestellt werden kann. Andere Punkte, wie die Administration, Konfiguration und Wartung einer Inhouse-Firewall, erfordern deutlich mehr internalisiertes Fachwissen als bei der Nutzung von Azure-Cloud-Services.

Aufgrund des geringen Funktionsumfangs in den zuvor genannten Bereichen punktet Access vor allem durch die einfache Handhabung. Weder die Integrität und Authentizität einer Access-Anwendung (einschließlich Logik und Daten) noch die Vertraulichkeit sowie die Verfügbarkeit lassen sich angemessen und nachvollziehbar sicherstellen. Die einzigen und halbherzigen Schutzmechanismen von Access sind die Dateisystemberechtigungen (*siehe Kapitel „4.1.6 Berechtigungen & Autorisierung“*), das Kompilieren

einer Access-Datei in das Dateiformat .accde (*siehe Kapitel „4.1.2 Konfiguration“*), die Dateiverschlüsselung mittels starkem Passwort und austauschbare Algorithmen (*siehe Kapitel „4.1.3 Kryptographie“*) sowie mögliche Überwachungsfunktionalitäten des Betriebs- oder Dateisystems (*siehe Kapitel „4.1.5 Logging & Auswertungsmöglichkeiten-“*).

Schwache Passwörter für die Dateiverschlüsselung bieten keinen Schutz, da sie leicht mit diversen Passwort-Cracking-Tools aus dem Internet und Brute-Force- oder Wörterbuchangriffen entschlüsselt werden können und Access auch sonst keine Schutzmaßnahmen gegen derartige Angriffe bietet.

Nach Eingabe des Passworts ist die Access-Datei aufgrund fehlender Sicherheitsmaßnahmen auf Benutzerebene mit den jeweiligen Dateisystemrechten verfügbar. Da nach Passworteingabe Vollzugriff auf die Datei besteht, ist nicht überprüfbar, ob die Berechtigungen bestimmungsgemäß verwendet werden. Zudem verhindert dieser Umstand den datenschutzkonformen Zugriff.

Negativ in die abschließende Bewertung fließt auch der Umstand ein, dass MS 365 während der *104. Datenschutzkonferenz* 2022 wiederholt für datenschutzwidrig erklärt wurde. Da MS nach eigenen Angaben im Rahmen der Telemetrie anonymisierte Daten nutzt, ist zu ergänzen, dass personenbezogene Daten für festgelegte, eindeutige und legitime Zwecke erhoben werden und nicht zweckentfremdet weiterverarbeitet werden dürfen (vergleiche Art. 5 Abs. 1 lit. b DS-GVO). Da auch eine Datenanonymisierung eine Verarbeitung darstellt, muss diese auf eine Rechtsgrundlage gestützt werden (vergleiche Art. 4 Abs. 2 DS-GVO). Es gibt Expertenmeinungen, welche einen Test unter restriktiven Voraussetzungen mit nicht anonymisierten Originaldaten nicht als Zweckänderung, sondern als mit dem Zweck vereinbar ansehen. Um die für den Zweck erhobene Verarbeitung überhaupt durchführen zu können, werden angemessen entwickelte und getestete IT-Systeme benötigt. Abstrahiert auf die von MS durchgeführte Telemetrie ist zu klären, ob auch bei der Datenanonymisierung eine Zweckvereinbarkeit gilt (vergleiche auch Art. 5 Abs. 1 lit. b DS-GVO in Verbindung mit Art. 6 Abs. 4 DS-GVO) [248, S. 165, S. 167].

Das MS im Microsoft Products and Services Data Protection Addendum die in der Vergangenheit genannten Ziele, wie die Bekämpfung von Betrug oder IT-Kriminalität, nicht mehr enthalten sind, steht im deutlichen Widerspruch zu der von Bill Gates im Jahre 2002 ins Leben gerufenen Initiative Trustworthy Computing. Dies kann ein Grund dafür sein, dass MS teilweise seit 2011 bekannte Schwachstellen nicht nachhaltig behoben hat, um die Sicherheit der Office-Produkte zu erhöhen (*siehe Kapitel „4.1.7 Datenschutzkonformer Zugriff“*).

Das Kompilieren in das .accde-Dateiformat ist nur bedingt sicherheitsrelevant. In .accde Dateien kann der VBA-Code zwar nicht angezeigt oder geändert werden, aber es können Datenmakroereignisse und AutoExec-Makros hinzugefügt werden, die beim Öffnen der Datei automatisch ausgeführt werden (*siehe Kapitel „4.1.2 Konfiguration“*). Außerdem sind hinterlegte Zeichenketten, wie Passwörter oder private Schlüssel zum Entschlüsseln von Daten, auch in .accde-Dateien (einschließlich VBA-Projekt) über den Hexadezimal-Editor einsehbar (*siehe Kapitel „4.3 Dateiformatanalyse mittels*

Hexadezimal-Editor (Access)“). Das AutoExec-Makros auch nach dem Deaktivieren der Funktion „Access-Spezialtasten verwenden“ nicht unterbrochen werden können, kann Fluch und Segen zugleich sein. Zum einen kann so die Ausführung von Schadcode abgebrochen werden, zum anderen können gewollte VBA-Eigenentwicklungen wie Logging-Features umgangen werden. Im Gegensatz dazu kann durch Deaktivieren der Funktion „Access-Spezialtasten verwenden“ das Abbrechen der eigentlichen Programmausführung
verhindert werden. Angreifende können so die Ausführung ihres Schadcodes schützen, wenn dieser nicht in einem AutoExec-Makro eingebettet ist. Möchten Angreifende schadhaften VBA-Code in eine .accde-Datei integrieren, muss die Struktur (inklusive Migration des Datenhaushalts) und Funktionalität der .accde-Datei im .accdb-Format nachgebaut werden. Im .accdb-Format kann dann beliebiger VBA-Schadcode eingebettet werden, bevor die manipulierte Datei ins .accde-Format konvertiert und mit dem Original ausgetauscht wird. Dieser Umstand erhöht den Aufwand für einen Angriff, wenn der Funktionsumfang der Originaldatei erhalten bleiben soll.

Da zur ordnungsgemäßen Ausführung von Access-Dateien Lese-, Schreib-, Erstellungs- und Löschrechte auf den Ordner und zum Schreiben Schreibrechte auf die Datei benötigt werden, sind ein Kopieren der Datei (Leserechte sind hierfür ausreichend, (*siehe Kapitel „2.5 Unterschied dateibasierte & Client-Server DBS“*)), ein unbemerkter Austausch einer Access-Datenquelle oder das Ablegen einer manipulierten Kopie keine unwahrscheinlichen Angriffsszenarien. Zumindest für den regulären Betrieb eines Access-Frontends sind Leserechte auf die Datei ausreichend, was einen Schutz zur Wahrung der Integrität der Logik darstellt. Da jedoch über das Frontend in den Access-Dateiformaten 2007-2016 hinterlegte Zeichenketten, wie das Passwort zum Entschlüsseln des Access-Backends, in einem Hexadezimal-Editor im Klartext eingesehen werden können (ohne Dateiverschlüsselung) und oftmals auf das Backend Schreibrechte vergeben werden müssen, birgt ein Access-Backend ein höheres Angriffs-, Manipulationsrisiko und Risiko die Verfügbarkeit zu beeinträchtigen als ein Access-Frontend. Je mehr Nutzende Schreibrechte auf den Ablageort besitzen, desto höher wird das Risiko. Je mehr Nutzende auf eine zur zentralen Nutzung abgelegte, manipulierte Datei zugreifen, umso leichter fällt es einem Angreifenden Schadcode zu verteilen.

Für einen Angriff werden nicht zwangsläufig Schreibrechte auf die Datei benötigt. Schreibrechte auf den Ablageort oder die Versendung manipulierter Dateien über eine interne E-Mail-Adresse und ein unaufmerksamer Nutzer, der die geschützte Ansicht durch Bestätigen der Warnmeldungen verlässt und somit die Ausführung von Makros und VBA-Code aktiviert, reichen aus. Einziges Warnsignal für Nutzende ist, wenn eine Unternehmensrichtlinie vorschreibt, dass Office-Dateien nur Aktionen enthalten dürfen, die auch in nicht vertrauenswürdigen Anwendungen unter nicht vertrauenswürdigen Pfaden ausgeführt werden können. Ansonsten ist nicht transparent, ob das Aktivieren der Inhalte die erwarteten Funktionalitäten oder Schadcode freischaltet. Noch schlimmer ist es, wenn vertrauenswürdige Pfade oder Herausgeber konfiguriert sind, denn dann entfällt der menschliche Schutzmechanismus und die Datei wird nicht in der geschützten Ansicht geöffnet.

Dabei sind einem Angreifenden bei der Schadcodeerstellung nahezu keine Grenzen gesetzt, denn über VBA sind Zugriffe auf das Betriebssystem, wie das Ausführen von Kommandozeilenbefehlen, oder auf das Dateisystem problemlos über den angemeldeten Windows-Nutzer und den jeweiligen Rechten möglich.

Eine digitale Signatur (*siehe Kapitel „4.1.3 Kryptographie“*), ein Passwortschutz des VBA-Projekts bei .accdb-Dateien (*siehe Kapitel „4.1.4 Passwörter & Authentifizierung“ sowie Kapitel „4.3 Dateiformatanalyse mittels Hexadezimal-Editor (Access)“*), ein benutzerdefiniertes oder ausgeblendetes Menüband (*siehe Kapitel „4.1.2 Konfiguration“*), die angebotenen Eingabeprüfungen in Form von Gültigkeitsregeln und Eingabeformaten sowie die von Access angebotenen Protokollierungsfunktionalitäten sind nur für die Benutzerführung relevant, schützen lediglich vor unbeabsichtigten Manipulationen und lassen ohne fundierte IT-Kenntnisse falsche Sicherheit annehmen (*siehe Kapitel „4.1.1 SQL-Dialekt Funktionsumfang & Schutz vor SQL-Injection-Angriffen“*). Ähnlich verhält es sich mit der temporären Sperrdatei, die nur wenig interessante Informationen beinhaltet und aufgrund der erforderlichen Rechte zur ordnungsgemäßen Nutzung leicht manipuliert werden kann (*siehe Kapitel „4.1.5 Logging & Auswertungsmöglichkeiten“*). Da digitale Signaturen bei Änderungen an der Datei nicht entfernt werden, bergen sie sogar das Risiko, dass manipulierte Dateien nicht in einer geschützten Ansicht geöffnet werden, wenn ein entsprechender vertrauenswürdiger Herausgeber hinterlegt ist. Gültigkeitsregeln zur Eingabeprüfung können leicht über Drittanbietertools, wie DbVisualizer, umgangen werden. Entsprechend verhält es sich mit den unzureichenden und unzuverlässigen Protokollierungsfunktionalitäten von Access, VBA-Eigenimplementierungen, Makros oder Ereignissen, die durch Nutzung von DbVisualizer leicht umgangen werden können. Dieser Umstand wirkt sich negativ auf die Datenintegrität, -authentizität sowie -vertraulichkeit aus und erschwert deutlich das Implementieren von Verfahren zur Sicherstellung der Qualität von Modelldaten.

Auch die Tatsache, dass in den Access-Dateieigenschaften kein „Zuletzt gespeichert von“-Attribut geführt wird, dass sich das letzte Änderungsdatum bereits beim bloßen Öffnen der Datei ändert und, dass Access keine Datensicherungsfunktionalitäten anbietet, fließen negativ in die Gesamtbewertung ein (*siehe Kapitel „4.1.8 Datensicherung“*).

Zwar veröffentlicht MS keine detaillierten Informationen zum Dateiformat 2007-2016, allerdings sind die Ähnlichkeiten zu JET auffällig. Entgegen dem Kerckhoffs’schen Prinzip sollte die Sicherheit eines Verfahrens nicht von seiner Geheimhaltung abhängen.

Auch die Anforderungen, die sich aus Banking 4.0 ergeben, werden von Access nicht erfüllt (*siehe Kapitel „4.1.9 Banking 4.0“*). Banking 4.0 und die damit einhergehenden Entwicklungen, wie Cloud-Computing, KI oder die Grundsäulen der Industrialisierung (darunter Standardisierung, Robustheit/Ausfallsicherheit oder Selbstkorrektur), tragen zur Erhöhung der IT-Sicherheit insgesamt bei. Der Einsatz von Access erhöht technische Schulden und wirkt sich negativ auf die Grundsäulen der Industrialisierung aus. Access ist nicht skalierbar und für einen Überblick über die in jeder Access-Datei gespeicherten Daten, muss jede Datei separat geöffnet und analysiert werden, da Access von Haus aus keine derartigen Analysefunktionalitäten bietet. Bei einer großen Dateianzahl wird

dieser Umstand schnell zu einer komplexen Herausforderung und kann nur durch eine eigens für die Analyse entwickelte Softwarelösung bewältigt werden.

Aber auch die Access Excel-Schnittstellenanalyse hat gravierende Schwachstellen aufgezeigt, die einem Angreifenden den vollständigen Zugriff auf das gesamte Excel-Frontend inklusive aller abgelegten Daten ermöglichen wie das im VBA-Quellcode oder in ausgeblendeten Zellen hinterlegte Passwort zum Entschlüsseln des Access-Backends (*siehe Kapitel „4.2 Schnittstellenanalyse (Access Excel)“*). Excel ist definitiv ein bevorzugtes erstes Angriffsziel, um Informationen für eine anschließende Kompromittierung des dahinterliegenden MS Access-Backends zu erhalten.

Neben den bereits erwähnten Möglichkeiten den VBA-Projekt-Passwortschutz bei .accdb-Dateien durch Manipulation der Dateiinterna zu entfernen und über einen Hexadezimal-Editor die in den Access-Dateien hinterlegten Zeichenketten im Klartext einzusehen, hat die Dateiformatanalyse ergeben, dass auch gelöschte Daten noch über den Hexadezimal-Editor einsehbar sind, bis die Funktion „Datenbank komprimieren und reparieren“ oder die automatische Alternative bei jedem Schließen der Datei genutzt wird (*siehe Kapitel „4.3 Dateiformatanalyse mittels Hexadezimal-Editor (Access)“*). Werden personenbezogene Daten nach Entfall des Zwecks (vergleiche Art. 5 Abs. 1 lit. e DS-GVO, Speicherbegrenzung) nicht gelöscht, liegt ein Verstoß gegen die gemäß DS-GVO geforderte Löschpflicht (vergleiche Art. 17 Abs. 1 DS-GVO, Recht auf Löschung) vor. Zweifellos ist die Nutzung der Funktion „Datenbank komprimieren und reparieren“ neben personenbezogenen Daten auch für gelöschte vertrauliche Daten relevant. Einzige Vorteile an dieser Besonderheit des Access-Dateiformats ist, dass Angreifenden die Unterscheidung zwischen gültigen und ungültigen Datensätzen erschwert oder die Datenwiederherstellung erleichtert wird.

Im Unterschied zu den anderen Analysen hat die forensische Analyse einer Access-Datei keine gravierenden Einschränkungen festgestellt. Alle Access-Dateien wurden gefunden und konnten erfolgreich exportiert werden (*siehe Kapitel „4.4 Forensische Analyse (Access)“*). Somit bietet Autopsy insgesamt eine gute Unterstützung für MS Access und es kann eine ausreichende Berücksichtigung von MS Access bei der forensischen Auswertung und Suche nach bisher unbekannten Access-Dateien mit Autopsy bestätigt werden. Die forensische Analyse hat drei Verbesserungsvorläge ergeben, um die Auswertung effizienter zu gestalten:

1. Erweiterung des „Embedded File Extractor“-Moduls, damit auch in einer Access-Relation gespeicherter Anhang erkannt wird.
2. Erweiterung des „PhotoRec Carver“-Moduls, damit gelöschte .accdc-Dateien mittels File Carving nicht nur im .cab-Format extrahiert werden können. Wird der zu einer gelöschten .accdc-Datei gehörende $MFT-Eintrag überschrieben, gehen Informationen zur verwendeten Signatur verloren.
3. Erweiterung der Office-Dateiansicht um alle möglichen Office-Dateiformate, wie von Access oder Excel, um die Analyse zu vereinfachen und alle Treffer in einer zentralen Ansicht zu kapseln. Alternativ ist auch eine Erweiterung der DBS-Dateiansicht um Access-Dateiformate möglich. Hauptziel soll sein, dass Office-Dateien nicht mehr über mehrere Ansichten verteilt werden.

Summa summarum lässt sich festhalten, dass Access nur in Anwendungsfällen eine Chance gegen „professionelle“ Client-Server-DBS mit vergleichbarer Funktionalität, wie die Azure SQL-DB, hat, in denen die einfache Handhabung im Vordergrund steht und die Vertraulichkeit, Verfügbarkeit, Integrität, Authentizität sowie die Sicherheit von Logik und Daten, das Need-to-Know- und Least-Privilege-Prinzip oder eine Rollentrennung von untergeordneter Bedeutung sind. Die Spurensicherung nach einem erfolgreichen Angriff, das Erkennen von Daten- und Logikmanipulationen ist allein mit Access-Bordmitteln mangels zuverlässiger Datenquellen eine unlösbare Aufgabe.

Fehlende Schnittstellen zur Vision Banking 4.0, fehlende Sicherheit auf Benutzerebene, fehlende zuverlässig funktionierende und vollautomatisierte Überwachungsfunktionalitäten, die Möglichkeit hinterlegte Zeichenketten im Klartext über den Hexadezimal-Editor einzusehen, die benötigten Lese-, Schreib-, Erstellungs- und Löschberechtigungen auf den Ablageort, das einfache Aushebeln des Projektschutzpasworts, die Möglichkeit bereits mit Leserechten die Datei kopieren zu können, fehlende Datensicherungsfunktionalitäten und fehlende Möglichkeiten, die zu speichernden Daten zu verschlüsseln oder die Integrität nachvollziehbar gewährleisten zu können, stellen die größten Defizite im Funktionsumfang von Access dar. Da der technische Fortschritt auch neue Angriffsmöglichkeiten für Hacker mit sich bringt, wird der Migration auf neue Konzepte, Technologien, Standardisierungen und Ansätze wie Banking 4.0 oder die Zero-Trust-Strategie eine hohe Bedeutung für die IT-Sicherheit beigemessen.

Die Risiken, die sich aus den Schwachstellen von Access ergeben, können abhängig vom Zweck der Anwendung und der Kritikalität des Prozesses schwerwiegende Folgen für ein Unternehmen haben. Die Eintrittswahrscheinlichkeit wird als hoch eingeschätzt. Die Risiken reichen von falschen Geschäftsentscheidungen aufgrund manipulierter Daten oder Logik über Schadprogramme, die unternehmensweit an dutzende Nutzende verteilt werden, bis hin zu Datenverlust oder Datendiebstahl, was sich negativ auf einen Wettbewerbsvorteil auswirken oder zu einem Reputationsschaden des Unternehmens führen kann. Nachfolgend eine Übersicht mit potenziellen Bedrohungsszenarien, für die in *Kapitel „7 Fazit“* mögliche TOM vorgeschlagen werden *(Folgen sie diesem Link zu den Handlungsempfehlungen im Fazit)*:

1. Unerkannte sowie schadhafte Manipulationen an einer Datei (Logik und Daten). Kaum Möglichkeiten die Einhaltung der Integrität, Verfügbarkeit, Authentizität und Vertraulichkeit der Daten sowie Logik überprüfbar sicherzustellen und nachweisen zu können.
2. Eingeschränkte Möglichkeiten das Need-to-Know-Prinzip und Least-Privilege-Prinzip einzuhalten, Nutzende sehen mehr als sie dürfen.
3. Offenlegung von vertraulichen Daten inklusive Passwörtern und sonstigen Zugangsdaten.
4. Eingeschränkte Möglichkeiten Aktivitäten von Nutzenden möglichst automatisiert sowie überprüfbar (Auditing) zu protokollieren und zu überwachen (Monitoring).
5. Eingeschränkte Unterstützung für Mitarbeitende mit geringem IT-Know-how und damit verbunden eine fälschlich suggerierte Sicherheit oder die Entstehung von vermeidbaren Schwachstellen.

6. Verstöße gegen regulatorische (MaRisk, BAIT) oder gesetzliche (DS-GVO) Anforderungen. Bedingt geeignet für:
    - personenbezogene Daten: Eingeschränkte Konformität der DS-GVO wie erschwerte Einhaltung von Löschpflichten.
    - Risikodaten: Daten können unter Umständen nicht zweifelsfrei identifiziert, zusammengeführt und ausgewertet werden. Einheitliche Namenskonventionen sind schwer umzusetzen und zu kontrollieren. Entstehen von Inkonsistenzen und Redundanzen.
    - Modelldaten: Aufgrund der gefundenen Schwachstellen keine Möglichkeit die Datenqualität nachvollziehbar zu gewährleisten.
7. Datenverlust aufgrund eingeschränkter Datensicherungsmöglichkeiten.
8. Aufbau neuer technischer Schulden durch ungeeignete Technologieauswahl und eine abnehmende Rendite, da Automatisierung allein keine nachhaltige Lösung darstellt.
9. Risiko, dass wichtige Access-Dateien bei der forensischen Analyse mit Autopsy übersehen werden (einschließlich in Tabellen eingebetteter Anhang) und erhöhter Auswertungsaufwand, da die Analyse teilweise unübersichtlich ist.

6. Handlungsempfehlungen & Ausblick

Aufbauend auf der Auswertung (*siehe Kapitel „5 Auswertung der Sicherheitsanalyse“*) der durchgeführten Analyse (*siehe Kapitel „4 Durchführung der Sicherheitsanalyse“*) werden in diesem Kapitel als Ergebnis der Sicherheitsanalyse konkrete Handlungsempfehlungen gegeben, um die IT-Sicherheit zu steigern.

Unabhängig von den folgenden Ausführungen wird generell empfohlen Standards, wie die frei zugänglichen BSI-Bausteine, die BSI-Normen zur Konfiguration der Computer- und Benutzerrichtlinien oder zur Konfiguration des Trust Centers, von MS 365 und Access, die technischen Sicherheitsanforderungen der Deutsche Telekom AG oder die OWASP-Cheat Sheet Series, zur weiteren Härtung zu berücksichtigen (*siehe Kapitel „3.2 Identifikation von gängigen Sicherheitsstandards für relationale DBSs (Kriterienanalyse)“*). Auf Standards sollte jedoch nicht blindlings vertraut werden, da sie technische Entwicklungen und neue Erkenntnisse, wie die Gefahren durch regelmäßige Passwortänderungen, unter Umständen erst mit Verzögerung berücksichtigen.

MS versichert zwar, dass aufgrund der in *Kapitel „4.1.4 Passwörter & Authentifizierung“* gefundenen Schwachstelle in Azure AD keine Nutzeraktionen erforderlich sind und MS diesbezüglich alle betroffenen Kunden informiert hat. Dem *Heise*-Redakteur *Jürgen Schmidt*, der mehr Transparenz und Details zu dem Vorfall fordert, kann aber dennoch nur zugestimmt werden [54]. Die Umstände erschüttern das Vertrauen in MS und die Sicherheit ihrer Produkte. Die besten und zukunftsweisenden Funktionalitäten rechtfertigen keinen laschen Umgang mit den (Kunden-)Daten und der IT-Sicherheit. Aus Sicht des Verfassers hat Sicherheit stets Vorrang vor Funktionalität, daher sollte MS so bald wie möglich zu der im Jahr 2002 gestarteten Initiative „Trustworthy Computing“ zurückkehren. Auch wenn die Ansätze von MS zukunftsweisend sind, scheinen die Produkte noch einige gravierende Schwächen zu haben. Im Hinblick auf den Funktionsumfang kann die Nutzung der Azure-Cloud-Dienste eindeutig empfohlen werden. Aus sicherheitstechnischer Sicht wird empfohlen mit der Migration in die Azure-Cloud einschließlich Azure AD und dem Dataverse zumindest für kritische Daten und Anwendungen noch so lange zu warten, bis die „Kinderkrankheiten“ behoben sind und MS maximale Transparenz geschaffen hat. MS und die Cloud-Technologie dürfen für Unternehmen keine Bedrohung sein, sondern müssen einen Mehrwert generieren.

Nach dem Motto des *Spring-Frameworks* „Konvention über Konfiguration“ werden viele sicherheitskritische Prozesse standardisiert und an MS in die Azure-Cloud ausgelagert [247]. Nun gilt es, die Funktionalitäten in einer Community nach dem Kerckhoffs’schen Prinzip, möglicherweise auch mit Unterstützung von KI so schnell wie möglich weiter zu verbessern, damit Schwachstellen, wie die kürzlich bekannt gewordene, der Vergangenheit angehören und die Sicherheit eines Verfahrens nicht von seiner Geheimhaltung abhängt. Nutzerdaten zum gezielten Training der KI helfen MS bei der Weiterentwicklung und Härtung sicherheitsrelevanter Funktionalitäten, im Gegenzug werden den Kunden standardisierte und robuste Lösungen zum Schließen potenzieller Einfallstore angeboten. Darunter etwa die Freischaltung von Zugangsberechtigten über Firewall-Regeln, AD- und Multi-Faktor-Authentifizierung, intelligente Passwortsperren oder die erweiterte Bedrohungserkennung. Neben Auftragskontrollen bei externen Dienstleistern

oder Auftragsverarbeitern, wie sie zur Einhaltung des Bundesdatenschutzgesetzes (BDSG) erforderlich sind (vergleiche § 64 Abs. 3 Nr. 12 BDSG, Auftragskontrolle), wird analog zur Always Encrypted-Funktionalität von der Azure SQL-DB empfohlen, Daten idealerweise nur in verschlüsselter Form auszulagern.

Vor Nutzung der Azure-Cloud wird empfohlen eine möglichst detaillierte Sicherheitsanalyse durchzuführen. Sollte MS die aus den Sicherheitsdefiziten resultierenden Anforderungen nicht erfüllen (können), so ist auch hier dem *Heise*-Redakteur *Jürgen Schmidt* zuzustimmen und über Alternativen nachzudenken sowie ein klares Signal an MS zu senden. Ein erster Schritt in die richtige Richtung ist die Ankündigung, dass der Vorfall nun vom *Cyber Safety Review Board* (analysiert bedeutende Cyber-Sicherheitsfälle und gibt konkrete Handlungsempfehlungen) untersucht und der Bericht unter anderem *Joe Biden* (Präsident der Vereinigte Staaten) vorgelegt wird [37]/[53]/[54].

Dabei sollte neben den neuesten MS-Produkten, wie der Azure-Cloud, auch die Office-Suite ein Sicherheitsupdate erhalten. Aufgrund der in dieser Arbeit aufgezeigten Schwachstellen wird empfohlen, in der Anwendungsentwicklung auf die Verwendung von um eigene Logik erweiterte Office-Programme generell zu verzichten.

Einige Schwachstellen, wie die Notwendigkeit des Schreibzugriffs auf das Backend, die leichte Kopierbarkeit oder die schlechte Datenzentralisierung, gelten auch für andere dateibasierte DBSs. Daher wird empfohlen, bei relevanten Daten zumindest für das Backend anstelle Access ein DBS mit Client-Server-Architektur einzusetzen, um die Vertraulichkeit, Verfügbarkeit, Authentizität und Integrität der Daten effektiver schützen zu können.

Wenn Office-Produkte mit eigener Logik erweitert werden müssen, wird empfohlen keinen vertrauenswürdigen Pfad oder Herausgeber zu konfigurieren und die Logik sollte gemäß MS-Empfehlung nur aus Makroaktionen bestehen, die auch in nicht vertrauenswürdigen Anwendungen und unter nicht vertrauenswürdigen Pfaden ausgeführt werden können. Ein wichtiges Indiz für die Vertrauenswürdigkeit einer Anwendung ist, wenn sie nicht in einer geschützten Ansicht geöffnet werden muss. MS empfiehlt aus Sicherheitsgründen auf VBA-Code zu verzichten, wenn die Datei von mehreren Nutzenden verwendet wird und auf einem Dateiserver abgelegt ist. Der Einsatz von VBA ist denkbar, wenn die Datei lokal abgespeichert ist und nur von einer Person genutzt wird [187]. Daher sollte bei Nutzung von VBA-Code überlegt werden, ob anstelle der zentralen Ablage eines Access-Frontends nicht ein dezentraler Ansatz verfolgt werden kann, indem die Nutzenden jeweils eine lokale Kopie ablegen. Somit sind auf Ablageordner und Datei Leserechte ausreichend, was die Sicherheit erhöht. Sofern Angreifende nicht die zentral abgelegte „Master-Datei“, sondern „nur“ eine lokale Kopie manipulieren, wird die Verbreitung von Schadcode an die übrigen Nutzenden verhindert. Nachteil an diesem Vorgehen ist, dass das Frontend verpflichtend dupliziert wird und Nutzende falsche Versionen nutzen können (vergleiche DLL-Hell). VBA-Funktionalitäten zur automatischen Abfrage neuer Versionen oder der Vergleich der Prüfsumme mit der Master-Datei sind denkbar.

Aber auch aus Gründen der Datenzentralisierung und Standardisierung, wozu auch einheitliche Namenskonventionen, die Datenklassifikation oder die zentrale Verwaltung

von Historien und Backups gehören, ist es generell empfehlenswert interessante Daten an zentraler Stelle für die Fachbereiche abzulegen, zu verwalten und die Zugriffe der Fachbereiche granular mittels Berechtigungen entsprechend den Verantwortlichkeiten zu steuern. Ein möglicher Anwendungsfall für ein zentrales Client-Server-DBS, das von mehreren Anwendungen genutzt wird, ist der Aufbau eines Data Lakes (zentraler Speicherort für alle strukturierten und unstrukturierten Daten) oder einer vergleichbaren Ablagestruktur, um alle relevanten Daten an einem zentralen Ort zu speichern. So können die Daten wiederverwendet, weiterverarbeitet und mit Data Mining-Ansätzen ausgewertet werden. Auf diese Weise kann neues Wissen generiert oder Trainingsdaten für KI-Anwendungen effizient erzeugt werden.

Allein die von der *Datenschutzkonferenz* festgestellte Datenschutzwidrigkeit von MS 365 und die damit zusammenhängende Telemetrie kann ein potenzieller Showstopper für einige Unternehmen, Daten und Anwendungsfälle sein. Dabei ist jedoch zu berücksichtigen, dass MS die gesammelten Daten auch zur Härtung von sicherheitsrelevanten Funktionalitäten, wie die Azure SQL-Überwachung, verwendet. Derartige Verarbeitungszwecke werden als sinnvoll und positiv bewertet.

Nach dem Standard-Datenschutzmodell stellt die Verschlüsselung von gespeicherten oder transferierten Daten sowie die Verfahren zur Verwaltung und zum Schutz der kryptografischen Informationen (Kryptokonzept) eine geeignete Referenzmaßnahme zur Gewährleistung der Vertraulichkeit dar. Daher wird empfohlen, dass vertrauliche Daten nur in verschlüsselter Form in MS-Dienste ausgelagert werden und der Schlüssel MS nicht bekannt gemacht wird [38, S. 32]. Es wird empfohlen die Azure SQL-DB daraufhin zu überprüfen, ob die Always Encrypted-Funktionalität eine geeignete TOM zur Risikominimierung darstellt. Bei der Analyse sollte auch der Always Encrypted-Treiber von MS auf potenzielle Telemetrie überprüft werden. Wie bei MS 365 ist auch hier der Einsatz eines zwischengeschalteten Filter-Proxys als Gegenmaßnahme denkbar. Bei Access und anderen (MS-)Diensten oder dateibasierten DBSs ist in einer separaten Analyse zu prüfen, ob die Verschlüsselung der Daten auf vergleichbare Weise umgesetzt werden kann. „Vault“ von *HashiCorp* könnte eine Option sein, bei der sich eine weitere Analyse lohnt. Mit Vault können unter anderem Daten in Dateisystemen, in DBSs und der Cloud in Form von Encryption-as-a-Service geschützt werden [52].

Wird nach einer kostenneutralen Client-Server-Alternative zu Access gesucht, empfiehlt MS den SQL Server Express [198]/[208]. Aber auch Open Source-DBSs, wie PostgreSQL, sollten neben kostenpflichtigen Varianten als Alternative für Access berücksichtigt werden. Interessant an Open Source-DBSs ist der der Umstand, dass das BSI im Jahr 2021 Codeanalysen von Open Source-Software gestartet hat, um hier die Sicherheit zu erhöhen und das Vertrauen in Open Source-Software zu steigern. Das Projekt wird unter dem Namen „CAOS 2.0“ fortgesetzt [6]/[180]. Dennoch wird auch bei Open Source-Software generell vor der Einführung eine detaillierte Sicherheitsanalyse inklusive Bestätigung der DS-GVO-Konformität und potenzieller Telemetrie empfohlen. Für die Sicherheitsanalyse von DBSs kann auf diese Arbeit aufgebaut werden. Für eine Eingrenzung möglicher Alternativen wird eine Markanalyse empfohlen.

Grundsätzlich und unabhängig von den gewählten Technologien wird eine Schichtentrennung mindestens zwischen Frontend (GUI und Logik) und Backend empfohlen. Auch

in Hinblick auf den Microservice- und API-Gedanken, der sich aus Banking 4.0 ergibt und, um möglichst keine neuen technischen Schulden zu generieren. Durch die Schichtentrennung ist es möglich, die eingesetzten Technologien flexibel, effizient und im besten Fall unabhängig von anderen Schichten austauschen zu können. Durch die Trennung der Schichten wird zudem der Austausch von Produktiv- mit Testdaten im Rahmen von Test- oder Entwicklungsaktivitäten erleichtert. Darüber hinaus wird aus Sicht des Verfassers empfohlen, dass alle (Office)-Programme unter Berücksichtigung einer Umgebungstrennung (Entwicklung, Test, Produktion) dokumentiert an einem zentralen Ort und nicht über Netzlaufwerke verteilt abgelegt werden. Die lesenden und schreibenden Zugriffsrechte auf die Ablageorte sind möglichst granular nach dem Need-to-Know-Prinzip und Least-Privilege-Prinzip zu vergeben.

Auf Passwörter zur Authentifizierung ist idealerweise zu verzichten, zumindest sollten diese genau wie Passwörter zum Entschlüsseln des Access-Backends oder sonstige Zugangsdaten nicht unverschlüsselt im Quelltext hinterlegt sein, sofern die ausgewählte Technologie hierfür keine expliziten Funktionalitäten anbietet. Zur Not sind derartige Verbindungsinformationen bei Nutzenden abzufragen. Um zu verhindern, dass die Nutzenden ihre Passwörter an ungeeigneten Orten aufbewahren, wird empfohlen, ein Programm zur Passwort- und Schlüsselverwaltung bereitzustellen, das gemeinsam von mehreren Nutzenden verwendet werden kann. Dies stellt gleichzeitig eine pragmatische Möglichkeit dar, Passwörter oder (private) Schlüssel zum Entschlüsseln der abgelegten Daten aus dem VBA-Quellcode auszulagern oder bei der Programmausführung über Eingabemasken von den berechtigten Nutzenden abzufragen. Auch hier kann sich eine Analyse von Vault lohnen. Anstelle einer Passwortauthentifizierung oder das manuelle Abfragen wird jedoch ohne die zuvor genannten Sicherheitsbedenken eine Authentifizierung über den AD-Ansatz oder ein vergleichbares Verfahren mit einer kontrollierbaren Umgebung, wie Vault (wenn geeignet), empfohlen [52]/[146].

Anstatt auf Verfahrensnutzer zu setzen, wird empfohlen die AD-Accounts von Nutzenden nach dem Need-to-Know-Prinzip auf der Azure SQL-DB granular mit den für ihre Arbeit notwendigen Rechten auszustatten. Nutzende mit gleichem Aufgabenbereich sollten zu AD-Gruppen zusammengefasst werden, um die Rechteverwaltung zu erleichtern.

Sollte es dennoch nach Abwägung erforderlich sein, mit dateibasiertem DBS und nativer Frontend-Anwendung anstelle einer Web-Applikation mit Client-Server-DBS zu arbeiten, so wird grundsätzlich empfohlen, dass Nutzende nur mit Leserechten auf die Frontend-Datei einschließlich des Ablageordners zugreifen dürfen (bei Access Widerspruch zu den für ordnungsgemäßen Betrieb benötigten Rechten). Dadurch wird verhindert, dass die Originaldatei gelöscht oder eine manipulierte, als neue Version getarnte Datei, wie „*Buchhaltungs-Tool_V2.0.0_ Ab dd.mm.yyyy zu nutzen.accde*“, abgelegt wird.

In Access-Datenquellen, Excel und vermutlich auch in anderen Office-Applikationen sind hinterlegte Zeichenketten, wie Passwörter zum Entschlüsseln des Backends oder (private) Schlüssel zum Entschlüsseln der im Backend gespeicherten Daten, ohne Dateiverschlüsselung über den Hexadezimal-Editor einsehbar (auch mit passwortgeschütztem VBA-Projekt). Daher wird auch bei der Wahl der Frontend-Technologie empfohlen, nicht auf ein Programm aus der MS 365-Suite zu setzen. In Anbetracht der Tatsache, dass die Dateiverschlüsselung ein zentraler Schutzmechanismus der Office-

Suite ist, sei an dieser Stelle darauf hingewiesen, dass nach MS ein Passwort allein eine Datei nicht zwangsläufig vor böswilligen Absichten schützt [138].

Die einzigen Vorteile von Excel gegenüber Access als Frontend-Technologie bestehen darin, dass für eine ordnungsgemäße, zentralisierte Nutzung Leserechte auf die Datei und den Speicherort ausreichend sind. Außerdem besitzt Excel viele nützliche Funktionen zur Datenanalyse und es wird mit „Office-Skripten“ eine sicherere Alternative zu VBA angeboten. Ein gravierender Nachteil gegenüber Access ist, dass im betrachteten Excel-Dateiformat .xlsm im Gegensatz zum .accde-Dateiformat von Access der VBA-Code einsehbar ist und damit auch die Erstellung manipulierter Kopien mit äquivalenter Funktionalität erleichtert wird. Bei der Auswahl von Excel oder Access sind die Vor- und Nachteile abzuwägen, wobei stets zu berücksichtigen ist, dass alle gespeicherten Zeichenketten ohne Dateiverschlüsselung eingesehen werden können. Da der VBA-Projekt-Passwortschutz leicht ausgehebelt werden kann, wird empfohlen ihn nicht als Schutzmechanismus anzusehen, sondern lediglich zur Benutzerführung zu verwenden. Liegt die Excel-Datei im .xltm-Dateiformat vor, kann bei zentraler Nutzung Multi-User-Support simuliert werden, da Nutzende jeweils eine Kopie der Datei öffnen. Zur Benutzerführung und, um zumindest unbeabsichtigte Manipulationen zu verhindern, ist trotz der einfachen Umgehung der Blattschutz, der Strukturschutz und der VBA-Projektschutz stets mit einem Passwort zu schützen. Gleiches gilt für die Deaktivierung des Menübands, der Spezialtasten-Kombinationen und den VBA-Projektschutz in Access. Die Verwendung von Excel als Backend-Technologie wird nicht empfohlen.

Ein weiterer Vorteil von Excel gegenüber Access ist die IRM-Kompatibilität. Das IRM besitzt im Vergleich zu Dateisystemberechtigungen einen erweiterten Funktionsumfang. Neben Dateien wie Word, Excel und E-Mails kann teilweise auch der Inhalt geschützt werden. Dies wird durch eine Form der Überwachung erreicht, die verhindert, dass autorisierte Empfänger die Daten für nicht autorisierte Zwecke kopieren, einfügen, weiterleiten oder drucken. Darüber hinaus ermöglicht das IRM die Einschränkung von Inhalten, das Festlegen eines Verfallsdatums für Dateien und die Steuerung und Kontrolle der Nutzung sowie Verteilung von Inhalten innerhalb des Unternehmens. IRM bietet keinen Schutz gegen manuelles Kopieren oder Abschreiben von Daten, Snipping Tools, Abfotografieren des Bildschirms und löschende oder datensammelnde Malware. Bei Verwendung von IRM in Verbindung mit einem MS Exchange Server kann auch der E-Mail-Anhang verschlüsselt werden, während die E-Mail-Nachricht im Klartext erhalten bleibt. Die von IRM angebotene Funktionalität macht einen sehr zukunftsweisenden ersten Eindruck und daher wird eine weitere (Sicherheits)Analyse von IRM empfohlen. Zu beachten ist, dass IRM auch auf AD-Rechten basiert. Es sollte in einer separaten Analyse geprüft werden, ob es eine vergleichbare Alternative für Access gibt [113]/[172].

Wenn Access zwingend als Frontend oder Backend-Technologie eingesetzt werden muss, wird angeraten stets eine Kompilierung in das .accde-Dateiformat vorzunehmen, da hier zumindest keine Einsicht in den VBA-Code möglich ist und einige Funktionen deaktiviert sind, was das Erstellen einer manipulierten Kopie der Datei erschwert. Aufgrund der vielen gefundenen Schwachstellen ist zu überprüfen, dass der in .accde-

Dateien hinterlegte VBA-Code nicht durch Dekompilieren der Access-Datei oder über andere Umwege wiederhergestellt werden kann.

Des Weiteren wird empfohlen zumindest das Access-Backend mit einem starken Passwort zu verschlüsseln, um Brute-Force-Angriffe zu erschweren und die Wahrung der Vertraulichkeit zu unterstützen. Da die Dateientschlüsselung nur mit einem Passwort möglich ist, wird ein Passwortschutz des Frontends nur dann als wirksam angesehen, wenn das Passwort unter den vertrauenswürdigen Nutzenden geheim bleibt, was kaum überprüfbar ist. Es wird empfohlen, eine Unternehmensrichtlinie für die Vergabe möglichst starker Passwörter zu veröffentlichen, da schwache Passwörter durch Brute-Force- oder Wörterbuchangriffe schnell entschlüsselt werden können. Als Referenz bietet MS diverse Quellen und Empfehlungen für die Passwortvergabe an. Demnach gelten Passwörter ab 18-20 Zeichen (Klein- und Großbuchstaben, Ziffern und Sonderzeichen) als wirklich robust gegen Brute-Force-Angriffe. Mehr als 10 Zeichen Mindestlänge für Passwörter soll laut MS zu vorhersagbarem Verhalten bei Nutzenden im Rahmen der Passwortwahl führen. Ähnlich wie bei regelmäßiger Änderung von Passwörtern ist davon auszugehen, dass Nutzende ohne die Verwendung von Passwortgeneratoren bei der Passwortwahl sich wiederholende Muster verwenden. Das BSI konkretisiert diesen Sachverhalt. Demnach ist ein starkes Passwort entweder kürzer und komplex (8-12 Zeichen, mit vier Zeichenarten wie Sonderzeichen und Ziffern) oder länger und dafür weniger komplex (20-25 Zeichen, mit zwei Zeichenarten). Ein 8-stelliges Passwort und drei Zeichenarten ist nur in Verbindung mit einem zusätzlichen Mehr-Faktor-Authentisierungsmerkmal sicher. Die Verwendung einer Multi-Faktor-Authentifizierung wird generell vom BSI empfohlen [15]/[133]/[135]/[224].

Da Nutzende nur lesend auf das Access- oder Excel-Frontend zugreifen, wird die zugriffsgeschützte Ablage einer Prüfsumme des Frontends zur Überprüfung der Programmintegrität (insbesondere des Quellcodes) empfohlen. Dasselbe gilt für andere Frontend-Technologien. Die Prüfsummen sollten in regelmäßigen Abständen zentralisiert auf Veränderungen geprüft werden, um etwaige Manipulationen erkennen zu können. Bei anderen Technologien ist auch der Einsatz von digitalen Signaturen denkbar, wenn diese bei Manipulationen an der Datei automatisch verworfen werden.

Als Alternative zu einem Access- oder Excel-Frontend und zur Erleichterung der Schichtentrennung wird eine Sicherheitsanalyse der Visual Studio Tools for Office (VSTO) und der „Excel Add-In“-Möglichkeiten empfohlen. VSTO nutzt in einem Visual-Studio-C#-Projekt die Mitarbeitenden vertrauten Office-Anwendungen beziehungsweise -Oberflächen, wie Excel oder Word, inklusive der dort verfügbaren Funktionen als Frontend. Dabei wird aus dem C#-Projekt eine DLL erstellt, die dann dem jeweiligen Frontend-Programm hinzugefügt wird. Das C#-Projekt ist getrennt von der jeweiligen Office-Datei. C#-Code kann in modernen integrierten Entwicklungsumgebungen wie Visual Studio geschrieben werden. Im Vergleich zum völlig veralteten, in Office-Produkten integrierten VBA-Editor bietet Visual Studio einen wesentlich größeren Funktionsumfang und ermöglicht ein einfacheres Refactoring. Im Vergleich zu VBA ist C# eine viel mächtigere Programmiersprache und unterstützt das Paradigma der objektorientierten Programmierung im vollen Umfang. Der C#-Quellcode kann durch Packages strukturiert werden, was mit VBA-Modulen nicht möglich ist. C#-Code kann mit professionellen Versions-

verwaltungstools, wie „Git“, versioniert werden. Git stößt bei Binärdateien, wie Access, schnell an seine Grenzen. Auch das Testen in Form von Modultests ist mit C# weitaus effizienter und automatisierbarer als in VBA (vergleiche diverse Drittanbieterbibliotheken wie NUnit und die Live Unit Testing-Funktionalität von Visual Studio). Dieser Umstand wirkt sich positiv auf die Forderung der MaRisk AT 7.2 Tz 3 aus, dass IT-Systeme vor ihrem erstmaligen Einsatz und nach wesentlichen Veränderungen von fachlich und technisch zuständigen Mitarbeitenden zu testen sind. Über die für C# angebotene Modultest-Funktionalität können auch bei unwesentlichen Änderungen, wie ein Refactoring, gemäß der Clean-Code „Pfadfinderregel“ (Boy Scout Rule, Hinterlasse einen Ort immer in einem besseren Zustand als du ihn vorgefunden hast) effizient Regressionstests automatisiert ausgeführt werden. Dadurch wird die Wartbarkeit gewährleistet und es entstehen nach der „Broken Windows Theorie“ (Verfall von Qualität beginnt allgemein mit Kleinigkeiten, die nur lange genug unbeachtet bleiben) keine „zerbrochenen Fenster“ (Code-Erosion). Die Programmqualität bleibt über einen längeren Zeitraum erhalten oder kann im Idealfall sogar verbessert werden [27]/[114]/[119].

MS empfiehlt die Erstellung einer individuellen Office-Lösung mittels VSTO anstelle der Nutzung einer Web-Anwendung für die folgenden Anwendungsfälle: Textverarbeitungsprogramme wie die Erstellung eines Vertragsgenerators (Word), Datenanalyseprogramme, wie ein automatisiertes Budgetarbeitsblatt, das von mehreren Projekten genutzt werden kann (Excel) oder E-Mail-Verwaltungsfunktionen (Outlook). Native Anwendungen haben gegenüber webbasierten Anwendungen den Vorteil, dass sie auch offline verfügbar sind. Außerdem ist dieser Ansatz nach Ansicht von MS bei komplexen Lösungen praktikabler als die Verwendung einer webbasierten Architektur [157].

VSTO-Anwendungen können dabei entweder mit einer bestimmten Office-Datei (Assembly wird einer einzelnen Datei zugeordnet und beim Öffnen der Datei geladen) oder auch als Add-In mit der Office-Anwendung selbst (Nachfolger des VSTO-Ansatzes, Assembly wird einer Office-Anwendung zugeordnet, die beim Starten der Office-Anwendung oder im Nachgang manuell durch Nutzende geladen wird) verknüpft sein. Das Add-In kann somit dateiübergreifend genutzt werden. Über das „*Trust Center → Add-Ins*“ kann festgelegt werden, dass alle Add-Ins von einem vertrauenswürdigen Herausgeber signiert sein müssen [157].

Im Gegensatz zu VSTO und Visual Basic handelt es sich bei VBA um nicht verwalteten Code. Das bedeutet, dass der VBA-Code nicht zuerst nach den Konventionen in die Zwischensprache (Intermediate Language) kompiliert und dann unter der Verwaltung der Common Language Runtime und der damit verbundenen „Just in Time-Kompilierung“ ausgeführt wird. VBA-Code ist zudem durch die Integration eng mit der jeweiligen Office-Datei verbunden. Beispiele für nicht verwalteten Code sind Zeiger und fehlende Dienste für die automatische Arbeitsspeicherverwaltung, für Sicherheitsgrenzen und die Typsicherheit. Darüber hinaus wurde VBA für die Makroaufzeichnung und einen vereinfachten Entwicklungsprozess konzipiert. VSTO wurde für den Einsatz in Unternehmen konzipiert und bietet Sicherheit, einfache Codewartung und die Möglichkeit, den vollen Funktionsumfang von Visual Studio als integrierte Entwicklungsumgebung zu nutzen. VBA hat nach Angaben von MS diverse Einschränkungen für Unternehmen, wie in den

Bereichen Sicherheit und Bereitstellung, die auch durch dieses Dokument aufgezeigt werden. MS empfiehlt VBA lediglich für benutzerdefinierte Arbeitsblattfunktionen und für die Aufzeichnung von Markos [159]/[164].

Die Hauptanforderung nach MS für die Arbeit mit VBA (gilt auch für alle anderen Programmiersprachen und Konzepte wie VSTO) ist, dass genügend Zeit und Raum für sorgfältiges und konzentriertes Arbeiten zur Verfügung steht. Eine unsaubere Programmierung unter Termindruck kann sich schnell rächen. Kann die Hauptanforderung nicht erfüllt werden, empfiehlt MS die Verwendung konventioneller Methoden anstelle einer Programmierung, auch wenn diese monoton sind und eine (manuelle) Aneinanderreihung sich ständig wiederholender Schritte bedeuten [109].

Mit den bereits erwähnten Office-Add-Ins können Office-Anwendungen mit individueller Logik erweitert werden, die mit Inhalten in Office-Dateien interagieren. Damit die Add-Ins im Gegensatz zu VSTO auch plattformübergreifend auf Windows, Mac, dem iPad oder online über den Browser funktionieren, wird hier auf Webtechnologien, wie HTML, CSS und JavaScript, anstelle C# gesetzt. So lassen sich webseitenähnliche Funktionalitäten realisieren, Office-Anwendungen uneingeschränkt erweitern und die erstellten Anwendungen im Gegensatz zu VSTO zentral durch einen Administrator bereitstellen. Die Add-Ins laufen dann im Kontext eines Browsers oder des „WebView“-Frameworks von MS in Form einer Sandbox.

Nach dieser kurzen Analyse scheinen der VSTO- und Add-In-Ansatz eine wesentlich geeignetere und modernere Wahl zu sein als die bisher vorgestellten integrierten VBA-Lösungen [136]. Eine detailliertere Sicherheitsanalyse im Sinne dieser Ausarbeitung auch unter Berücksichtigung der Anforderungen von Banking 4.0, wie dem API- und Microservice-Ansatz oder der Reduktion von technischen Schulden, wird dennoch empfohlen. Bereits nach dieser rudimentären Analyse kann eine deutliche Verbesserung gegenüber den um VBA erweiterten Office-Anwendungen festgestellt werden.

Bei der Analyse ist auch zu prüfen, ob Office-Add-Ins zentral über den Organisations-App-Store in MS Teams bereitgestellt werden können und falls ja, welche Vorteile sich daraus ergeben (*siehe Kapitel „1.3 Anwendungsszenario“*).

In der Analyse von Access-Alternativen sollte auch die Auswirkung auf den Erfüllungsgrad der Zero-Trust-Strategie berücksichtigt werden (*siehe Kapitel „Anlage 1: Zero-Trust-Strategie“*). Es wird empfohlen die Zero-Trust-Strategie bei (Architektur-)Entscheidungen zu berücksichtigen, da es laut MS nicht mehr zeitgemäß ist, dass Unternehmen den Netzwerkzugang nur mit VPN und Firewalls schützen. Nach Angaben von MS basieren sowohl MS 365 als auch MS Azure auf dieser Strategie. Nach der Zero-Trust-Strategie muss jede am Datenaustausch beteiligte Schicht („Identity“, „Endpoints“, „Applications“, „Network“, „Infrastructure“, „Data“) geschützt werden, da jede Schicht ein potenzielles Angriffsziel für den Abfluss sensibler Daten darstellen kann. Angesichts der Möglichkeiten, die neue Technologien, wie Cloud-Computing oder Homeoffice bieten, muss mehr für die Sicherheit getan werden. Die Grundprinzipien der Zero-Trust-Strategie folgen dem Leitsatz „niemals vertrauen und immer überprüfen“ und umfassen die explizite Authentifizierung einschließlich der Verifikation jeder Transaktion, Anwendung

des Least-Privilege-Prinzips sowie die ständige Annahme eines bösartigen Kommunikationspartners. Jede Anfrage, auch aus internen Netzwerken, wird überprüft, als käme sie aus einem öffentlichen Netz. Es kann nicht davon ausgegangen werden, dass alle Kommunikationspartner innerhalb eines Netzwerks vertrauenswürdig sind. Auch die weitere Analyse der Zero-Trust-Strategie wird als Ausblick empfohlen [174]/[193, S. 1]/[223, S. 1].

Um fehlendes IT-Know-how bei den Entwicklern von (Office-)Anwendungen, wie IDV-Anwendungen, zu kompensieren und bisher unbekannte IDV-Anwendungen zu identifizieren, empfiehlt es sich, die unternehmensweit im Einsatz befindlichen Prozesse durch zentrale Kontrollen in Anlehnung an den Datenschutzmanagement-Prozess/-zyklus (sowie den Sicherheitsprozess) schrittweise auf mit diesen Programmen verbundene Risiken oder vermeidbare technische Schulden zu analysieren. Risiken können unter anderem durch Fehlbedienung oder Schwachstellen entstehen. Außerdem liegt der Fokus bei den Kontrollen auf der Effizienzsteigerung, Digitalisierung, Verfolgung der Banking 4.0-Vision und dem Straight Through Processing-Gedanken (Interoperabilität). Dieses Vorgehen steht gleichzeitig im Einklang mit der MaRisk-Vorgabe, die eine regelmäßige Prüfung der IT-Systeme und zugehörigen Prozesse auf Eignung fordert. Eine enge Zusammenarbeit mit der zentralen IT-Abteilung und den professionellen Anwendungsentwicklenden wird bei der Durchführung der Kontrollaktivitäten empfohlen. Anders als im „Datenschutzmanagement-Prozess“ liegt der Fokus bei den Verarbeitungstätigkeiten nicht nur auf personenbezogene Daten, sondern auch auf vertraulichen Daten und kritischen Zwecken. Die IT- und Datensicherheit des Unternehmens hat Priorität. Da Sicherheit kein Zustand, sondern ein Prozess ist, werden die Kontrollmaßnahmen pro Prozess in wiederkehrenden Iterationen durchgeführt. Dies ermöglicht eine Erfolgskontrolle, eine kontinuierliche Verbesserung sowie Anpassbarkeit an neue (regulatorische) Anforderungen. Der Datenschutzmanagement-Prozess orientiert sich an dem aus der Qualitätssicherung bekannten Plan-Do-Check-Act (PDCA)-Zyklus. Werden Schwachstellen festgestellt, werden TOMs zur Behebung der Schwachstellen ergriffen (für personenbezogene Daten sind Referenzmaßnahmen im Standard-Datenschutzmodell zu finden). Können keine TOM angewendet werden oder werden Optimierungspotenziale und vermeidbare technische Schulden identifiziert, kann eine weitere Automatisierung oder Industrialisierung beziehungsweise Weiterentwicklung der IDV-Anwendung aber auch eine Neuentwicklung oder vollständige Neuausrichtung des (Teil)Prozesses oder der Anwendung in Auftrag gegeben werden. Im Idealfall kann die IDV-Anwendung in eine von der zentralen IT-Abteilung verwaltete Anwendung oder Umgebung migriert werden. Gleichzeitig findet eine hausweite Sensibilisierung der Mitarbeitenden für das Thema und die damit verbundenen Risiken statt. Bisher unbekannte Daten, die in den Fachbereichen entstehen, werden identifiziert. In einem zweiten Schritt erfolgt die Auslagerung relevanter Daten und somit neuem Wissen in eine zentrale Instanz wie einem Data Lake oder eine ähnliche Speicherstruktur [10]/[38]/[41]:

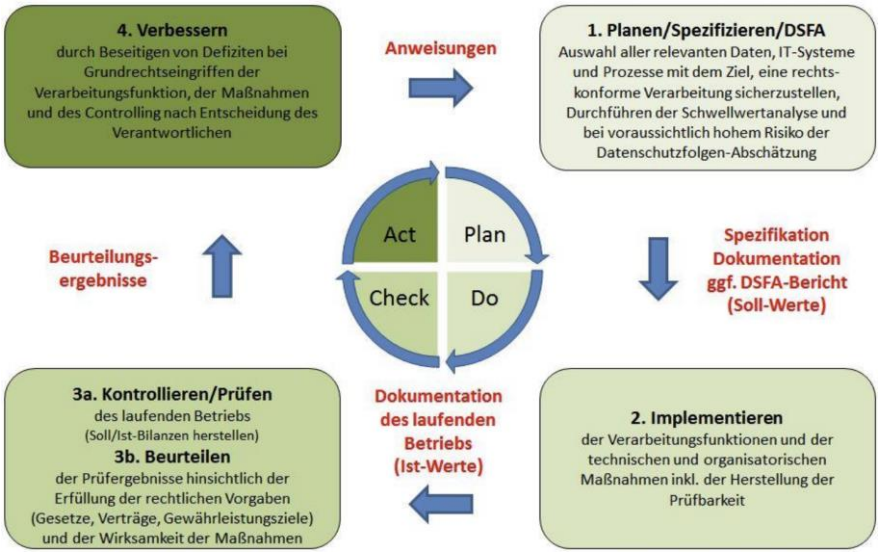


Bild 127: Datenschutzmanagement-Zyklus (angelehnt an PDCA-Zyklus) als mögliche Vorlage für ein potenzielles Vorgehen bei Kontroll- und Digitalisierungsmaßnahmen [38, S. 62]

In der Regel bestehen Programme zur Datenverarbeitung nach dem „EVA-Prinzip“ aus Eingabe, Verarbeitung und Ausgabe [62]. Daher wird ergänzend zu den Kontrollhandlungen empfohlen, in Anlehnung an den Ansatz „Konvention über Konfiguration“ des *Spring-Frameworks* zu prüfen, ob ein Muster-/Modellprogramm einschließlich einem Konzept für ein standardisiertes Vorgehen als Blaupause für alle Fachbereiche entwickelt werden kann [247]. Das Musterprogramm sollte idealerweise höhere Kritikalitätsanforderungen erfüllen, um proaktiv auf mögliche Verschärfungen in der Zukunft vorbereitet zu sein.

In diesem Zusammenhang und als mögliche Alternative zu Office-Programmen, die mit VBA erweitert werden, wird die Analyse (einschließlich Sicherheits- und Datenschutzaspekten) von „Low-Code-Plattform“-Ansätzen, wie MS Power Apps, Power Automate und dem Dataverse, empfohlen, um die Effizienz zu steigern und die technischen Schulden zu verringern. Mit Power Apps können IT-fremde Fachbereiche selbstständig (IDV-)Anwendungen erstellen, ohne dabei über die Kenntnisse eines professionellen Anwendungsentwickelnden verfügen zu müssen. Im Idealfall können über derartige Plattformen möglichst viele (regulatorische) Anforderungen standardisiert erfüllt werden, sodass sich die Fachbereiche auf die Umsetzung der Geschäftslogik konzentrieren können. Gleichzeitig wird das Risiko einer Schatten-IT verringert, da die Anwendungsentwicklung in einer Umgebung erfolgt, die zentral von einer IT-Abteilung verwaltet und überwacht werden kann. Anstelle eines zugrundeliegenden DBS verfügen Power Apps-Anwendungen über ein eigenständiges „Data Management System“ (Common Data Service), auf dem Nutzende auch Validierungsregeln konfigurieren können. Bei Low-Code-Plattformen erfolgt die Anwendungsentwicklung mit einfach zu bedienenden (grafischen) Werkzeugen und in der Regel ohne Schreiben von klassischem Quellcode wie C# [210, S. 4-5, S. 9].

Wird nach einer Abwägung dennoch der klassische Ansatz gewählt, wird empfohlen, über das Musterprogramm möglichst standardisierbare Funktionalitäten für viele Anwendungsfälle anzubieten. Mögliche Anwendungsfälle sind das zuverlässige Einlesen der

Rohdaten aus unterschiedlichen Formaten, wie Excel oder CSV-Dateien (inklusive Eingabeprüfung), das Zwischenspeichern oder das Persistieren der Ergebnisse in einer DB-Relation, um in der weiteren Verarbeitung mit SQL auf die Daten zugreifen zu können („Einlese-Service“, „Ausgabe-Service“). Wenn die Eingabedaten aus anderen Systemen bezogen werden können, ist eine dauerhafte und redundante Speicherung der Eingabedateien in der Regel nicht zu empfehlen. Zumal Dateien ohne IRM schnell kopiert werden können. Zusätzlich erschweren redundante Datenhaltungen die Einhaltung von Löschpflichten. Denkbar sind auch Historisierungsfunktionalitäten zur Protokollierung der einzelnen Programmläufe, ähnlich der von der Azure SQL-DB angebotenen Change Data Capture-Funktionalität, um importierte Daten, Zwischenergebnisse oder generierte Endergebnisse einer DB nachvollziehbar zu persistieren. Auch ein Anonymisierungsdienst, der personenbezogene Daten in einer Datenquelle, wie Access anonymisiert oder löscht, wenn der Zweck erfüllt ist, ist denkbar. Es kann auch ein Service angeboten werden, der die zu speichernden Daten ver- und entschlüsselt, ein Dienst, der anonymisierte Testdaten erzeugt oder ähnliche Funktionen, wie die in dieser Arbeit vorgestellten Azure-Cloud-Dienste und Funktionen der Azure SQL-DB, zur Überwachung oder Protokollierung. Ob eine KI-Unterstützung bei der Testdatengenerierung möglich ist, muss perspektivisch ebenso analysiert werden wie die konkrete Art der Bereitstellung.

Soll weiterhin mit VBA gearbeitet werden, können die standardisierten Funktionen als DLL für Entwickler bereitgestellt werden. Die DLL-Dateien können in den eigenen VBA-Code importiert und leicht durch neue Versionen der DLL ausgetauscht werden. Durch derartige Standardisierungen werden die Algorithmen zunehmend robuster, da Nutzende parallel als Testende fungieren und möglicherweise selbst Weiterentwicklungen vornehmen, von denen wiederrum andere Nutzende profitieren können. Im besten Fall findet ein reger Austausch und Wissenstransfer zwischen den Entwicklern der unterschiedlichen Fachbereiche eines Unternehmens statt, einschließlich professionellen Anwendungsentwickelnden aus der zentralen IT-Abteilung. Dies würde die allgemeine IT-Kompetenz zum Nutzen vieler erhöhen. Zur Kollaboration ist auch ein firmeninternes *Stack Overflow* oder eine ähnliche Plattform für den Austausch denkbar. Die Sicherheit der standardisierten Funktionalitäten hängt nach dem Kerckhoffs’schen Prinzip nicht von der Geheimhaltung des Algorithmus ab und die allgemeine Sicherheit der verwendeten Funktionen wird erhöht.

Alle .NET-Programmiersprachen, wie Visual Basic.NET und C#, werden in eine einheitliche Zwischensprache („Intermediate Language“) kompiliert, die dann von der Common Language Runtime während der Programmausführung von einem Just in Time-Compiler in prozessorabhängigen Maschinencode übersetzt wird, den der Prozessor dann ausführt. Aus diesem Grund laufen .NET-Applikationen mit der Zeit schneller. Denn mit der Zeit wird immer mehr Code kompiliert, der bei erneuten Aufrufen wiederverwendet werden kann. Da alle .NET-Sprachen in die einheitliche Intermediate Language kompiliert werden, können Objekte, die in verschiedenen .NET-Sprachen geschrieben werden, miteinander interagieren, sich gegenseitig aufrufen und voneinander erben [240]. Der Austausch wird auch durch die standardisierte COM-Schnittstelle erleichtert (*siehe Kapitel „2.2 Beispiele für Datenzugriffe in Access“*). Es ist daher möglich Objekte und DLLs in der Programmiersprache C# zu programmieren, die dann in den VBA-Code für

die zuvor beschriebene Verwendung importiert werden können. Stattdessen wird jedoch der bereits skizzierte VSTO-Ansatz empfohlen.

7. Fazit

Abschließend wird in diesem Kapitel auf Basis der zuvor durchgeführten (*siehe Kapitel „4 Durchführung der Sicherheitsanalyse“*) und bewerteten (*siehe Kapitel „5 Auswertung der Sicherheitsanalyse“*) Sicherheitsanalyse ein Fazit in Form eines Überblicks gegeben. Dabei wird auch die Frage aus der ursprünglichen Aufgabenstellung aufgegriffen, ob und gegebenenfalls wann MS Access eine Alternative zu „professionellen“ DBSs wie die Azure SQL-DB sein kann. Außerdem werden als Zusammenfassung für die Ergebnisse der Sicherheitsanalyse (*siehe Kapitel „6 Handlungsempfehlungen & Ausblick“*) mögliche TOM gegen die identifizierten Bedrohungsszenarien vorgestellt.

Stillstand ist der Anfang allen Übels:

Tabelle 8: Gegenüberstellung der Vor- und Nachteile von Access und dem Vergleichs-DBS Azure SQL-DB

	Access	Azure SQL-DB
	Einfache und unkomplizierte Bedienung (auch für Mitarbeitende mit wenig IT-Know-how) einschließlich vieler nutzbarer grafischer Werkzeuge über die Access-Oberfläche.	Reduzierung von technischer Komplexität durch viele Standardisierungen und technische Unterstützungen. Das Konzept setzt einen Standard zur Erhöhung der IT-Sicherheit. Verfolgung des *Spring-Framework*-Ansatzes „Konvention über Konfiguration“.
	Anwendungen können ohne Quellcode über die Oberfläche erstellt werden.	Platform-as-a-Service-Lösung und kontrollierbare Umgebung. Daten, Hardware und IT-Sicherheit werden an MS ausgelagert. Reduzierter Aufwand und weniger internes Wissen, wie aufgrund fehlender Hardware-Wartung, benötigt.
	-	Banking 4.0-Konform. Viele zukunftsweisende Funktionalitäten und Dienste darunter: TDE, Azure AD-Authentifizierung (inklusive granulare Berechtigungsvergabe und Multi-Faktor-Authentifizierung), intelligenter Passwortschutz, automatische Überwachungsfunktionalität (Azure SQL-Überwachung), erweiterte Bedrohungserkennung inklusive KI, automatisierte Sicherheitswarnungen und -empfehlungen, Change Data Capture-Funktionalität, Ledger sowie Tipps und Empfehlungen zur Einhaltung

	Access	Azure SQL-DB
		gesetzlicher Bestimmungen wie zum Datenschutz.
	-	Die Dienste der Azure-Cloud lassen sich je nach Anforderung zu einer standardisierten Infrastruktur kombinieren.
	-	Zentrale Verwaltung aller Azure Dienste über das Azure-Portal. Zentralisierung der Datenhaltung.
	Viele Schwachstellen und Erhöhung von technischen Schulden. Monolithische Anwendungen sind nicht mehr zeitgemäß. Geringer Funktionsumfang, Zugriffe nicht kontrollier- und protokollierbar. Die Vertraulichkeit, Verfügbarkeit, Integrität und Authentizität von Logik und Daten kann kaum überprüfbar sichergestellt werden.	Schwachstelle in Azure AD, teilweise intransparenter Umgang von MS.
	Über DbVisualizer oder Eigenentwicklungen, die den UCanAccess-Treiber nutzen, können sämtliche Schutzmechanismen, wie benutzerdefinierte Menübänder, Gültigkeitsprüfungen und die Ausführung von VBA-Code oder (Daten)Makros, umgangen werden.	Relativ neuartiger Ansatz. Weitere „Kinderkrankheiten“ sind aufgrund der aktuellen Ereignisse zu erwarten.
	Viele Schwachstellen, die vor Nutzenden mit geringen technischen Kenntnissen versteckt werden, sodass falsche Sicherheit angenommen wird.	Abgabe von Verantwortung aufgrund der Auslagerung von Daten, Hardware und IT-Sicherheit an MS. Es muss vertrauen in MS herrschen.
	Im .accdb-Dateiformat kann das VBA-Projektschutzpasswort leicht ausgehebelt werden.	Intransparente Telemetrie.
	Einziger Schutz besteht in Dateisystemberechtigungen und der Dateiverschlüsselung. Kompilieren in das .accde-Dateiformat ist nur bedingt sicherheitsrelevant. Zur ordnungsgemäßen Nutzung ohne exklusives Öffnen werden umfangreiche Rechte auf den Ablageort der Access-Datei benötigt. Eine Access-Datenquelle kann mit Leserechten kopiert werden.	-
	Über den Hexadezimal-Editor können ohne Dateiverschlüsselung im .accdb	-

	Access	Azure SQL-DB
	sowie .accde-Dateiformat hinterlegte Zeichenketten im Klartext eingesehen werden (auch in passwortgeschützten VBA-Projekten).	
	Gelöschte Daten werden erst nach Nutzung der „Datenbank komprimieren und reparieren“-Funktion bereinigt und sind erst dann nicht mehr über den Hexadezimal-Editor einsehbar (kann auch gültige Daten löschen).	-
	Keine Möglichkeit die Datenhaltung zu zentralisieren oder alle Access-Datenquellen zentral in einer kontrollierten Umgebung zu verwalten. Daher erhöhte Gefahr für Redundanzen, Inkonsistenzen und Schwachstellen.	-
	Intransparente Telemetrie.	-

Für den Einsatz beziehungsweise Umgang mit Access und als Zusammenfassung für die gegebenen Handlungsempfehlungen werden auf dieser Basis nachfolgend konkrete TOM für die identifizierten Bedrohungsszenarien gegeben (Aufzählung nicht erschöpfend). Manche der Empfehlungen sind nur in Kombination mit anderen Maßnahmen wirksam. Andere Handlungsempfehlungen schützen gleichzeitig vor mehreren Bedrohungen. Die <*„Nummern am Ende“*> stehen für die Nummern relevanter Bedrohungsszenarien. Als Fazit lässt sich festhalten, dass es mit Access schwierig ist, alle Bedrohungen wirksam zu bekämpfen. Oft können nur Teilaspekte der Bedrohungen adressiert werden und es müssen Abstriche gemacht werden, wie aus den Kommentaren hinter den jeweiligen Nummern ersichtlich ist *(Folgen sie diesem Link zu den Bedrohungsszenarien)*:

- Kompilieren der produktiven Version in das .accde-Dateiformat *<1 (bezogen auf VBA-Code), 2 (siehe vorheriger Kommentar), 3 (siehe vorheriger Kommentar), 5 (Standardisiertes Vorgehen trägt zur Sicherheit bei)>*.
- Keine Konfiguration von vertrauenswürdigen Pfaden oder Herausgebern. Die Logik sollte nur aus Makroaktionen bestehen, die auch in nicht vertrauenswürdigen Anwendungen und unter nicht vertrauenswürdigen Pfaden ausgeführt werden können *<1, 5>*.
- Verbot von SQL Client und Database Management Software wie DbVisualizer für Nutzende. Nur Anwendungsentwickelnde haben Zugriff auf die Vollversion von Access, Nutzende haben lediglich die Access Runtime installiert *<wenn Nutzung über Access-Oberfläche erzwungen wird: 1, 2, 3, 4, 5>*.
- Aufteilen der Anwendung in ein separates Access-Frontend und -Backend *<1 (Logik und Daten sind getrennt. Passwort zum Entschlüsseln des Access-Backends darf nicht über Frontend eingesehen werden können), 2 (siehe vorheriger Kommentar), 3*

(bezogen auf Daten im Backend, wenn Entschlüsselungspasswort nicht eingesehen werden kann), 6 (siehe vorheriger Kommentar)>.

- Standardisierter, zentraler Ablageort für alle nativen Anwendungen und der zugehörigen Access-Backends zur Datenhaltung. Bestmögliche Standardisierung und Bereitstellung eines Musterprogramms einschließlich Konzepts für ein standardisiertes Vorgehen, das kontinuierlich und gemeinschaftlich weiterentwickelt wird *<1, 2, 3, 5 (ohne Musterprogramm erhöhen Punkte, wie eine Schichtentrennung, die Komplexität und die Gefahr entsteht, dass Vorgaben nicht eingehalten werden), 6, 7; wenn beim Modellprogramm nicht auf Office-Produkte gesetzt wird: 1-9 (nicht auf Autopsy bezogen, sondern auf die Vermeidung von Access-Anwendungen aufgrund Anbietens von Alternativen>*.
- Umgebungstrennung (gegebenenfalls mittels Ordner) zwischen Entwicklung, Test und Produktion. Etablierung eines geregelten Softwareentwicklungsprozesses *<2 (keine Entwicklung in der Produktivumgebung), 3 (Produktivdaten), 6>*.
- Die Berechtigungsvergabe für Lese- und Schreibzugriffe sind je Umgebung nach dem Need-to-Know-Prinzip und dem Least-Privilege-Prinzip geregelt. Nutzende erhalten ausschließlich Leserechte auf die Frontend-Datei *<1 (bezogen auf VBA-Code), 2 (nur auf Ebene von Lese- und Schreibrechten möglich. Entwickler sollten nur in Ausnahmefällen Zugriff auf die Produktionsumgebung besitzen), 3, 4 (Zugriffe können über Funktionen des Dateisystems überwacht werden) 5, 6>*.
- Bereitstellung einer fälschungssicher abgelegten Prüfsumme für das Frontend, da Nutzende nur Leserechte auf die Frontend-Datei haben und sich somit das letzte Änderungsdatum beim alleinigen Öffnen nicht ändert (BAIT fordert Vorkehrungen zu treffen, die versehentliche oder absichtliche Manipulationen transparent machen). Alle zentral abgelegten Dateien sollten von einer zentralen Einheit in einem festgelegten Intervall auf Unterschiede zur Prüfsumme kontrolliert werden, um Manipulationen zu erkennen *<1, 5, 6 (Integrität der Logik)>*.
- Regelmäßige Datensicherungen von Front- sowie Access-Backend. Für das Backend wird empfohlen zusätzlich auch ein Upload in ein Client-Server-DBS in Betracht zu ziehen, um die dort verfügbaren Funktionalitäten, wie zur Datensicherung, zu nutzen. Idealerweise wird ein Client-Server-DBS als Backend genutzt und Access wird nur in legitimen Anwendungsfällen als Frontend eingesetzt. Die Datenhistorisierung und -archivierung wird ebenfalls an das Client-Server-DBS ausgelagert. Bei einer Prozessüberarbeitung sollte im Sinne einer durchgehenden Datenverarbeitung und Banking 4.0 auf so wenig manuelle Datenabzüge wie möglich gesetzt werden, auch um die Datenhaltung in einer kontrollierbaren Umgebung zu zentralisieren *<1 (bezogen auf Daten), 2, 3 (bezogen auf Daten), 4, 6, 7, 8, 9 (nicht auf Autopsy bezogen, sondern auf das Auffinden von Daten im Allgemeinen>*.
- Verschlüsselung der im Backend gespeicherten Daten und der Access-Datenquelle mit einem starken Passwort. Der Schlüssel zum Entschlüsseln der Daten oder der Datei ist an einem separat zugriffsgeschützten Ort, wie einem Passwortmanager, hinterlegt und muss im ungünstigsten Fall bei jeder Programmausführung eingegeben werden, damit die Information aus dem Office-Frontend ausgelagert wird. Eine Alternative können Windows-Zertifikatspeicher oder Vault sein, bei der Nutzende nicht zwangsläufig mit den jeweils

erforderlichen Sicherheitskomponenten in Berührung kommen. Festlegen von verpflichtend zu erfüllenden Vorgaben zur Passwortwahl. Aufgrund der Schwachstellen bei Access idealerweise keine Ablage von Passwörtern im Quellcode, zumindest nicht im Klartext. Bei Nutzung von Drittsystemen Verwendung anderer Authentifizierungsmerkmale wie AD. Vorausgesetzt es wird trotz der kürzlichen Ereignisse ausreichende Robustheit angenommen *<1, 2, 3, 6 (gilt nur für den Schutz personenbezogener Daten), 5 (nur, wenn Lösung einfach zu nutzen ist), 9 (verschlüsselte Dateien werden in Autopsy in einer separaten Ansicht gekapselt)>*.

- Neue Versionen werden vor wesentlichen Änderungen getestet. Nutzung eines Versionsverwaltungstools, Durchführung von Code-Reviews, Trennung von Zuständigkeiten wie Entwickler und Tester *<1 (schützt nicht vor nachträglichen Manipulationen in der Produktivumgebung), 3 (Code-Review dient als Kontrolle), 4 (zusätzlich manuelle Dokumentation der Aktivitäten notwendig. Nur bei Versionsverwaltung automatisiert möglich), 5 (Entwicklungstätigkeiten können während dem Code-Review überwacht und unterstützt werden), 6 (geregelter Softwareentwicklungsprozess)>*.
- Analyse von Auswertungs- und Überwachungsmöglichkeiten von IRM und etwaigen Alternativen wie die des Betriebssystems (Windows-Events oder Konfigurationsmöglichkeiten über die Registry). Zum Schutz vor Datenverlust sind die Datensicherungsfunktionalitäten des Betriebssystems, von RAID-Systemen oder sonstige Drittanbietersoftware zu analysieren *<1, 2, 3, 4, 5, 6, 7, 9 (nicht auf Autopsy bezogen, sondern auf das Auffinden im Allgemeinen und auffälligen Nutzeraktivitäten wie der Start von unerlaubten Dateiformaten>*.
- Unternehmensweites Angebot regelmäßiger Austauschtermine zur Vernetzung und zum Wissenstransfer für IT-Experten und Nicht-IT-Fachleute zu Themen wie sichere Softwareentwicklung, Clean Code, neue Technologien oder sonstigen Best Practices. Ein firmeninternes *Stack Overflow* oder eine ähnliche Plattform für den Austausch zwischen Anwendungsentwickelnden ist denkbar *<3, 5, 6, 8 (übergreifend und nicht auf Access bezogen), 9 (nicht auf Autopsy bezogen, sondern auf die Sensibilisierung der Mitarbeitenden>*.
- Ein Access-Backend oder eine um eigene Logik erweiterte Access- oder andere Office-Anwendung sollte nur in begründeten Ausnahmefällen eingesetzt werden *<wenn keine kritischen Daten in Access gespeichert werden: 1-4, 6-9>*.
- Unterstützung für Fachbereiche durch eine zentrale Einheit in Kooperation mit der Anwendungsentwicklung der zentralen IT-Abteilung im Rahmen der beschriebenen Prozesskontrollmaßnahmen (MaRisk fordert regelmäßige Prüfungen der IT-Systeme und der zugehörigen Prozesse auf Eignung). Der Prozess ist an den Datenschutzmanagement-Prozess beziehungsweise den PDCA-Zyklus angelehnt. Ziele der Kontrollmaßnahmen sind die Identifizierung von Risiken, wie Schwachstellen aufgrund falscher Technologiewahl (darunter Access für kritische Zwecke), Datenschutzverletzungen, Optimierungs- und Industrialisierungspotenzial sowie die Reduzierung von technischen Schulden. Darüber hinaus wird empfohlen alle relevanten IDV-Anwendungen auf Basis ungeeigneter Technologien mit der Zeit in eine zentral von der IT-Abteilung kontrollierbare Umgebung (einschließlich granulare Autorisierungsmöglichkeiten), wie Power Apps oder eine vergleichbare moderne Alternative, zu migrieren.

Zielbild der zu kontrollierenden Prozesse ist die Ausrichtung nach dem Straight Through Processing-Ansatz (durchgehende Datenverarbeitung) und der Vision Banking 4.0. Diese Maßnahme wird gleichzeitig auch als Alternative zu einem Scan aller Speichermedien nach potenziellen Access-Dateien oder anderen mit Logik erweiterten Office-Programmen empfohlen. Ein Scan ist nur dann zu empfehlen, wenn die Ergebnismenge vollständig automatisiert, belastbar und auf das Wesentliche reduziert werden kann. Anders als bei der Prozessanalyse wird mit mehr False-Positive-Treffern gerechnet, es gehen Informationen zum Anwendungsumfeld verloren, was die Identifikation von Optimierungsmöglichkeiten erschwert. *<1-9 (nicht auf Autopsy bezogen, sondern auf das Auffinden und Vermeiden im Allgemeinen)>*.

- Suchen nach einer Alternative zu Access oder vergleichbaren Programmen aufgrund der in dieser Arbeit identifizierten Schwachstellen und technischen Schulden. Die in dieser Arbeit genannte Alternativen sind VSTO und Office-Skripte, Low-Code-Plattformen, IRM, Client-Server-DBSs, die Nutzung von sicheren und datenschutzkonformen Platform-as-a-Service Cloud-Lösungen. Es gilt eine möglichst standardisierbare Lösung zu finden, die auf möglichst allen Ebenen vollständig automatisierbar ist und die IT-Komplexität bei der Anwendungsentwicklung auf Seiten der Fachbereiche reduziert (siehe auch TOM zu Musterprogramm und -konzept). Ziel sollte sein, technische Schulden zu verringern und einen schrittweisen Übergang zum Banking 4.0-Ansatz zu ermöglichen (inklusive Ausrichtung der IT-Strategie des Unternehmens). Suchen nach einer Alternative zu MS, sollten sich die Umstände nicht bessern *<1-9 (nicht auf Autopsy bezogen, sondern auf die Vermeidung von Access-Anwendungen aufgrund Anbietens von Alternativen)>*.
- Umsetzung der in dieser Arbeit mit Autopsy identifizierten Verbesserungsvorschläge. Vergleich von Autopsy mit weiteren forensischen Werkzeugen *<4 (wenn zur Suche nach Access-Dateien verwendet), 6 (siehe vorheriger Kommentar), 8 (siehe vorheriger Kommentar und, wenn gefundene Missstände nachhaltig behoben werden), 9>*.

Aus Sicht des Verfassers empfiehlt es sich Access und andere um eigene Logik erweiterte Office-Anwendungen nur im Rahmen des Prototypings oder für unkritische Anwendungsfälle, Geschäftslogik und Daten einzusetzen (beispielsweise Arbeitserleichterungen). Von den Anwendungsfällen darf nur ein vertretbares Risiko ausgehen und im Schadensfall dürfen nur geringe Auswirkungen zu befürchten sein. Derartige Programme sollten daher nicht oder nur in begründeten Ausnahmefällen für die Ablage und Verarbeitung von Risiko-, Modell- und personenbezogene Daten verwendet werden (für diese Datenkategorien gelten verschärfte Anforderungen). Sie sind auch nicht für die Ablage oder Verarbeitung sonstiger vertraulicher beziehungsweise (geschäfts)kritischer Daten oder Geschäftslogik geeignet.

Access kann lediglich mit der einfachen und unkomplizierten Handhabung punkten, wobei dieser Vorteil durch den Database-as-a-Serivce-Ansatz der Azure-Cloud im Verhältnis zum On-Premises-Modell geschmälert wird. Dennoch ist Access, sofern keine Low-Code-Plattformen genutzt werden, in derartigen Anwendungsfällen, in denen die einfache Handhabung im Vordergrund steht, und die Vertraulichkeit, Verfügbarkeit,

Integrität, Authentizität von Logik und Daten sowie die Sicherheit, das Need-to-Know-Prinzip und Least-Privilege-Prinzip oder eine Rollentrennung von untergeordneter Bedeutung sind, eine praktikable Alternative zu „professionellen“ DBSs wie der Azure SQL-DB. In einer nicht vertrauenswürdigen Umgebung sollten keine um VBA-Code erweiterten Office-Anwendungen eingesetzt werden. Der Begriff „kritisch“ ist dabei so präzise und eindeutig wie möglich zu definieren, um ihn nicht zur Interpretationssache zu machen. So können klare Vorgaben für die Entwicklung mit Office-Programmen erstellt werden und gleichzeitig wird eine Grundlage für die Kontrolle mit technischer KI-Unterstützung geschaffen. Bei der Wahl von Office-Produkten ist beispielsweise zu hinterfragen, wie groß die Auswirkungen oder der Schaden sind, wenn die Geschäftslogik oder die Daten veröffentlicht oder unbemerkt verändert werden. Je nach Anwendungsfall, der Kritikalität der Daten sowie des Prozesses, in dem die Anwendung eingesetzt wird, ist abzuwägen, ob Access, Excel oder die Azure SQL-DB aufgrund der aufgezeigten Schwachstellen eine geeignete Technologiewahl darstellen oder, ob die daraus resultierenden Risiken gegen einen Einsatz sprechen.

Es wird daher empfohlen die Fachbereiche so gut es geht zu unterstützen und eine möglichst standardisierte, flexible und zukunftsfähige IT-Umgebung anzubieten, die möglichst wenig negative Folgen aufgrund einer unpassenden oder altmodischen technischen Umsetzung hat (Vermeidung von technischen Schulden oder Schwachstellen). Die Ausarbeitung hat gezeigt, dass die Verfolgung von Banking 4.0 durchaus einen positiven Einfluss auf die IT-Sicherheit haben kann. Für ein natives Frontend sollte zumindest eine Technologie gewählt werden, bei der vertrauliche Zeichenketten, wie Passwörter zur Entschlüsselung eines Access-Backends, nicht im Klartext eingesehen werden können. Im Rahmen der Zusammenarbeit zwischen IT und anderen Fachbereichen wird empfohlen, Prozesse und relevante Systeme regelmäßig auf Optimierungs- und Industrialisierungsmöglichkeiten, Schwachstellen und vermeidbare technische Schulden zu überprüfen.

Literaturverzeichnis

[1] Ahmad Osama & Shashikant Shakya: Professional Azure SQL Managed Database Administration. 3., Auflage, Packt Publishing Ltd., Birmingham 2021

[2] Autopsy (o. V.): Autopsy – Digital Forensics. O. J., URL: https://www.autopsy.com/, letzter Zugriff 23.08.2023

[3] Baeldung (o. V.): File Access: Sequential vs. Direct vs. Indexed. 2023, URL: https://www.baeldung.com/cs/file-access, letzter Zugriff 23.08.2023

[4] bobcares (Nicky Mathew): View and Close Open Files in Windows Server SMB Share. 2020, URL: https://bobcares.com/blog/open-files-in-windows-server/, letzter Zugriff 23.08.2023

[5] Bundesamt für Sicherheit in der Informationstechnik (o. V.): BSI veröffentlicht Empfehlungen zur sicheren Konfiguration von Microsoft-Office-Produkten (Microsoft Access 2013/2016/2019). 2019, URL: https://www.bsi.bund.de/DE/Service-Navi/Presse/Pressemitteilungen/Presse2019/Empfehlungen_Microsoft_190619.html, letzter Zugriff 23.08.2023

[6] Bundesamt für Sicherheit in der Informationstechnik (o. V.): BSI will Sicherheit von Open-Source-Software erhöhen. 2023, URL: https://www.bsi.bund.de/DE/Service-Navi/Presse/Pressemitteilungen/Presse2023/230801_Projekt_CAOS.html, letzter Zugriff 23.08.2023

[7] Bundesamt für Sicherheit in der Informationstechnik (o. V.): Evaluierung der Telemetrie von Microsoft Office 365. O. J., URL: https://www.bsi.bund.de/DE/Service-Navi/Publikationen/Studien/Office_Telemetrie/telemetrie.html, letzter Zugriff 23.08.2023

[8] Bundesamt für Sicherheit in der Informationstechnik (o. V.): IT-Grundschutz-Kompendium – IT-Grundschutz-Bausteine (Edition 2023). 2023, URL: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/IT-Grundschutz/IT-Grundschutz-Kompendium/IT-Grundschutz-Bausteine/Bausteine_Download_Edition_node.html, letzter Zugriff 23.08.2023

[9] Bundesamt für Sicherheit in der Informationstechnik (o. V.): IT-Grundschutz-Kompendium (Edition 2023). 2023, URL: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/IT-Grundschutz/IT-Grundschutz-Kompendium/it-grundschutz-kompendium_node.html, letzter Zugriff 23.08.2023

[10] Bundesamt für Sicherheit in der Informationstechnik (o. V.): Lerneinheit 2.1: Der Sicherheitsprozess. O. J., URL: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/IT-Grundschutz/Zertifizierte-Informationssicherheit/IT-Grundschutzschulung/Online-Kurs-IT-Grundschutz/Lektion_2_Sicherheitsmanagement/Lektion_2_01/Lektion_2_01_node.html, letzter Zugriff 23.08.2023

[11] Bundesamt für Sicherheit in der Informationstechnik (o. V.): Lerneinheit 4.1: Grundlegende Definitionen. O. J., URL: https://www.bsi.bund.de/DE/Themen/Unternehmen-und-Organisationen/Standards-und-Zertifizierung/IT-Grundschutz/Zertifizierte-Informationssicherheit/IT-Grundschutzschulung/Online-Kurs-IT-Grundschutz/Lektion_4_Schutzbedarfsfeststellung/Lektion_4_01/Lektion_4_01_node.html, letzter Zugriff 23.08.2023

[12] Bundesamt für Sicherheit in der Informationstechnik (o. V.): Mehrere Schwachstellen in MS Exchange. 2021, URL: https://www.bsi.bund.de/SharedDocs/Cybersicherheitswarnungen/DE/2021/2021-197772-1132.pdf, letzter Zugriff 23.08.2023

[13] Bundesamt für Sicherheit in der Informationstechnik (o. V.): Migration zu Post-Quanten-Kryptografie – Handlungsempfehlungen des BSI. 2020, URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Krypto/Post-Quanten-Kryptografie.pdf?__blob=publicationFile&v=1, letzter Zugriff 23.08.2023

[14] Bundesamt für Sicherheit in der Informationstechnik (o. V.): Richtlinie zur Risikoanalyse. 2019, URL:

https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Hilfsmittel/Recplast/A02_Richtlinie_Risikoanalyse.pdf?__blob=publicationFile&v=1, letzter Zugriff 23.08.2023

[15] Bundesamt für Sicherheit in der Informationstechnik (o. V.): Sichere Passwörter erstellen. O. J., URL: https://www.bsi.bund.de/DE/Themen/Verbraucherinnen-und-Verbraucher/Informationen-und-Empfehlungen/Cyber-Sicherheitsempfehlungen/Accountschutz/Sichere-Passwoerter-erstellen/sichere-passwoerter-erstellen_node.html, letzter Zugriff 23.08.2023

[16] Bundesamt für Sicherheit in der Informationstechnik (o. V.): Was sind Kritische Infrastrukturen?. O. J., URL: https://www.bsi.bund.de/DE/Themen/KRITIS-und-regulierte-Unternehmen/Kritische-Infrastrukturen/Allgemeine-Infos-zu-KRITIS/allgemeine-infos-zu-kritis.html, letzter Zugriff 23.08.2023

[17] Bundesamt für Sicherheit in der Informationstechnik (o. V.): Zuordnungstabelle ISO zum IT-Grundschutz. 2021, URL: https://www.bsi.bund.de/SharedDocs/Downloads/DE/BSI/Grundschutz/Kompendium/Zuordnung_ISO_und_IT_Grundschutz.html, letzter Zugriff 23.08.2023

[18] Bundesanstalt für Finanzdienstleistungsaufsicht (o. V.): Bankenaufsicht. 2019, URL: https://www.bafin.de/DE/DieBaFin/AufgabenGeschichte/Bankenaufsicht/bankenaufsicht_node.html, letzter Zugriff 23.08.2023

[19] Bundesanstalt für Finanzdienstleistungsaufsicht (o. V.): IT-Sicherheit: Aufsicht konkretisiert Anforderungen an die Kreditwirtschaft. 2018, URL: https://www.bafin.de/SharedDocs/Veroeffentlichungen/DE/Fachartikel/2018/fa_bj_1801_BAIT.html, letzter Zugriff 23.08.2023

[20] Bundesbank (o. V.): Aufgaben. 2020, URL: https://www.bundesbank.de/de/aufgaben/zentralbank-der-bundesrepublik-deutschland-597738, letzter Zugriff 23.08.2023

[21] Bundesbank (o. V.): Bankaufsichtliche Anforderungen an die IT (Rundschreiben 10/2017 (BA)). 2021, URL: https://www.bundesbank.de/de/aufgaben/bankenaufsicht/einzelaspekte/risikomanagement/bait/bankaufsichtliche-anforderungen-an-die-it-598580, letzter Zugriff 23.08.2023

[22] Bundesbank (o. V.): Mindestanforderungen an das Risikomanagement – MaRisk (Rundschreiben 05/2023 (BA)). 2023, URL: https://www.bundesbank.de/de/aufgaben/bankenaufsicht/einzelaspekte/risikomanagement/marisk/mindestanforderungen-an-das-risikomanagement-799520, letzter Zugriff 23.08.2023

[23] Bundesbank (o. V.): Operationelles Risiko. o. J., URL: https://www.bundesbank.de/de/aufgaben/bankenaufsicht/einzelaspekte/eigenmittelanforderungen/operationelles-risiko/operationelles-risiko-598534, letzter Zugriff 23.08.2023

[24] Capterra (o. V.): Microsoft Access. o. J., URL: https://www.capterra.com/p/233275/Microsoft-Access/, letzter Zugriff 23.08.2023

[25] Christian Baun: Computer Networks/Computernetze – Bilingual Edition: English – German/
Zweisprachige Ausgabe: Englisch – Deutsch. 1. Auflage, Springer Vieweg, Wiesbaden 2019

[26] Clean Code Developer (o. V.): Open Closed Principle (OCP). O. J., URL: https://clean-code-developer.de/die-grade/gruener-grad/#Open_Closed_Principle_OCP, letzter Zugriff 23.08.2023

[27] Clean Code Developer (o. V.): Roter Grad, Boy Scout Rule. O. J., URL: https://clean-code-developer.de/die-grade/roter-grad/#Boy_Scout_Rule, letzter Zugriff 23.08.2023

[28] Codekabinett (Philipp Stiefel): Authentication Mechanisms for Access + SQL-Server-Applications. 2019, URL: https://codekabinett.com/rdumps.php?Lang=2&targetDoc=sql-server-authentication-access-application, letzter Zugriff 23.08.2023

[29] Codekabinett (Philipp Stiefel): Transactions and how to use them in Microsoft Access. 2015, URL: https://codekabinett.com/rdumps.php?Lang=2&targetDoc=how-to-access-transaction, letzter Zugriff 23.08.2023

[30] Com! (Klaus Hauptfleisch): Zentrale versus dezentrale IT-Infrastruktur. 2015, URL: https://www.com-magazin.de/praxis/business-it/zentrale-versus-dezentrale-it-infrastruktur-968695.html?page=1_pro---contra-dezentrale-oder-zentrale-it-strukturen, letzter Zugriff 23.08.2023

[31] ComputerWeekly (Adam Hughes & Rich Castagna): Flat File. 2022, URL: https://www.computerweekly.com/de/definition/Flat-File, letzter Zugriff 23.08.2023
[32] CVE (o. V.): CVE-2021-26855. 2021, URL: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2021-26855, letzter Zugriff 23.08.2023
[33] Dark Reading (Ericka Chickowski): Flat-File Databases Often Overlooked In Security Schemes. 2010, URL: https://www.darkreading.com/risk/flat-file-databases-often-overlooked-in-security-schemes, letzter Zugriff 23.08.2023
[34] DB-Engines (o. V.): DB-Engines Ranking of Relational DBMS. 2023, URL: https://db-engines.com/en/ranking/relational+dbms, letzter Zugriff 23.08.2023
[35] DB-Engines (o. V.): DB-Engines Ranking. 2023, URL: https://db-engines.com/en/ranking, letzter Zugriff 23.08.2023
[36] DeftPDF (o. V.): What happens when you open a file?. 2021, URL: https://deftpdf.com/de/blog/what-happens-when-you-open-a-file, letzter Zugriff 23.08.2023
[37] Departmend of Homeland Security (o. V.): Department of Homeland Security's Cyber Safety Review Board to Conduct Review on Cloud Security. 2023, URL: https://www.dhs.gov/news/2023/08/11/department-homeland-securitys-cyber-safety-review-board-conduct-review-cloud, letzter Zugriff 23.08.2023
[38] Der Landesbeauftragte für Datenschutz und Informationsfreiheit Mecklenburg-Vorpommern (o. V.): Standard-Datenschutzmodell (Version 3.0). O. J., URL: https://www.datenschutz-mv.de/datenschutz/datenschutzmodell/, letzter Zugriff 23.08.2023
[39] Dirk Labudde & Michael Spranger (Hrsg.): Forensik in der digitalen Welt – Moderne Methoden der forensischen Fallarbeit in der digitalen und digitalisierten realen Welt. 1. Auflage, Springer Vieweg, Wiesbaden 2020
[40] Easy-Software (o. V.): Was ist Self-Service? Bedeutung, Vorteile, Technologien: der Goldstandard einfach erklärt. O. J., URL: https://easy-software.com/de/newsroom/warum-self-service-neue-chancen-bei-der-interaktion-mit-geschaeftspartnern-und-mitarbeitern-eroeffnet/, letzter Zugriff 23.08.2023
[41] Gabler Wirtschaftslexikon (Jochen Metzger): Straight-through Processing (STP). 2018, URL: https://wirtschaftslexikon.gabler.de/definition/stp-44650/version-267956, letzter Zugriff 23.08.2023
[42] Gartner (o. V.): 2021-2023 Emerging Technology Roadmap for Large Enterprises. 2021, URL: https://www.gartner.com/en/publications/emerging-technology-roadmap-for-large-enterprises, letzter Zugriff: 23.08.2023
[43] Gartner (o. V.): Die 10 wichtigsten strategischen Technologie-Trends von Gartner für 2023. 2022, URL: https://www.gartner.de/de/artikel/gartner-top-10-strategische-technologie-trends-2023, letzter Zugriff: 23.08.2023
[44] Gartner (o. V.): What is New in the 2022 Gartner Hype Cycle for Emerging Technologies. 2022, URL: https://www.gartner.co.uk/en/articles/what-s-new-in-the-2022-gartner-hype-cycle-for-emerging-technologies, letzter Zugriff: 23.08.2023
[45] Gartner (Raj Bala, Dennis Smith, Kevin Ji et al.): Magic Quadrant for Cloud Infrastructure and Platform Services. 2022, URL: https://www.gartner.com/doc/reprints?id=1-29B7RDWN&ct=220304&st=sb, letzter Zugriff: 23.08.2023
[46] GetApp (o. V.): Microsoft Access. O. J., URL: https://www.getapp.com/it-management-software/a/microsoft-access/reviews/, letzter Zugriff 23.08.2023
[47] GetData Forensics (o. V.): About FEX Imager™ (free). O. J., URL: https://getdataforensics.com/product/fex-imager/, letzter Zugriff: 23.08.2023
[48] Guru99 (Richard Peterson): File System vs DBMS – Difference Between Them. 2023, URL: https://www.guru99.com/difference-between-file-system-and-dbms.html, letzter Zugriff 23.08.2023
[49] Handelsblatt (Christof Kerkmann): Unternehmen heben Informations-Schätze – Datenversteher verzweifelt gesucht. 2019, URL: https://www.handelsblatt.com/unternehmen/it-medien/handelsblatt-tagung-it-management-unternehmen-heben-informations-schaetze-datenversteher-verzweifelt-gesucht/23894960.html, letzter Zugriff 23.08.2023
[50] Hartmut Ernst, Jochen Schmidt & Gerd Beneken: Grundkurs Informatik: Grundlagen und Konzepte für die erfolgreiche IT-Praxis – Eine umfassende, praxisorientierte Einführung. 7. erweiterte und aktualisierte Auflage, Springer Vieweg, Wiesbaden 2020

[51] Has Altaiar, Ingrid Babel, Jack Lee, Josh Garverick, Mustafa Toroman & Vahe Minasyan: The Developer's Guide to Azure. 2. Auflage, Microsoft Press, Washington 2022
[52] HashiCorp (o. V.): Data Encryption & Tokenization. O. J., URL: https://www.vaultproject.io/use-cases/data-encryption, letzter Zugriff 23.08.2023
[53] Heise Online (Jürgen Schmidt): Microsofts gestohlener Master-Key: USA stellen Cloud-Security auf den Prüfstand. 2023, URL: https://www.heise.de/news/Microsofts-gestohlener-Master-Key-USA-stellen-Cloud-Security-auf-den-Pruefstand-9244487.html, letzter Zugriff 23.08.2023
[54] Heise Online (Jürgen Schmidt): Microsofts gestohlener Schlüssel mächtiger als vermutet. 2023, URL: https://www.heise.de/news/Neue-Erkenntnisse-Microsofts-Cloud-Luecken-viel-groesser-als-angenommen-9224640.html, letzter Zugriff 23.08.2023
[55] Heise Online (Jürgen Schmidt): Passwörter: BSI verabschiedet sich vom präventiven, regelmäßigen Passwort-Wechsel. 2020, URL: https://www.heise.de/news/Passwoerter-BSI-verabschiedet-sich-vom-praeventiven-Passwort-Wechsel-4652481.html, letzter Zugriff 23.08.2023
[56] Heise Online (Stefan Krempl): Datenschutzkonferenz: Microsoft 365 ist und bleibt datenschutzwidrig. 2022, URL: https://www.heise.de/news/Datenschutzkonferenz-Microsoft-365-ist-und-bleibt-datenschutzwidrig-7352065.html, letzter Zugriff 23.08.2023
[57] Helmut Herold, Bruno Lurz, Jürgen Wohlrab, Matthias Hopf: Grundlagen der Informatik. 3. aktualisierte Auflage. Hallbergmoos: Paerson Deutschland GmbH, 2017
[58] Information Security Forum Ltd (o. V.): Standard of Good Practice for Information Security (SOGP). O. J., URL: https://www.securityforum.org/solutions-and-insights/standard-of-good-practice-for-information-security/, letzter Zugriff 23.08.2023
[59] Infosec (o. V.): Free & open source computer forensics tools. 2019, URL: https://resources.infosecinstitute.com/topics/digital-forensics/free-open-source-computer-forensics-tools/, letzter Zugriff 23.08.2023
[60] International Electrotechnical Commission (o. V.): EC everywhere for a safer and more efficient world. O. J., URL: https://www.iec.ch/homepage, letzter Zugriff 23.08.2023
[61] International Organization for Standardization (o. V.): We're ISO, the International Organization for Standardization. We develop and publish International Standards. O. J., URL: https://www.iso.org/home.html, letzter Zugriff 23.08.2023
[62] Ionos (o. V.): Was ist das EVA-Prinzip?. 2022, URL: https://www.ionos.de/digitalguide/server/knowhow/eva-prinzip/, letzter Zugriff 23.08.2023
[63] Ionos (o. V.): XLSB: Was steckt hinter dem Excel XLSB-Dateiformat?. 2021, URL: https://www.ionos.de/digitalguide/online-marketing/verkaufen-im-internet/xlsb/, letzter Zugriff 23.08.2023
[64] Isladogs on Access (Colin Riddington): Compare Access File Security: MDB/MDE vs ACCDB/ACCDE. 2022, URL: https://isladogs.co.uk/compare-access-file-security/index.html, letzter Zugriff 23.08.2023
[65] ISO – Online Browsing Platform (o. V.): ISO/IEC 27001:2022(en) Information security, cybersecurity and privacy protection – Information security management systems – Requirements. 2022, URL: https://www.iso.org/obp/ui/en/#iso:std:iso-iec:27001:ed-3:v1:en, letzter Zugriff 23.08.2023
[66] ISO – Online Browsing Platform (o. V.): ISO/IEC 27002:2022(en) Information security, cybersecurity and privacy protection – Information security controls. 2022, URL: https://www.iso.org/obp/ui/en/#iso:std:iso-iec:27002:ed-3:v2:en, letzter Zugriff 23.08.2023
[67] IT-Finanzmagazin (Lenildo Morais): BaaS – Banking-as-a-Service: Was kommt auf Banken und Unternehmen zu? Das Dossier. 2020, URL: https://www.it-finanzmagazin.de/baas-banking-as-a-service-dossier-100077/, letzter Zugriff 23.08.2023
[68] IT-Forensik Wiki (o. V.): Forensik-Software – Test von Funktionalitäten auf Dateiidentifikations- & Dateirekonstruktionsebene in NTFS & APFS-Dateisystemen. Projektarbeit im Bachelor-Fernstudiengan „IT-Forensik“, Hochschule Wismar, 2022, URL: https://it-forensik.fiw.hs-wismar.de/index.php/Anonym, letzter Zugriff 23.08.2023

[69] IT-Forensik Wiki (Stefan Augustin & Nils Majewski): SQLInjection Angriffe ausführen und forensisch nachweisen. Projektarbeit im Bachelor-Fernstudiengan „IT-Forensik“, Hochschule Wismar, 2022, URL: https://it-forensik.fiw.hs-wismar.de/images/3/37/DBII-NilsMajewskiStefanAugustin.pdf, letzter Zugriff 23.08.2023
[70] Jürgen Cleve & Uwe Lämmel: Data Mining. 3. Auflage, Walter de Gruyter GmbH, Berlin/Boston 2020
[71] Justin Clarke: SQL Hacking – SQL-Injektion auf relationale Datenbanken im Detail verstehen und abwehren. 1. Auflage, Franzis Verlag GmbH, Haar bei München, 2016
[72] LogicalRead Powered By SolarWinds (Janis Griffin): SQL Server fn_dblog() Function Details and Example. 2012, URL: https://logicalread.com/sql-server-dbcc-log-command-tl01/, letzter Zugriff 23.08.2023
[73] Microsoft (Charlie Bell): Mitigation for China-based threat actor activity. 2023, URL: https://blogs.microsoft.com/on-the-issues/2023/07/11/mitigation-china-based-threat-actor/, letzter Zugriff 23.08.2023
[74] Microsoft (Irene Nadler): Die Geschichte von Windows. 2020, URL https://news.microsoft.com/de-de/features/windows-geschichte/, letzter Zugriff 23.08.2023
[75] Microsoft (MS et al.): "File sharing lock count exceeded…" error during large transaction processing. 2022, URL: https://learn.microsoft.com/en-us/office/troubleshoot/access/file-sharing-lock-count-exceeded, letzter Zugriff 23.08.2023
[76] Microsoft (MS et al.): @@Version: Transact SQL-Konfigurationsfunktionen. 2023, URL: https://learn.microsoft.com/de-de/sql/t-sql/functions/version-transact-sql-configuration-functions?view=sql-server-ver16, letzter Zugriff 23.08.2023
[77] Microsoft (MS et al.): About the Microsoft Dataverse for Teams environment. 2023, URL: https://learn.microsoft.com/en-us/power-platform/admin/about-teams-environment, letzter Zugriff 23.08.2023
[78] Microsoft (MS et al.): ADD SIGNATURE (Transact-SQL). 2023, URL: https://learn.microsoft.com/en-us/sql/t-sql/statements/add-signature-transact-sql?view=sql-server-ver16, letzter Zugriff 23.08.2023
[79] Microsoft (MS et al.): Aktive Georeplikation. 2023, URL: https://learn.microsoft.com/de-de/azure/azure-sql/database/active-geo-replication-overview?view=azuresql-db, letzter Zugriff 23.08.2023
[80] Microsoft (MS et al.): Always Encrypted. 2023, URL: https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/always-encrypted-database-engine?view=sql-server-ver16, letzter Zugriff 23.08.2023
[81] Microsoft (MS et al.): An overview of Azure SQL Database and SQL Managed Instance security capabilities. 2023, URL: https://learn.microsoft.com/en-us/azure/azure-sql/database/security-overview?view=azuresql-mi, letzter Zugriff 23.08.2023
[82] Microsoft (MS et al.): Anfügen einer Datenbank. 2023, URL: https://learn.microsoft.com/de-de/sql/relational-databases/databases/attach-a-database?view=sql-server-ver16, letzter Zugriff 23.08.2023
[83] Microsoft (MS et al.): Announcing the new release of OLE DB Driver for SQL Server. 2017, URL: https://learn.microsoft.com/de-de/archive/blogs/sqlnativeclient/announcing-the-new-release-of-ole-db-driver-for-sql-server, letzter Zugriff 23.08.2023
[84] Microsoft (MS et al.): Anpassen der Standardeinstellungen einer Datenbank. O. J., URL: https://support.microsoft.com/de-de/office/anpassen-der-standardeinstellungen-einer-datenbank-b573d022-cc98-4e64-a0ed-595d179a9cbb, letzter Zugriff 23.08.2023
[85] Microsoft (MS et al.): Anzeigen und Speichern von Ausführungsplänen. 2023, URL: https://learn.microsoft.com/de-de/sql/relational-databases/performance/display-and-save-execution-plans?view=sql-server-ver16, letzter Zugriff 23.08.2023
[86] Microsoft (MS et al.): Auditing for Azure SQL Database and Azure Synapse Analytics. 2023, URL: https://learn.microsoft.com/en-us/azure/azure-sql/database/auditing-overview?view=azuresql-db#next-steps, letzter Zugriff 23.08.2023
[87] Microsoft (MS et al.): Authentication in SQL Server. 2018, URL: https://learn.microsoft.com/en-us/previous-versions/dotnet/framework/data/adonet/sql/authentication-in-sql-server, letzter Zugriff 23.08.2023

[88] Microsoft (MS et al.): Azure SQL-Konnektivitätseinstellungen. 2023, URL: https://learn.microsoft.com/de-de/azure/azure-sql/database/connectivity-settings?view=azuresql&tabs=azure-portal#deny-public-network-access, letzter Zugriff 23.08.2023
[89] Microsoft (MS et al.): Cabinet Files. 2021, URL: https://learn.microsoft.com/en-us/windows/win32/msi/cabinet-files, letzter Zugriff 23.08.2023
[90] Microsoft (MS et al.): Cloud Security Alliance (CSA) STAR Selbsteinschätzung. 2023, URL: https://learn.microsoft.com/de-de/compliance/regulatory/offering-csa-star-self-assessment, letzter Zugriff 23.08.2023
[91] Microsoft (MS et al.): Column-level security. 2022, URL: https://learn.microsoft.com/en-us/azure/synapse-analytics/sql-data-warehouse/column-level-security, letzter Zugriff 23.08.2023
[92] Microsoft (MS et al.): COM Technical Overview. 2021, URL: https://learn.microsoft.com/en-us/windows/win32/com/com-technical-overview, letzter Zugriff 23.08.2023
[93] Microsoft (MS et al.): Common Language Runtime Integration. 2023, URL: https://learn.microsoft.com/en-us/sql/relational-databases/clr-integration/common-language-runtime-integration-overview?view=sql-server-ver16, letzter Zugriff 23.08.2023
[94] Microsoft (MS et al.): Configure and manage Azure AD authentication with Azure SQL. 2023, URL: https://learn.microsoft.com/en-us/azure/azure-sql/database/authentication-aad-configure?view=azuresql&tabs=azure-powershell, letzter Zugriff 23.08.2023
[95] Microsoft (MS et al.): Create a self-signed public certificate to authenticate your application. 2022, URL: https://learn.microsoft.com/en-us/azure/active-directory/develop/howto-create-self-signed-certificate, letzter Zugriff 23.08.2023
[96] Microsoft (MS et al.): CreateParameter-Methode (ADO). 2023, URL: https://learn.microsoft.com/de-de/office/client-developer/access/desktop-database-reference/createparameter-method-ado, letzter Zugriff 23.08.2023
[97] Microsoft (MS et al.): Cryptography and encryption in Office 2016. 2023, URL: https://learn.microsoft.com/en-us/deployoffice/office2016/security/cryptography-encryption, letzter Zugriff 23.08.2023
[98] Microsoft (MS et al.): Database Engine Instances (SQL Server). 2023, URL: https://learn.microsoft.com/en-us/sql/database-engine/configure-windows/database-engine-instances-sql-server?view=sql-server-ver16, letzter Zugriff 23.08.2023
[99] Microsoft (MS et al.): Database.CreateQueryDef-Methode (DAO) . 2023, URL: https://learn.microsoft.com/de-de/office/client-developer/access/desktop-database-reference/database-createquerydef-method-dao, letzter Zugriff 23.08.2023
[100] Microsoft (MS et al.): Datenmakroereignisse. 2023, URL: https://learn.microsoft.com/de-de/office/client-developer/access/desktop-database-reference/data-macro-events?redirectedfrom=MSDN, letzter Zugriff 23.08.2023
[101] Microsoft (MS et al.): Difference between MS Access in 365 and 2016?. 2020, URL: https://answers.microsoft.com/en-us/msoffice/forum/all/difference-between-ms-access-in-365-and-2016/1e177c44-d7f5-4653-9730-7d3370e25003, letzter Zugriff 23.08.2023
[102] Microsoft (MS et al.): DoCmd.CopyDatabaseFile-Methode. 2023, URL https://learn.microsoft.com/de-de/office/vba/api/Access.DoCmd.CopyDatabaseFile, letzter Zugriff 23.08.2023
[103] Microsoft (MS et al.): DoCmd.SetParameter-Methode (Access). 2023, URL: https://learn.microsoft.com/de-de/office/vba/api/access.docmd.setparameter, letzter Zugriff 23.08.2023
[104] Microsoft (MS et al.): Dynamic data masking. 2023, URL: https://learn.microsoft.com/en-us/sql/relational-databases/security/dynamic-data-masking?view=sql-server-ver16#creating-a-dynamic-data-mask, letzter Zugriff 23.08.2023
[105] Microsoft (MS et al.): Einführung in Sperrdateien (laccdb und ldb) in Access, URL: https://learn.microsoft.com/de-de/office/troubleshoot/access/lock-files-introduction. 2023, letzter Zugriff 23.08.2023
[106] Microsoft (MS et al.): Einrichten von Azure App Service-Zugriffseinschränkungen. 2023, URL: https://learn.microsoft.com/de-de/azure/app-service/app-service-ip-restrictions?tabs=azurecli, letzter Zugriff 23.08.2023

[107] Microsoft (MS et al.): Entfernen oder Zurücksetzen von Dateikennwörtern in Office 2016. 2023, URL: https://learn.microsoft.com/de-de/deployoffice/office2016/security/remove-reset-file-passwords?redirected-from=MSDN, letzter Zugriff 23.08.2023
[108] Microsoft (MS et al.): Entwickeln Ihrer Benennungs- und Kennzeichnungsstrategie für Azure-Ressourcen, URL: https://learn.microsoft.com/de-de/azure/cloud-adoption-framework/ready/azure-best-practices/naming-and-tagging. 2023, letzter Zugriff 23.08.2023
[109] Microsoft (MS et al.): Erste Schritte mit VBA in Office. 2023, URL: https://learn.microsoft.com/de-de/office/vba/library-reference/concepts/getting-started-with-vba-in-office#doing-things-another-way, letzter Zugriff 23.08.2023
[110] Microsoft (MS et al.): GRANT (Transact-SQL). 2023, URL: https://learn.microsoft.com/de-de/sql/t-sql/statements/grant-transact-sql?view=sql-server-ver16, letzter Zugriff 23.08.2023
[111] Microsoft (MS et al.): Granting Row-Level Permissions in SQL Server. 2017, URL: https://learn.microsoft.com/en-us/previous-versions/dotnet/framework/data/ado-net/sql/granting-row-level-permissions-in-sql-server, letzter Zugriff 23.08.2023
[112] Microsoft (MS et al.): Hyperscale-Dienstebene. 2023, URL: https://learn.microsoft.com/de-de/azure/azure-sql/database/service-tier-hyperscale?view=azuresql, letzter Zugriff: 23.08.2023
[113] Microsoft (MS et al.): Information Rights Management in Exchange Server. 2023, URL: https://learn.microsoft.com/en-us/exchange/policy-and-compliance/information-rights-management?view=exchserver-2019, letzter Zugriff: 23.08.2023
[114] Microsoft (MS et al.): Install unit test frameworks. 2023, URL: https://learn.microsoft.com/en-us/visualstudio/test/install-third-party-unit-test-frameworks?view=vs-2022, letzter Zugriff: 23.08.2023
[115] Microsoft (MS et al.): ISO/IEC 27001:2013 Information Security Management Standards. 2023, URL: https://learn.microsoft.com/de-de/compliance/regulatory/offering-iso-27001, letzter Zugriff: 23.08.2023
[116] Microsoft (MS et al.): Konfigurieren von Datenaufbewahrungs- und Archivierungsrichtlinien in Azure Monitor-Protokollen. 2023, URL: https://learn.microsoft.com/de-de/azure/azure-monitor/logs/data-retention-archive?tabs=portal-1%2Cportal-2, letzter Zugriff: 23.08.2023
[117] Microsoft (MS et al.): Ledgerübersicht. 2023, URL: https://learn.microsoft.com/de-de/sql/relational-databases/security/ledger/ledger-overview?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[118] Microsoft (MS et al.): Leitfaden zur Architektur der Abfrageverarbeitung. 2023, URL: https://learn.microsoft.com/de-de/sql/relational-databases/query-processing-architecture-guide?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[119] Microsoft (MS et al.): Live Unit Testing overview. 2022, URL: https://learn.microsoft.com/en-us/visualstudio/test/live-unit-testing-intro?view=vs-2022, letzter Zugriff: 23.08.2023
[120] Microsoft (MS et al.): LogEvent-Makroaktion. 2023, URL: https://learn.microsoft.com/de-de/office/client-developer/access/desktop-database-reference/logevent-macro-action?redirectedfrom=MSDN, letzter Zugriff: 23.08.2023
[121] Microsoft (MS et al.): Machen Sie einen Access-Ausflug durch SQL Server. O. J., URL: https://support.microsoft.com/de-de/office/migrieren-einer-access-datenbank-zu-sql-server-7bac0438-498a-4f53-b17b-cc22fc42c979, letzter Zugriff: 23.08.2023
[122] Microsoft (MS et al.): Microsoft Defender für SQL. 2023, URL: https://learn.microsoft.com/de-de/azure/azure-sql/database/azure-defender-for-sql?view=azuresql-mi&WT.mc_id=Portal-Microsoft_Azure_Security, letzter Zugriff 23.08.2023
[123] Microsoft (MS et al.): Microsoft is Aligning with ODBC for Native Relational Data Access. 2011, URL: https://learn.microsoft.com/de-de/archive/blogs/sqlnativeclient/microsoft-is-aligning-with-odbc-for-native-relational-data-access, letzter Zugriff: 23.08.2023
[124] Microsoft (MS et al.): Microsoft Jet Database Engine Programmer's Guide - Introduction. 2010, URL: https://learn.microsoft.com/en-us/previous-versions/cc966378(v=technet.10)?redirectedfrom=MSDN, letzter Zugriff: 23.08.2023

[125] Microsoft (MS et al.): Microsoft OLE DB. 2016, URL: https://learn.microsoft.com/en-us/previous-versions/windows/desktop/ms722784(v=vs.85), letzter Zugriff: 23.08.2023
[126] Microsoft (MS et al.): Microsoft OLE DB-Treiber für SQL Server. 2023, URL: https://learn.microsoft.com/de-de/sql/connect/oledb/oledb-driver-for-sql-server?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[127] Microsoft (MS et al.): Microsoft Open Database Connectivity (ODBC). 2023, URL: https://learn.microsoft.com/en-us/sql/odbc/microsoft-open-database-connectivity-odbc?view=sql-server-ver16, letzter Zugriff 23.08.2023
[128] Microsoft (MS et al.): Microsoft SQL documentation – Learn how to use SQL Server and Azure SQL, both on-premises and in the cloud. O. J., URL: https://learn.microsoft.com/en-us/sql/?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[129] Microsoft (MS et al.): Monitor Azure resources with Azure Monitor. 2023, URL: https://learn.microsoft.com/en-us/azure/azure-monitor/essentials/monitor-azure-resource, letzter Zugriff 23.08.2023
[130] Microsoft (MS et al.): ODBC and the Standard CLI. 2023, URL: https://learn.microsoft.com/en-us/sql/odbc/reference/odbc-and-the-standard-cli?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[131] Microsoft (MS et al.): OLE DB Glossary. 2017, URL: https://learn.microsoft.com/en-us/previous-versions/windows/desktop/ms713672(v=vs.85), letzter Zugriff 23.08.2023
[132] Microsoft (MS et al.): Overview of OLE DB. 2017, URL: https://learn.microsoft.com/en-us/previous-versions/windows/desktop/ms718124(v=vs.85), letzter Zugriff 23.08.2023
[133] Microsoft (MS et al.): Password must meet complexity requirements. 2023, URL: https://learn.microsoft.com/en-us/windows/security/threat-protection/security-policy-settings/password-must-meet-complexity-requirements, letzter Zugriff 23.08.2023
[134] Microsoft (MS et al.): Password policies and account restrictions in Azure Active Directory. 2023, URL: https://learn.microsoft.com/en-us/azure/active-directory/authentication/concept-sspr-policy, letzter Zugriff 23.08.2023
[135] Microsoft (MS et al.): Password policy recommendations for Microsoft 365 passwords. 2023, URL: https://learn.microsoft.com/en-us/microsoft-365/admin/misc/password-policy-recommendations?view=o365-worldwide, letzter Zugriff 23.08.2023
[136] Microsoft (MS et al.): Plattformübersicht für Office-Add-Ins. 2023,, URL: https://learn.microsoft.com/de-de/office/dev/add-ins/overview/office-add-ins, letzter Zugriff 23.08.2023
[137] Microsoft (MS et al.): Prevent Cross-Site Scripting (XSS) in ASP.NET Core. 2023,, URL: https://learn.microsoft.com/en-us/aspnet/core/security/cross-site-scripting?view=aspnetcore-7.0, letzter Zugriff 23.08.2023
[138] Microsoft (MS et al.): Protect an Excel file. O. J., URL: https://support.microsoft.com/en-us/office/protect-an-excel-file-7359d4ae-7213-4ac2-b058-f75e9311b599, letzter Zugriff 23.08.2023
[139] Microsoft (MS et al.): Protection and security in Excel. O. J., URL: https://support.microsoft.com/en-us/office/protection-and-security-in-excel-be0b34db-8cb6-44dd-a673-0b3e3475ac2d?ui=en-us&rs=en-us&ad=us, letzter Zugriff 23.08.2023
[140] Microsoft (MS et al.): Publish a custom app by uploading an app package. 2023, URL: https://learn.microsoft.com/en-us/microsoftteams/upload-custom-apps, letzter Zugriff 23.08.2023
[141] Microsoft (MS et al.): Query Performance Insight for Azure SQL Database. 2023, URL: https://learn.microsoft.com/en-us/azure/azure-sql/database/query-performance-insight-use?view=azuresql-db, letzter Zugriff 23.08.2023
[142] Microsoft (MS et al.): Query Plan Viewer in Azure Data Studio. 2023, URL: https://learn.microsoft.com/de-de/sql/azure-data-studio/query-plan-viewer?view=sql-server-ver16, letzter Zugriff 23.08.2023
[143] Microsoft (MS et al.): QueryDef-Objekt (DAO). 2023, URL: https://learn.microsoft.com/de-de/office/client-developer/access/desktop-database-reference/querydef-object-dao, letzter Zugriff 23.08.2023
[144] Microsoft (MS et al.): RaiseError-Makroaktion. 2023, URL: https://learn.microsoft.com/de-de/office/client-developer/access/desktop-database-reference/raiseerror-macro-action, letzter Zugriff: 23.08.2023

[145] Microsoft (MS et al.): Row-Level Security. 2023, URL: https://learn.microsoft.com/en-us/sql/relational-databases/security/row-level-security?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[146] Microsoft (MS et al.): Safe storage of app secrets in development in ASP.NET Core. 2023, URL: https://learn.microsoft.com/en-us/aspnet/core/security/app-secrets?view=aspnetcore-7.0&tabs=windows, letzter Zugriff: 23.08.2023
[147] Microsoft (MS et al.): SendEmail-Makroaktion. 2023, URL: https://learn.microsoft.com/de-de/office/client-developer/access/desktop-database-reference/sendemail-macro-action, letzter Zugriff: 23.08.2023
[148] Microsoft (MS et al.): Sperren und Entsperren von Bytebereichen in Dateien, both on-premises and in the cloud. 2023, URL: https://learn.microsoft.com/de-de/windows/win32/fileio/locking-and-unlocking-byte-ranges-in-files, letzter Zugriff: 23.08.2023
[149] Microsoft (MS et al.): SQL Advanced Threat Protection. 2023, URL: https://learn.microsoft.com/en-us/azure/azure-sql/database/threat-detection-overview?view=azuresql, letzter Zugriff: 23.08.20233
[150] Microsoft (MS et al.): SQL Server technical documentation – Technical documentation to help you get started, administer, develop, and work with SQL Server and associated products. O. J., URL: https://learn.microsoft.com/en-us/sql/?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[151] Microsoft (MS et al.): SQL Server Transaktionsprotokollarchitektur und Verwaltungsleitfaden. 2023, URL: https://learn.microsoft.com/de-de/sql/relational-databases/memory-management-architecture-guide?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[152] Microsoft (MS et al.): sys.database_principals (Transact-SQL). 2023, URL: https://learn.microsoft.com/en-us/sql/relational-databases/system-catalog-views/sys-database-principals-transact-sql?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[153] Microsoft (MS et al.): sys.dm_exec_cached_plans (Transact-SQL). 2023, URL: https://learn.microsoft.com/de-de/sql/relational-databases/system-dynamic-management-views/sys-dm-exec-cached-plans-transact-sql?view=sql-server-ver16, letzter Zugriff: 23.08.2023
[154] Microsoft (MS et al.): sys.dm_exec_query_plan (Transact-SQL). 2023, URL: https://learn.microsoft.com/de-de/sql/relational-databases/system-dynamic-management-views/sys-dm-exec-query-plan-transact-sql?view=azuresqldb-current&viewFallbackFrom=sql-server-ver16, letzter Zugriff: 23.08.2023
[155] Microsoft (MS et al.): Tutorial: Secure a database in Azure SQL Database. 2023, URL: https://learn.microsoft.com/en-us/azure/azure-sql/database/secure-database-tutorial?view=azuresql, letzter Zugriff 23.08.2023
[156] Microsoft (MS et al.): Übersicht über den Ausführungsplan. 2023, URL: https://learn.microsoft.com/de-de/sql/relational-databases/performance/execution-plans?view=sql-server-ver16, letzter Zugriff 23.08.2023
[157] Microsoft (MS et al.): Übersicht über die Entwicklung von Office-Lösungen (VSTO). 2023, URL: https://learn.microsoft.com/de-de/visualstudio/vsto/office-solutions-development-overview-vsto?view=vs-2022, letzter Zugriff 23.08.2023
[158] Microsoft (MS et al.): Übersicht über NTFS. 2023, URL: https://learn.microsoft.com/de-de/windows-server/storage/file-server/ntfs-overview, letzter Zugriff 23.08.2023
[159] Microsoft (MS et al.): VBA- und Office-Lösungen in Visual Studio im Vergleich. 2023, URL: https://learn.microsoft.com/de-de/visualstudio/vsto/vba-and-office-solutions-in-visual-studio-compared?view=vs-2022, letzter Zugriff 23.08.2023
[160] Microsoft (MS et al.): VBA-Referenz für Office. 2023, URL: https://learn.microsoft.com/de-de/office/vba/api/overview/, letzter Zugriff 23.08.2023
[161] Microsoft (MS et al.): Verbindungsfehler bei Verwendung von CurrentProject.Connection oder CurrentDB.Connection in Access. 2023, URL: https://learn.microsoft.com/de-de/office/troubleshoot/access/database-connection-error, letzter Zugriff 23.08.2023
[162] Microsoft (MS et al.): Verbindungshandles. 2023, URL: https://learn.microsoft.com/de-de/sql/odbc/reference/develop-app/connection-handles?view=sql-server-ver16, letzter Zugriff 23.08.2023

[163] Microsoft (MS et al.): Verwenden der Azure Active Directory-Authentifizierung. 2023, URL: https://learn.microsoft.com/de-de/azure/azure-sql/database/authentication-aad-overview?view=azuresql, letzter Zugriff 23.08.2023
[164] Microsoft (MS et al.): Was ist „verwalteter Code“?. 2023, URL: https://learn.microsoft.com/de-de/dotnet/standard/managed-code, letzter Zugriff 23.08.2023
[165] Microsoft (MS et al.): Was ist Microsoft Defender für Cloud?. 2023, URL: https://learn.microsoft.com/de-de/azure/defender-for-cloud/defender-for-cloud-introduction, letzter Zugriff 23.08.2023
[166] Microsoft (MS et al.): What About the Microsoft JET Engine?. 2014, URL: https://learn.microsoft.com/en-us/previous-versions/office/developer/office-2007/cc811599(v=office.12)?redirectedfrom=MSDN#what-about-the-microsoft-jet-engine, letzter Zugriff 23.08.2023
[167] Microsoft (MS et al.): What are Azure Monitor alerts?. 2023, URL: https://learn.microsoft.com/en-us/azure/azure-monitor/alerts/alerts-overview, letzter Zugriff 23.08.2023
[168] Microsoft (MS et al.): What is change data capture (CDC). 2023, URL: https://learn.microsoft.com/en-us/sql/relational-databases/track-changes/about-change-data-capture-sql-server?view=sql-server-ver16&source=recommendations, letzter Zugriff 23.08.2023
[169] Microsoft (MS et al.): What is the database ledger?. 2023, URL: https://learn.microsoft.com/en-us/sql/relational-databases/security/ledger/ledger-database-ledger?view=sql-server-ver16, letzter Zugriff 23.08.2023
[170] Microsoft (MS et al.): Windows-Ereignisprotokoll. 2023, URL: https://learn.microsoft.com/de-de/windows/win32/wes/windows-event-log, letzter Zugriff 01.04.2023
[171] Microsoft (MS et al.): Workspace.BeginTrans-Methode (DAO). 2023, URL: https://learn.microsoft.com/de-de/office/client-developer/access/desktop-database-reference/workspace-begintrans-method-dao, letzter Zugriff 23.08.2023
[172] Microsoft (o. V): Einschränken des Zugriffs auf Arbeitsmappen mit Information Rights Management in Excel. O. J., URL: https://support.microsoft.com/de-de/office/einschr%C3%A4nken-des-zugriffs-auf-arbeitsmappen-mit-information-rights-management-in-excel-3525d8fd-4313-4645-b60e-5ec0e1b9c317, letzter Zugriff 23.08.2023
[173] Microsoft (o. V.): Access specifications. O. J., URL: https://support.microsoft.com/en-us/office/access-specifications-0cf3c66f-9cf2-4e32-9568-98c1025bb47c, letzter Zugriff: 23.08.2023
[174] Microsoft (o. V.): Adopting a Zero Trust approach is a technology and business imperative. 2021, URL: https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RW10GBb?culture=en-us&country=us, letzter Zugriff 23.08.2023
[175] Microsoft (o. V.): Automate startup events with a macro. O. J., URL: https://support.microsoft.com/en-us/office/automate-startup-events-with-a-macro-b08d4f22-b517-4ccf-bd14-8670416628b0?ui=en-us&rs=en-us&ad=us#__toc325350998, letzter Zugriff 23.08.2023
[176] Microsoft (o. V.): Azure Monitor overview. 2023, URL: https://learn.microsoft.com/en-us/azure/azure-monitor/overview#high-level-architecture, letzter Zugriff 23.08.2023
[177] Microsoft (o. V.): Bereitstellen einer Access-Anwendung. O. J., URL: https://support.microsoft.com/de-de/office/bereitstellen-einer-access-anwendung-7bb4f2ba-30ee-458c-a673-102dc34bf14f, letzter Zugriff 23.08.2023
[178] Microsoft (o. V.): Calling a stored procedure with a command. O. J., URL: https://support.microsoft.com/en-us/office/create-procedure-statement-91c700d1-8076-4040-896a-a0b7cf9d9888, letzter Zugriff 23.08.2023
[179] Microsoft (o. V.): Chapter 1: Understanding Microsoft Access 2000 Client/Server Development. 2014, URL: https://learn.microsoft.com/de-de/previous-versions/office/developer/office2000/aa139930(v=office.10), letzter Zugriff 23.08.2023
[180] Microsoft (o. V.): Codearme und codefreie App-Entwicklung im Vergleich. O. J., URL: https://powerapps.microsoft.com/de-de/low-code-no-code-development-platforms/, letzter Zugriff 23.08.2023
[181] Microsoft (o. V.): Create a macro that runs when you open a database. O. J., URL: https://support.microsoft.com/en-us/office/create-a-macro-that-runs-when-you-open-a-

database-98ba1508-dcc6-4e0f-9698-a4755e548124?ui=en-us&rs=en-us&ad=us, letzter Zugriff 23.08.2023
[182] Microsoft (o. V.): CREATE VIEW Statement. O. J., URL: https://support.microsoft.com/en-us/office/create-view-statement-ffcd67a0-047f-448d-a069-24c8f3e165ba?ui=en-us&rs=en-us&ad=us, letzter Zugriff 23.08.2023
[183] Microsoft (o. V.): Diagnosedaten in Microsoft 365. O. J., URL: https://support.microsoft.com/de-de/office/diagnosedaten-in-microsoft-365-f409137d-15d3-4803-a8ae-d26fcbfc91dd?ui=de-de&rs=de-de&ad=de, letzter Zugriff 23.08.2023
[184] Microsoft (o. V.): Die zehn besten Gründe für die Verwendung von Access Excel. O. J., URL: https://support.microsoft.com/de-de/office/die-zehn-besten-gr%C3%BCnde-f%C3%BCr-die-verwendung-von-access-excel-2a454445-13cc-4b39-bc2f-d27fd12ca414, letzter Zugriff 23.08.2023
[185] Microsoft (o. V.): Durch Hinzufügen einer digitalen Signatur Vertrauen ausdrücken. O. J., URL: https://support.microsoft.com/de-de/office/durch-hinzuf%C3%BCgen-einer-digitalen-signatur-vertrauen-ausdr%C3%BCcken-5f4ebff3-360d-4b61-b2f8-ce0dfb53adf6, letzter Zugriff 23.08.2023
[186] Microsoft (o. V.): Einführung in Abfragen. O. J., URL: https://support.microsoft.com/de-de/office/einf%C3%BChrung-in-abfragen-a9739a09-d3ff-4f36-8ac3-5760249fb65c, letzter Zugriff 23.08.2023
[187] Microsoft (o. V.): Einführung in die Access-Programmierung. O. J., URL: https://support.microsoft.com/de-de/office/einf%C3%BChrung-in-die-access-programmierung-92eb616b-3204-4121-9277-70649e33be4f, letzter Zugriff 23.08.2023
[188] Microsoft (o. V.): Entdecken Sie die Neuigkeiten in Microsoft 365. O. J., URL: https://www.microsoft.com/de-DE/Microsoft-365, letzter Zugriff 23.08.2023
[189] Microsoft (o. V.): Erreichen Sie Ihre persönlichen Ziele mit Microsoft 365. O. J., URL: https://www.microsoft.com/de-de/microsoft-365/explore-microsoft-365-for-home, letzter Zugriff 23.08.2023
[190] Microsoft (o. V.): Erste Schritte: Migrieren von Access-Daten zu Dataverse. O. J., URL: https://support.microsoft.com/de-de/office/erste-schritte-migrieren-von-access-daten-zu-dataverse-013c8bab-7737-46ca-ad2e-892bbf26287d, letzter Zugriff 23.08.2023
[191] Microsoft (o. V.): Erstellen eines benutzerdefinierten Menübands in Access. O. J., URL: https://support.microsoft.com/de-de/office/erstellen-eines-benutzerdefinierten-men%C3%BCbands-in-access-45e110b9-531c-46ed-ab3a-4e25bc9413de, letzter Zugriff 23.08.2023
[192] Microsoft (o. V.): Erstellen eines Datenmakros. O. J., URL: https://support.microsoft.com/de-de/office/erstellen-eines-datenmakros-b1b94bca-4f17-47ad-a66d-f296ef834200, letzter Zugriff: 23.08.2023
[193] Microsoft (o. V.): Evolving Zero Trust – How real-world deployments and attacks are shaping the future of Zero Trust strategies. 2021, URL: https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RWJJdT, letzter Zugriff 23.08.2023
[194] Microsoft (o. V.): Herunterladen und Installieren von Microsoft 365 Access Runtime. O. J., URL: https://support.microsoft.com/de-de/office/herunterladen-und-installieren-von-microsoft-365-access-runtime-185c5a32-8ba9-491e-ac76-91cbe3ea09c9, letzter Zugriff: 23.08.2023
[195] Microsoft (o. V.): KB5002984: Configuring Jet Red Database Engine and Access Connectivity Engine to block access to remote databases. O. J., URL: https://support.microsoft.com/en-us/topic/kb5002984-configuring-jet-red-database-engine-and-access-connectivity-engine-to-block-access-to-remote-databases-56406821-30f3-475c-a492-208b9bd30544, letzter Zugriff: 23.08.2023
[196] Microsoft (o. V.): Machen Sie einen Access-Ausflug durch SQL Server. O. J., URL: https://support.microsoft.com/de-de/office/machen-sie-einen-access-ausflug-durch-sql-server-532584f3-f2c6-47e0-9387-361c5873348e, letzter Zugriff 23.08.2023
[197] Microsoft (o. V.): Microsoft 365 – Alles, was Sie brauchen, um in weniger Zeit mehr zu erreiche. 2023, URL: https://www.microsoft.com/de-de/microsoft-365/buy/compare-all-microsoft-365-products?ocid=oo_support_mix_marvel_ups_support_smcofficelifecycle_inline&rtc=1, letzter Zugriff 23.08.2023

[198] Microsoft (o. V.): Microsoft Access Database Engine 2016 Redistributable. O. J., URL: https://www.microsoft.com/en-us/download/details.aspx?id=54920, letzter Zugriff: 23.08.2023
[199] Microsoft (o. V.): Microsoft Office – Office Professional 2021. O. J., URL: https://www.microsoft.com/de-de/microsoft-365/p/office-professional-2021/CFQ7TTC0HHJ9?activetab=pivot:%C3%BCbersichttab, letzter Zugriff 23.08.2023
[200] Microsoft (o. V.): Migrieren einer Access-Datenbank zu SQL Server. O. J., URL: https://support.microsoft.com/de-de/office/migrieren-einer-access-datenbank-zu-sql-server-7bac0438-498a-4f53-b17b-cc22fc42c979, letzter Zugriff: 23.08.2023
[201] Microsoft (o. V.): Möglichkeiten der Freigabe einer Access-Desktopdatenbank. O. J., URL: https://support.microsoft.com/de-de/office/m%C3%B6glichkeiten-der-freigabe-einer-access-desktopdatenbank-03822632-da43-4d8f-ba2a-68da245a0446, letzter Zugriff: 23.08.2023
[202] Microsoft (o. V.): Open XML-Formate und -Dateinamenerweiterungen. O. J., URL: https://support.microsoft.com/de-de/office/open-xml-formate-und-dateinamenerweiterungen-5200d93c-3449-4380-8e11-31ef14555b18, letzter Zugriff 23.08.2023
[203] Microsoft (o. V.): Optimieren von Access. O. J., URL: https://support.microsoft.com/de-de/office/optimieren-von-access-f6827763-bb5c-4f48-8457-7a14addab6be#bm2_2, letzter Zugriff 23.08.2023
[204] Microsoft (o. V.): Perfektionieren Sie Ihre Daten, O. J., URL: https://www.microsoft.com/de-de/microsoft-365/access, letzter Zugriff 23.08.2023
[205] Microsoft (o. V.): Restrict access to workbooks with Information Rights Management in Excel. O. J., URL: https://support.microsoft.com/en-us/office/restrict-access-to-workbooks-with-information-rights-management-in-excel-3525d8fd-4313-4645-b60e-5ec0e1b9c317#ID0EBBD=Windows, letzter Zugriff: 23.08.2023
[206] Microsoft (o. V.): Schützen einer Arbeitsmappe. O. J., URL: https://support.microsoft.com/de-de/office/sch%C3%BCtzen-einer-arbeitsmappe-7e365a4d-3e89-4616-84ca-1931257c1517, letzter Zugriff: 23.08.2023
[207] Microsoft (o. V.): Schützen von Daten mit Sicherungs- und Wiederherstellungsprozessen. O. J., URL: https://support.microsoft.com/de-de/office/sch%C3%BCtzen-von-daten-mit-sicherungs-und-wiederherstellungsprozessen-96539a81-5984-4d56-99ca-ee81f8d6356c?ui=de-de&rs=de-de&ad=de, letzter Zugriff: 23.08.2023
[208] Microsoft (o. V.): SQL Server Licensing Datasheet. 2022, URL: https://download.microsoft.com/download/0/f/4/0f4c1b3c-cbc4-4495-97e4-2050543f49b3/SQL_Server_2022_Licensing_Datasheet.pdf, letzter Zugriff: 23.08.2023
[209] Microsoft (o. V.): Steuern von Dateneingabeformaten mithilfe von Eingabeformaten. O. J., URL: https://support.microsoft.com/de-de/office/steuern-von-dateneingabeformaten-mithilfe-von-eingabeformaten-e125997a-7791-49e5-8672-4a47832de8da, letzter Zugriff: 23.08.2023
[210] Microsoft (o. V.): Understanding the Impact of Low-Code Development with Power Apps + Power Automate. 2020, URL: https://info.microsoft.com/ww-Landing-Low-Code-Development-Impact-ebook.html, letzter Zugriff 23.08.2023
[211] Microsoft (o. V.): Unterschiede zwischen Office-Skripts und VBA-Makros. 2023, URL: https://learn.microsoft.com/de-de/office/dev/scripts/resources/vba-differences, letzter Zugriff 23.08.2023
[212] Microsoft (o. V.): Using Access or Excel to manage your datao. O. J., URL: https://support.microsoft.com/en-us/office/using-access-or-excel-to-manage-your-data-09576147-47d1-4c6f-9312-e825227fcaea, letzter Zugriff 23.08.2023
[213] Microsoft (o. V.): Verbergen von VBA-Code vor Benutzern. O. J., URL: https://support.microsoft.com/de-de/office/verbergen-von-vba-code-vor-benutzern-ce6ab610-af07-4008-91e0-1ef1b796ff18, letzter Zugriff 23.08.2023
[214] Microsoft (o. V.): Verbinden von Access mit SQL Server. O. J., URL: https://support-uat.microsoft.com/de-de/office/verbinden-von-access-mit-sql-server-050d88f3-b2d6-4e76-b6f9-f3c556f139ea, letzter Zugriff: 23.08.2023
[215] Microsoft (o. V.): Verhindern und Beheben von Datenbankdateiproblemen mithilfe von Komprimieren und Reparieren. O. J., URL: https://support.microsoft.com/de-

de/office/verhindern-und-beheben-von-datenbankdateiproblemen-mithilfe-von-komprimieren-und-reparieren-6ee60f16-aed0-40ac-bf22-85fa9f4005b2, letzter Zugriff: 23.08.2023

[216] Microsoft (o. V.): Verschlüsseln einer Datenbank mithilfe eines Datenbankkennworts. O. J., URL: https://support.microsoft.com/de-de/office/verschl%C3%BCsseln-einer-datenbank-mithilfe-eines-datenbankkennworts-12aa0e5c-34c6-4957-af3b-b5f5cfa9a766?ns=msaccess&version=90&syslcid=1031&uilcid=1031&appver=zac900&helpid=116589&ui=de-de&rs=de-de&ad=de, letzter Zugriff 23.08.2023

[217] Microsoft (o. V.): Verwalten von ODBC-Datenquellen. O. J., URL: https://support.microsoft.com/de-de/office/verwalten-von-odbc-datenquellen-b19f856b-5b9b-48c9-8b93-07484bfab5a7, letzter Zugriff 23.08.2023

[218] Microsoft (o. V.): Verwenden des Diagnosedaten-Viewers mit Office. O. J., URL: https://support.microsoft.com/de-de/office/verwenden-des-diagnosedaten-viewers-mit-office-cf761ce9-d805-4c60-a339-4e07f3182855, letzter Zugriff 23.08.2023

[219] Microsoft (o. V.): Video: Erstellen einer Access-App. O. J., URL: https://support.microsoft.com/de-de/office/video-erstellen-einer-access-app-477edd92-e1ed-4796-8a05-db3fffa81791, letzter Zugriff: 23.08.2023

[220] Microsoft (o. V.): Was ist Azure SQL?. 2023, URL: https://learn.microsoft.com/de-de/azure/azure-sql/azure-sql-iaas-vs-paas-what-is-overview?view=azuresql, letzter Zugriff 23.08.2023

[221] Microsoft (o. V.): Was ist der Unterschied zwischen Microsoft 365 und Office 2021?. O. J., URL: https://support.microsoft.com/de-de/office/was-ist-der-unterschied-zwischen-microsoft-365-und-office-2021-ed447ebf-6060-46f9-9e90-a239bd27eb96, letzter Zugriff 23.08.2023

[222] Microsoft (o. V.): What is IaaS?. O. J., URL: https://azure.microsoft.com/en-ca/resources/cloud-computing-dictionary/what-is-iaas/, letzter Zugriff 23.08.2023

[223] Microsoft (o. V.): Zero Trust Essentials eBook. O. J., URL: https://query.prod.cms.rt.microsoft.com/cms/api/am/binary/RWIrfk, letzter Zugriff 23.08.2023

[224] Microsoft (Robyn Hicock): Password Guidance. 2016, URL: https://www.microsoft.com/en-us/research/publication/password-guidance/, letzter Zugriff 23.08.2023

[225] Mircrosoft (MS et al.): System dynamic management views. 2023, URL: https://learn.microsoft.com/en-us/sql/relational-databases/system-dynamic-management-views/system-dynamic-management-views?view=sql-server-ver16, letzter Zugriff 23.08.2023

[226] Mircrosoft (MS et al.): TDE (Transparent Data Encryption) für SQL-Datenbank, SQL Managed Instance und Azure Synapse Analytics. 2023, URL: https://learn.microsoft.com/de-de/azure/azure-sql/database/transparent-data-encryption-tde-overview?view=azuresql-mi&tabs=azure-portal, letzter Zugriff 23.08.2023

[227] Mircrosoft (MS et al.): The Transaction Log (SQL Server). 2023, URL: https://learn.microsoft.com/en-US/sql/relational-databases/logs/the-transaction-log-sql-server?view=sql-server-ver16, letzter Zugriff 23.08.2023

[228] Mircrosoft (MS et al.): Transparent data encryption (TDE). 2023, URL: https://learn.microsoft.com/en-us/sql/relational-databases/security/encryption/transparent-data-encryption?view=sql-server-ver16, letzter Zugriff 23.08.2023

[229] Mircrosoft (MS et al.): T-SQL differences between SQL Server and Azure SQL Database. 2022, URL: https://learn.microsoft.com/en-us/azure/azure-sql/database/transact-sql-tsql-differences-sql-server?view=azuresql, letzter Zugriff 23.08.2023

[230] Mircrosoft (MS et al.): Verwenden von Transaktionen in einem DAO-Recordset. 2023, URL: https://learn.microsoft.com/de-de/office/vba/access/concepts/data-access-objects/use-transactions-in-a-dao-recordset, letzter Zugriff 23.08.2023

[231] Mircrosoft (MS et al.): What Is ODBC?. 2023, URL: https://learn.microsoft.com/en-us/sql/odbc/reference/what-is-odbc?view=sql-server-ver16, letzter Zugriff 23.08.2023

[232] Mohan Bhatia: Banking 4.0 – The Industrialised Bank of Tomorrow. 1. Auflage, Springer Nature, Singapur 2022

[233] NTFS.com (o. V.): File and Folder Advanced Permissions. O. J., URL: https://www.ntfs.com/ntfs-permissions-file-advanced.htm, letzter Zugriff 23.08.2023

[234] NTFS.com (o. V.): NTFS File Types. O. J., URL: https://www.ntfs.com/ntfs-files-types.htm, letzter Zugriff 23.08.2023
[235] NTFS.com (o. V.): NTFS Master File Table (MFT). O. J., URL: https://www.ntfs.com/ntfs-mft.htm, letzter Zugriff 23.08.2023
[236] NTFS.com (o. V.): NTFS Overview. O. J., URL: https://www.ntfs.com/ntfs_basics.htm, letzter Zugriff 23.08.2023
[237] Oracle (o. V.): Database Development Guide - What is ODBC?. O. J., URL: https://docs.oracle.com/en/database/oracle/oracle-database/19/adfns/odbc-driver.html#GUID-7931EDFB-7A70-4BBE-903E-8A2BB09DBE9D, letzter Zugriff: 23.08.2023
[238] Oracle (o. V.): Database PL/SQL Language Reference. O. J., https://docs.oracle.com/en/database/oracle/oracle-database/21/lnpls/overview.html#GUID-2FBCFBBE-6B42-4DB8-83F3-55B63B75B1EB, letzter Zugriff: 23.08.2023
[239] Oracle (o. V.): Developer's Guide for Microsoft Windows – 1 Introduction to Oracle Provider for OLE DB. O. J., https://docs.oracle.com/en/database/oracle/oracle-data-access-components/19.3/oledb/introduction-to-oracle-provider-for-oledb.html#GUID-FFC7020D-27B0-41AA-9D91-35EACB743106, letzter Zugriff: 23.08.2023
[240] O'Reilly (o. V.): Compilation and the MSIL. O. J., URL: https://www.oreilly.com/library/view/programming-visual-basic/0596004389/ch01s04.html, letzter Zugriff 23.08.2023
[241] OWASP (Jim Manico, Jakub Maćkowski, Kevin W. Wall, Shlomo Zalman Heigh et al.): OWASP Cheat Sheet Series. O. J., URL: https://cheatsheetseries.owasp.org/index.html, letzter Zugriff 23.08.2023
[242] Password Depot (o. V.): Brute-Force-Angriffe. O. J., URL: https://www.password-depot.de/de/know-how/brute-force-angriffe.htm, letzter Zugriff 23.08.2023
[243] Peter Fischer & Peter Hofer: Lexikon der Informatik. 15. überarbeitete Auflage. Heidelberg: Springer-Verlag, 2011
[244] SANS (Rob Lee): SIFT Workstation. O. J., URL: https://www.sans.org/tools/sift-workstation/, letzter Zugriff 23.08.2023
[245] Sascha Kersken: IT-Handbuch für Fachinformatiker – Der Ausbildungsbegleiter. 8., aktualisierte Auflage 2017, 1. Nachdruck 2018, Rheinwerk Computing, Bonn 2017
[246] SOURCEFORGE (o. V.): Microsoft Access. O. J., URL: https://sourceforge.net/software/product/Microsoft-Access/, letzter Zugriff 23.08.2023
[247] Spring (o. V.): 15.10 Convention over configuration support. O. J., URL: https://docs.spring.io/spring-framework/docs/3.0.0.M4/reference/html/ch15s10.html, letzter Zugriff 23.08.2023
[248] Springer Link (Ralf Kneuper & Sven Jacobs): Softwaretest mit Originaldaten – Eine Analyse aus Sicht des Datenschutzes. 2021, URL: https://link.springer.com/content/pdf/10.1007/s11623-021-1411-8.pdf, letzter Zugriff 23.08.2023
[249] SQLite (o. V.): SQLite As An Application File Format. 2022, URL: https://www.sqlite.org/appfileformat.html, letzter Zugriff 23.08.2023
[250] SQLShack (Minette Steynberg): Reading the SQL Server Transaction Log. 2015, URL: https://www.sqlshack.com/reading-sql-server-transaction-log/, letzter Zugriff 23.08.2023
[251] Stack Overflow (Stack Overflow et al.): Removing the password from a VBA project. 2008, URL: https://stackoverflow.com/questions/272503/removing-the-password-from-a-vba-project, letzter Zugriff 23.08.2023
[252] Statista (DB-Engines): Ranking of the most popular database management systems worldwide, as of February 2023. 2023, URL: https://www.statista.com/statistics/809750/worldwide-popularity-ranking-database-management-systems/, letzter Zugriff 23.08.2023
[253] Statista (edgescan): Distribution of web application critical vulnerabilities worldwide as of 2022. 2022, URL: https://www.statista.com/statistics/806081/worldwide-application-vulnerability-taxonomy/, letzter Zugriff 23.08.2023
[254] Statista (Mathias Brandt): Office-Software dominiert deutsche Büros. 2015, URL: https://de.statista.com/infografik/3367/nutzung-von-office-software-in-deutschland/, letzter Zugriff 23.08.2023

[255] Statista (Nielsen & Empower GmbH): Meistgenutzte Office-Software von Büromitarbeitern in Unternehmen in Deutschland im Jahr 2020. 2020, URL: https://de.statista.com/statistik/daten/studie/77226/umfrage/internetnutzer-verbreitung-von-office-software-in-deutschland/, letzter Zugriff 23.08.2023
[256] Statista (R&D WORLD): Ranking of technologies by expected importance worldwide in 2025. 2022, URL: https://www.statista.com/statistics/732288/worldwide-research-and-development-important-technologies/, letzter Zugriff 23.08.2023
[257] Statista (Statista Research Department): Statistiken zu Microsoft. 2023, URL: https://de.statista.com/themen/239/microsoft/#topicOverview, letzter Zugriff 23.08.2023
[258] Steven Roman: Access Database Design and Programming – Creating Programmable Database Applications with Access 97, 2000, 2002 & 2003. 3. Auflage, O'Reilly Media Inc., Sebastopol 2002
[259] Telekom (o. V.): Sicherheit als Designkriterium. 2012, URL: https://www.telekom.com/de/konzern/datenschutz-und-sicherheit/archiv-datenschutznews/news/sicherheit-als-designkriterium-342980, letzter Zugriff 23.08.2023
[260] Tenfold Softwareentwicklung GmbH & Co. KG (Nele Nikolaisen): Least-Privilege-Prinzip: Access managen, Daten schützen. 2020, URL: https://www.tenfold-security.com/least-privilege-user-access/, letzter Zugriff 23.08.2023
[261] The Sleuth Kit® (o. V.): Open Source Digital Forensics. O. J., URL: https://sleuthkit.org/, letzter Zugriff 23.08.2023
[262] Thegrideon Software (o. V.): ACCDB PASSWORD RECOVERY. O. J., URL: https://www.thegrideon.com/accdb-password-recovery.html, letzter Zugriff 23.08.2023
[263] Thegrideon Software (o. V.): ACCESS FORENSICS, URL: https://www.thegrideon.com/access-forensics.html, letzter Zugriff 23.08.2023
[264] TrustRadius (o. V.): Microsoft Access. O. J., URL: https://www.trustradius.com/products/microsoft-access/reviews?qs=product-usage&sr=1%2C2%2C3&lu=4#overview, letzter Zugriff 23.08.2023
[265] Veikko Krypczyk & Olena Bochkor: Handbuch für Softwareentwickler. 1., Auflage, Rheinwerk Computing, Bonn 2018
[266] Velociraptor (Mike Cohen): Recovering deleted NTFS Files with Velociraptor – Deep forensics on the endpoint. 2019 URL: https://velociraptor.velocidex.com/recovering-deleted-ntfs-files-with-velociraptor-1fcf09855311, letzter Zugriff 23.08.2023
[267] Volexity (Josh Grunzweig, Matthew Meltzer, Sean Koessel, Steven Adair & Thomas Lancaster): Operation Exchange Marauder: Active Exploitation of Multiple Zero-Day Microsoft Exchange Vulnerabilities. 2021, URL: https://www.volexity.com/blog/2021/03/02/active-exploitation-of-microsoft-exchange-zero-day-vulnerabilities/, letzter Zugriff 23.08.2023
[268] Volker Brühl & Joachim Dorschel (Hrsg.): Praxishandbuch – Digital Banking. 1. Auflage, Springer Gabler, Wiesbaden 2018
[269] Wie-funktioniert.com (o. V.): Die vier Generationen der Programmiersprachen. 2016, URL: https://www.wie-funktioniert.com/die-vier-generationen-der-programmiersprachen/, letzter Zugriff 23.08.2023
[270] Wikipedia (Wikipedia et al.): Instructions per second. 2023, URL: https://en.wikipedia.org/wiki/Instructions_per_second, letzter Zugriff 23.08.2023
[271] Wired (o. V.): Bill Gates: Trustworthy Computing. 2002, URL: https://www.wired.com/2002/01/bill-gates-trustworthy-computing/, letzter Zugriff 23.08.2023
[272] WirtschaftsWoche (Thomas Kuhn): Microsoft wird zum Sicherheitsrisiko. 2023, URL: https://www.wiwo.de/technologie/digitale-welt/cybersicherheits-gau-microsoft-wird-zum-sicherheitsrisiko/29284812.html, letzter Zugriff 23.08.2023
[273] WirtschaftsWoche (Thomas Kuhn): Microsofts gefährliches Schweigen. 2021, URL: https://www.wiwo.de/technologie/digitale-welt/cybersecurity-microsofts-gefaehrliches-schweigen/26992710.html, letzter Zugriff 23.08.2023
[274] Wiz (Shir Tamari): Compromised Microsoft Key: More Impactful Than We Thought. 2023, URL: https://www.wiz.io/blog/storm-0558-compromised-microsoft-key-enables-authentication-of-countless-micr, letzter Zugriff 23.08.2023

[275] Wolfram Langer: Access – Das umfassende Handbuch. 3. aktualisierte Auflage, Rheinwerk Verlag GmbH, Bonn 2023

Bilderverzeichnis

Tabellenverzeichnis

Formelverzeichnis

Abkürzungsverzeichnis

Abkürzung	Langform
$MFT	Master File Table
ACE	Access Connectivity Engine
AD	Active Directory
ADE	Access Database Engine
ADO	ActiveX Data Objects
AES	Advanced Encryption Standard
AI	Artificial Intelligence
API	Application Programming Interfaces
AT	Allgemeiner Teil
BaFin	Bundesanstalt für Finanzdienstleistungsaufsicht
BAIT	Bankaufsichtliche Anforderungen an die IT
BDSG	Bundesdatenschutzgesetz
BSI	Bundesamt für Sicherheit in der Informationstechnik
CNG	Crypto API: Next Generation
COM	Component Object Model
CSA	Cloud Security Alliance
CSP	Crypto Service Provider
CSV	Comma-Separated Values
DAO	Data Access Object
DB	Datenbank
DBMS	Datenbankmanagementsystem
DBS	Datenbanksystem
DDL	Data Definition Language
DLL	Dynamic Link Library
DMF	System Dynamic Management Functions
DML	Data Manipulation Language
DMV	System Dynamic Management Views
DS-GVO	Datenschutz-Grundverordnung
EPUB	Electronic Publication
GIF	Graphics Interchange Format
GUI	Graphical User Interface
GUID	Globally Unique Identifier
IANA	Internet Assigned Numbers Authority
ID	Identifikator
IDV	Individuelle Datenverarbeitung
IEC	International Electrotechnical Commission
IID	Abbreviation Forinterface Identifier
IP	Internetprotokoll
IRM	Information Rights Management
ISAM	Internal Indexed Sequential Access Method
ISF	Information Security Forum
ISMS	Information Security Management System
ISO	International Organization for Standardization
IT	Informationstechnologie
JDBC	Java Database Connectivity
JET	Microsoft Joint Engine Technology
JPEG	Joint Photographic Experts Group
KI	Künstliche Intelligenz
KWG	Kreditwesengesetz
MaRisk	Mindestanforderungen an das Risikomanagement
MIME Type	Multipurpose Internet Mail Extensions
MS	Microsoft

NTFS	New Technology File System
OCP	Open Closed Principle
ODBC	Open Database Connectivity
OLE DB	Object Linking and Embedding, Database
PDCA	Plan-Do-Check-Act
PDF	Portable Document Format
RAID	Redundant Array of Independent Disks
RAM	Random-Access Memory
RDBMS	Relationales Datenbankmanagementsystem
RFC	Request for Comments
SANS	SysAdmin, Audit, Networking and Security
SOGP	Standard of Good Practice For Information Security
SQL	Structured Query Language
TCP	Transmission Control Protocol
TDE	Transparent Data Encryption
TLS	Transport Layer Security
TOM	Technisch-organisatorische Maßnahmen
Tz	Teilziffer
URL	Uniform Resource Locator
VPN	Virtual Private Network
VSTO	Visual Studio Tools for Office
WSL	Windows-Subsystem für Linux
XML	Extensible Markup Language
ZIP	Zipper

Anlagenverzeichnis

Anlagen

Anlage 1: Zero-Trust-Strategie

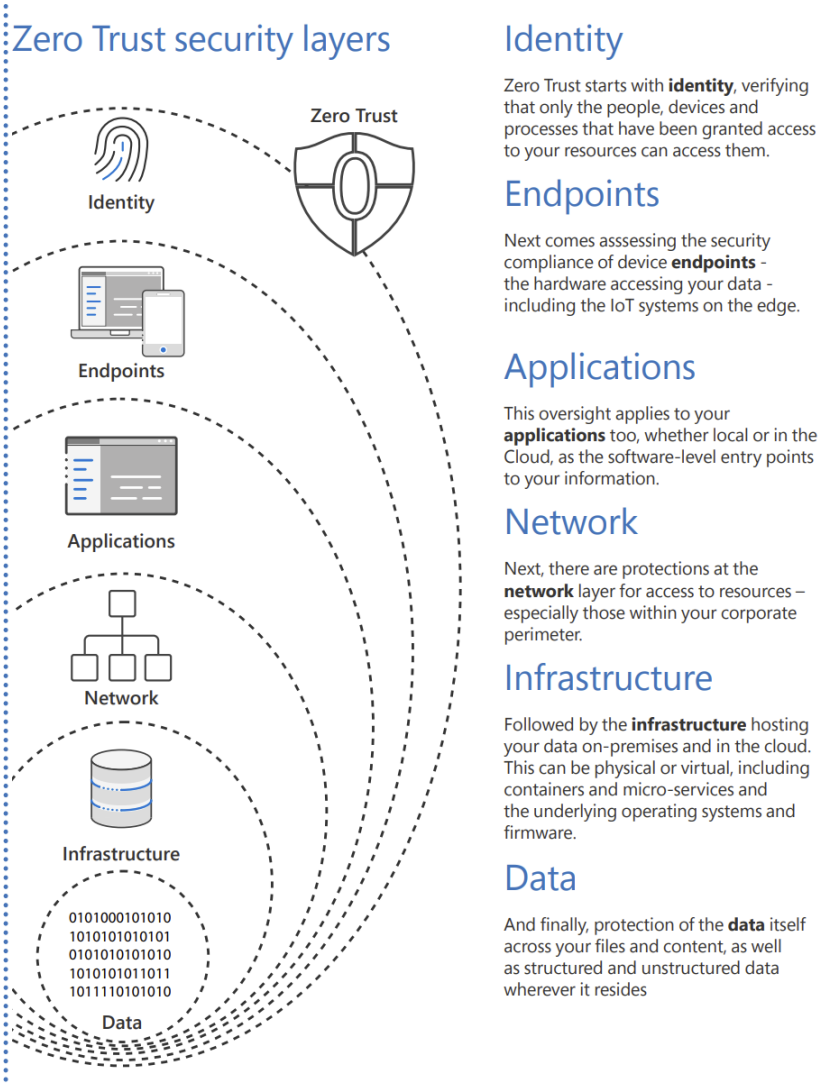


Bild 128: Schichten der Zero-Trust-Strategie [223, S. 2]

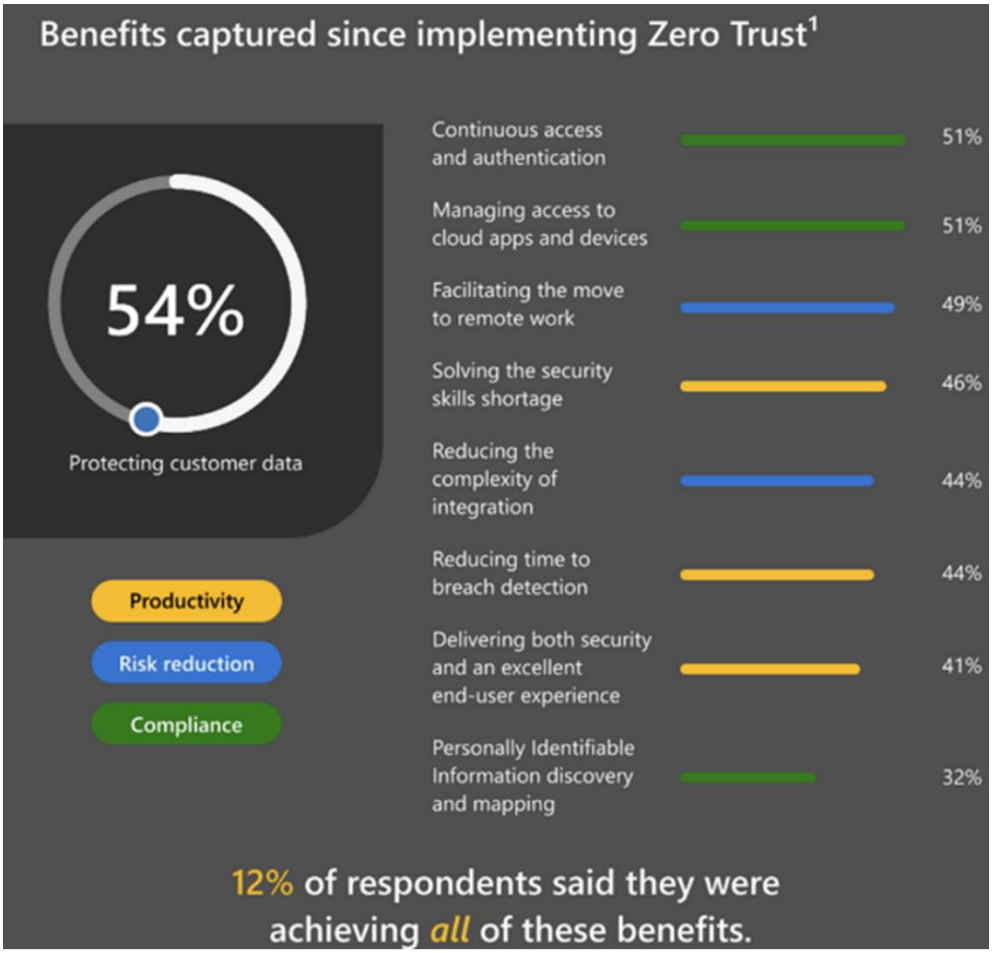


Bild 129: Von MS propagierte Vorteile bei Anwendung der Zero-Trust-Strategie [174]

Anlage 2: Banking 3.0 & 4.0

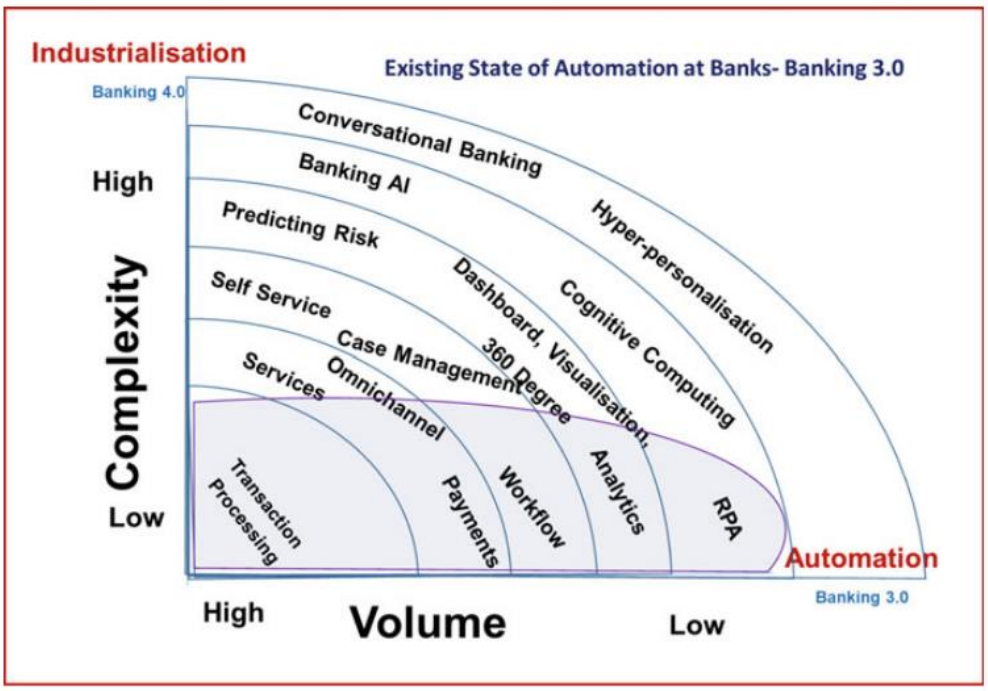


Bild 130: Scope von Banking 3.0 [232, S. 3]

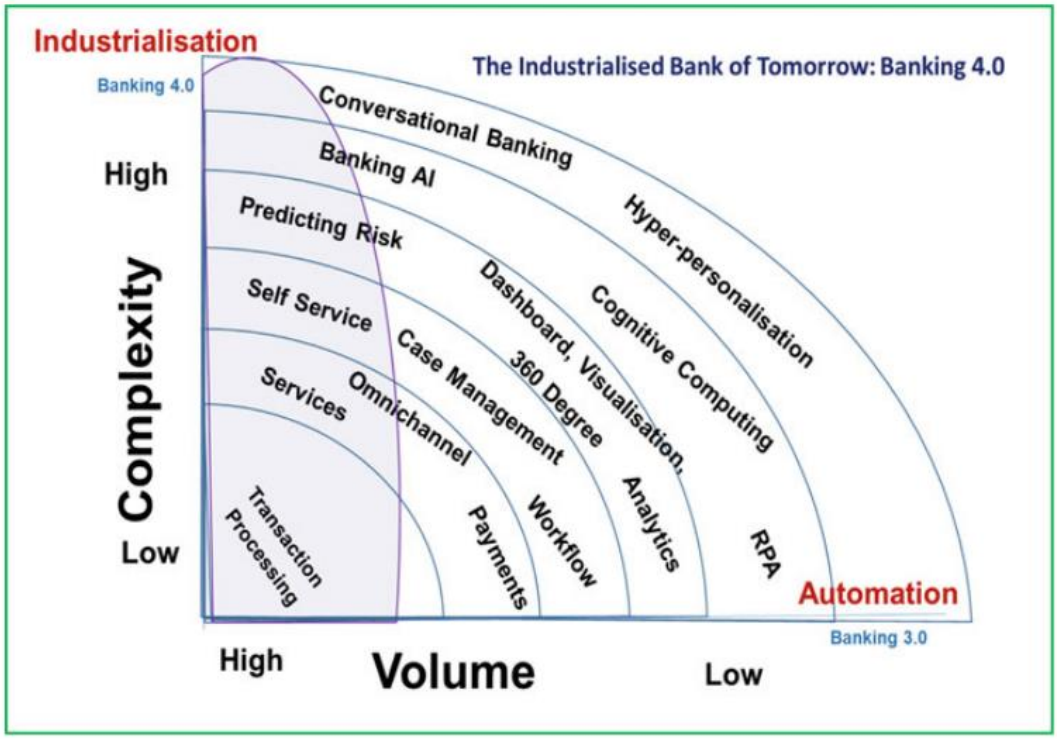


Bild 131: Scope von Banking 4.0 [232, S. 10]

Die folgende Grafik zeigt wesentliche Einflussfaktoren der Digitalisierung auf das Bankgeschäft der Zukunft sowie die sich daraus ergebenden Anforderungen an die technologische Infrastruktur der Institute:

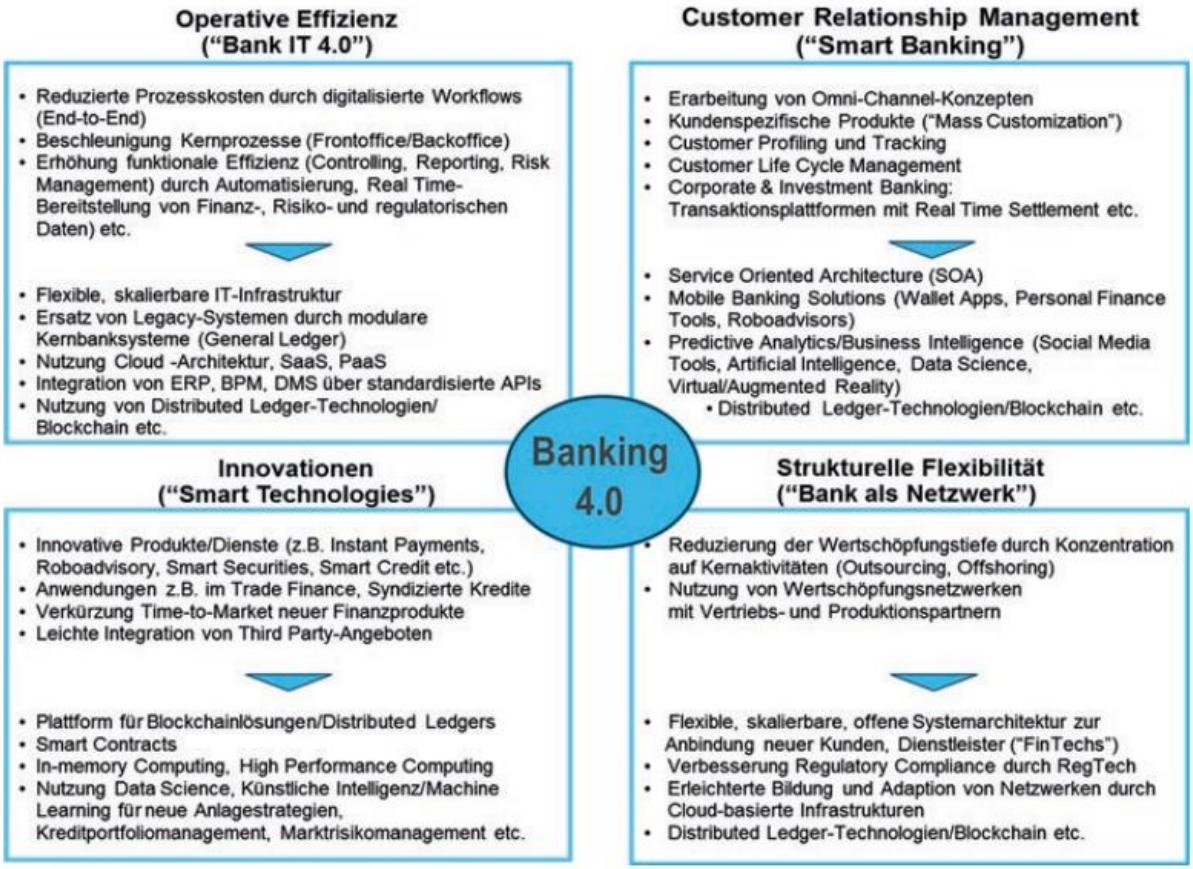


Bild 132: Übersicht Banking 4.0 [268, S. 5]

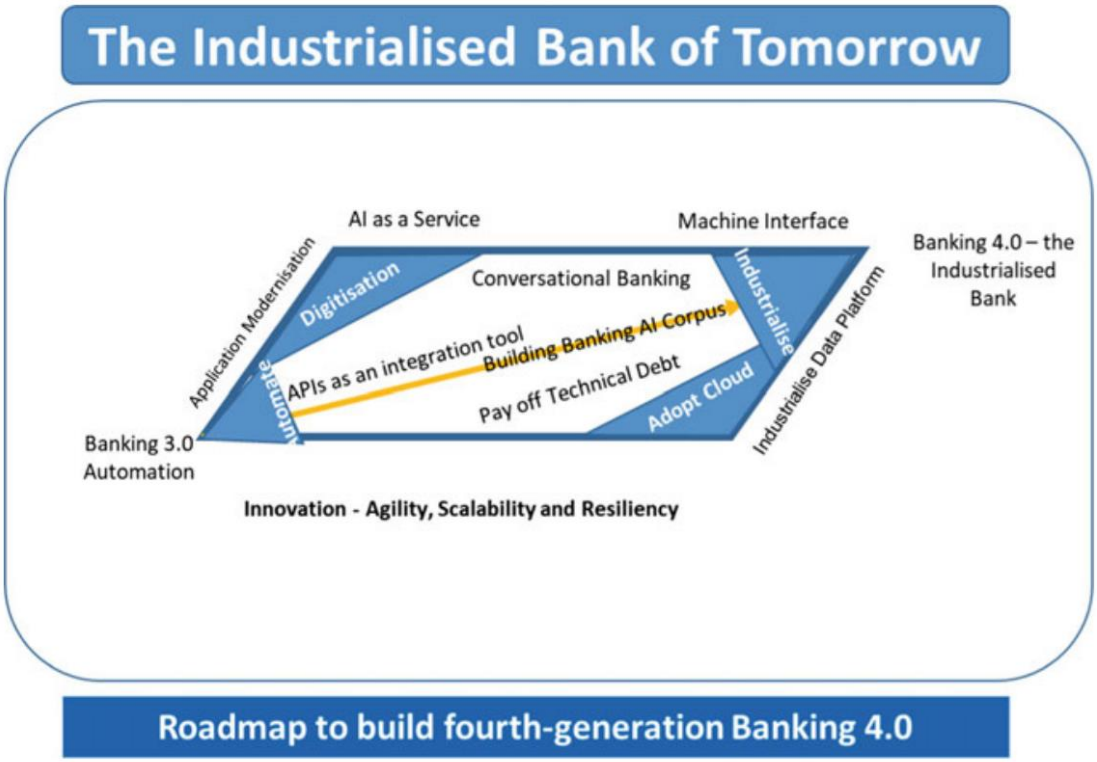


Bild 133: Roadmap zu Banking 4.0 0 [232, S. 11]

Anlage 3: Über das Entwurfsfenster konfigurierbare Eingabeprüfungen für Freitext-Steuerelemente zur Eingabe & Gültigkeitsregeln für Attribute in einer Relation (Access)

Hinzufügen von Gültigkeitsregeln für Eingabe-Steuerelemente über die Access-Oberfläche:

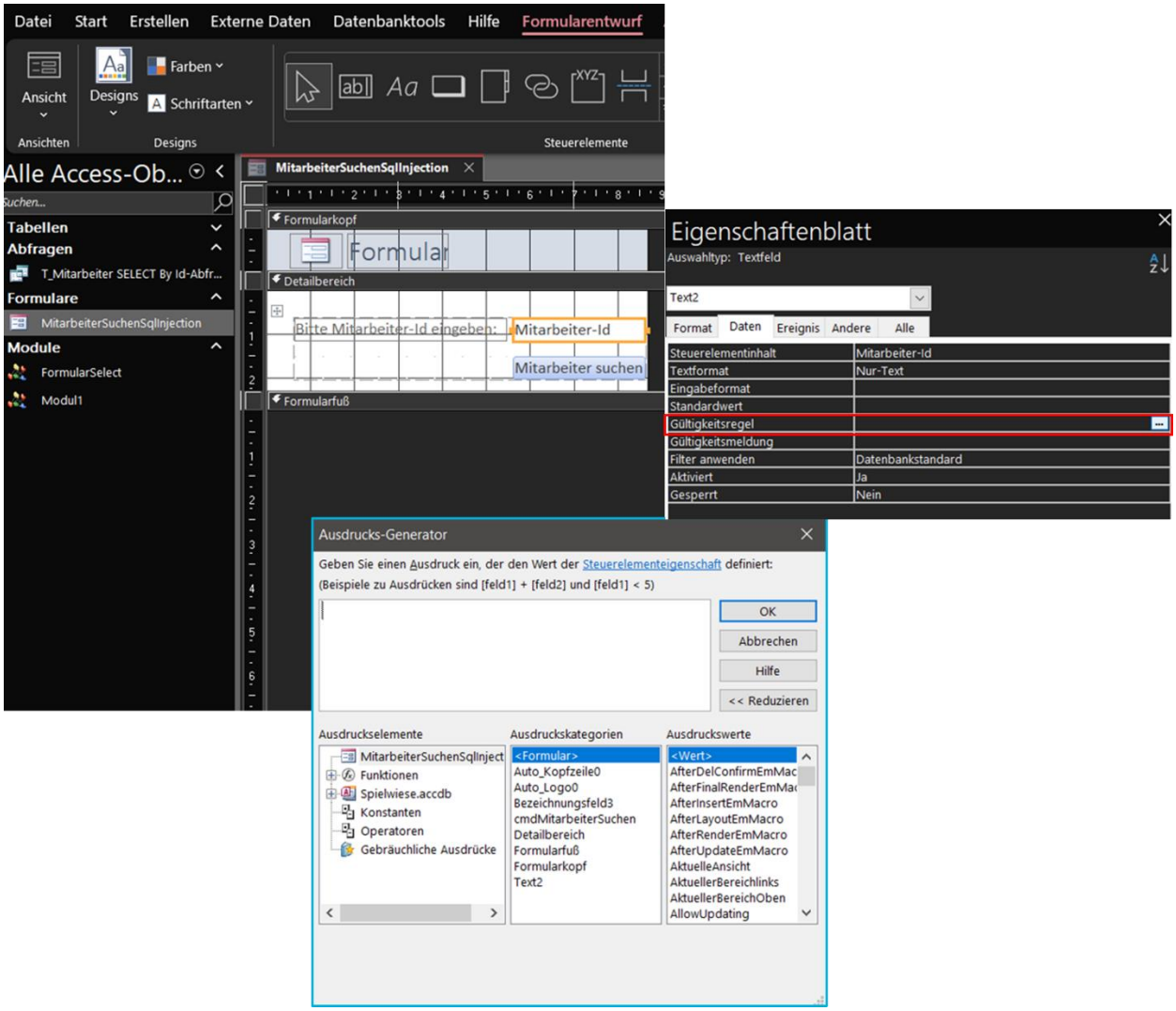


Bild 134: Hinzufügen von Gültigkeitsregeln für Eingabe-Steuerelemente über die Access-Oberfläche

Hinzufügen eines Ereignisses mit zugehörigem VBA-Code, das bei Änderungen im Steuerelement eintritt:

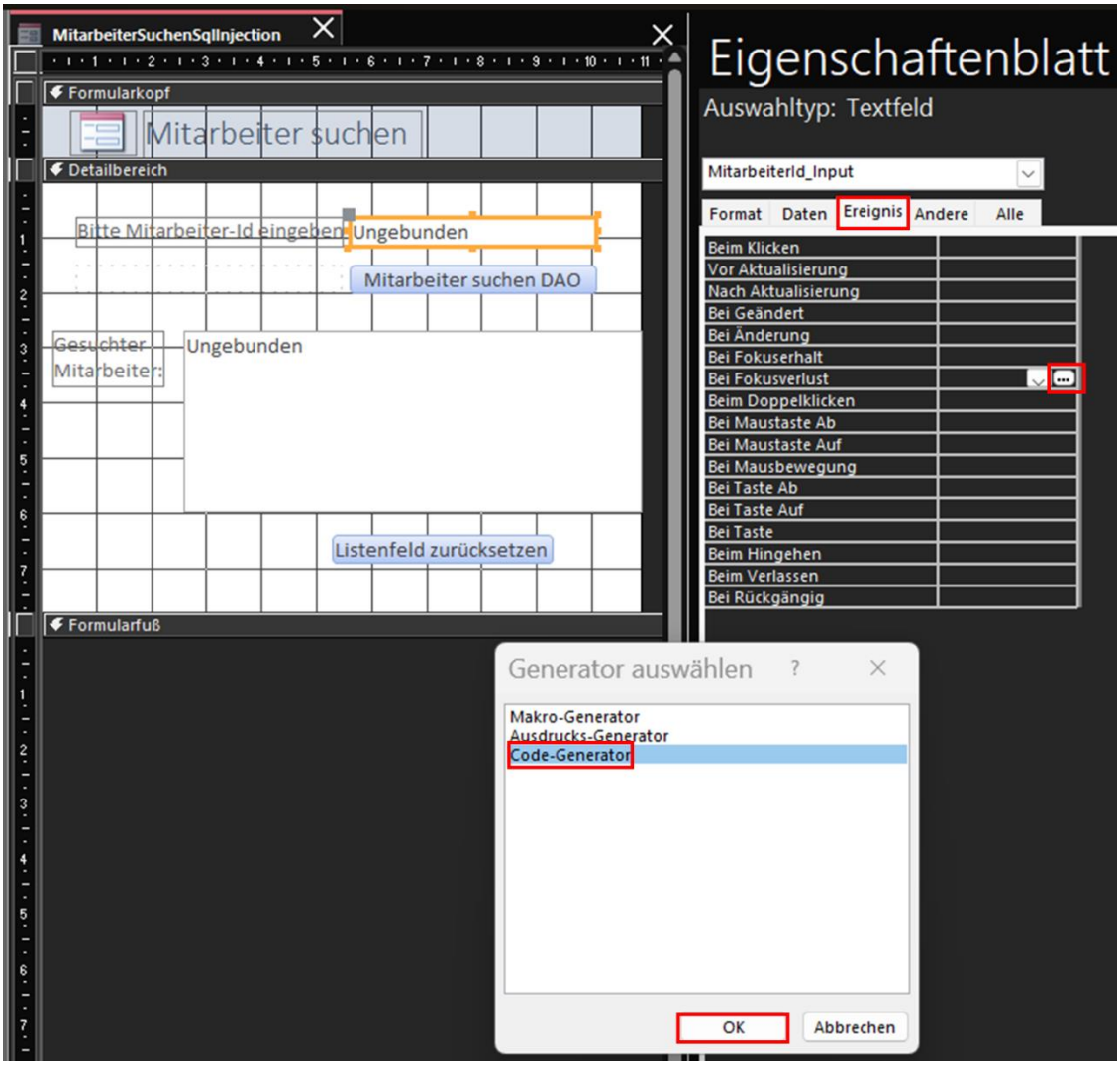


Bild 135: Aktivierung eines Ereignisses bei Änderungen im Eingabe-Steuerelement (hier bei Fokusverlust) über die Entwurfsansicht auf der Access-Oberfläche

Angreifende starten einen SQL-Injection-Angriff über in das in die Access-Datei eingebettete GUI:

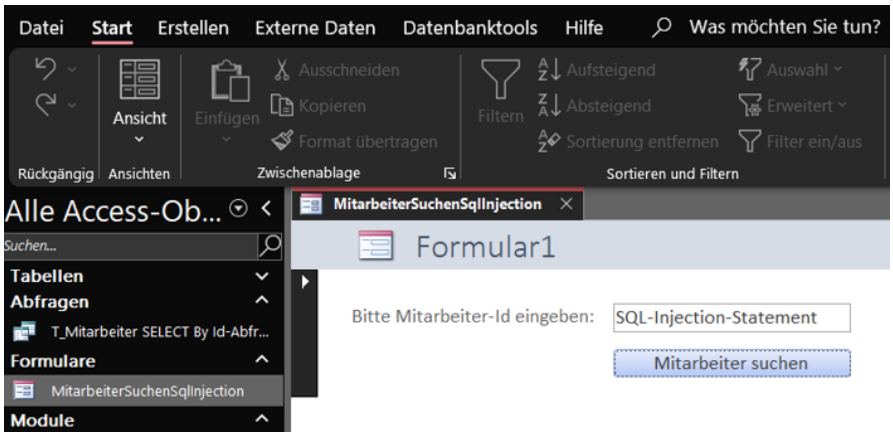


Bild 136: Eingabe eines SQL-Injection-Statements in das zuvor durch ein Ereignis geschützte Eingabe-Steuerelement in Access

Beim Eintreten des zuvor konfigurierten Ereignisses wird der VBA-Code aufgerufen:

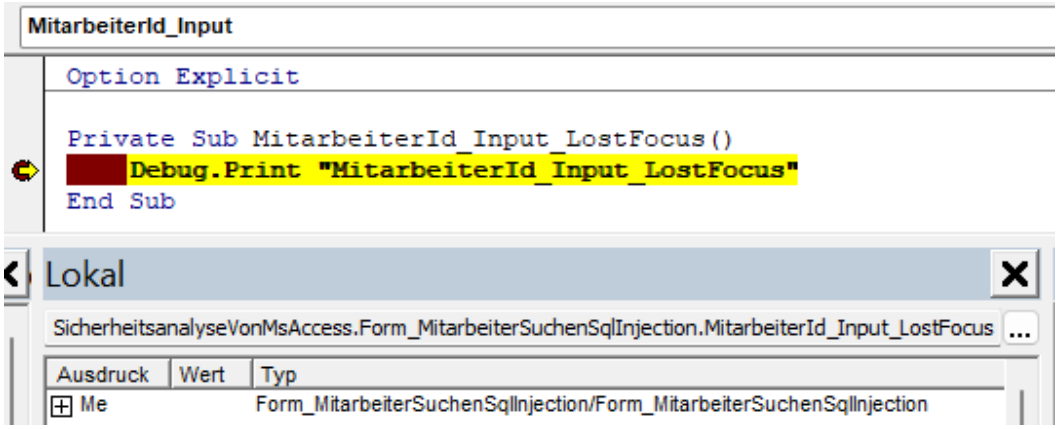


Bild 137: Auslösen des bei Fokusverlust-Ereignisses des geschützten Eingabe-Steuerelements in Access

Hinzufügen eines Eingabeformats für Eingabe-Steuerelemente über die Access-Oberfläche. Das Eingabeformat erlaubt nur Ziffern, wobei mindestens eine Ziffer eingegeben werden muss. „*0*“ steht für ein obligatorisches, „*9*“ für ein optionales Zeichen:

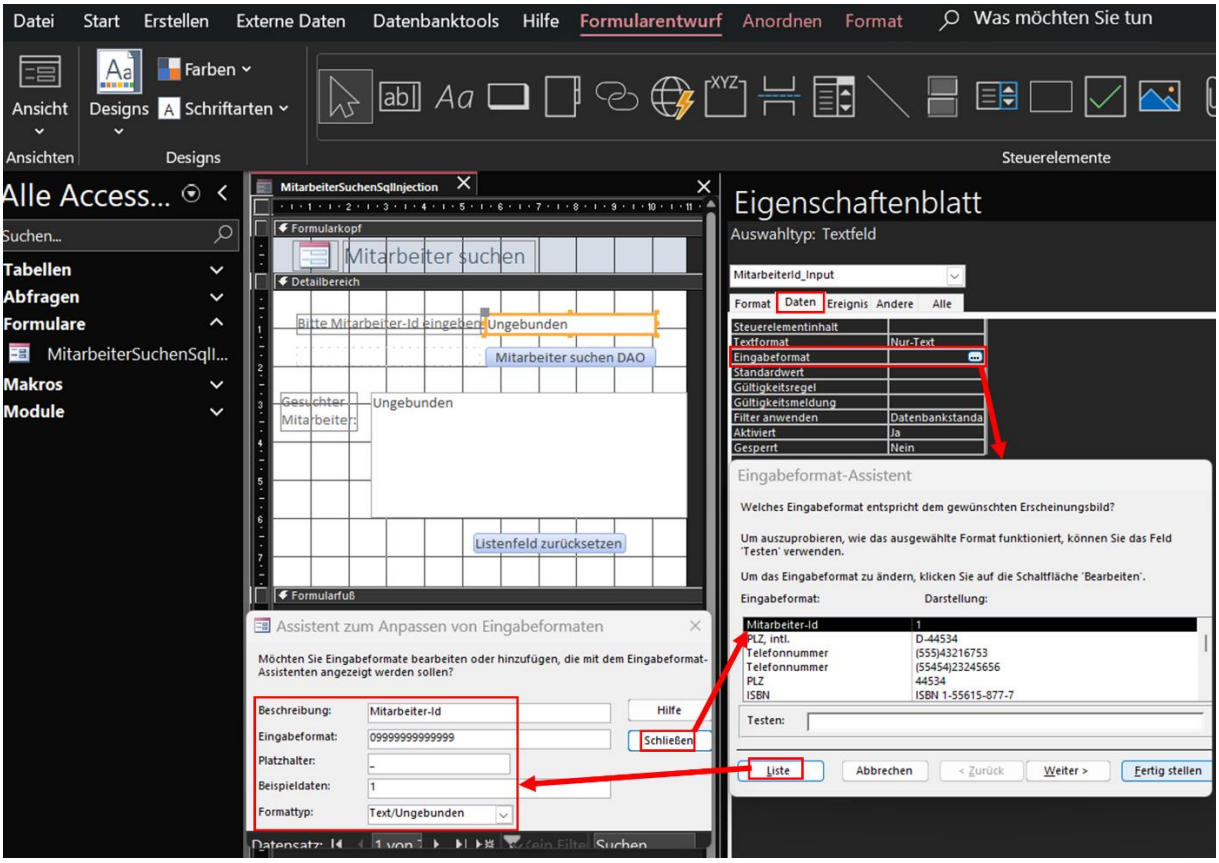


Bild 138: Hinzufügen und Konfigurieren eines Eingabeformats für Eingabe-Steuerelemente über die Entwurfsansicht der Access-Oberfläche [209]

Nach Hinzufügen des Eingabeformats sind keine Eingabe mehr von anderen Zeichen, wie „=“ oder „*OR*“, möglich:

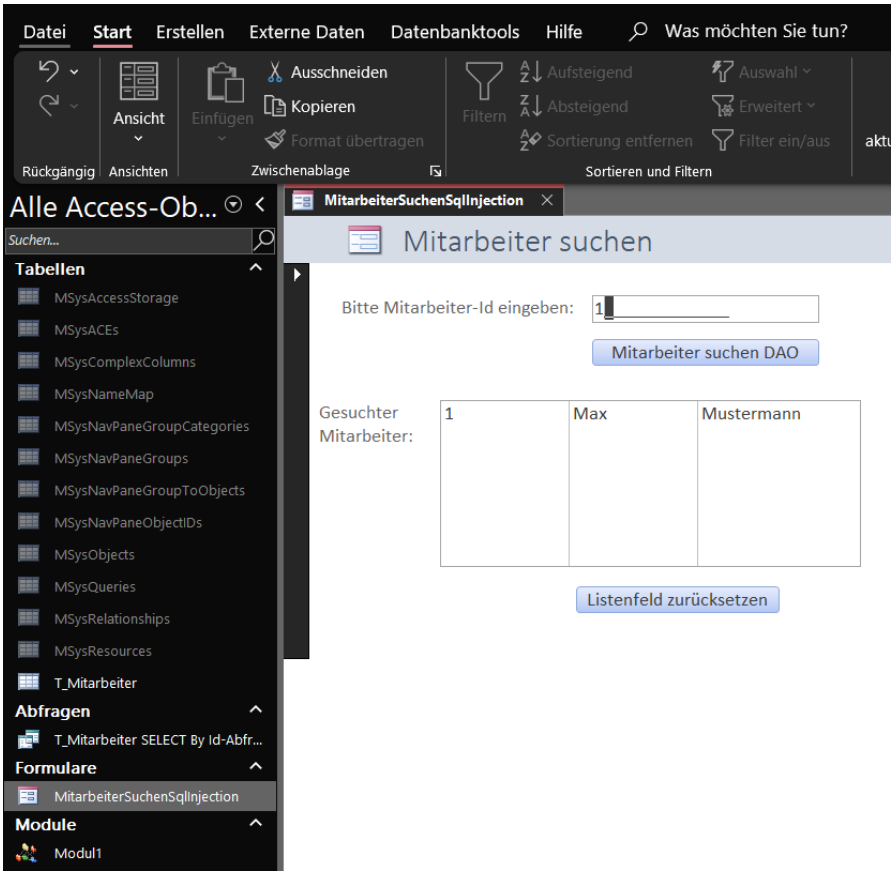


Bild 139: Eingabe eines Wertes in das zuvor durch ein Eingabeformat geschützte Eingabe-Steuerelement in Access

Ähnlich wie das Hinzufügen von Gültigkeitsregeln für Eingabe-Steuerelemente über die Access-Oberfläche funktioniert auch das Hinzufügen von Gültigkeitsregeln für Attribute in einer Access-Relation. Dabei können Gültigkeitsregeln attributübergreifend für einen gesamten Datensatz (inklusive einer benutzerdefinierten Fehlermeldung) konfiguriert werden. Die folgende Gültigkeitsregel verhindert das Einfügen von Datensätzen, bei

denen sowohl der Vor- als auch der Nachname leer oder nur mit Leerzeichen gefüllt sind:

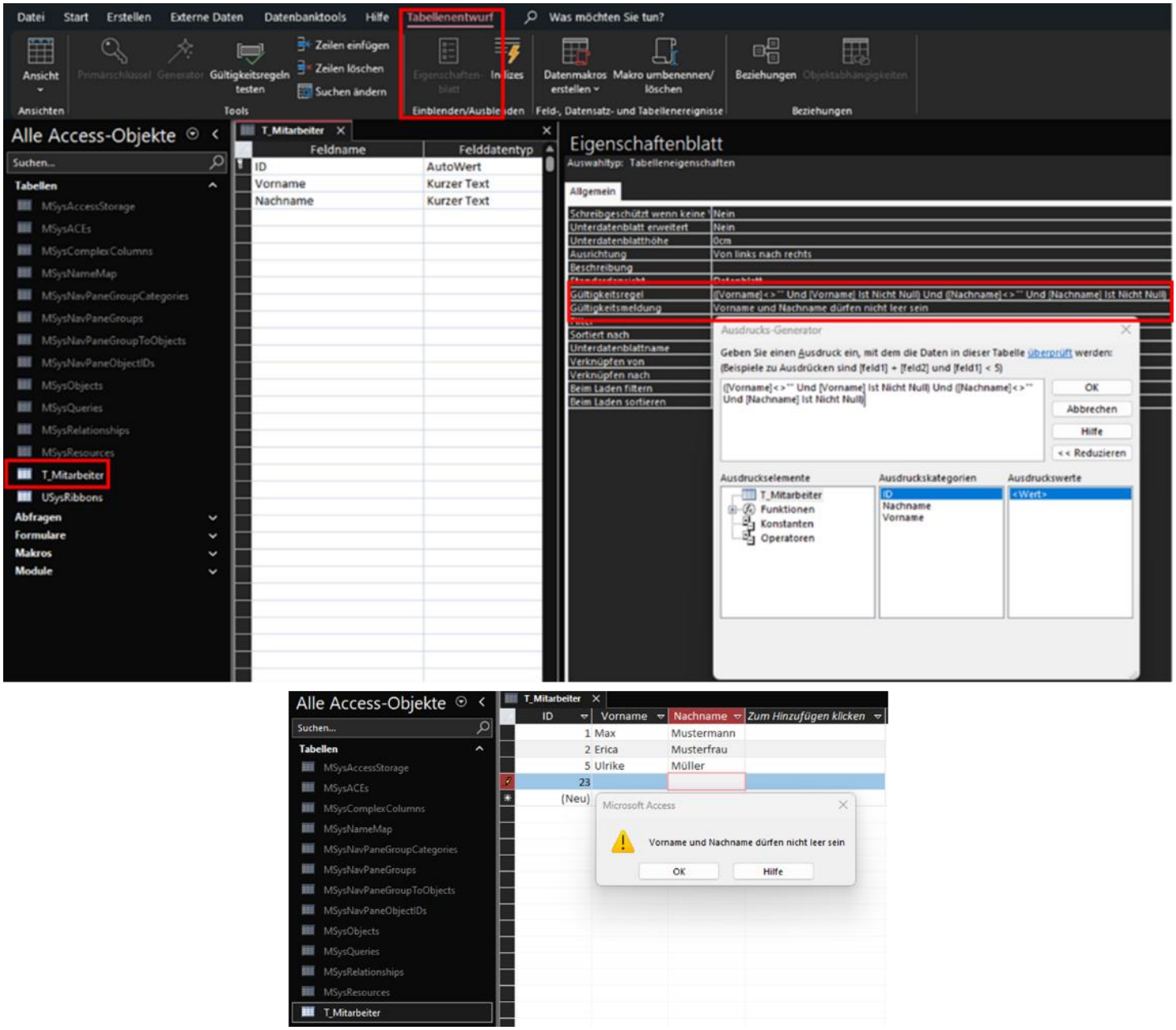


Bild 140: Konfiguration einer attributübergreifenden Gültigkeitsregel für Attribute in einer Access-Relation zur Prüfung des gesamten Datensatzes (inklusive einer benutzerdefinierten Fehlermeldung) und Erfassen eines Datensatzes, der gegen die Gültigkeitsregel verstößt

Anlage 4: Benutzerdefinierte Menüband-Konfiguration & sonstige Einstellungen (Access)

Nachfolgend sind die notwendigen Schritte zum Erstellen und Aushebeln eines benutzerdefinierten Menübandes bei .accdb- sowie .accde-Dateien skizziert. Falls die Relation „*USysRibbons*“ nicht existiert, muss sie manuell angelegt werden. Das Tabellenattribut „*ID*“ ist dabei der Primärschlüssel, die Attribute besitzen folgende Datentypen:

USysRibbons

Feldname	Felddatentyp
ID	AutoWert
RibbonName	Kurzer Text
RibbonXml	Langer Text

Bild 141: Benutzerdefinierte Menüband-Konfiguration – Attribute der USysRibbons-Tabelle

Anschließend kann für das Tabellenattribut „*RibbonXml*“ der folgende Attributwert festlegt werden. Mit dem Attribut „*startFromScratch*“ des Elements „*ribbon*“ kann festgelegt werden, ob alle vorhandenen Registerkarten entfernt und nur die Registerkarten angezeigt werden, die über die XML aktiviert sind („*startFromScratch == true*“). Das Element „*backstage*“ bezieht sich auf den Kopfreiter „*Datei*“. Das Button-Element mit der Ausprägung „*ApplicationOptionDialog*“ des Attributs „*idMso*“ deaktiviert die Optionen („*Datei → Optionen*“):

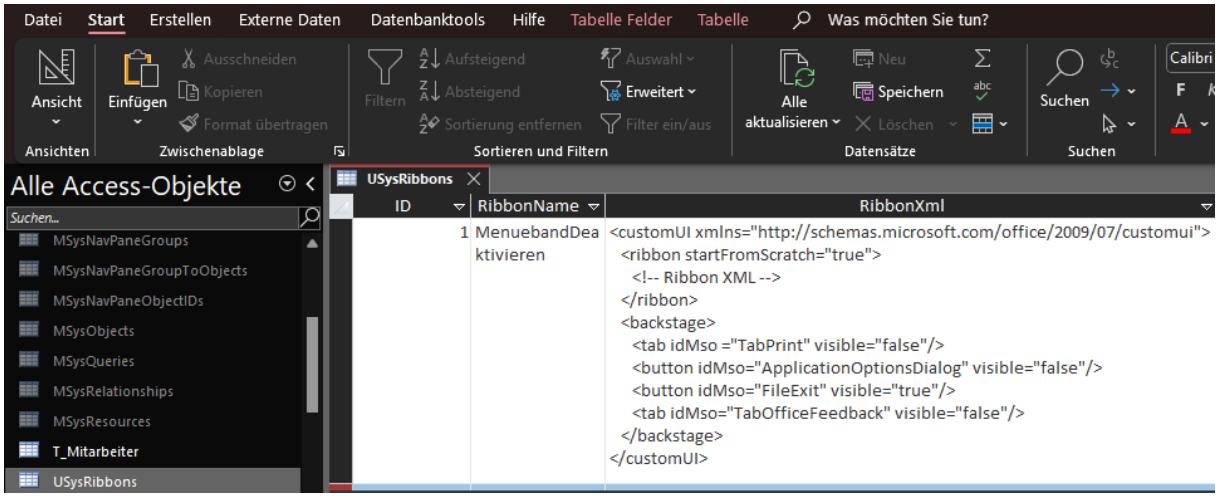


Bild 142: Benutzerdefinierte Menüband-Konfiguration – Inhalt der USysRibbons-Tabelle mit der XML-Konfiguration des Menübands

Damit das benutzerdefinierte Menüband in Form des erstellten XML in der Relation „*USysRibbons*“ aktiviert wird, muss es unter „*Datei → Optionen → Aktuelle Datenbank*“ unter „Name des Menübands:“ ausgewählt werden [191]:

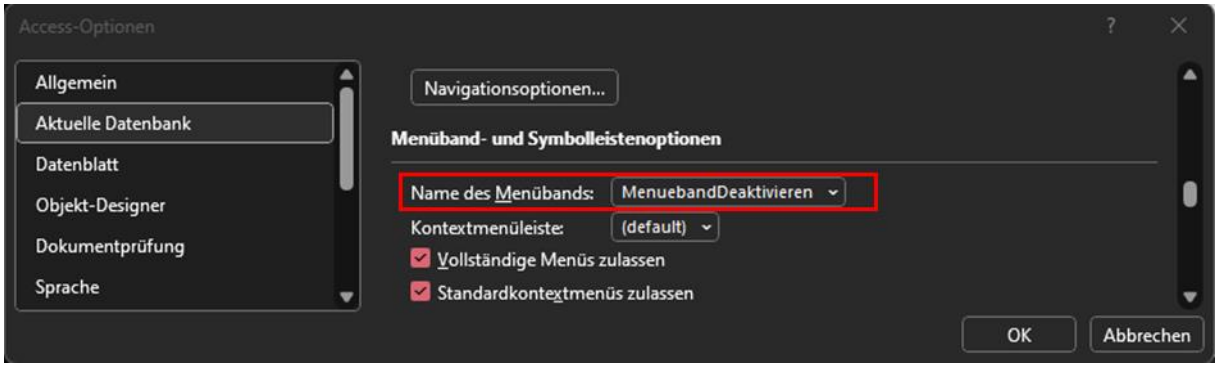


Bild 143: Benutzerdefinierte Menüband-Konfiguration – Aktivierung des benutzerdefinierten Menübands über die Access-Oberfläche

Nach dem Neustart der Access-Datei ist das benutzerdefinierte Menüband aktiv. Alle Kopfreiter bis auf „*Datei*“ sind nicht mehr sichtbar:

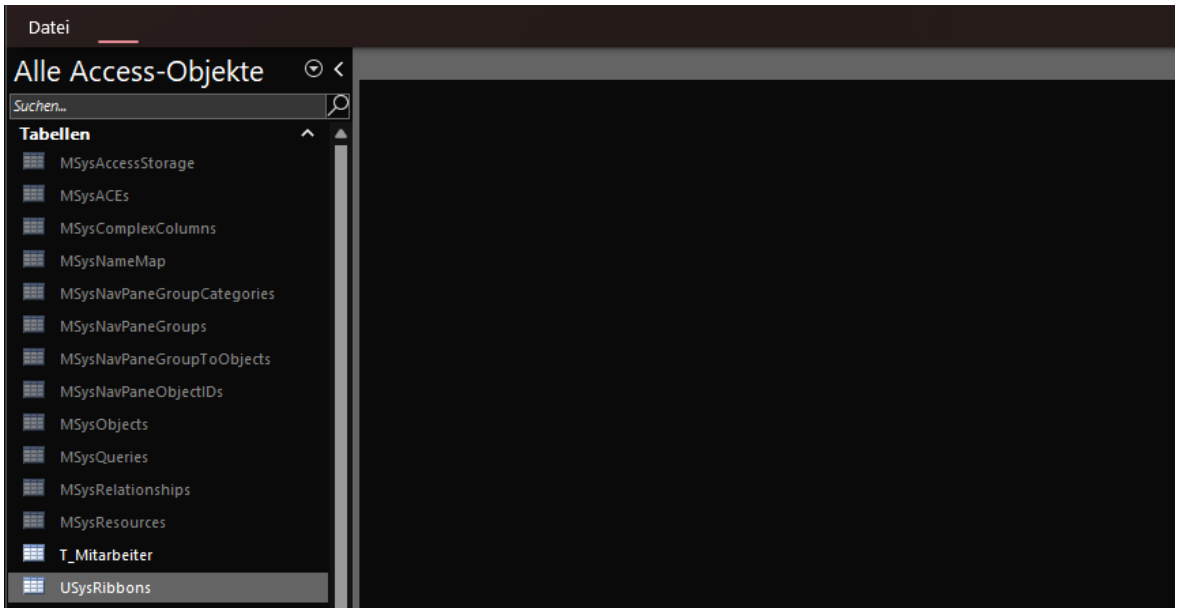


Bild 144: Benutzerdefinierte Menüband-Konfiguration – Bestätigung der Aktivierung 1/2

Im Kopfreiter „*Datei*“ sind die Funktionen wie „*Optionen*“ nicht auswählbar:

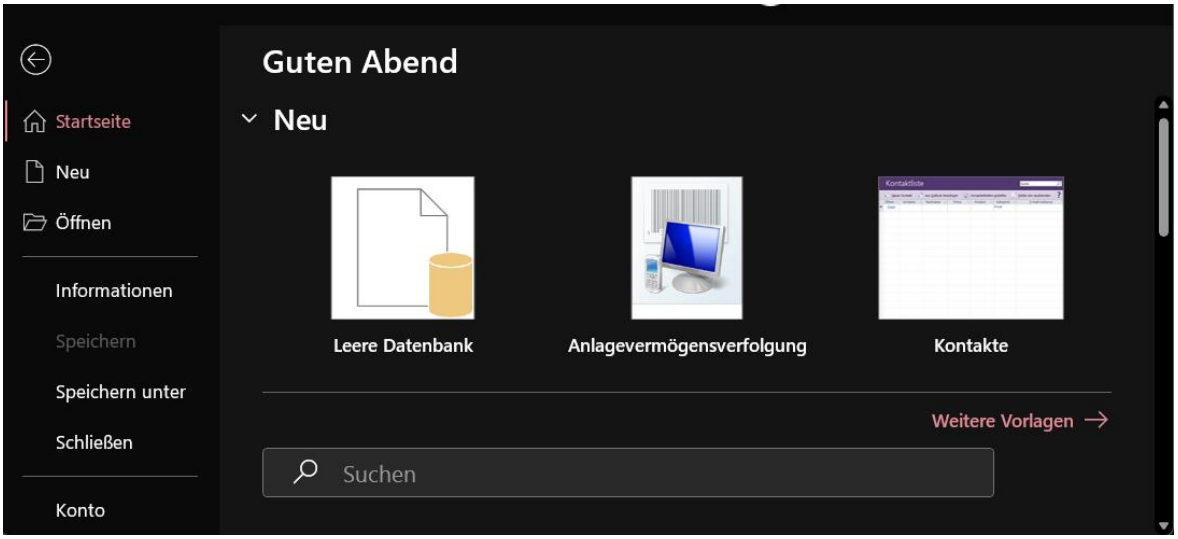


Bild 145: Benutzerdefinierte Menüband-Konfiguration – Bestätigung der Aktivierung (Kopfreiter „Datei“ aka. <Backstage>) 2/2

Um den Navigationsbereich und weitere Funktionen, wie Tastenkürzel der Tastatur oder das Kontextmenü zu deaktivieren, muss der Button „*Optionen*“ unter „*Datei*“ wieder eingeblendet werden. Dazu wird das in der Relation „`USysRibbons`“ im Tabellenattribut „`RibbonXml`“ hinterlegte XML erneut angepasst und dabei das Button-Element mit der Attributausprägung „`ApplicationOptionsDialog`“ des Attributs „`idMso`“ auf „`true`“ gesetzt und gespeichert:

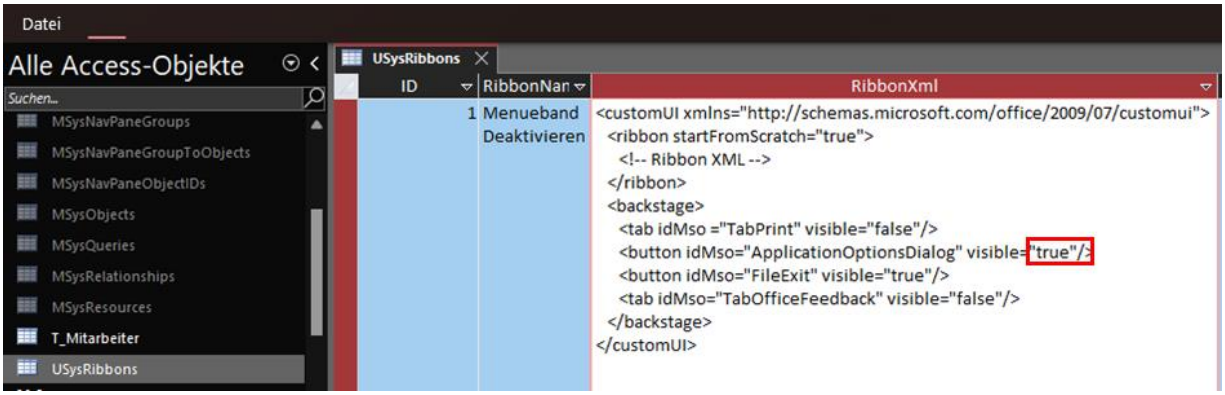


Bild 146: Benutzerdefinierte Menüband-Konfiguration – Erneute Aktivierung von „Optionen“ unter „Datei“ im Menüband

Nach dem Neustart sind die „*Optionen*“ unter „*Datei*“ wieder aufrufbar:

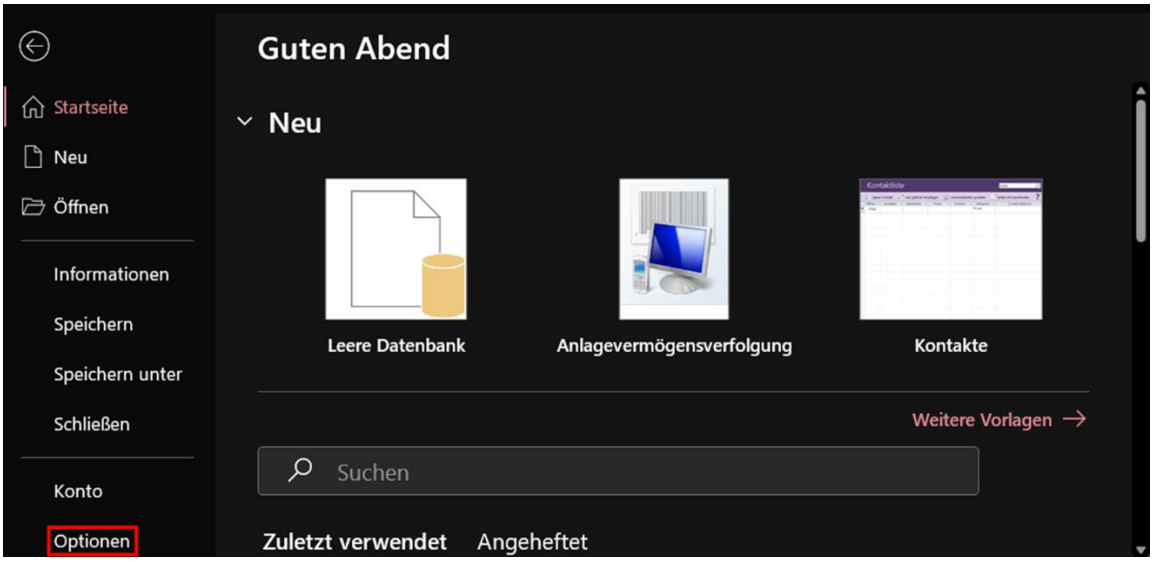


Bild 147: Benutzerdefinierte Menüband-Konfiguration – Erneut aktivierte „Optionen“ unter „Datei“ im Menüband

Jetzt werden die Konfigurationen unter „*Datei → Optionen → Aktuelle Datenbank*“ vorgenommen, um Kontextmenü, Tastenkürzel der Tastatur, den Navigationsbereich und Co. zu deaktivieren:

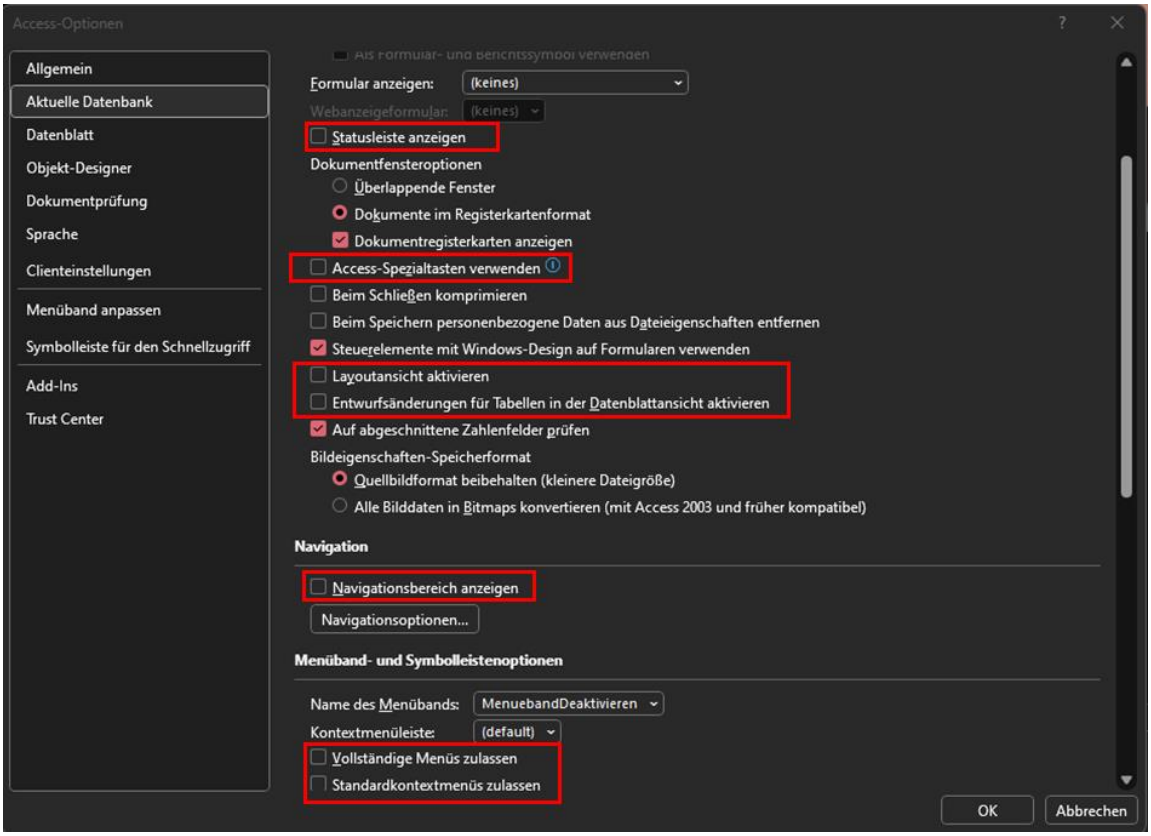


Bild 148: Benutzerdefinierte Menüband-Konfiguration – Getätigte Konfigurationen unter „Datei → Optionen → Aktuelle Datenbank“

Damit der Options-Button beim nächsten Neustart unter „*Datei*“ wieder ausgeblendet wird, wird in der Relation „*USysRibbons*“ das Button-Element mit der Attributausprägung „*ApplicationOptionsDialog*“ im Attribut „*idMso*“ wieder auf „*false*“ gesetzt und anschließend gespeichert:

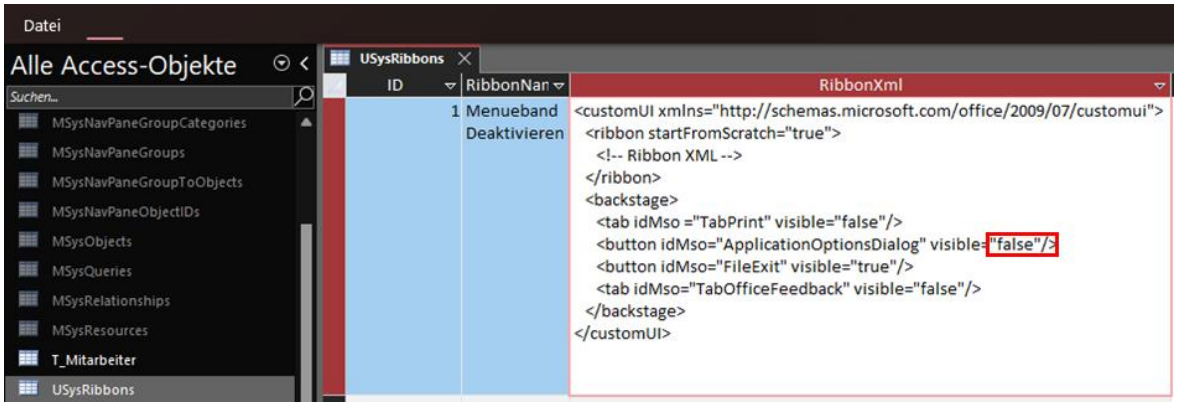


Bild 149: Benutzerdefinierte Menüband-Konfiguration – Erneute Deaktivierung der „Optionen“ unter „Datei“

Nach dem Neustart sind das Menüband und die Auswahlmöglichkeiten unter „*Datei*“ nun vollständig ausgeblendet:

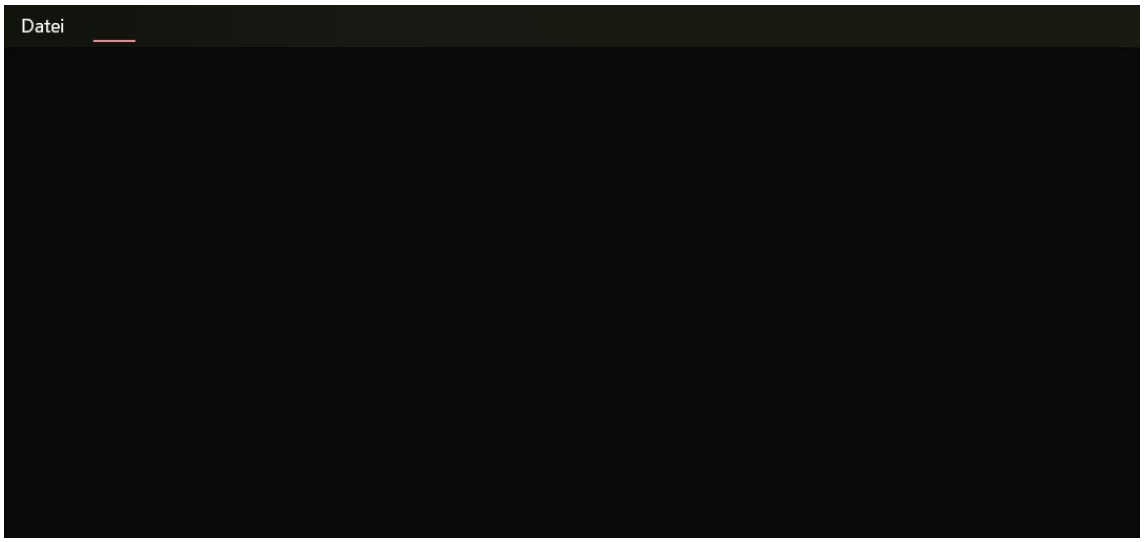


Bild 150: Benutzerdefinierte Menüband-Konfiguration – Bestätigung der erneuten Aktivierung

Die Access-Datei kann nun geschlossen werden und über einen SQL Client, wie DbVisualizer, geöffnet werden. Hier wird die zuvor manuell angelegte Relation „*USysRibbons*“ aufgerufen und der Attributwert des „*ribbon*“-Element-Attributs „*startFromScratch*“ auf „*false*“ geändert sowie das gesamte „*backstage*“-Element inklusive Kind-Elemente auskommentiert:

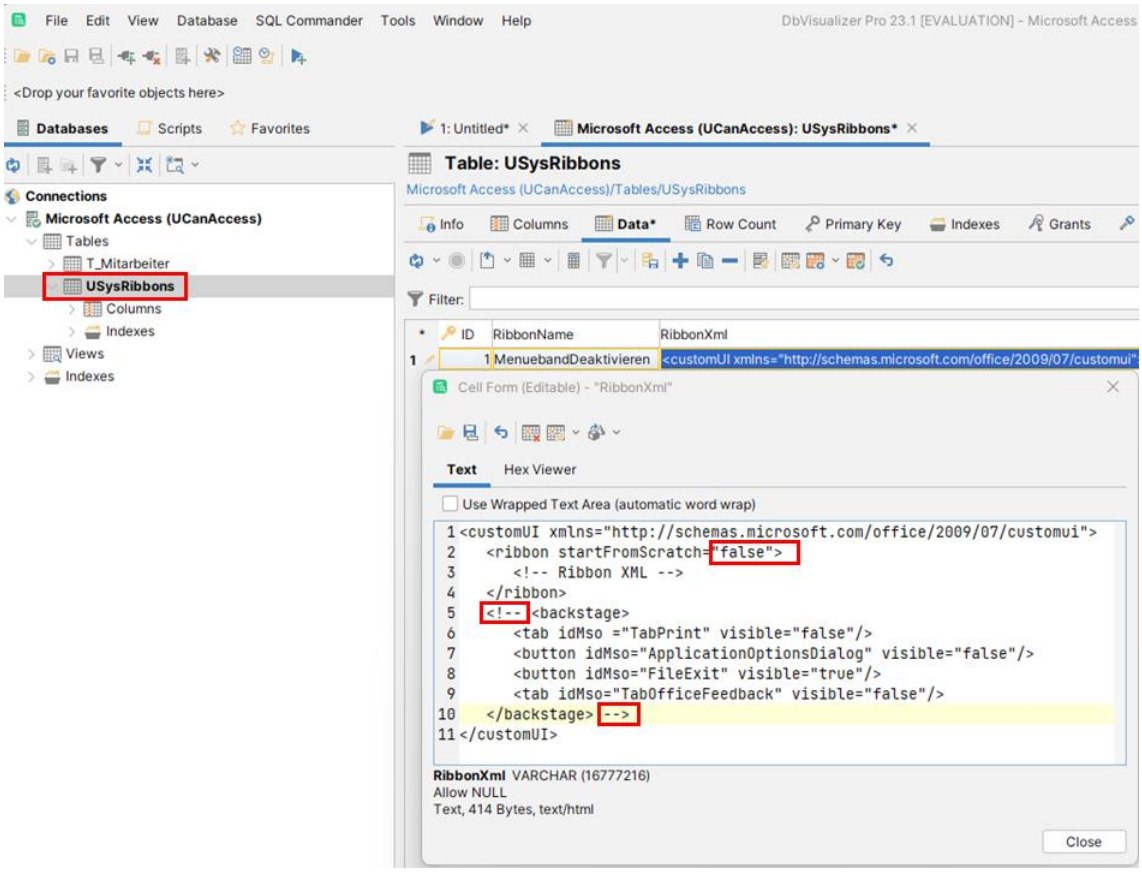


Bild 151: Benutzerdefinierte Menüband-Konfiguration – Deaktivieren des benutzerdefinierten Menübands via DbVisualizer

Nach dem Speichern kann die Access-Datei wieder geöffnet werden. Das Menüband wird zwar nicht wieder eingeblendet, aber die Optionen können unter „*Datei → Datenschutzoptionen*“ wieder aufgerufen werden:

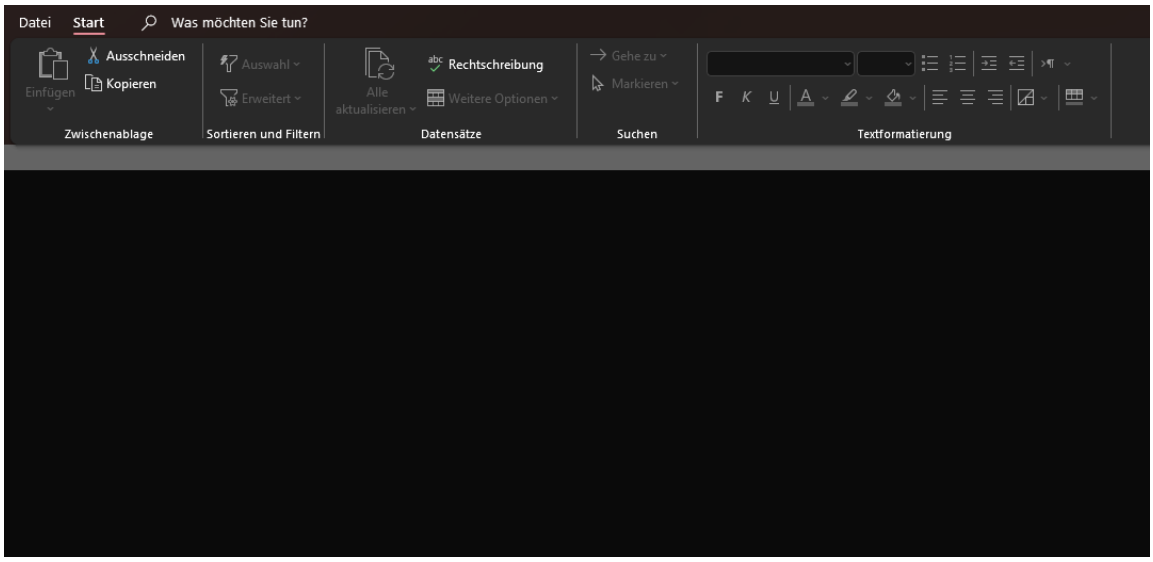


Bild 152: Benutzerdefinierte Menüband-Konfiguration – Bestätigen der Deaktivierung nach der Manipulation via DbVisualizer 1/2

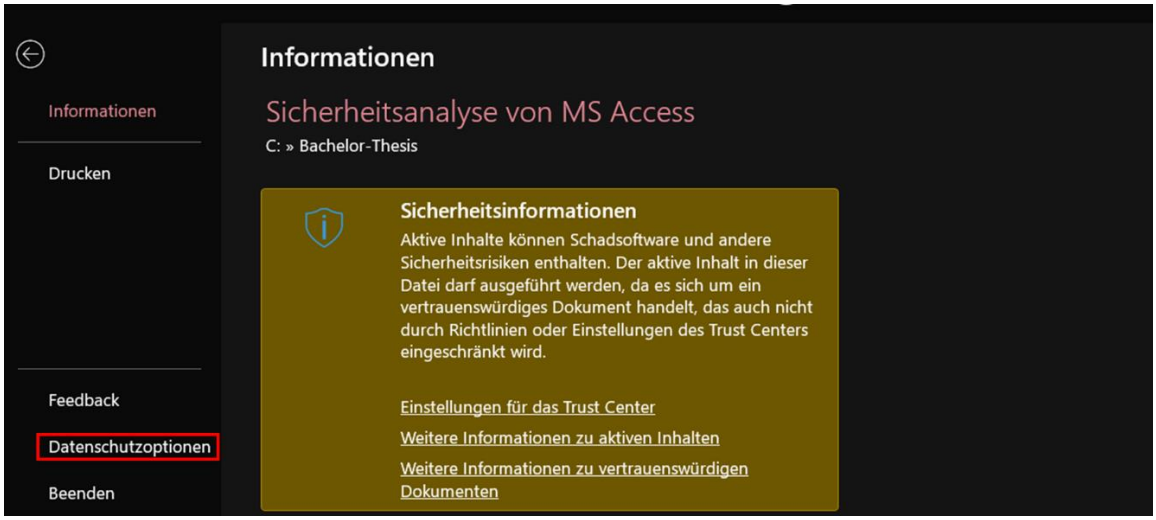


Bild 153: Benutzerdefinierte Menüband-Konfiguration – Die Optionen sind nach Deaktivieren via DbVisualizer unter „Datei → Datenschutzoptionen“ aufrufbar 2/2

Die zuvor unter „*Datei → Optionen → Aktuelle Datenbank*“ vorgenommenen Einstellungen, wie das Ausblenden des Navigationsbereichs und das Deaktivieren der Tastenkürzel der Tastatur, können rückgängig gemacht werden:

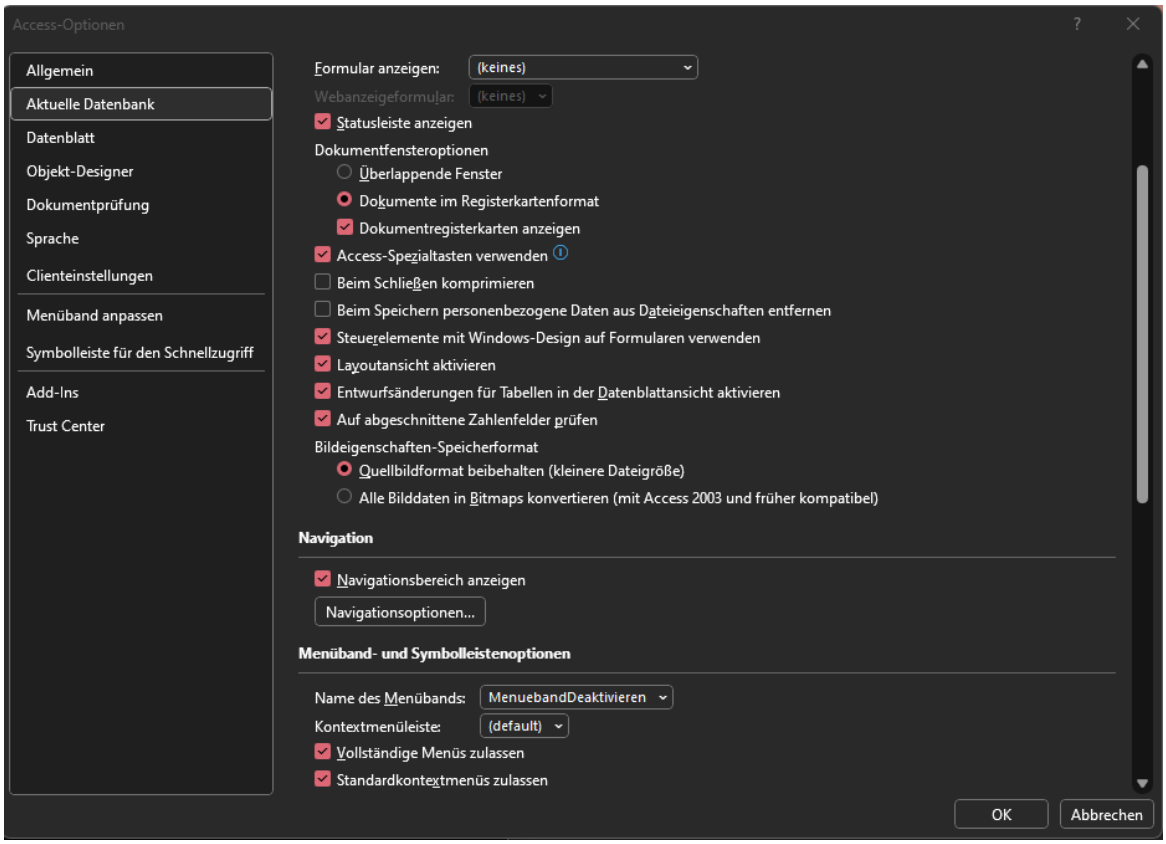


Bild 154: Benutzerdefinierte Menüband-Konfiguration – Wiederherstellen der Standardkonfigurationen unter „Datei → Optionen → Aktuelle Datenbank“

Nach dem Neustart der Access-Datei ist der Vollzugriff wiederhergestellt. Es sind alle Kopfreiter des Standard-Menübands wieder sichtbar und die Datei kann beliebig manipuliert werden:

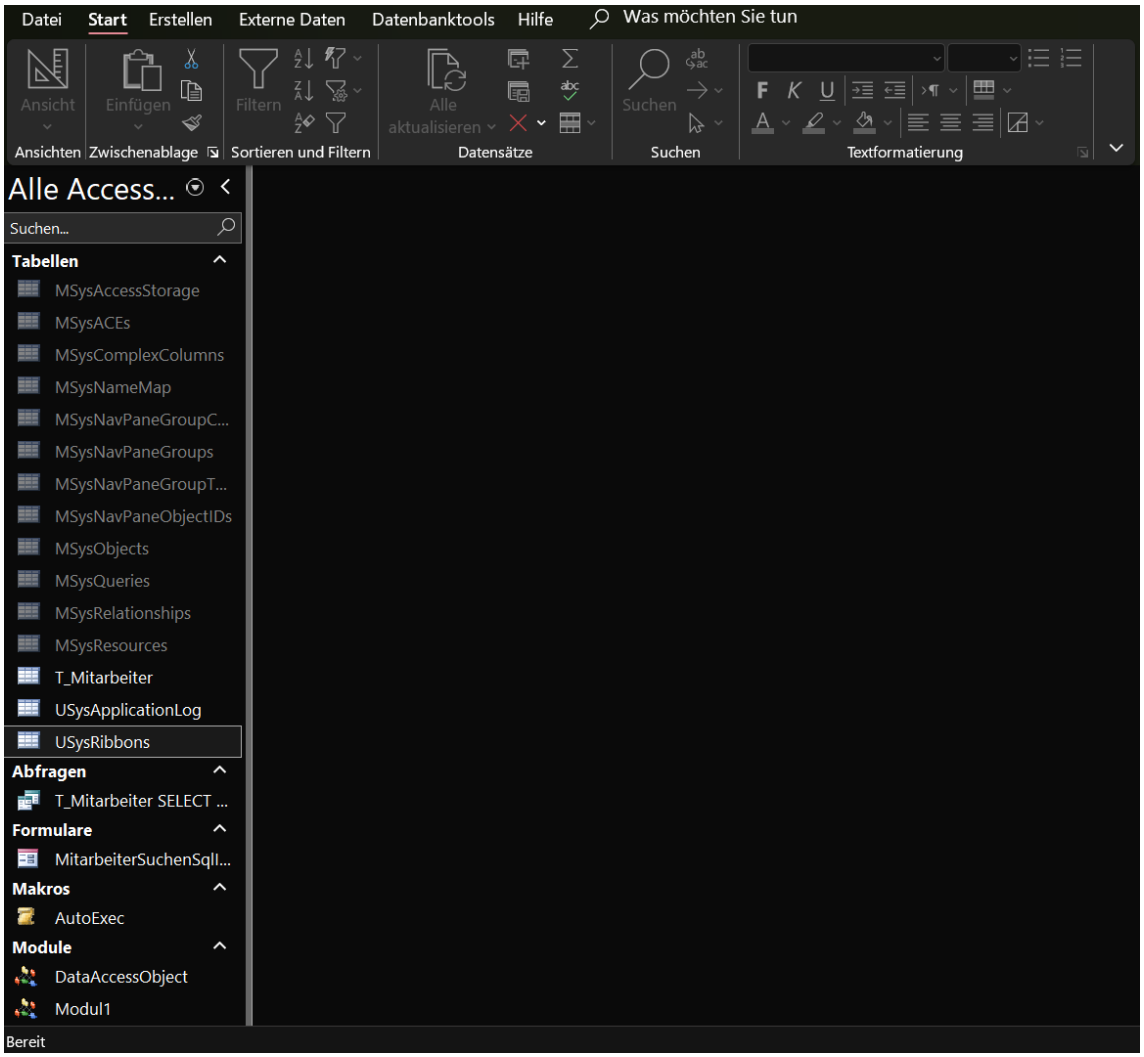


Bild 155: Benutzerdefinierte Menüband-Konfiguration – Menüband nach Wiederherstellen der Standard-konfiguration 1/2

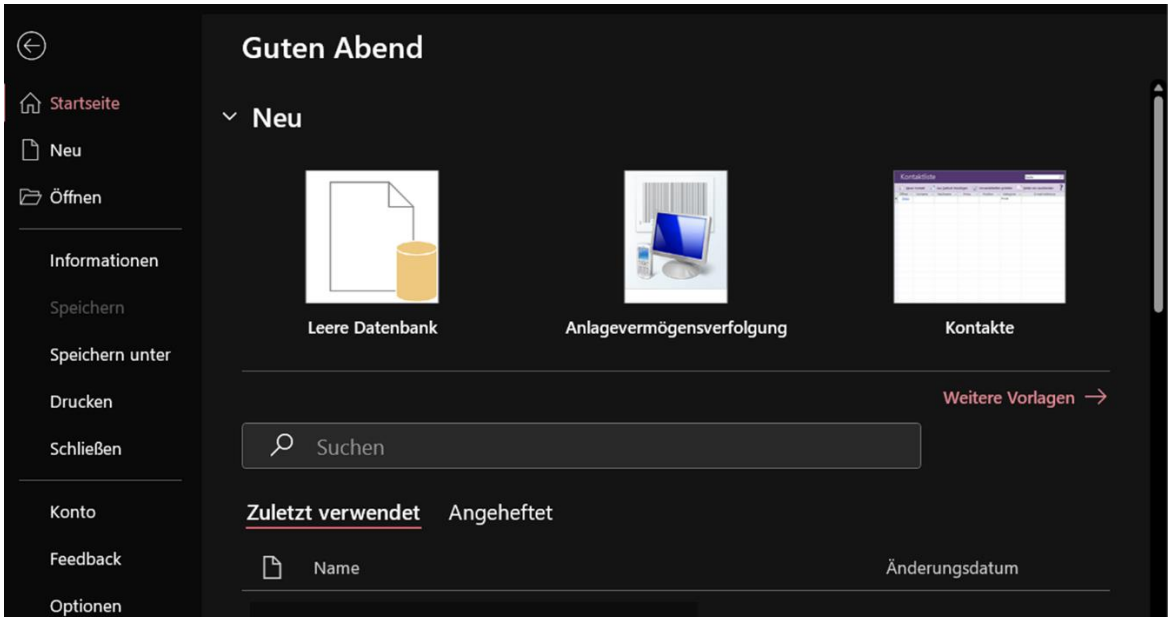


Bild 156: Benutzerdefinierte Menüband-Konfiguration – Menüband nach Wiederherstellen der Standard-konfiguration („Datei → Optionen“) 2/2

Anlage 5: Digitale Signaturen (Access)

Ein Test mit einem selbstsignierten Test-Zertifikat unter Verwendung des PowerShell-Befehls „*New-SelfSignedCertificate*“ hat gezeigt, dass trotz nachträglicher Manipulation, Speichern, Schließen und erneutem Öffnen der Datei die VBA-Code-Signatur aktiv bleibt und es keinen Hinweis auf die nachträgliche Manipulation der Logik gibt.

Im ersten Schritt wird mittels PowerShell-Befehls „*New-SelfSignedCertificate*“ ein selbstsigniertes Zertifikat erstellt:

```
PS C:\Users\NM> $certname = "Bachelor-Thesis"
PS C:\Users\NM> $cert = New-SelfSignedCertificate -DNSName "www.BachelorThesis.de" -CertStoreLocation Cert:\CurrentUse
r\My -Type CodeSigningCert -Subject "Sicherheitsanalyse von Microsoft Access"
```

Bild 157: Erstellen eines selbstsignierten Zertifikats zum Test der digitalen Signatur-Funktionalitäten in Access

Das erstellte Zertifikat wird dann in Access über „*Datei → Speichern unter → Packen und signieren*“ zum Signieren der Datei ausgewählt:

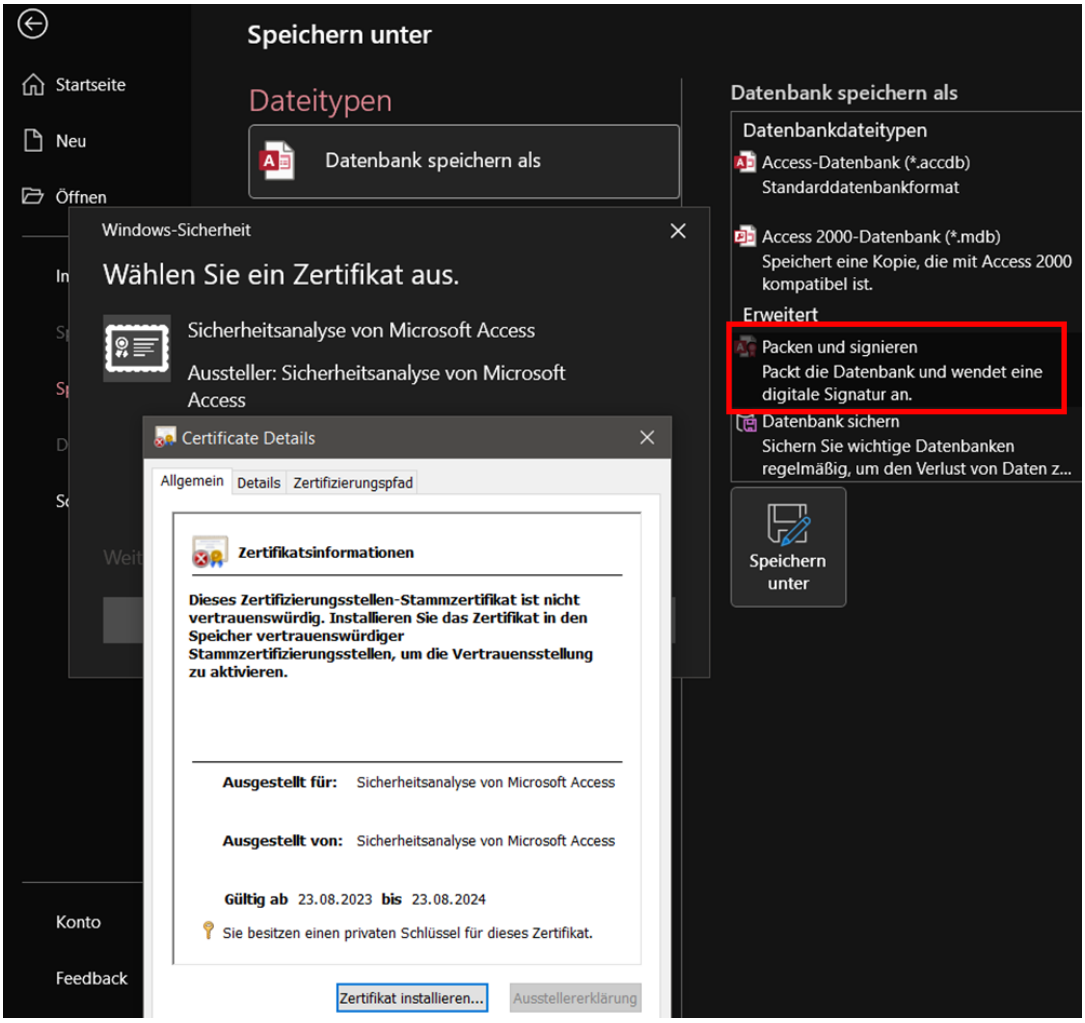


Bild 158: Packen und signieren der Access-Datei mit dem selbstsignierten Zertifikat zum Test der digitalen Signatur-Funktionalitäten in Access

Mit dem selbstsignierten Zertifikat kann auch das VBA-Projekt unter *„Extras → Digitale Signatur…“* geschützt werden:

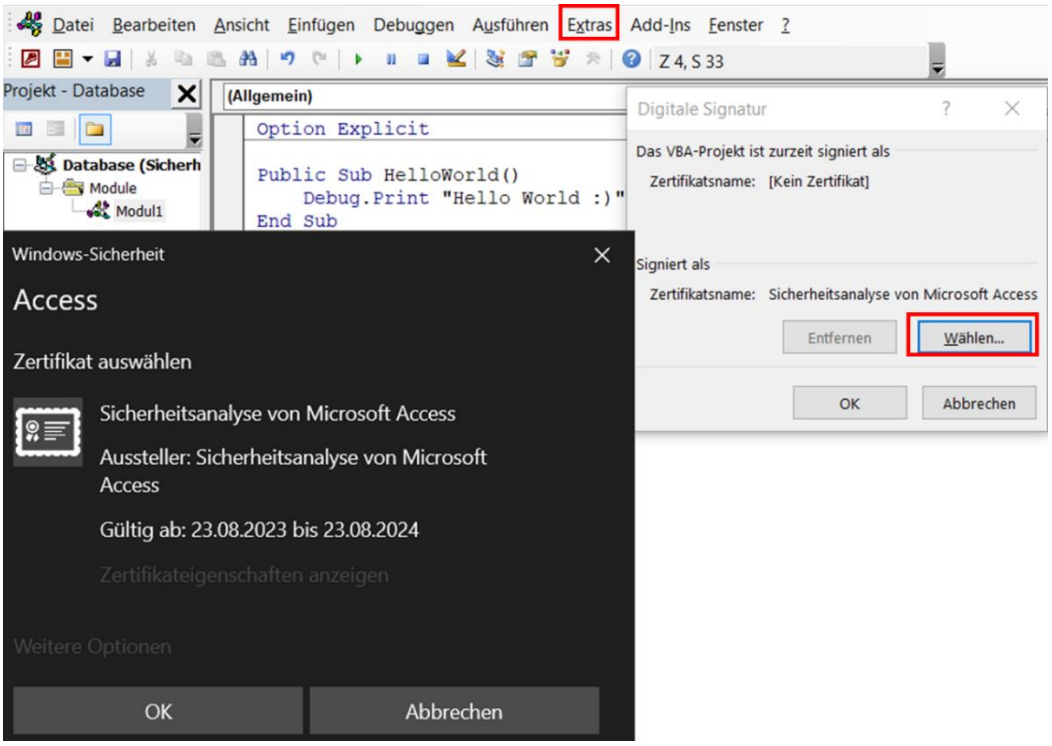


Bild 159:Signieren des VBA-Projekts mit dem selbstsignierten Zertifikat zum Test der digitalen Signatur-Funktionalitäten in Access

Weder das Signieren des VBA-Projekts noch das Packen und Signieren der Datei bieten Sicherheit. Trotz nachträglicher Manipulation, Speichern, Schließen und erneutem Öffnen der Datei bleibt die Signatur des VBA-Projekts aktiv und es gibt keinen Hinweis auf die nachträgliche Manipulation der Logik. Auch bei aktiviertem Projektschutz inklusive Passwort kann die hinterlegte digitale Signatur über den Button „Entfernen“ oder „Wählen…“ manipuliert werden:

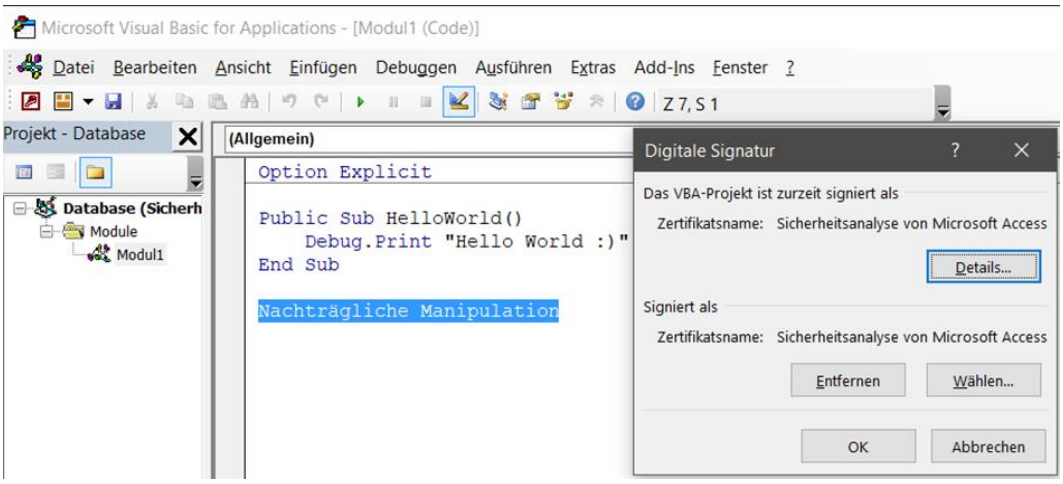


Bild 160: Trotz nachträglicher Manipulation, Speichern, Schließen und erneutem Öffnen der Datei bleibt die Signatur des VBA-Projekts aktiv und es gibt keinen Hinweis auf die nachträgliche Manipulation der Logik

Anlage 6: Tabelle ausblenden (Access)

Auch wenn eine Tabelle ausgeblendet wurde, ist sie noch über den DbVisualizer zugänglich:

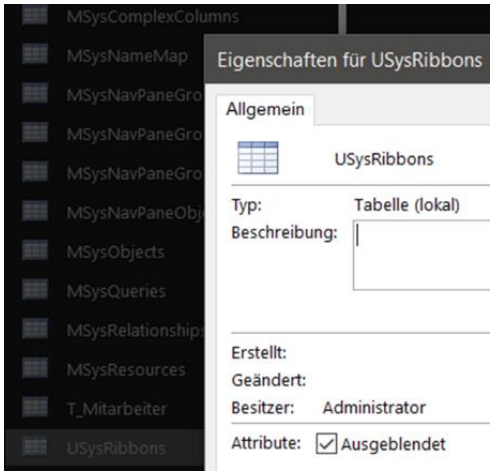


Bild 161: Access-Tabelle über Tabellen-Eigenschaften ausblenden

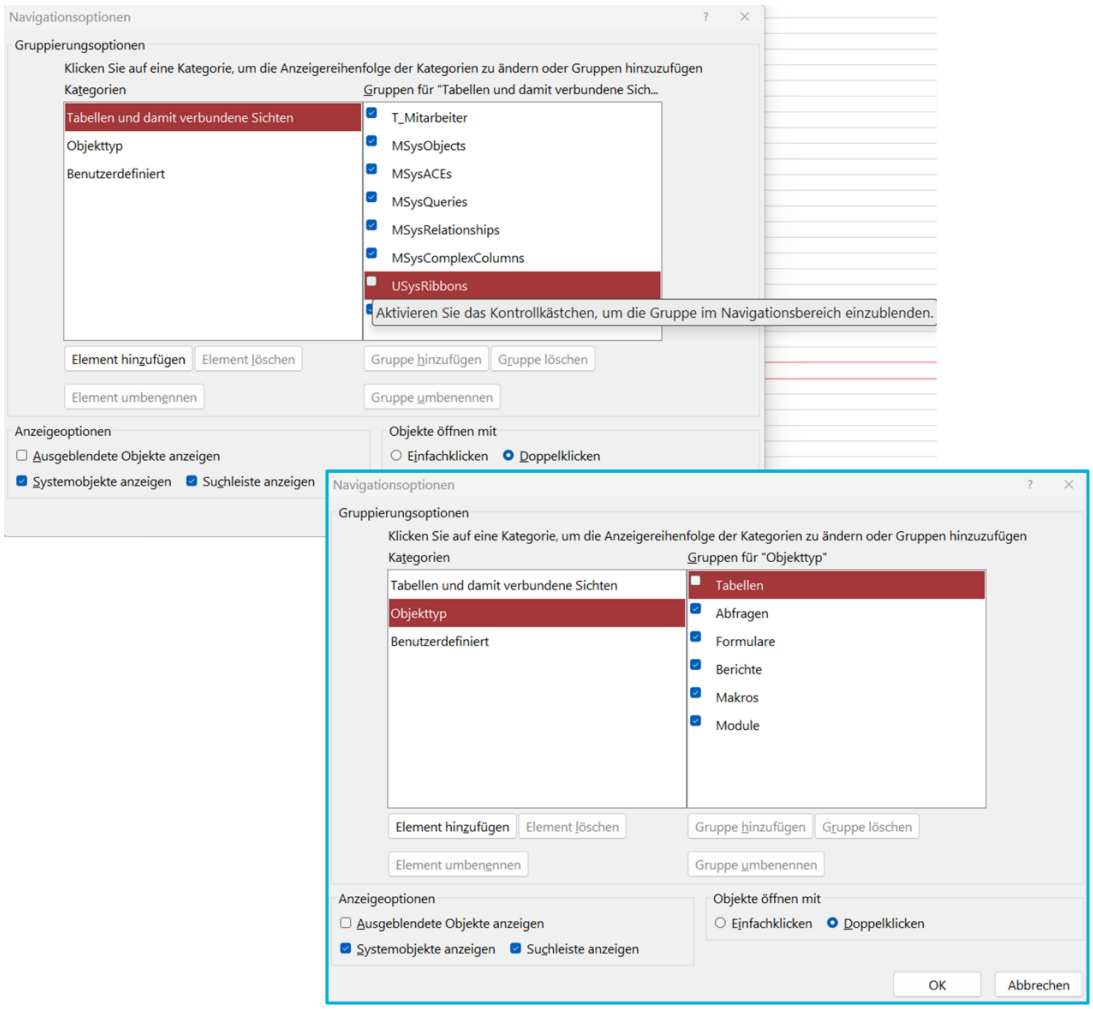


Bild 162: Access-Tabelle(n) über die Navigationsoptionen ausblenden (Links: Ausblenden der Relation „USysRibbons"; Rechts: Ausblenden aller Tabellenobjekte)

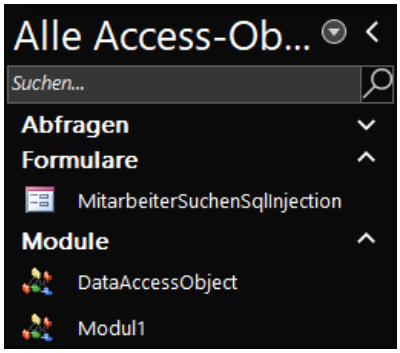


Bild 163: Access-Tabelle(n) über „Navigationsoptionen → Anzeigeoptionen → Ausgeblendete Objekte anzeigen" ausblenden

Bei allen zuvor skizzierten Vorgehensweisen zum Ausblenden einer Relation ist die Relation „`USysRibbons`" über den DbVisualizer weiterhin zugänglich und manipulierbar. Die Access-Systemrelationen können nicht über DbVisualizer und den UCanAccess-Treiber eingesehen werden:

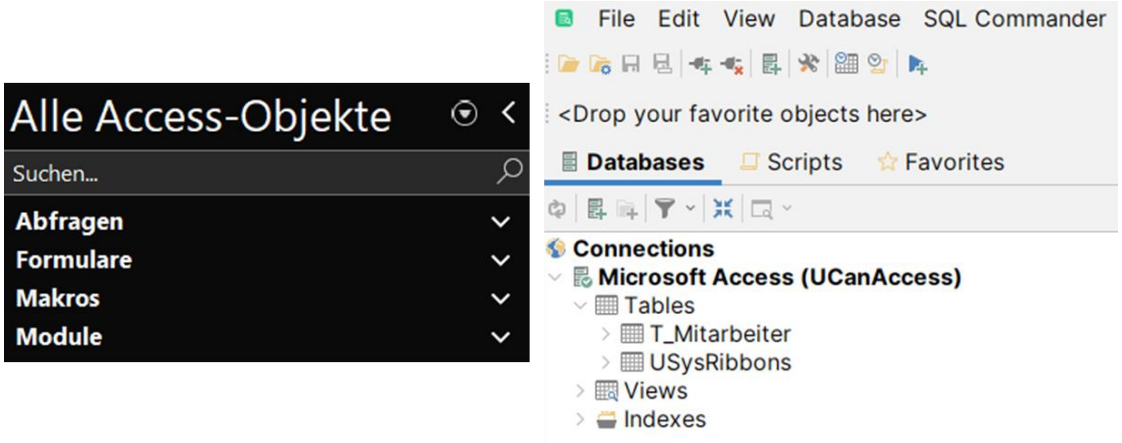


Bild 164: Links: Ausblenden der Tabellenobjekte über die Access-Oberfläche; Rechts: Trotz des Ausblendens der Tabellenobjekte auf verschiedene Arten über die Access-Oberfläche, ist der Vollzugriff über DbVisualizer weiterhin möglich. Die Access-Systemrelationen sind nicht einsehbar

Anlage 7: Entfernen von VBA-Projekt-Passwortschutz in Excel (Access Excel)

Ausgangslange: .xlsm-Frontend (Excel), das sich über VBA mit dem Access-.accdb-Backend (Access) verbindet. Das VBA-Projekt des Frontends ist durch ein Passwort geschützt und enthält im Quellcode das Passwort zum Entschlüsseln des Backends. Die Excel-Datei selbst ist nicht verschlüsselt:

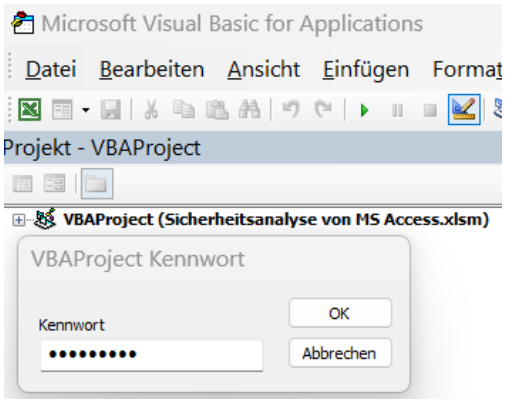


Bild 165: Excel – Aktivierter VBA-Projekt-Passwortschutz

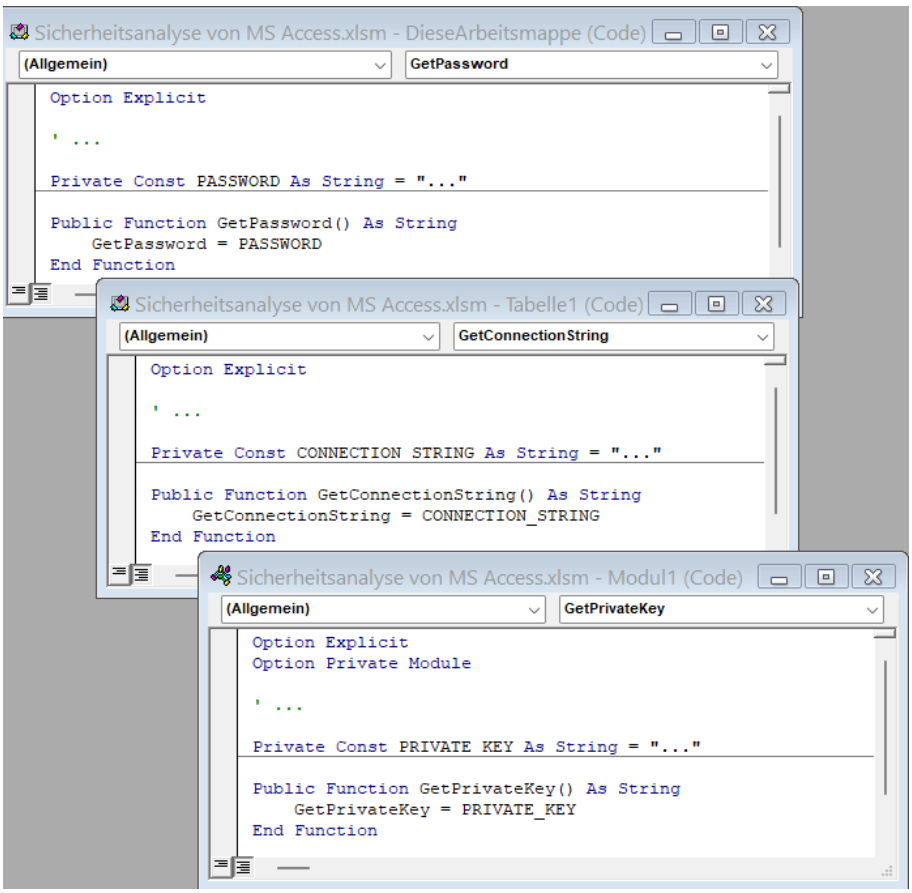


Bild 166: Excel – Nach Passworteingabe besteht Vollzugriff auf das VBA-Projekt. Vertrauliche Informationen, wie das Passwort zu einem verschlüsseltem Access-Backend, können eingesehen oder Schadcode kann hinzugefügt werden

Die Datei wird geschlossen und die Dateierweiterung im Dateinamen von .xlsm zu .zip geändert:

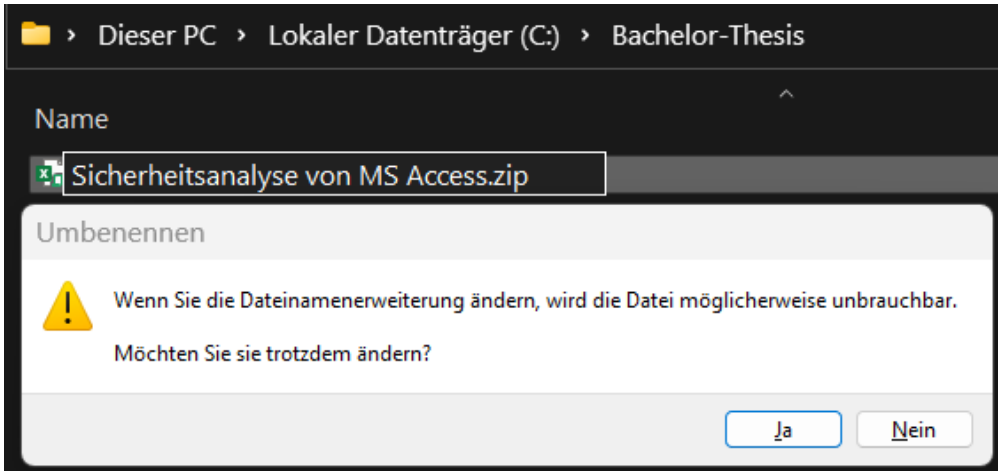


Bild 167: Excel – Ändern der Dateierweiterung im Dateinamen von .xlsm auf .zip

Das Archiv wird in WinRAR geöffnet:

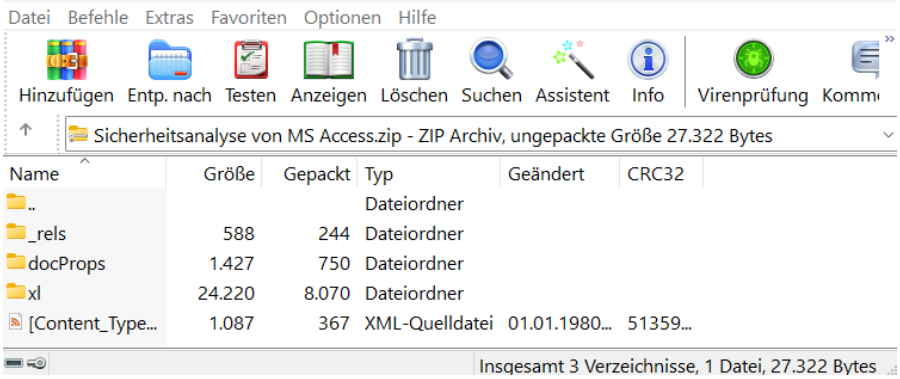


Bild 168: Excel – Inhalt des entpackten .zip-Archivs

Die Datei unter „`xl\vbaProject.bin`“ wird mit WinRAR und dem Standard-Hexadezimal-Editor (Visual Studio Code, *siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software“*) angepasst:

Bild 169: Excel – Änderungen an der Datei "xl\vbaProject.bin" im Hexadezimal-Editor, um den VBA-Projekt-Passwortschutz zu entfernen (Oben: Vor Anpassung; Unten: Nach Anpassung)

Nach dem Speichern der Manipulation wird die Dateierweiterung im Dateinamen des über WinRAR und Visual Studio Code manipulierten Archivs wieder von .zip auf .xlsm geändert:

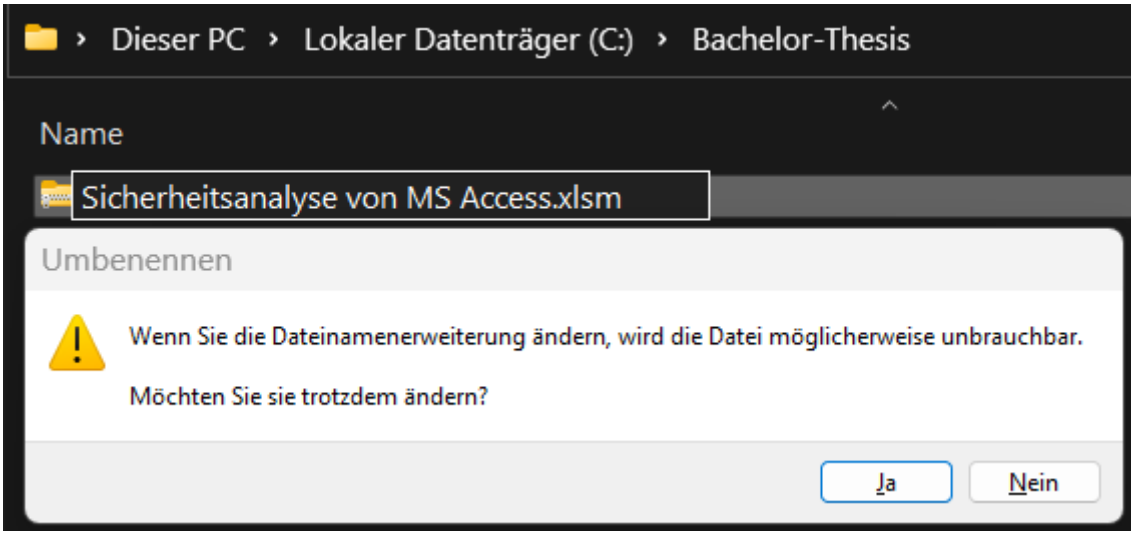


Bild 170: Excel – Rückgängig machen der Änderung in der Dateierweiterung (von .zip auf .xlsm)

Die Datei kann problemlos in Excel geöffnet werden:

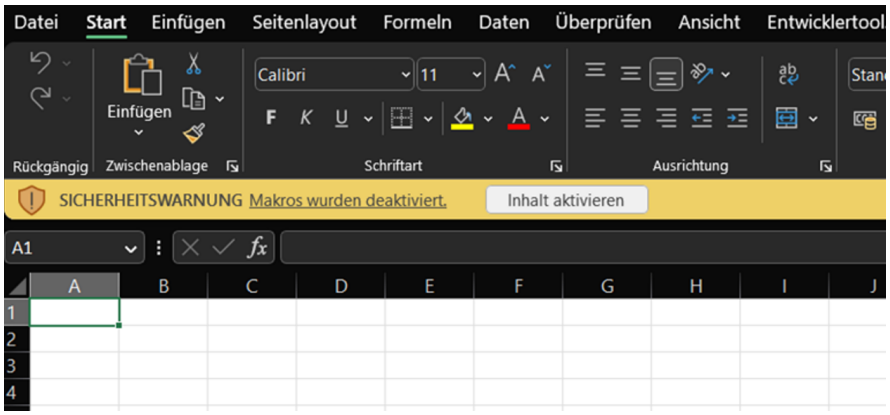


Bild 171: Excel – Öffnen der manipulierten .xslm-Datei

Beim Klick auf „*Entwicklertools → Visual Basic*“ erscheint folgender Dialog:

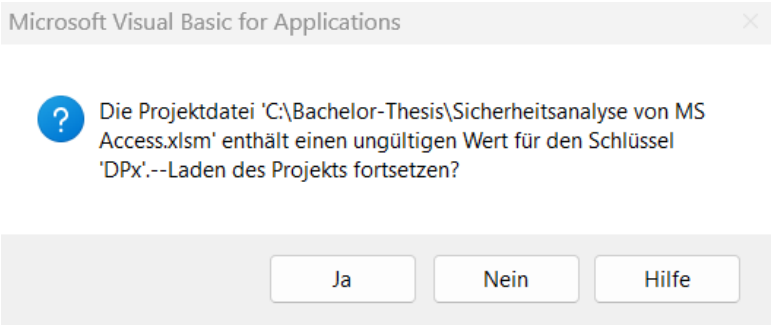


Bild 172: Excel – Fehlermeldung nach den Änderungen an der Datei „xl\vbaProject.bin“, um den Passwortschutz des VBA-Projekts zu entfernen

Nach Bestätigen des Dialogs erscheint folgende Fehlermeldung:

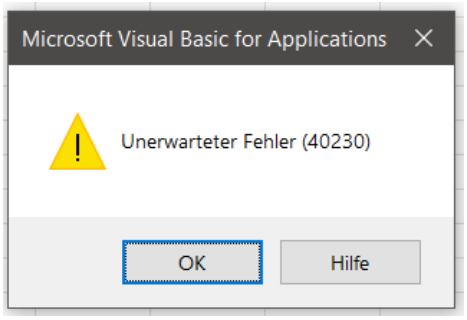


Bild 173: Excel – Fehlermeldung nach dem Öffnen des VBA-Editors der manipulierten Datei

Nach mehrmaliger Bestätigung der Fehlermeldung öffnet sich der VBA-Editor. Unter „*Extras → Eigenschaften von … → Kopfreiter Schutz*“ wird die Option „Projekt für die Anzeige sperren“ deaktiviert. Zu beachten ist, dass das zuvor unter „Kennwort“ vergebene Passwort nicht mehr existiert:

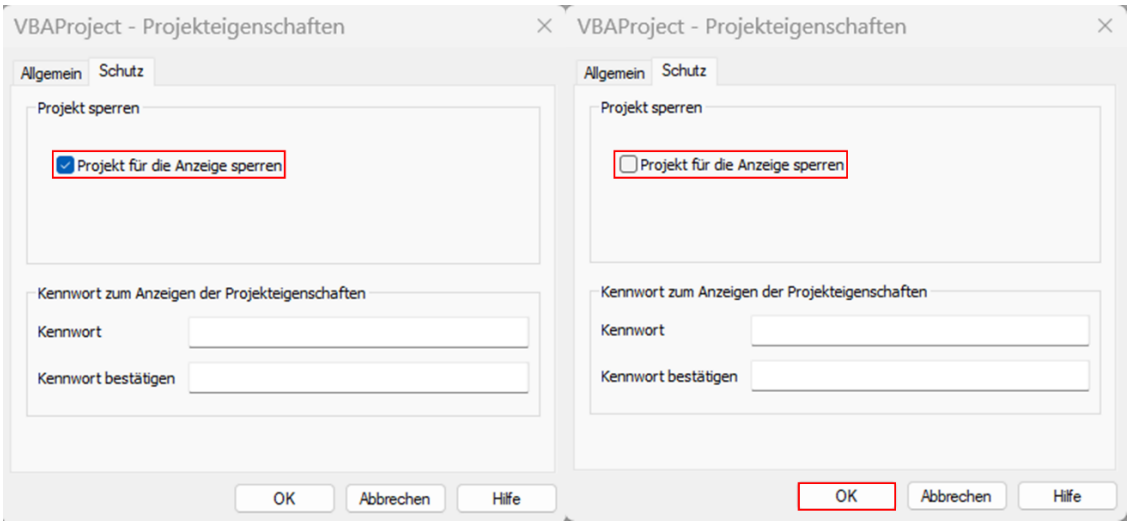


Bild 174: Excel – Ergebnis der erfolgreichen Änderung an der Datei „xl\vbaProject.bin“, um den Passwortschutz des VBA-Projekts zu entfernen (Links: Vor Anpassung; Rechts: Nach Anpassung)

Es wird eine Kopie der Datei gespeichert und geöffnet. Der Passwortschutz des VBA-Projekts ist nun entfernt und der Zugriff auf den Quellcode möglich:

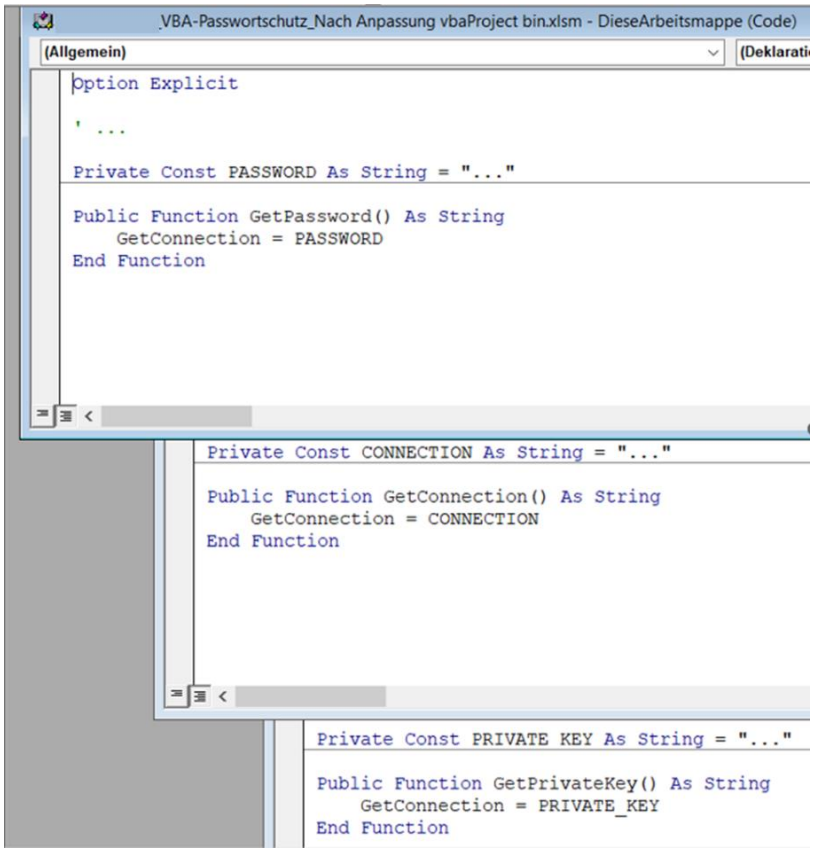


Bild 175: Excel – Bestätigung des Vollzugriffs auf das VBA-Projekt

Wird die Änderung an der Datei „`xl/vbaProject.bin`“ nicht in einem Hexadezimal-Editor, sondern in einem Text-Editor, wie dem Default Text-Editor von Visual Studio Code, geöffnet, kann es anders als bei .xml-Dateien zu Problemen kommen, die zum automatischen Löschen des VBA-Quellcodes führen:

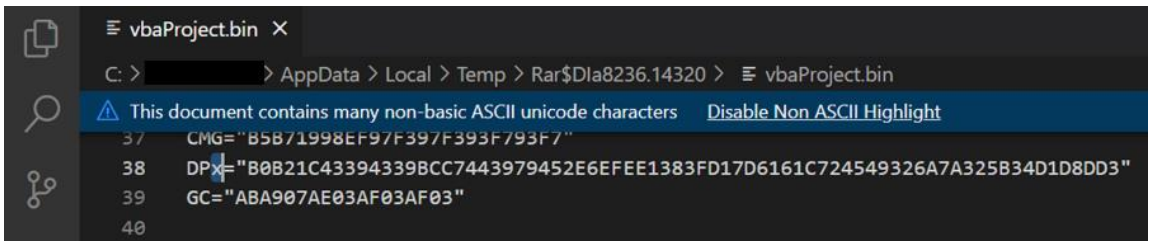


Bild 176: Excel – Änderungen an der Datei „xl\vbaProject.bin“ im Text-Editor, um den VBA-Projekt-Passwortschutz zu entfernen

In diesem Fall erscheint beim Öffnen der angepassten Datei der aus *Kapitel „4.2 Schnittstellenanalyse (Access Excel)“* bekannte Dialog:

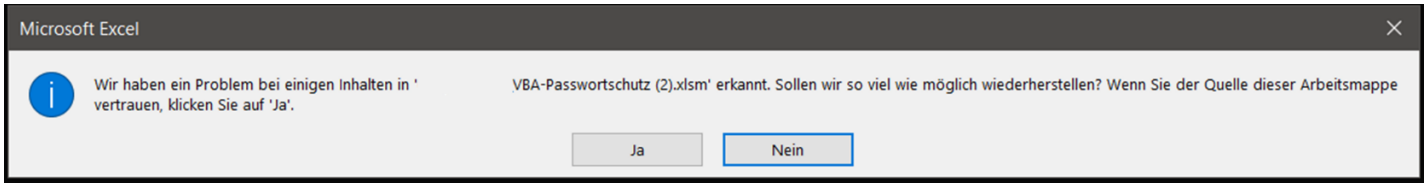


Bild 177: Excel – Dialog nach Öffnen des VBA-Editors der über einen Text-Editor manipulierten Datei

Nach dem Bestätigen des Dialogs werden Nutzende über Probleme mit der manipulierten Datei und über die Löschung des Quellcodes informiert:

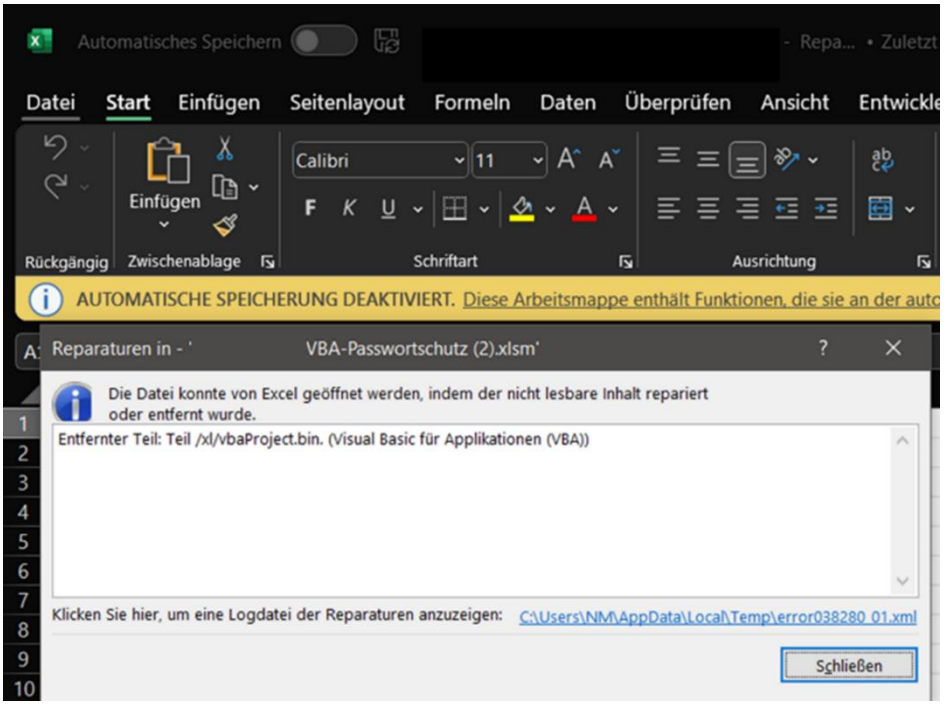


Bild 178: Excel – Warnmeldung nach Öffnen des VBA-Editors der über einen Text-Editor manipulierten Datei

Das Öffnen des VBA-Editors bestätigt die Löschung:

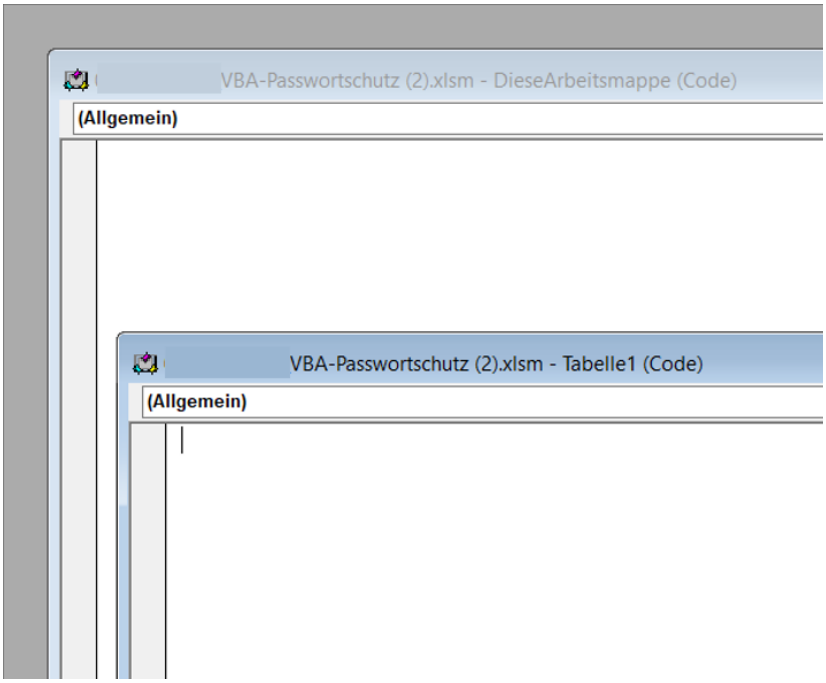


Bild 179: Excel – Bestätigung des gelöschten VBA-Codes

Die Excel-Schnittstelle weist ein gravierendes Sicherheitsrisiko auf, über das Angreifende im schlimmsten Fall an das Passwort zum Entschlüsseln des Access-Backends oder an Anmeldedaten für ein Client-Server-DBS wie Azure SQL oder den SQL Server gelangen können. Die Excel-Datei kann auch unbemerkt mit Schadcode erweitert werden, der dann auf allen Endgeräten der Nutzenden ausgeführt wird.

Anlage 8: Entfernen von Blattschutz in Excel (Access Excel)

Ausgangslange: .xlsx oder .xlsm-Frontend das vertrauliche Informationen, wie personenbezogene Daten oder Verbindungsinformationen, in ausgeblendeten Zellen speichert. Die Struktur und der Inhalt eines Worksheets sind mit einem Blattschutz inklusive Passwort geschützt. Die Excel-Datei selbst ist nicht verschlüsselt:

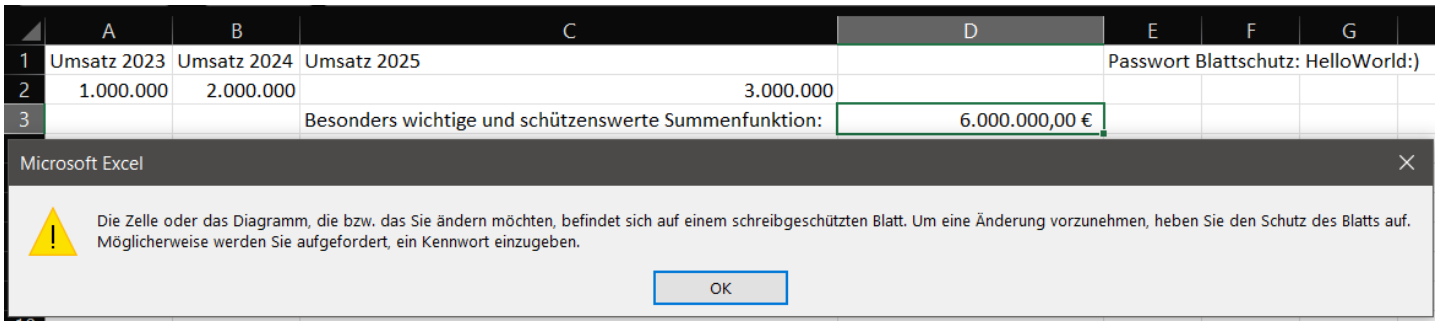


Bild 180: Excel – Aktivierter Blattschutz

Die Dateierweiterung wird von .xlsx oder .xlsm auf .zip geändert und das Archiv, wie im *Kapitel „Anlage 7: Entfernen von VBA-Projekt-Passwortschutz in Excel (Access Excel)“* beschrieben, mit WinRAR geöffnet:

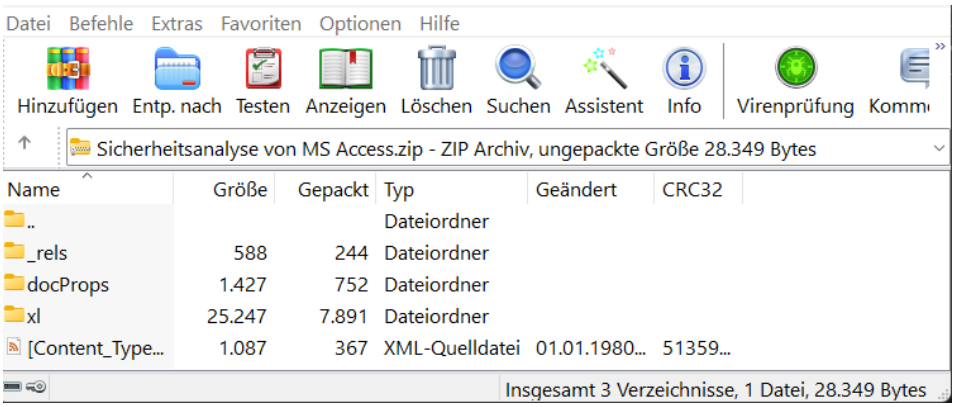


Bild 181: Excel – Inhalt des entpackten .zip-Archivs

Die zum Worksheet mit Passwortschutz gehörende Datei unter *„xl\worksheets\sheet1.xml“* wird über WinRAR und den Standard-Text-Editor (Visual Studio Code (*siehe Kapitel „2.3 Informationen zu MS, Access & weitere eingesetzte Software“*)) geändert. Es wird das gesamte Element gelöscht:

```
<sheetProtection algorithmName="SHA-512" hashVa-
lue="JBDhMA8zpcPhS1iGc39YWbO5LxkZ0/r/I3JSFyb+aIWmEfnZMJE0Nt5WdjKyaAvBG
lDA0jyGJpjusvlYEyDyPQ==" saltValue="oNKnZ90e3Oa4jxXMLTPH3A=="
spinCount="100000" sheet="1" objects="1" scenarios="1"/>
```

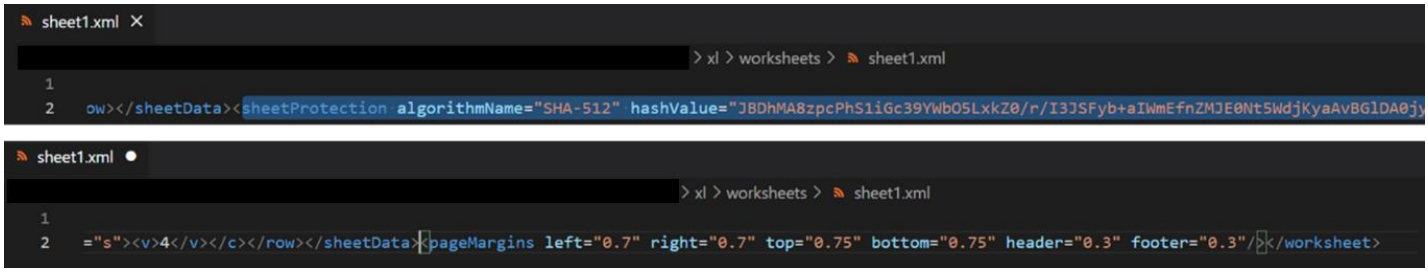


Bild 182: Excel – Änderungen an der Datei „xl\worksheets\sheet1.xml“ im Hexadezimal-Editor, um den Blattschutz zu entfernen (Oben: Vor Anpassung; Unten: Nach Anpassung)

Nach dem Speichern der Manipulation wird die Dateierweiterung im Dateinamen des über WinRAR und Visual Studio Code manipulierten Archivs wieder von .zip auf .xlsx oder .xlsm geändert und die Datei geöffnet. Der Blattschutz ist nun entfernt. Die Manipulation der Formeln oder das Einblenden von Zellen mit sensiblen Daten ist problemlos möglich:

A	B	C	D
Umsatz 2023	Umsatz 2024	Umsatz 2025	
1.000.000	2.000.000	3.000.000	
		Besonders wichtige und schützenswerte Summenfunktion:	=A2+B2+C2

Bild 183: Excel – Änderungen an der Datei „xl\worksheets\sheet1.xml“ im Hexadezimal-Editor haben den Blattschutz erfolgreich entfernt

Anlage 9: Entfernen von VBA-Projekt-Passwortschutz in Access (nur .accdb)

Nachfolgende Ausführungen gelten nicht für.accde-Dateien, da hier nicht auf das VBA-Projekt mit Quellcode und die Entwurfsansicht bei Formularen zugegriffen werden kann. Anders als bei Excel- und Word-Dateien handelt es sich bei Access-Dateien nicht um ein .zip-Archiv. Daher führt die Anpassung der Dateierweiterung .zip zu einem Fehler. Das VBA-Projekt der .accdb-Datei ist durch ein Passwort geschützt, die Access-Datei selbst ist nicht verschlüsselt:

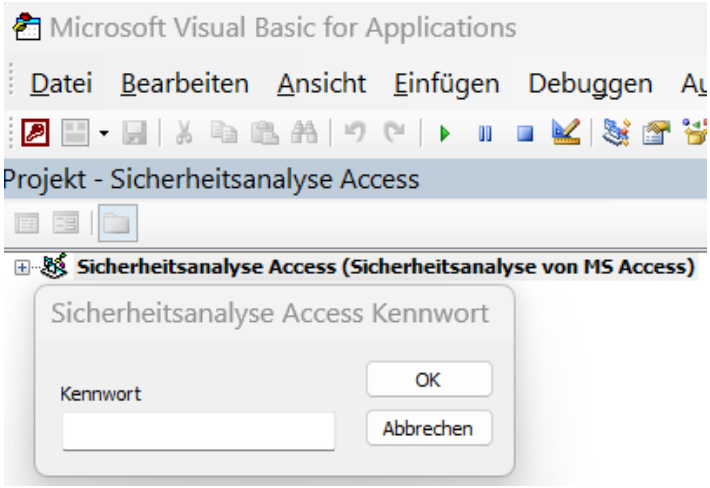


Bild 184: Aktivierter VBA-Projekt-Passwortschutz

Anstelle eine Access-Datei in ein .zip-Archiv umzubenennen, kann sie ohne Umwege direkt im Hexadezimal-Editor geöffnet und die Manipulationen vorgenommen werden. Äquivalent zu Excel-Dateien (*siehe Kapitel „Anlage 7: Entfernen von VBA-Projekt-Passwortschutz in Excel (Access Excel)“*) wird wieder nach „*DPB*“ gesucht und das letzte Zeichen „*B*“ durch ein beliebiges Zeichen ersetzt. Im Gegensatz zu Excel gibt es nun drei Suchtreffer, die einzeln nacheinander durchprobiert werden:

Bild 185: Änderungen an der .accdb-Datei im Hexadezimal-Editor, um den VBA-Projekt-Passwortschutz zu entfernen (Oben: Vor Anpassung; Unten: Nach Anpassung)

Erst nach Ändern des dritten Suchtreffers erscheint der aus der Excel-Manipulation bekannte Dialog. Nach Bestätigen des Dialogs und Öffnen des VBA-Editors erscheint eine Fehlermeldung:

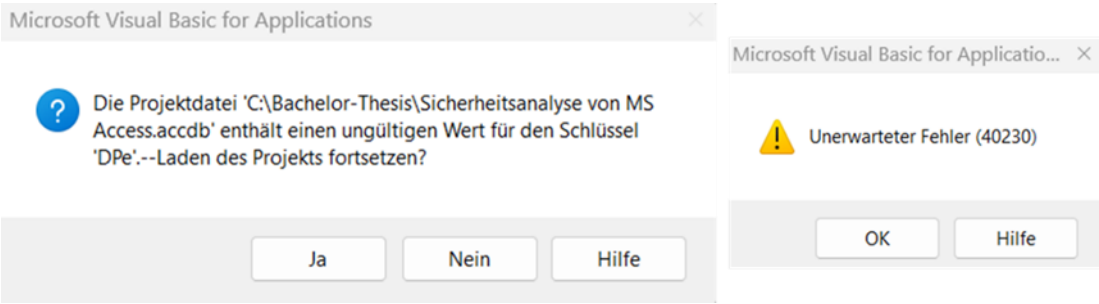


Bild 186: Fehlermeldung nach den Änderungen an der .accdb-Datei, um den Passwortschutz des VBA-Projekts zu entfernen

Nach mehrmaligem Bestätigen öffnet sich der VBA-Editor. Unter „*Extras → Eigenschaften von … → Kopfreiter Schutz*“ kann der Projektschutz deaktiviert werden, da das Passwort durch die Manipulation entfernt wurde. Der Projektschutz wird deaktiviert:

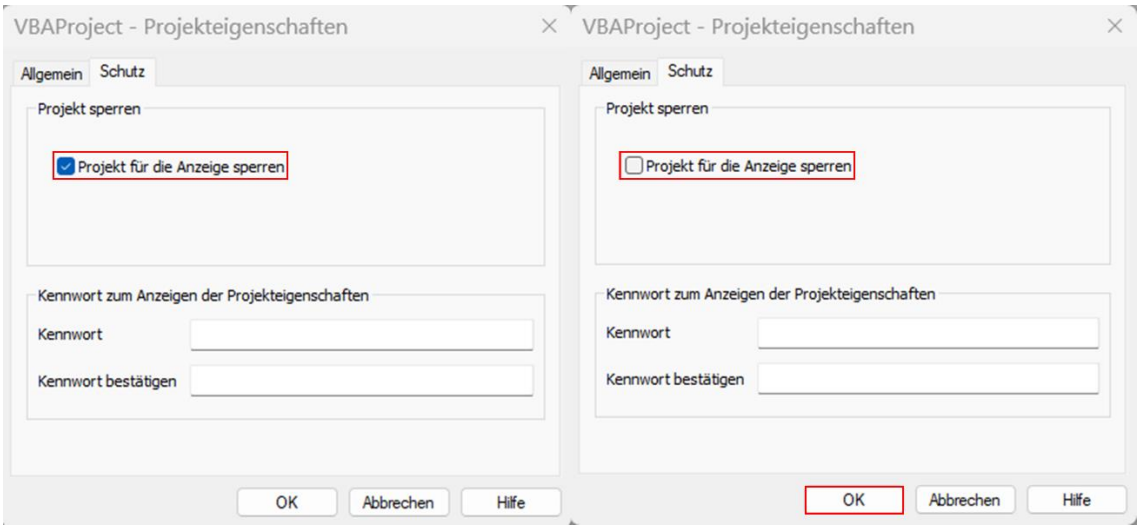


Bild 187: Ergebnis der erfolgreichen Änderung an der .accdb-Datei, um den VBA-Projekt-Passwortschutz zu entfernen (Links: Vor Anpassung; Rechts: Nach Anpassung)

Eine Kopie der Datei wird gespeichert und geöffnet. Beim Öffnen des VBA-Editors erscheint wieder die vorherige Fehlermeldung:

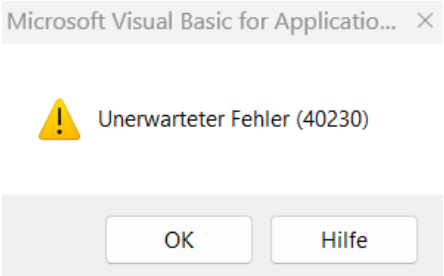


Bild 188: Nicht behobene Fehlermeldung nach deaktivieren der Funktion „Projekt für Anzeige sperren“ in der manipulierten Datei

Nach mehrmaligem Bestätigen öffnet sich der VBA-Editor und der Objektbaum lässt sich öffnen. Beim Klick auf ein zufällig ausgewähltes Modul oder Objekt erscheint erneut die bereits bekannte Fehlermeldung:

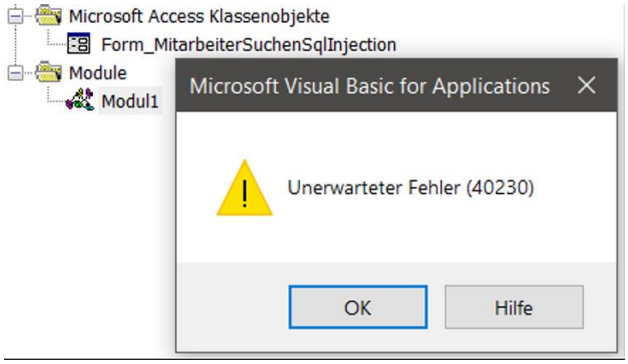


Bild 189: Der VBA-Code in der manipulierten Datei kann nach Deaktivieren der Funktion „Projekt für Anzeige sperren" nicht angezeigt werden

Der VBA-Quellcode ist leider anders als in Excel nicht direkt in der Access-Datei einseh- und manipulierbar, kann aber über „*Rechtsklick → Datei exportieren …*" aus der Datei extrahiert werden:

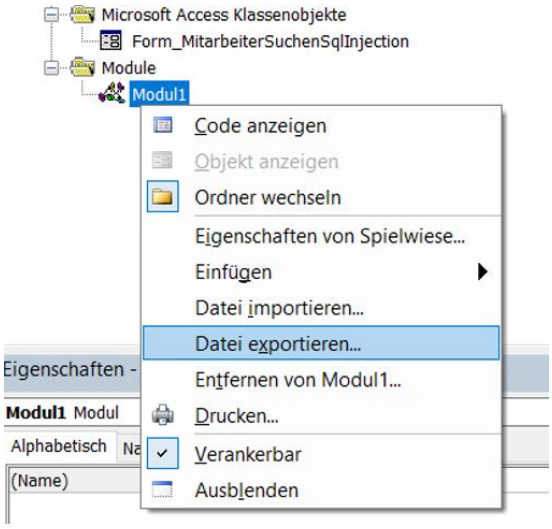


Bild 190: Exportieren des VBA-Codes aus der manipulierten Datei

Der exportierte Quellcode liegt im Klartext vor, inklusive aller Geheimnisse wie ein Connection-String mit SQL-Authentifizierung:

```
1   Option Explicit
2
3   Private Const CONNECTION_STRING As String = "This is a Connection-String"
```

Bild 191: Anzeige des aus der manipulierten Datei exportierten VBA-Codes

Mit den bis zu diesem Punkt gesammelten Informationen können nun weitere Systeme, wie verbundene SQL Server, angegriffen werden. Vorausgesetzt es sind entsprechende Verbindungsinformationen im Quellcode hinterlegt und es wird nicht die Windows oder Azure AD-Authentifizierung verwendet. Als empfohlenes Architekturmodell wirbt MS für

den Einsatz von Access als Präsentationsschicht (Frontend) und dem SQL Server als Datenhaltungsschicht (Backend):

Microsoft | Support Microsoft 365 Office Windows Surface Xbox Sonderangebote Microsoft 365 kaufen

Microsoft 365-Support Produkte Geräte Neuerungen Microsoft 365 installieren Konto und Abrechnung Vorlagen

Access / SQL Server-Migration / Machen Sie einen Access-Ausflug durch SQL Server

Machen Sie einen Access-Ausflug durch SQL Server

Access für Microsoft 365, Access 2021, Access 2019, Access 2016, Access 2013, Access 2010

Nachdem Sie Ihre Daten von Access zu SQL Server migriert haben, verfügen Sie jetzt über eine Client/Server-Datenbank, bei der es sich entweder um eine lokale oder hybride Azure-Cloudlösung handeln kann. In beiden Fällen ist Access nun die Präsentationsschicht und SQL Server die Datenschicht. Jetzt ist ein guter Zeitpunkt, um Aspekte Ihrer Lösung, insbesondere Abfrageleistung, Sicherheit und Geschäftskontinuität, zu überdenken, damit Sie Ihre Datenbanklösung verbessern und skalieren können.

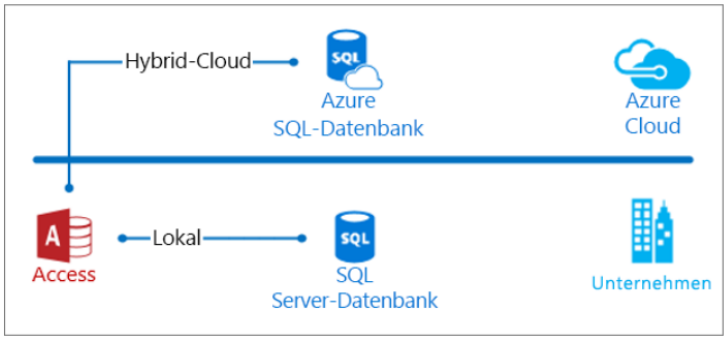


Bild 192: MS Access-Ausflug durch den SQL Server (Access als Frontend, SQL Server als Backend) [196]

Anlage 10: Betrachtung des Dateiformats im Hexadezimal-Editor (Access)

Beim Öffnen einer .accdb sowie .accde-Datei im Hexadezimal-Editor können im VBA-Quelltext hinterlegte Zeichenketten oder in Relationen hinterlegte Daten im Klartext eingesehen werden. Ohne regelmäßiges Ausführen der „Datenbank komprimieren und reparieren“-Funktion oder der automatischen Alternative beim Schließen der Datei unter „*Datei → Optionen → Aktuelle Datenbank → Beim Schließen komprimieren*“ sind gelöschte Daten im Hexadezimal-Editor einsehbar, die nicht mehr über die Access-Oberfläche angezeigt werden (siehe „`Gelöschter Datensatz`“):

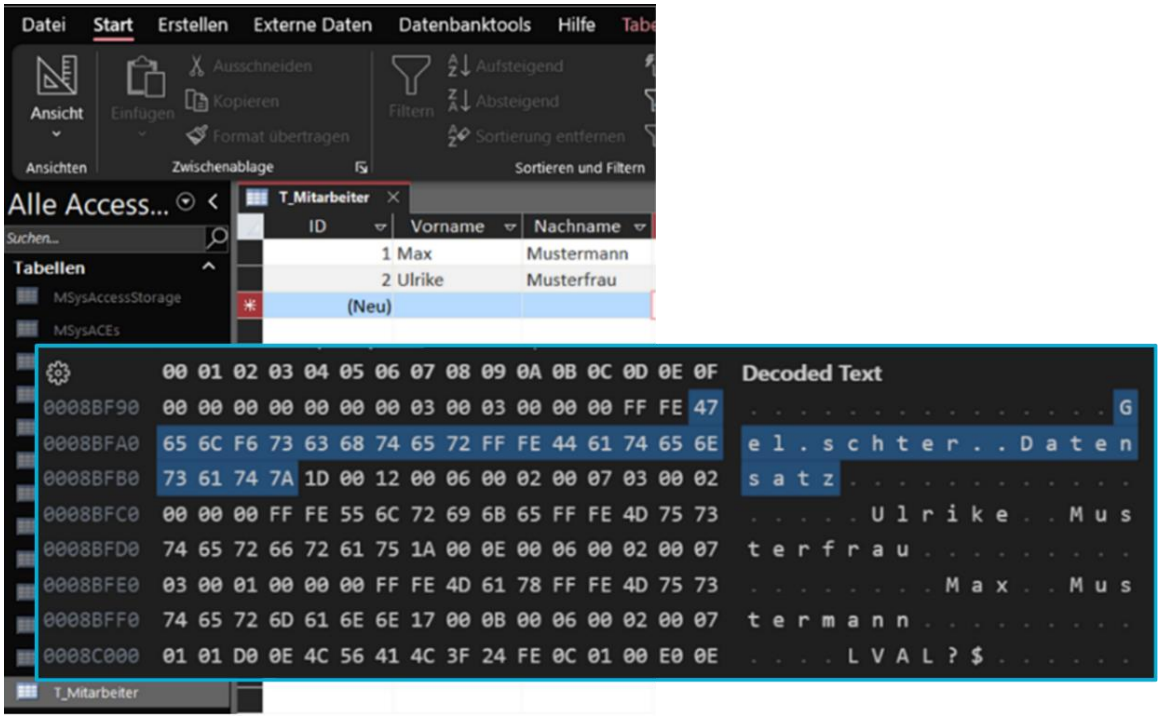


Bild 193: Hexadezimal-Editor-Analyse – Speicherung von Datensätzen im Klartext und Verfügbarkeit von gelöschten Datensätzen in .accdb sowie .accde-Dateien („Gelöschter Datensatz“)

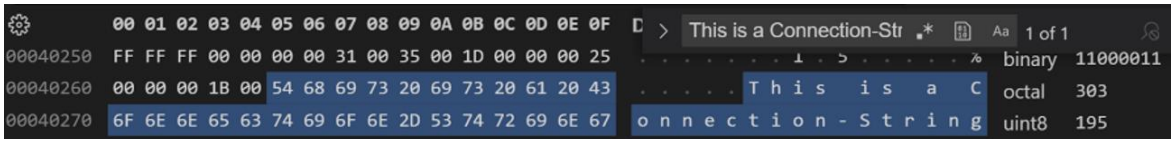


Bild 194: Hexadezimal-Editor-Analyse – Speicherung von Zeichenketten im VBA-Code bei .accdb und .accde-Dateien im Klartext

Weitere Härtungsmaßnahmen gegen diese potenzielle Schwachstelle stellen das Verschlüsseln von Access-Dateien im Dateiformat 2007-2016 mittels Passworts oder der Daten selbst dar. Die .accdb und .accde-Dateien sind mit dem AES-256-Algorithmus (Standard) verschlüsselt. Das Passwort ist mit einem Salt verknüpft und als Hash (SHA-512) mitverschlüsselt im Datei-Header eingebettet. Während des Verschlüsselungsvorgangs wird eine unkritische Warnmeldung angezeigt, dass eine Blockchiffrierung nicht mit der Sperrung auf Datensatzebene kompatibel ist. Unter dem Kopfreiter „*Datei → Optionen → Clienteinstellungen → Datenbanken mit Sperrung auf Datensatzebene öffnen*“ kann die verwendete Sperreinstellung konfiguriert werden, mit der Access bestimmte

Datenmengen bei der Bearbeitung von Datensätzen sperrt. Es kann zwischen einer Sperrung auf Seitenebene und Datensatzebene gewählt werden, wobei die Sperrung auf Seitenebene zur Leistungsoptimierung beiträgt. Die gesperrte Datenmenge hängt von der verwendeten Sperreinstellung ab. Bei Sperrung auf Datensatzebene sperrt Access nur den Datensatz, der bearbeitet wird. Andere Datensätze sind nicht betroffen. Bei der Sperrung auf Seitenebene sperrt Access dagegen die gesamte Seite (Speicherbereich im Arbeitsspeicher), die den Datensatz enthält. Eine aktivierte Sperrung auf Seitenebene kann dazu führen, dass andere Datensätze, die in der Nähe im Arbeitsspeicher gespeichert sind, ebenfalls gesperrt werden und somit die Datenverfügbarkeit negativ beeinflusst wird [203]:

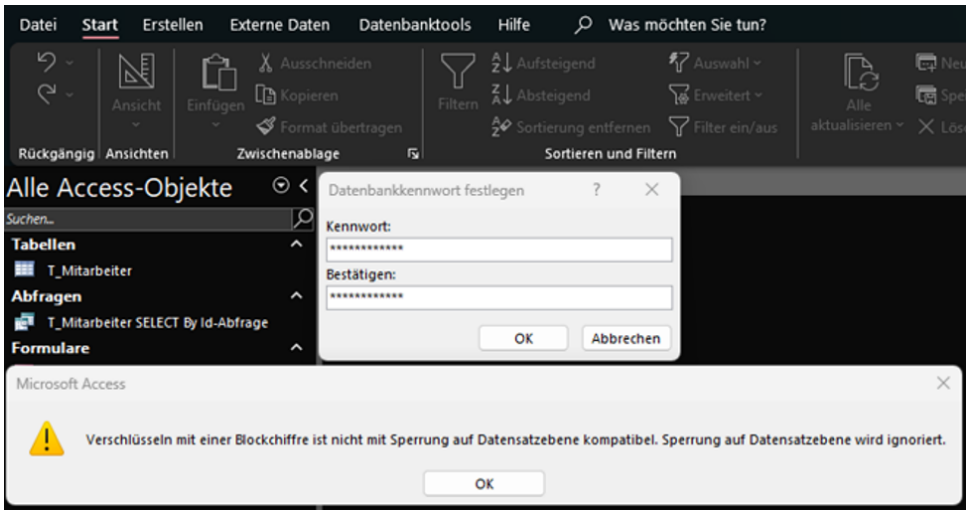


Bild 195: Hexadezimal-Editor-Analyse – Dateiverschlüsselung

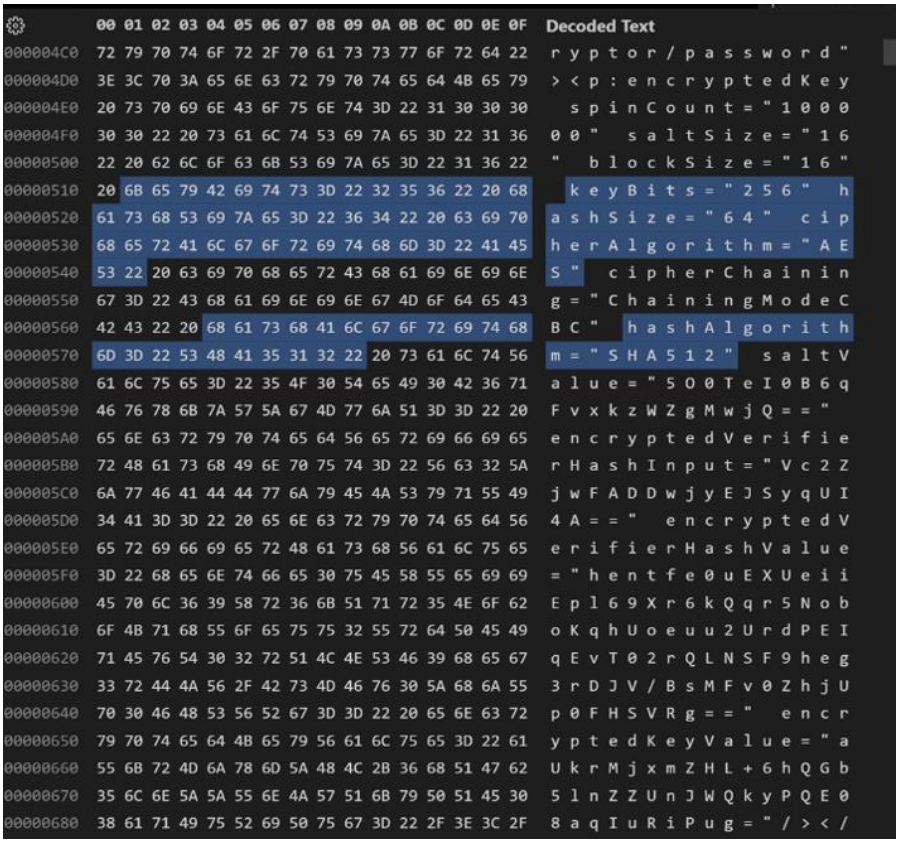


Bild 196: Hexadezimal-Editor-Analyse – Verschlüsselungsinformationen im Datei-Header

Sämtliche vorher lesbaren Daten liegen nach der Dateiverschlüsselung nun in verschlüsselter und somit unlesbarer Form vor. Es können somit keine Informationen über verknüpfte Tabellen, Passwörter oder Connections-Strings im VBA-Quellcode gesammelt werden. Der Dateiinhalt ist sicher, solange Angreifende keinen Zugriff auf das Passwort zum Öffnen und Entschlüsseln der Datei oder den Treuhandschlüssel (*siehe Kapitel „4.1.3 Kryptographie“*) besitzen:

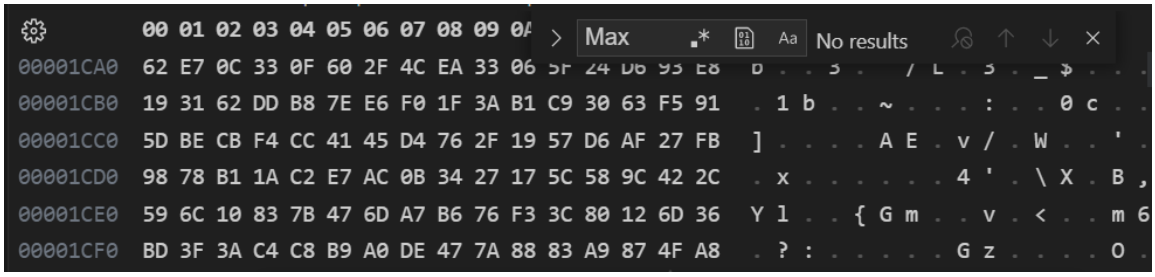


Bild 197: Hexadezimal-Editor-Analyse – Verschlüsselter Dateiinhalt nach Aktivierung der Funktion „Mit Kennwort verschlüsseln“ (Suche nach: „Max Mustermann“) 1/2

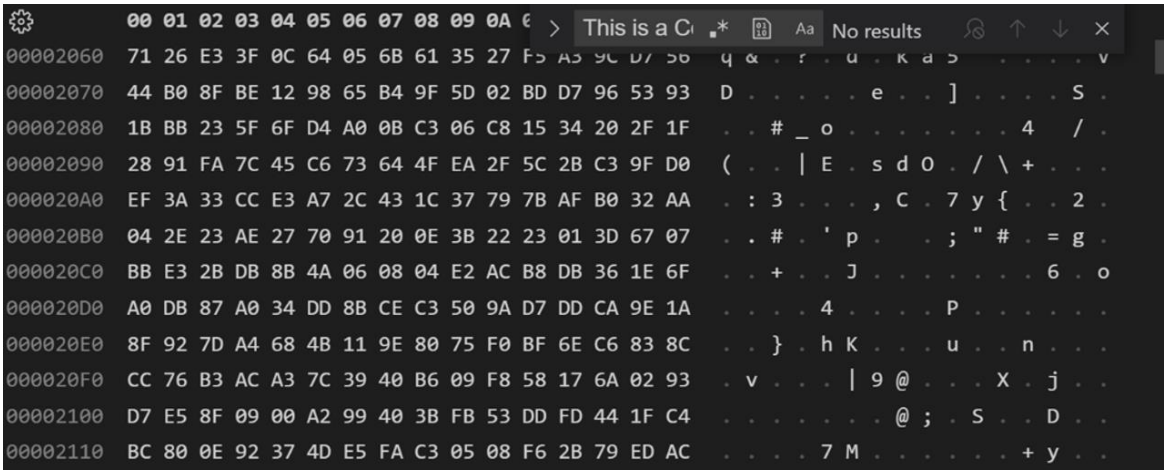


Bild 198: Hexadezimal-Editor-Analyse – Verschlüsselter VBA-Code nach Aktivierung der Funktion „Mit Kennwort verschlüsseln“ (Suche nach: „This is a Connection String“) 2/2

Die Analyse zeigt, dass eine Verschlüsselung einer Access-Datei (Front- sowie Backend) unabdingbar für die (Daten)Sicherheit und der einzige Schutz gegen das Speichern von Daten im Klartext ist. Neben der Nutzung von starken Passwörtern kann die Speicherung von bereits verschlüsselten Daten als zusätzliche Härtungsmaßnahme dienen.

Anlage 11: Vorbereitung der Testumgebung für die forensische Analyse einer Access-Datei

Als Dummy-Image dient ein 512 Megabyte großer USB-Stick mit zufällig ausgewählten Verzeichnissen und Ordnern aus „*C:\Windows*“ und NTFS-Dateisystem:

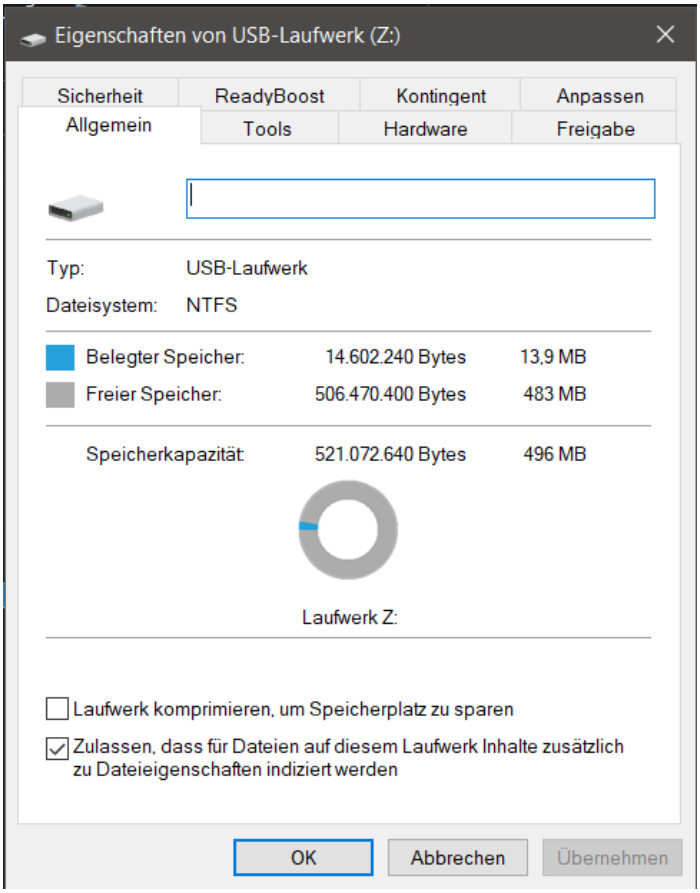


Bild 199: Eigenschaften des Dummy-USB-Sticks

Vor Befüllung mit den Dummy-Dateien ist der USB-Stick mit dem Kommandozeilen-Befehl „*format Z: /fs:NTFS /p:1*“ formatiert worden, um jeden Sektor auf dem Volume zu „nullen“, indem er einmal mit einer Zufallszahl überschrieben wird und somit alle Daten unwiederbringlich gelöscht werden (Disk Wipe):

```
C:\>format Z: /fs:NTFS /p:1
Legen Sie eine neue Diskette in Laufwerk Z: ein,
und drücken Sie die EINGABETASTE.
Der Typ des Dateisystems ist NTFS.
Überprüfung von 496,9 MB
Volumebezeichnung (32 Zeichen, EINGABETASTE für keine)? Z
Struktur des Dateisystems wird erstellt.
Formatieren beendet.
     496,9 MB Speicherplatz auf dem Datenträger insgesamt.
     492,6 MB sind verfügbar.
```

Bild 200: Wipe des Dummy-USB-Sticks für die forensische Analyse und zugehörige Informationen

Eine aufwändigere Alternative dazu stellt der „*diskpart clean all*“-Komanndozeilenbefehl dar. Nach Auswahl des zu formatierenden USB-Sticks mittels „*select disk*“-Befehl, folgt der „*clean all*“-Befehl zum Löschen aller Partitions- oder Volumeformatierungen. Der Parameter „*all*“ gibt an, dass jeder Sektor auf dem Datenträger auf null gesetzt wird. Danach erfolgt über den „*create partition primary*“-Befehl die Anlage einer primären Partition. Im letzten Schritt wird mittels bereits bekanntem „*format fs=ntfs*“-Befehl die Formatierung des USB-Sticks in das NTFS durchgeführt, damit wieder Dateien abgelegt werden können. Optional kann auch über den „*assign letter=<Laufwerksbuchstabe>*“-Befehl ein Laufwerksbuchstabe vergeben werden:

```
Administrator: Eingabeaufforderung
C:\>diskpart

Microsoft DiskPart-Version 10.0.19041.964

Copyright (C) Microsoft Corporation.
Auf Computer: DESKTOP-RB4UI7G

DISKPART> list disk

  Datenträger ###  Status         Größe    Frei     Dyn  GPT
  ---------------  -------------  -------  -------  ---  ---
  Datenträger 0    Online          476 GB  2048 KB
  Datenträger 1    Online          497 MB   496 MB

DISKPART> sel dis 1

Datenträger 1 ist jetzt der gewählte Datenträger.

DISKPART> clean all

Der Datenträger wurde bereinigt.

DISKPART> create partition primary

Die angegebene Partition wurde erfolgreich erstellt.

DISKPART> format fs=ntfs

  100 Prozent bearbeitet

DiskPart hat das Volume erfolgreich formatiert.

DISKPART> assign letter=Z

Der Laufwerkbuchstabe oder der Bereitstellungspunkt wurde zugewiesen.

DISKPART> exit

Datenträgerpartitionierung wird beendet...
```

Bild 201: Alternatives Vorgehen zum Wipe des Dummy-USB-Sticks

Nach Vorbereitung des USB-Sticks und Befüllung mit den zufällig ausgewählten Dateien sind Testdateien unter folgenden, ebenso zufällig ausgewählten Verzeichnissen versteckt worden. Im Dateinamen ist angegeben, ob die Dateien Eigenschaften, wie signierten Code, besitzen oder gelöscht wurden. In allen Access- Testdateien ist in einer Tabelle eine Datei eingebettet (in Relationen eingebettete Dateien werden in Access als Anhang bezeichnet), der Übersichtlichkeit halber wurde dieser Zusatz nicht im Dateinamen ergänzt:

Tabelle 9: Ablageorte der Testdateien in Dummy-USB-Image

Dateiname inklusive Eigenschaft	Ablageort
Passwort für verschlüsselte Dateien & Ablageorte.txt	*Z:\Boot\PCAT\en-US*
Testdatei Access 1 Code signiert.accdb	Z:\5EK7S0EPQC
Testdatei Access 2.accde	Z:\Boot\Misc\PCAT\a\b\c\d\e
Testdatei Access 3 Dateiverschlüsse-lung.accde	Z:\Fonts
Testdatei Access 4 Datei signiert.accdc	Z:\Fonts
Testdatei Access 5 Datei signiert & ge-löscht.accdc	Z:\7HJBIAP3W0 (Ablageort vor Löschung)
Testdatei Access 6 Code signiert & ge-löscht.accdb	Z:\7HJBIAP3W0 (Ablageort vor Löschung)
Testdatei Access 7 gelöscht.accde	Z:\7HJBIAP3W0 (Ablageort vor Löschung)
Testdatei Excel 1 eingebettete Da-tei.xlsx	Z:\Wallpaper
Testdatei Excel 2.xlsm	Z:\Boot\Misc\PCAT\a\b\c\d\e
Testdatei Excel 3 Dateiverschlüsse-lung.xlsm	Z:\
Testdatei Word 1.docx	*Z:\5EK7S0EPQC*
Testdatei Word 2.docm	*Z:\Boot\Misc\PCAT\a\b\c\d\e*
Testdatei Word 3 Dateiverschlüsse-lung.docm	*Z:\Wallpaper\Theme1*

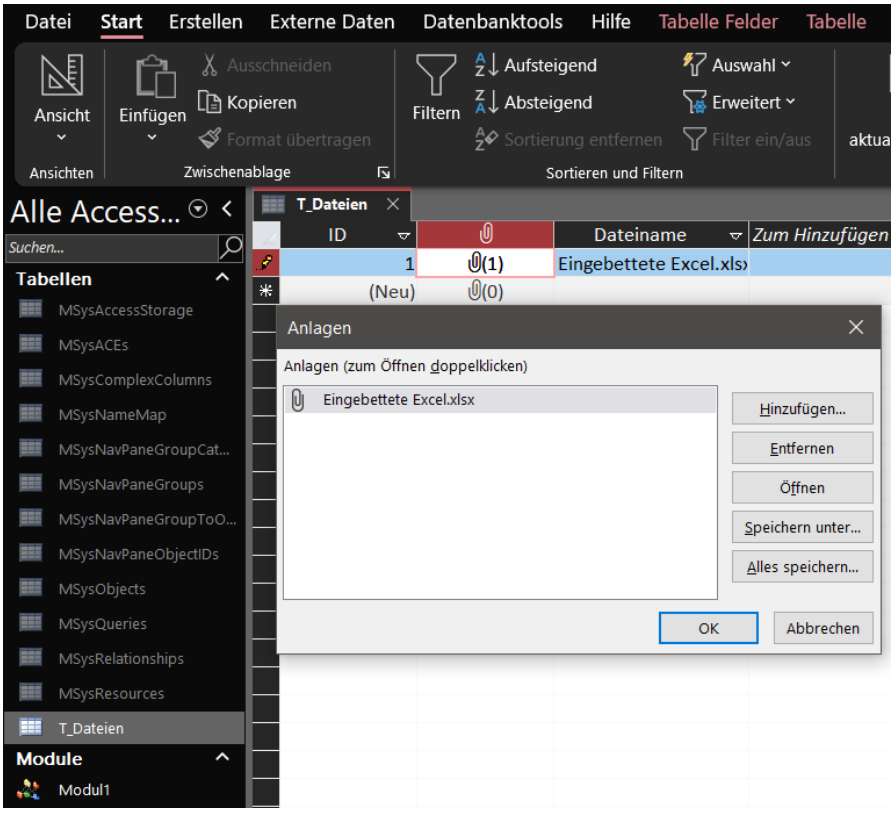


Bild 202: In alle Access-Testdateien ist eine eingebettete Datei (Anhang) in der Relation „T_Dateien“ abgelegt

Nach Vorbereitung des Dummy-USB-Sticks wird mit dem forensischen Programm „FEX Imager™“ von *GetData Forensics* ein .E01-Image des USB-Sticks erstellt. Dabei ist die standardmäßig aktivierte Komprimierung deaktiviert [47]:

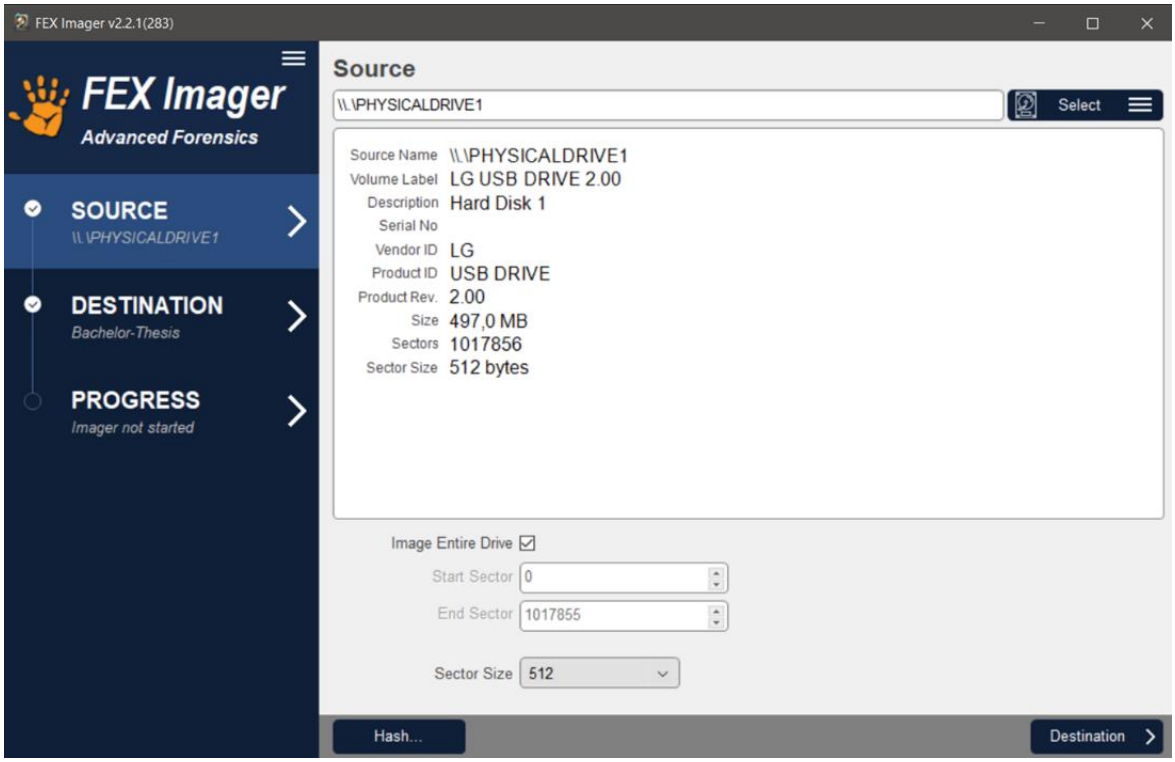


Bild 203: .E01-Image-Erstellung mit Hilfe des Programms FEX Imager™ 1/2

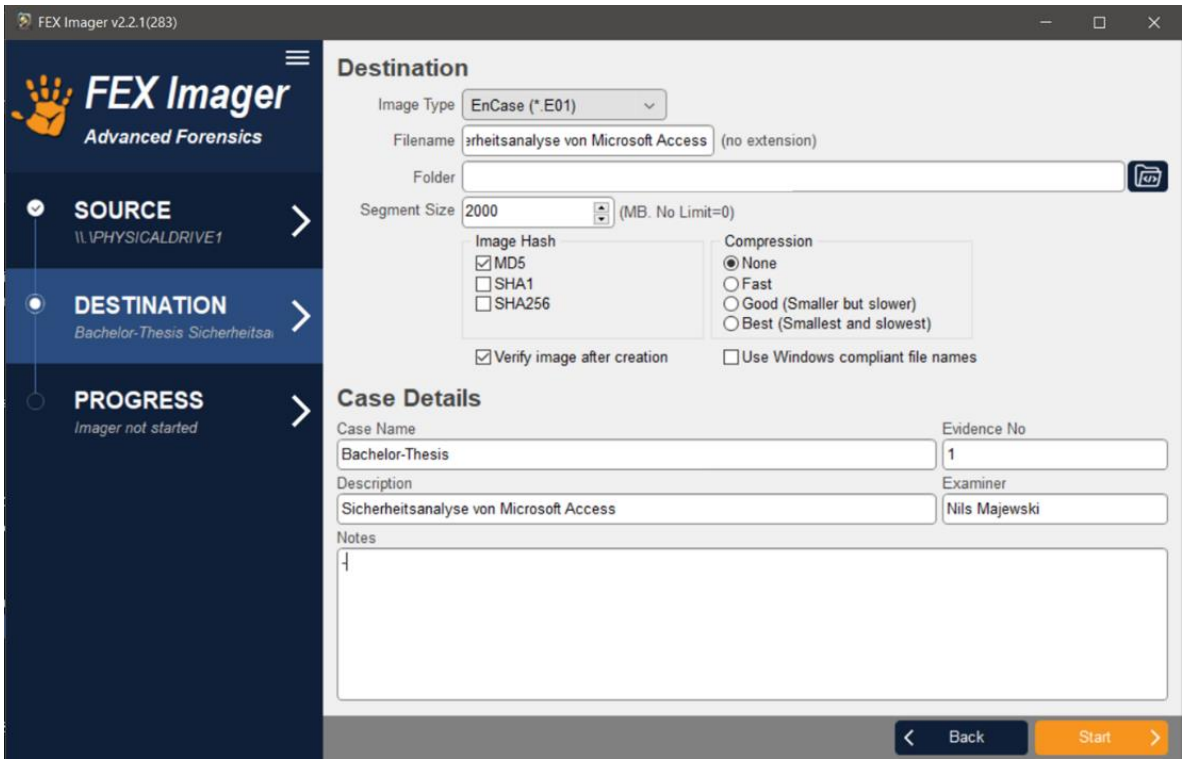


Bild 204: .E01-Image-Erstellung mit Hilfe des Programms FEX Imager™ 2/2

Das erstellte .E01-Image wird als Datenquelle einem Fall in Autopsy hinzugefügt. Die Standardkonfigurationen werden beibehalten, es wird nach allen Dateien und Ordnern gesucht. Inklusive Berücksichtigung des Unallocated-Space, also gelöschten Dateien

via PhotoRec Carver-Moduls sowie eingebetteten Dateien mittels des Embedded File Extractor-Moduls. Das PhotoRec Carver-Modul analysiert den Unallocated-Space mit Hilfe von File Carving, bei dem die Datei-Metadaten wie der Dateiname verlorengehen. Beim File Carving wird nicht allokierter Speicherbereich durchsucht und versucht Datei-Header, -Footer sowie Dateisignaturen zu ermitteln. Die Dateisignatur (magische Zahl) ist im Datei-Header zu finden. Dabei handelt es sich um eine für jeden Dateityp eindeutige Folge von Bytes (bei JPEG „*0xFFD8FF*“). Auch das Dateiende kann standardisiert sein (bei JPEG „*0xFFD900*“). Ist keine Endsequenz, wie bei Word-Dokumenten, vorhanden, muss die Dateigröße anders bestimmt werden. Ist die Dateigröße im Header zu finden, kann sie dort ausgelesen werden. Sind Header und Footer ermittelt, werden die Daten zwischen Header und Footer ausgelesen, um die Datei wiederherzustellen. Im Optimalfall sind die Daten nicht fragmentiert und die zur Datei gehörenden Daten liegen somit auf dem Datenträger in aufeinanderfolgenden Datenblöcken [39, S. 137]:

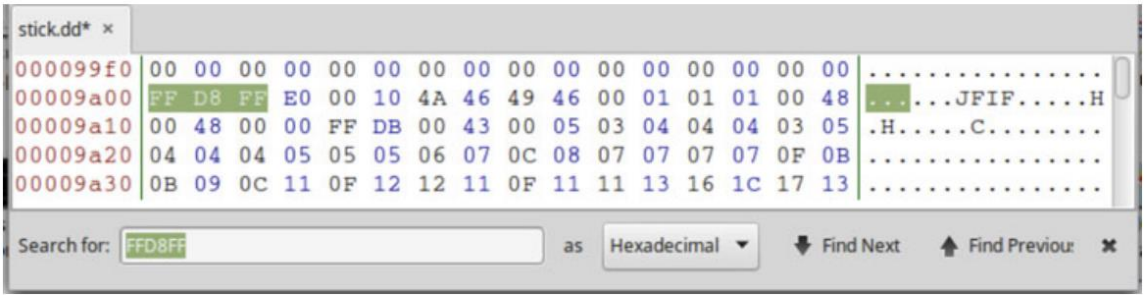


Bild 205: Dateisignatur (magische Zahl) einer JPEG-Datei [39, S. 138]

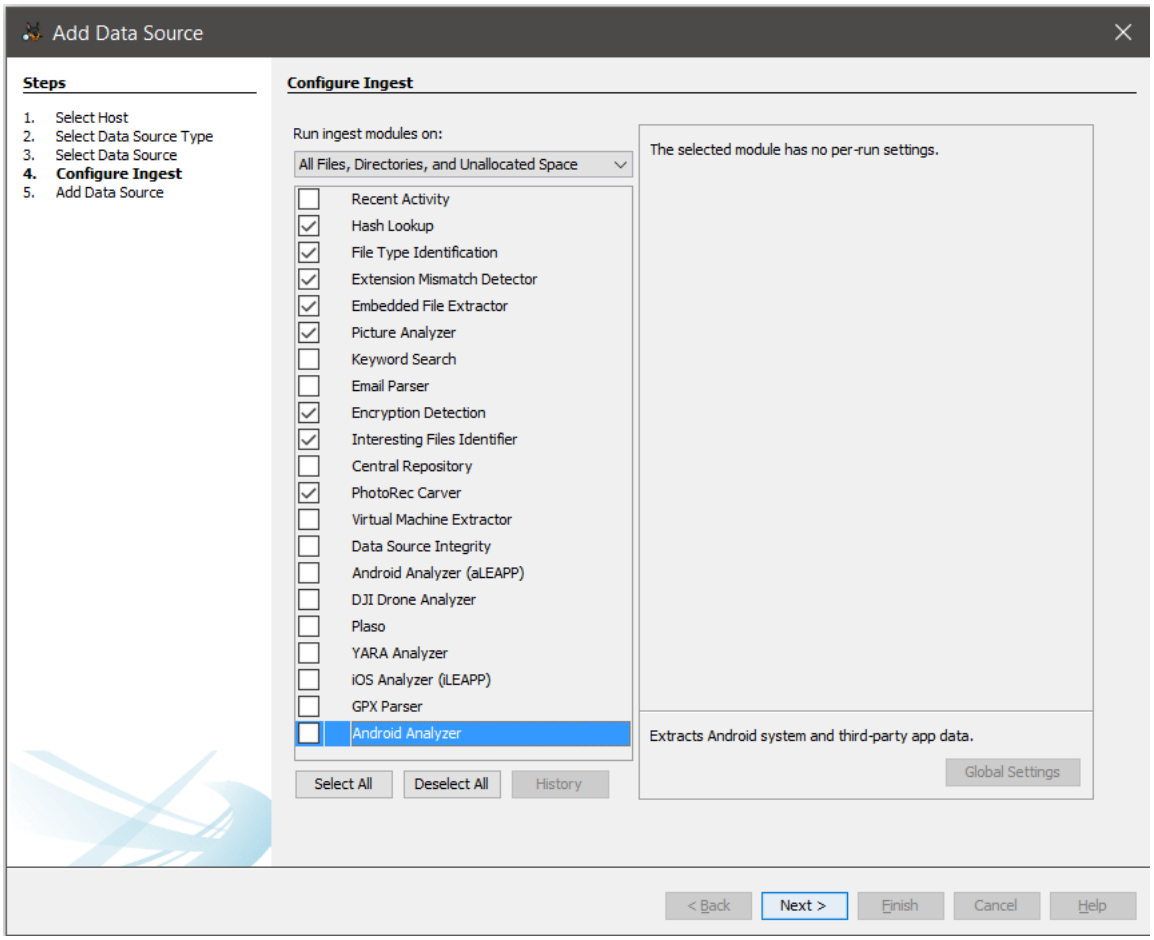


Bild 206: Autopsy – Aktivierte Ingest Modules