APP.4.8 Graphdatenbanken

1. Beschreibung

1.1. Einleitung

Graphdatenbanken wie Neo4j, ArangoDB oder Amazon Neptune sind für die Speicherung und Abfrage von stark vernetzten Daten konzipiert. Anstatt Daten in Tabellen oder Dokumenten zu organisieren, modellieren sie Informationen in Knoten (Entitäten) und Kanten (Beziehungen). Dieser Ansatz ist ideal für Anwendungsfälle wie soziale Netzwerke, Betrugserkennung, Wissensgraphen und Empfehlungssysteme.
Die Stärke von Graphdatenbanken liegt in ihrer Fähigkeit, komplexe Beziehungen effizient zu durchlaufen (Traversierung). Dies birgt jedoch auch spezifische Sicherheitsrisiken. Leistungsfähige Abfragesprachen wie Cypher oder Gremlin können bei unsachgemäßer Anwendung zu schwerwiegenden Leistungsproblemen führen. Zudem ermöglicht die vernetzte Datenstruktur Angreifern, durch das Verfolgen von Beziehungen auf Informationen zu schließen, die nicht direkt für sie sichtbar sind (Inference Attacks). Die Absicherung dieser Systeme erfordert daher einen anderen Fokus als bei relationalen Datenbanken.

1.2. Zielsetzung

Dieser Baustein zeigt einen systematischen Weg auf, um Graphdatenbanken sicher zu konfigurieren, zu betreiben und zu nutzen. Ziel ist es, die Vertraulichkeit und Integrität der gespeicherten Knoten und Beziehungen zu schützen, die Verfügbarkeit der Datenbank gegen ressourcenintensive Abfragen zu härten und unautorisierte Datenerkenntnisse durch die Analyse von Graphenstrukturen zu unterbinden.

1.3. Abgrenzung und Modellierung

Der Baustein APP.4.8 ist auf alle eingesetzten Graphdatenbanken anzuwenden.
Dieser Baustein vervollständigt das Spektrum der Datenbank-Bausteine im IT-Grundschutz-Kompendium und behandelt die spezifischen Aspekte von Graphdatenbanken. Er baut auf den grundlegenden Server-Absicherungen der Schicht OPS.1.1 auf. Anforderungen an die sichere Nutzung von datenbankeigenen Erweiterungen und die Entwicklung von Anwendungen, die Graphdatenbanken verwenden, finden sich in CON.1 Softwareentwicklung.

2. Gefährdungslage

Für den Baustein APP.4.8 sind folgende spezifische Bedrohungen und Schwachstellen von besonderer Bedeutung:

Unautorisierte Datenermittlung durch Graph-Traversierung

Angreifer mit begrenztem Lesezugriff können durch das systematische Verfolgen von Beziehungen (Traversierung) im Graphen versteckte Zusammenhänge aufdecken und sensible Informationen ableiten. Selbst wenn sie einzelne Knoten nicht direkt abfragen dürfen, kann der Pfad zwischen zwei Knoten bereits kritische Informationen preisgeben und so zu einer ungewollten Offenlegung von Wissen führen.

Denial-of-Service durch komplexe Graph-Abfragen

Die Abfragesprachen von Graphdatenbanken sind darauf ausgelegt, Pfade beliebiger Tiefe zu analysieren. Eine unsauber formulierte oder böswillige Abfrage, die beispielsweise nach Pfaden mit unbegrenzter Länge sucht ("variable length path traversals"), kann eine Kaskade von Operationen auslösen, die den Arbeitsspeicher und die CPU des Servers vollständig auslasten und die Datenbank zum Stillstand bringen.

Kompromittierung über unsichere Erweiterungen und Prozeduren

Viele Graphdatenbanken erlauben die Erweiterung ihrer Funktionalität durch benutzerdefinierte Prozeduren oder Plugins. Werden diese Erweiterungen aus nicht vertrauenswürdigen Quellen bezogen oder sind sie selbst fehlerhaft programmiert, können sie als Einfallstor dienen, um Sicherheitskontrollen der Datenbank zu umgehen oder unautorisierten Zugriff auf das darunterliegende Betriebssystem zu erlangen.

Unzureichende Zugriffskontrolle auf Graphenebene

Fehlt ein granulare Zugriffskontrollmodell, erhalten Benutzer möglicherweise Zugriff auf den gesamten Graphen, obwohl sie nur einen kleinen Ausschnitt (einen Subgraphen) sehen dürften. Dies ist besonders in mandantenfähigen Systemen kritisch, wo eine strikte Trennung von Daten verschiedener Benutzer oder Gruppen zwingend erforderlich ist.

Preisgabe von Informationen über ungesicherte APIs

Graphdatenbanken werden häufig über APIs (z. B. GraphQL oder REST-basierte Endpunkte) für Anwendungen bereitgestellt. Wenn diese APIs keine ausreichende Authentifizierung, Autorisierung und Eingabevalidierung durchführen, bieten sie Angreifern einen direkten und oft sehr mächtigen Hebel, um Daten unkontrolliert abzufragen oder zu manipulieren.

3. Anforderungen

Im Folgenden sind die spezifischen Anforderungen des Bausteins APP.4.8 aufgeführt.

3.1. Basis-Anforderungen

Die folgenden Anforderungen MÜSSEN für diesen Baustein vorrangig erfüllt werden.

APP.4.8.A1 Absicherung des Netzwerkzugriffs (B)

Der Netzwerkzugriff auf die Graphdatenbank und ihre administrativen Schnittstellen MUSS durch eine Firewall auf die zwingend erforderlichen Systeme beschränkt sein. Die Dienste MÜSSEN nur an vertrauenswürdige Netzwerkschnittstellen gebunden werden.

APP.4.8.A2 Einrichtung einer verpflichtenden Authentifizierung (B)

Für jeden Zugriff auf die Graphdatenbank MUSS eine Authentifizierung erzwungen werden. Ein anonymer Zugriff MUSS deaktiviert und alle Standard-Benutzerkonten MÜSSEN mit sicheren, nicht standardisierten Passwörtern versehen oder deaktiviert werden.

APP.4.8.A3 Zuweisung grundlegender Rollen (B)

Es MUSS ein grundlegendes Rollenkonzept umgesetzt werden, das mindestens zwischen administrativen Benutzern, Benutzern mit Schreibzugriff und Benutzern mit reinem Lesezugriff unterscheidet. Anwendungsbenutzer MÜSSEN mit den geringstmöglichen Rechten ausgestattet werden.

APP.4.8.A4 Sicherer Umgang mit Erweiterungen (B)

Es MUSS eine Regelung geben, die festlegt, dass Erweiterungen (Plugins, Prozeduren) nur aus vertrauenswürdigen Quellen bezogen und installiert werden dürfen.

3.2. Standard-Anforderungen

Gemeinsam mit den Basis-Anforderungen entsprechen die folgenden Anforderungen dem Stand der Technik. Sie SOLLTEN grundsätzlich erfüllt werden.

APP.4.8.A5 Verschlüsselung der Datenübertragung (S)

Die gesamte Kommunikation mit der Graphdatenbank, einschließlich der Verbindungen von Anwendungen und der Replikations-Kommunikation in einem Cluster, SOLLTE mittels TLS verschlüsselt werden.

APP.4.8.A6 Umsetzung einer granularen Zugriffskontrolle (S)

Es SOLLTE ein detailliertes Berechtigungskonzept umgesetzt werden, das Zugriffe auf der Ebene von Knoten-Labels, Beziehungstypen und Eigenschaften regelt. Dadurch SOLLTE sichergestellt werden, dass Benutzer nur die Daten sehen und bearbeiten können, für die sie autorisiert sind.

APP.4.8.A7 Härtung der Abfragesprache (S)

Um Denial-of-Service-Angriffe durch missbräuchliche Abfragen zu verhindern, SOLLTEN technische Schutzmaßnahmen konfiguriert werden. Dies SOLLTE Limits für die Laufzeit von Abfragen und den dabei verbrauchten Arbeitsspeicher umfassen. Transaktionen SOLLTEN überwacht und bei Überschreitung von Schwellenwerten automatisch beendet werden.

APP.4.8.A8 Regelmäßige Protokollierung von Abfragen und Zugriffen (S)

Alle ausgeführten Abfragen, fehlgeschlagenen Anmeldeversuche und administrativen Änderungen SOLLTEN protokolliert werden. Die Protokolldaten SOLLTEN an ein zentrales System übermittelt und regelmäßig auf Anomalien analysiert werden.

3.3. Anforderungen bei erhöhtem Schutzbedarf

Die folgenden Anforderungen sind exemplarische Vorschläge für ein Schutzniveau, das über den Stand der Technik hinausgeht. Sie SOLLTEN bei erhöhtem Schutzbedarf in Betracht gezogen werden.

APP.4.8.A9 Implementierung von Zugriffskontrollen auf Subgraph-Ebene (H)

Für Umgebungen mit hohen Anforderungen an die Datentrennung (z. B. Mandantenfähigkeit) SOLLTE eine Zugriffskontrolle implementiert werden, die Benutzern nur Zugriff auf einen definierten Teilausschnitt des Graphen (Subgraph) gewährt. Abfragen, die die Grenzen dieses Subgraphen überschreiten würden, MÜSSEN blockiert werden.

APP.4.8.A10 Verschlüsselung der gespeicherten Daten (Encryption at Rest) (H)

Werden in der Graphdatenbank besonders schützenswerte Daten und Beziehungen gespeichert, SOLLTEN die auf dem Datenträger abgelegten Datenbankdateien verschlüsselt werden.

APP.4.8.A11 Einsatz von Schema-Constraints (H)

Um die Datenintegrität zu gewährleisten, SOLLTE ein striktes Schema für den Graphen definiert und technisch erzwungen werden. Dies stellt sicher, dass Knoten und Beziehungen nur mit den vordefinierten Eigenschaften und Typen erstellt werden können.

APP.4.8.A12 Automatische Analyse und Überwachung von Abfragen (H)

Es SOLLTEN Werkzeuge eingesetzt werden, die Abfragen vor ihrer Ausführung analysieren, um übermäßig komplexe oder potenziell schädliche Traversierungen zu erkennen. Verdächtige Abfragen SOLLTEN blockiert oder zur manuellen Überprüfung gemeldet werden.