# APP.4.9 Objektbasierte Speicherlösungen

## 1. Beschreibung

### 1.1. Einleitung

Objektbasierte Speicherlösungen (Object Storage) sind ein zentraler Baustein moderner IT-Architekturen, insbesondere in Cloud-Umgebungen (z. B. S3-kompatible Dienste) und in großen On-Premises-Infrastrukturen (z. B. mit Ceph oder OpenStack Swift). Im Gegensatz zu traditionellen Datei- oder Blockspeichern organisieren sie Daten als einzelne Objekte in flachen Hierarchien, sogenannten Buckets. Jedes Objekt besteht aus den eigentlichen Daten, einer eindeutigen ID und Metadaten.
Der Zugriff auf Objektspeicher erfolgt ausschließlich über programmatische Schnittstellen (APIs), typischerweise über HTTP. Diese API-zentrierte Natur und ein sehr flexibles Berechtigungsmodell sind Stärken, aber auch die Quelle spezifischer Risiken. Eine einzige Fehlkonfiguration in einer Bucket-Richtlinie kann dazu führen, dass Terabytes an sensiblen Daten ungeschützt im Internet stehen. Die Sicherheit von Objektspeichern hängt daher entscheidend von der korrekten Konfiguration der Zugriffsrichtlinien und der Absicherung der API-Endpunkte ab.

### 1.2. Zielsetzung

Dieser Baustein zeigt einen systematischen Weg auf, um objektbasierte Speicherlösungen sicher zu konfigurieren und zu betreiben. Ziel ist es, die Vertraulichkeit, Integrität und Verfügbarkeit der gespeicherten Objekte zu gewährleisten. Schwerpunkte sind die rigorose Kontrolle von Zugriffsberechtigungen auf Bucket- und Objektebene, die durchgängige Verschlüsselung der Daten, die Absicherung der API-Kommunikation und das lückenlose Logging aller Zugriffe.

### 1.3. Abgrenzung und Modellierung

Der Baustein APP.4.9 ist auf alle eingesetzten objektbasierten Speicherlösungen anzuwenden.
Dieser Baustein konkretisiert die allgemeinen Anforderungen aus APP.4.4 Speicherlösungen für den spezifischen Fall des Objektspeichers. Er behandelt nicht die Absicherung der zugrundeliegenden Server-Infrastruktur bei On-Premises-Lösungen (siehe OPS.1.1-Bausteine). Bei der Nutzung von Cloud-Diensten sind zusätzlich die Anforderungen aus OPS.2.3 Nutzung von Cloud-Diensten zu beachten.

## 2. Gefährdungslage

Für den Baustein APP.4.9 sind folgende spezifische Bedrohungen und Schwachstellen von besonderer Bedeutung:

#### Unbeabsichtigte Veröffentlichung von Daten durch öffentliche Buckets

Dies ist eine der häufigsten und schwerwiegendsten Schwachstellen bei Objektspeichern. Durch eine fehlerhafte Konfiguration der Zugriffsrichtlinien (ACLs oder Bucket Policies) wird

ein Bucket für die gesamte Welt lesbar oder sogar beschreibbar. Dies führt zur vollständigen Offenlegung aller darin enthaltenen Daten.

#### Kompromittierung von API-Zugangsdaten

Der Zugriff auf Objektspeicher erfolgt über API-Schlüssel (Access Keys). Werden diese Schlüssel kompromittiert, beispielsweise weil sie in öffentlich zugänglichem Quellcode hinterlegt wurden, erlangt ein Angreifer die gleichen Rechte wie der legitime Besitzer der Schlüssel und kann Daten unbemerkt auslesen, manipulieren oder löschen.

#### Unzureichende Zugriffskontrolle auf Objektebene

Selbst wenn ein Bucket nicht vollständig öffentlich ist, können zu weitreichende interne Berechtigungen zu Datenlecks führen. Erlaubt eine Richtlinie beispielsweise allen authentifizierten Benutzern einer Organisation den Lesezugriff auf alle Objekte, können Daten über die Abteilungsgrenzen hinweg unautorisiert eingesehen werden.

#### Verlust oder Manipulation von Daten ohne Versionierung

Ist die Objekt-Versionierung nicht aktiviert, wird bei einer Überschreibung eines Objekts die alte Version unwiederbringlich gelöscht. Dies kann durch Angreifer, aber auch durch fehlerhafte Skripte oder menschliches Versagen zum permanenten Datenverlust oder zur unbemerkten Manipulation von Daten führen.

#### Ausspähen von Daten während der Übertragung

Erfolgt die Kommunikation mit den API-Endpunkten des Objektspeichers unverschlüsselt über HTTP, können sowohl die übertragenen Daten als auch die zur Authentifizierung genutzten API-Schlüssel im Netzwerk mitgelesen werden.

## 3. Anforderungen

Im Folgenden sind die spezifischen Anforderungen des Bausteins APP.4.9 aufgeführt.

### 3.1. Basis-Anforderungen

Die folgenden Anforderungen MÜSSEN für diesen Baustein vorrangig erfüllt werden.

#### APP.4.9.A1 Blockieren des öffentlichen Zugriffs (B)

Es MUSS auf der obersten Ebene des Speichersystems (z. B. auf Account-Ebene) eine technische Einstellung aktiviert sein, die jeglichen öffentlichen Zugriff auf Buckets standardmäßig blockiert. Ausnahmen MÜSSEN explizit und nach einem Genehmigungsverfahren definiert werden.

#### APP.4.9.A2 Grundlegendes Berechtigungskonzept für Buckets (B)

Für jeden Bucket MUSS eine Zugriffsrichtlinie (Bucket Policy oder ACL) definiert sein, die den Zugriff auf einen explizit definierten und minimalen Kreis von Benutzern oder Diensten beschränkt.

### APP.4.9.A3 Erzwingung der Transportverschlüsselung (B)

Es MUSS sichergestellt sein, dass alle Verbindungen zu den API-Endpunkten des Objektspeichers ausschließlich über TLS-verschlüsselte Verbindungen (HTTPS) erfolgen. Unverschlüsselte Verbindungen MÜSSEN serverseitig abgewiesen werden.

### APP.4.9.A4 Schutz der administrativen Zugangsdaten (B)

Die primären Zugangsdaten (Root-Account, Administrator) des Objektspeichersystems MÜSSEN besonders geschützt und DÜRFEN NICHT für den regulären, automatisierten Zugriff durch Anwendungen verwendet werden.

---

## 3.2. Standard-Anforderungen

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

### APP.4.9.A5 Umsetzung eines Prinzips der geringsten Rechte (Least Privilege) (S)

Für jede Anwendung und jeden Benutzer SOLLTE ein eigener technischer Benutzer (z. B. IAM-Benutzer) mit dedizierten API-Schlüsseln erstellt werden. Die Berechtigungen SOLLTEN so granular wie möglich auf die benötigten Aktionen (z. B. GetObject, PutObject) und Ressourcen (spezifische Buckets oder Pfade) beschränkt werden.

### APP.4.9.A6 Aktivierung der serverseitigen Verschlüsselung (Server-Side Encryption) (S)

Alle in einem Bucket gespeicherten Objekte SOLLTEN standardmäßig serverseitig verschlüsselt werden (Encryption at Rest). Es SOLLTE sichergestellt sein, dass unverschlüsselte Objekte beim Speichern automatisch abgewiesen oder verschlüsselt werden.

### APP.4.9.A7 Aktivierung der Zugriffs-Protokollierung (S)

Für alle Buckets, die schützenswerte Daten enthalten, SOLLTE die Zugriffs-Protokollierung aktiviert werden. Alle API-Zugriffe (lesend, schreibend, löschend) SOLLTEN protokolliert und die Logs an ein zentrales System zur Auswertung weitergeleitet werden.

### APP.4.9.A8 Nutzung der Objekt-Versionierung (S)

Für Buckets mit kritischen Daten SOLLTE die Objekt-Versionierung aktiviert werden, um Objekte vor unbeabsichtigtem Überschreiben oder Löschen zu schützen und die Wiederherstellung früherer Versionen zu ermöglichen.

### APP.4.9.A9 Regelmäßige Überprüfung der Bucket-Berechtigungen (S)

Die Konfiguration aller Bucket-Richtlinien SOLLTE regelmäßig, vorzugsweise automatisiert, daraufhin überprüft werden, ob sie öffentliche oder übermäßig freizügige Zugriffe erlauben.

---

## 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.4.9.A10 Einsatz von clientseitiger Verschlüsselung (Client-Side Encryption) (H)

Für Objekte mit hochsensiblen Daten SOLLTE eine clientseitige Verschlüsselung eingesetzt werden. Die Daten MÜSSEN von der Anwendung verschlüsselt werden, bevor sie an den Objektspeicher übertragen werden. Die Schlüssel dürfen dem Betreiber des Speichersystems nicht bekannt sein.

### APP.4.9.A11 Durchsetzung von zentralen Sicherheitsrichtlinien (Policy Enforcement) (H)

Es SOLLTEN übergeordnete, technische Richtlinien (z. B. Service Control Policies) durchgesetzt werden, die es Benutzern unmöglich machen, unsichere Konfigurationen vorzunehmen. Solche Richtlinien können beispielsweise die Erstellung öffentlicher Buckets oder das Speichern unverschlüsselter Objekte für die gesamte Organisation unterbinden.

### APP.4.9.A12 Absicherung des Zugriffs über private Endpunkte (H)

Der Zugriff auf die API des Objektspeichers SOLLTE, wann immer möglich, über private Netzwerkverbindungen erfolgen und nicht über das öffentliche Internet. Dies kann durch private Endpunkte innerhalb eines virtuellen Netzwerks oder über dedizierte Leitungen realisiert werden.

### APP.4.9.A13 Aktivierung der Objektsperre (Object Lock) (H)

Für Daten, die aus Compliance- oder Revisionsgründen unveränderbar gespeichert werden müssen (WORM: Write Once, Read Many), SOLLTE die Funktion zur Objektsperre (Object Lock) genutzt werden. Diese verhindert die Löschung oder Änderung eines Objekts für einen festgelegten Zeitraum.