



CON: Konzepte und Vorgehensweisen

# CON.10: Entwicklung von Webanwendungen

## 1 Beschreibung

### 1.1 Einleitung

Webanwendungen bieten Anwendern bestimmte Funktionen und dynamische (sich verändernde) Inhalte. Dazu stellen Webanwendungen auf einem Server Dokumente und Benutzeroberflächen, z. B. in Form von Eingabemasken, bereit und liefern diese auf Anfrage an entsprechende Programme auf den Clients aus, z. B. an Webbrowser. Webanwendungen werden gewöhnlich auf der Grundlage von Frameworks (Rahmenstrukturen) entwickelt. Frameworks sind wiederverwendbare Programmschablonen für häufig wiederkehrende Aufgaben. Auch für Sicherheitskomponenten gibt es Frameworks.

Webanwendungen müssen Sicherheitsmechanismen umsetzen, die den Schutz der verarbeiteten Informationen gewährleisten und deren Missbrauch verhindern. Typische Sicherheitskomponenten bzw. -mechanismen sind Authentisierung, Autorisierung, Eingabevalidierung, Ausgabekodierung, Session-Management, Fehlerbehandlung und Protokollierung.

Entwickler müssen mit den relevanten Sicherheitsmechanismen einer Webanwendung vertraut sein. Aus diesem Grund hat der Entwicklungsprozess einen entscheidenden Einfluss auf die Sicherheit der späteren Software.

### 1.2 Zielsetzung

Ziel dieses Bausteins ist es, sichere Webanwendungen zu entwickeln sowie Informationen zu schützen, die durch eine Webanwendung verarbeitet werden.

### 1.3 Abgrenzung und Modellierung

Der Baustein ist auf den Informationsverbund anzuwenden, wenn Webanwendungen entwickelt werden.

In diesem Baustein werden die spezifischen Gefährdungen und Anforderungen betrachtet, die bei der Entwicklung von Webanwendungen relevant sind. Anforderungen an den sicheren Einsatz von Webanwendungen werden nicht in diesem Baustein betrachtet. Sie sind im Baustein APP.3.1 *Webanwendungen* zu finden. Ebenso werden allgemeine Anforderungen an die sichere Entwicklung von Software an anderer Stelle behandelt. Sie werden im Baustein CON.8 *Softwareentwicklung* zu betrachtet.

## 2 Gefährdungslage

Folgende spezifische Bedrohungen und Schwachstellen sind für den Baustein CON.10 *Entwicklung von Webanwendungen* von besonderer Bedeutung.

### 2.1 Umgehung der Autorisierung bei Webanwendungen

Angreifer versuchen häufig, auf Funktionen oder Daten von Webanwendungen zuzugreifen, die nur für eine eingeschränkte Benutzergruppe verfügbar sind. Ist die Autorisierung fehlerhaft umgesetzt, kann ein Angreifer unter Umständen die Berechtigungen eines anderen Benutzers mit umfangreicheren Rechten erlangen und somit auf geschützte Bereiche und Daten zugreifen. Dies geschieht beispielsweise, indem der Angreifer seine eigenen Eingaben gezielt manipuliert.

### 2.2 Unzureichende Eingabvalidierung und Ausgabekodierung

Verarbeitet eine Webanwendung ungeprüfte Eingabedaten, können möglicherweise Schutzmechanismen umgangen werden. Dieses Risiko erhöht sich, wenn gleichzeitig die Ausgabedaten der Webanwendung ohne ausreichende Kodierung direkt an den Webbrowser des Benutzers, die aufrufende Anwendung oder an nachgelagerte IT-Systeme übermittelt werden. Solche Ausgabedaten können Schadcode enthalten, der auf den Zielsystemen interpretiert oder ausgeführt wird. Beispielsweise kann ein Angreifer Javascript Code in Formulardaten eingeben. Dieser Schadcode wird dann ungewollt vom IT-System eines anderen Benutzers ausgeführt.

### 2.3 Fehlende oder mangelhafte Fehlerbehandlung durch Webanwendungen

Wenn während des Betriebs einer Webanwendung Fehler auftreten, die nicht korrekt behandelt werden, können sowohl der Betrieb einer Webanwendung als auch der Schutz ihrer Funktionen und Daten beeinträchtigt werden. Beispielsweise kann ein Fehler dazu führen, dass die Webanwendung nicht mehr korrekt ausgeführt wird und für Clients nicht mehr erreichbar ist. Außerdem werden unter Umständen Aktionen nur noch unvollständig durchgeführt, zwischengespeicherte Aktionen und Daten gehen verloren oder Sicherheitsmechanismen fallen aus.

### 2.4 Unzureichende Protokollierung von sicherheitsrelevanten Ereignissen

Wenn sicherheitsrelevante Ereignisse von der Webanwendung unzureichend protokolliert werden, können Sicherheitsvorfälle zu einem späteren Zeitpunkt nur schwer nachvollzogen werden. Die Ursachen für ein Vorfall sind dann möglicherweise nicht mehr ermittelbar. Beispielsweise können Konfigurationsfehler übersehen werden, wenn sie nicht zu Fehlermeldungen in den Log-Dateien führen. Auch Schwachstellen sind bei unzureichender Protokollierung schwer oder gar nicht zu erkennen und zu beheben.

### 2.5 Offenlegung sicherheitsrelevanter Informationen durch Webanwendungen

Webseiten und Daten, die von einer Webanwendung generiert und ausgeliefert werden, können Informationen zu den Hintergrundsystemen enthalten, z. B. Angaben zu Datenbanken oder Versionsständen von Frameworks. Diese Informationen können es einem Angreifer erleichtern, gezielte Angriffe auf die Webanwendung durchzuführen.

### 2.6 Missbrauch einer Webanwendung durch automatisierte Nutzung

Wenn ein Angreifer Funktionen einer Webanwendung automatisiert nutzt, kann er zahlreiche Vorgänge in kurzer Zeit ausführen. Mithilfe eines wiederholt durchgeführten Login-Prozesses kann der Angreifer z. B. versuchen, gültige Kombinationen von Benutzernamen und Passwörtern zu erraten (Brute-Force). Außerdem kann er Listen mit gültigen Benutzernamen erzeugen (Enumeration), falls die Webanwendung Informationen über vorhandene Benutzer zurück gibt. Darüber hinaus können wiederholte Aufrufe von ressourcenintensiven Funktionen wie z. B. komplexen Datenbankabfragen für Denial-of-Service-Angriffe auf Anwendungsebene missbraucht werden.

## 2.7 Unzureichendes Session-Management von Webanwendungen

Unzureichendes Session-Management kann es einem Angreifer ohne spezielle Zugriffsrechte ermöglichen, die Session-ID eines Benutzers mit umfangreichen Zugriffsrechten zu ermitteln. Anschließend kann der Angreifer mit dieser ID auf geschützte Funktionen und Ressourcen der Webanwendung zugreifen, z. B. in Form eines Session-Fixation-Angriffs. Hier lässt sich der Angreifer zunächst eine Session-ID mit eingeschränkten Benutzerrechten von der Webanwendung zuweisen. Diese ID übermittelt er, z. B. über einen Link in einer E-Mail, an den höher legitimen Benutzer. Falls der höher legitimierte Benutzer diesem Link folgt und sich gegenüber der Webanwendung unter der Session-ID des Angreifers authentisiert, kann der Angreifer die Anwendung anschließend mit den vollen Zugriffsrechten des legitimen Benutzers verwenden.

## 3 Anforderungen

Im Folgenden sind die spezifischen Anforderungen des Bausteins CON.10 *Entwicklung von Webanwendungen* aufgeführt. Grundsätzlich ist der Entwickler für die Erfüllung der Anforderungen zuständig. Der Informationssicherheitsbeauftragte (ISB) ist bei strategischen Entscheidungen stets einzubeziehen. Außerdem ist der ISB dafür zuständig, dass alle Anforderungen gemäß dem festgelegten Sicherheitskonzept erfüllt und überprüft werden. Zusätzlich kann es noch andere Rollen geben, die weitere Zuständigkeiten bei der Erfüllung von Anforderungen haben. Diese sind dann jeweils explizit in eckigen Klammern in der Überschrift der jeweiligen Anforderungen aufgeführt.

Zuständigkeiten	Rollen
Grundsätzlich zuständig	Entwickler
Weitere Zuständigkeiten	

### 3.1 Basis-Anforderungen

Die folgenden Anforderungen MÜSSEN für den Baustein CON.10 *Entwicklung von Webanwendungen* vorrangig erfüllt werden:

#### CON.10.A1 Authentisierung bei Webanwendungen (B)

Die Entwickler MÜSSEN sicherstellen, dass sich Benutzer gegenüber der Webanwendung sicher und angemessen authentisieren, bevor diese auf geschützte Funktionen oder Inhalte zugreifen können. Es MUSS eine angemessene Authentisierungsmethode ausgewählt werden. Der Auswahlprozess MUSS dokumentiert werden.

Eine zentrale Authentisierungskomponente MUSS verwendet werden. Die zentrale Authentisierungskomponente SOLLTE mit etablierten Standardkomponenten (z. B. aus Frameworks oder Programmbibliotheken) umgesetzt werden. Falls eine Webanwendung Authentisierungsdaten auf einem Client speichert, MUSS der Benutzer explizit auf die Risiken der Funktion hingewiesen werden und zustimmen („Opt-In“).

Die Webanwendung MUSS die Möglichkeit bieten, Grenzwerte für fehlgeschlagene Anmeldeversuche festzulegen. Die Webanwendung MUSS Benutzer sofort informieren, wenn ihr Passwort zurückgesetzt wurde.

#### CON.10.A2 Zugriffskontrolle bei Webanwendungen (B)

Die Entwickler MÜSSEN mittels einer Autorisierungskomponente sicherstellen, dass Benutzer ausschließlich solche Aktionen durchführen können, zu denen sie berechtigt sind. Jeder Zugriff auf geschützte Inhalte und Funktionen MUSS kontrolliert werden, bevor er ausgeführt wird.

Die Autorisierungskomponente MUSS sämtliche Ressourcen und Inhalte berücksichtigen, die von der

Webanwendung verwaltet werden. Ist die Zugriffskontrolle fehlerhaft, MÜSSEN Zugriffe abgelehnt werden. Es MUSS eine Zugriffskontrolle bei URL-Aufrufen und Objekt-Referenzen geben.

#### **CON.10.A3            Sicheres Session-Management (B)**

Die Entwickler MÜSSEN sicherstellen, dass Session-IDs geeignet geschützt werden. Session-IDs MÜSSEN zufällig und mit ausreichender Entropie erzeugt werden. Falls das Framework der Webanwendung sichere Session-IDs generieren kann, MUSS diese Funktion des Frameworks von den Entwicklern verwendet werden. Sicherheitsrelevante Konfigurationsmöglichkeiten des Frameworks MÜSSEN berücksichtigt werden. Wenn Session-IDs übertragen und von den Clients gespeichert werden, MÜSSEN sie ausreichend geschützt übertragen werden.

Eine Webanwendung MUSS den Benutzern die Möglichkeit geben, eine bestehende Sitzung explizit zu beenden. Nachdem sich der Benutzer angemeldet hat, MUSS eine bereits bestehende Session-ID durch eine neue ersetzt werden. Sitzungen MÜSSEN eine maximale Gültigkeitsdauer besitzen (Timeout). Inaktive Sitzungen MÜSSEN automatisch nach einer bestimmten Zeit ungültig werden. Nachdem die Sitzung ungültig ist, MÜSSEN alle Sitzungsdaten ungültig und gelöscht sein.

#### **CON.10.A4            Kontrolliertes Einbinden von Inhalten bei Webanwendungen (B)**

Die Entwickler MÜSSEN sicherstellen, dass eine Webanwendung ausschließlich vorgesehene Daten und Inhalte einbindet und an den Benutzer ausliefert.

Die Ziele der Weiterleitungsfunktion einer Webanwendung MÜSSEN ausreichend eingeschränkt werden, sodass Benutzer ausschließlich auf vertrauenswürdige Webseiten weitergeleitet werden. Verlässt ein Benutzer die Vertrauensdomäne, MUSS ihn die Webanwendung darüber informieren.

#### **CON.10.A5            Upload-Funktionen (B)**

Die Entwickler MÜSSEN sicherstellen, dass ein Benutzer Dateien nur im vorgegebenen Pfad speichern kann. Die Entwickler MÜSSEN sicherstellen, dass der Benutzer den Ablageort der Uploads nicht beeinflussen kann. Die Entwickler MÜSSEN Funktionen in die Webanwendung integrieren, mit denen der spätere Betreiber der Webanwendung die Uploads konfigurieren kann.

#### **CON.10.A6            Schutz vor unerlaubter automatisierter Nutzung von Webanwendungen (B)**

Die Entwickler MÜSSEN Sicherheitsmechanismen implementieren, die die Webanwendung vor automatisierten Zugriffen schützen. Die Entwickler MÜSSEN bei der Implementierung der Sicherheitsmechanismen berücksichtigen, wie sich diese auf die Nutzungsmöglichkeiten berechtigter Benutzer auswirken.

#### **CON.10.A7            Schutz vertraulicher Daten (B)**

Die Entwickler MÜSSEN sicherstellen, dass vertrauliche Daten von den Clients zu den Servern nur mit der HTTP-Post-Methode übertragen werden.

Die Entwickler MÜSSEN durch Direktiven in der Webanwendung gewährleisten, dass clientseitig keine schützenswerten Daten zwischengespeichert werden. Die Entwickler MÜSSEN sicherstellen, dass in Formularen keine vertraulichen Formulardaten im Klartext angezeigt werden. Die Webanwendung SOLLTE verhindern, dass vertrauliche Daten vom Webbrowser unerwartet gespeichert werden. Sämtliche Zugangsdaten der Webanwendung MÜSSEN serverseitig mithilfe von sicheren kryptografischen Algorithmen vor unbefugtem Zugriff geschützt werden (Salted Hash). Die Dateien mit den Quelltexten der Webanwendung MÜSSEN vor unerlaubten Abrufen geschützt werden.

#### **CON.10.A8            Umfassende Eingabvalidierung und Ausgabekodierung (B)**

Die Entwickler MÜSSEN sämtliche an eine Webanwendung übergebenen Daten als potenziell gefährlich behandeln und geeignet filtern. Sämtliche Eingabedaten sowie Datenströme und Sekundärdaten, wie z. B. Session-IDs, MÜSSEN serverseitig validiert werden.

Fehleingaben SOLLTEN möglichst nicht automatisch behandelt werden (Sanitizing). Lässt es sich

jedoch nicht vermeiden, MUSS Sanitizing sicher umgesetzt werden.

Ausgabedaten MÜSSEN so kodiert werden, dass schadhafter Code auf dem Zielsystem nicht interpretiert oder ausgeführt wird.

#### **CON.10.A9 Schutz vor SQL-Injection (B)**

Falls Daten an ein Datenbankmanagementsystem (DBMS) weitergeleitet werden, MÜSSEN die Entwickler Stored Procedures bzw. Prepared SQL Statements einsetzen. Falls Daten an ein DMBS weitergeleitet werden und weder Stored Procedures noch Prepared SQL Statements von der Einsatzumgebung unterstützt werden, MÜSSEN die SQL-Queries separat abgesichert werden.

#### **CON.10.A10 Restriktive Herausgabe sicherheitsrelevanter Informationen (B)**

Die Entwickler MÜSSEN sicherstellen, dass Webseiten, Rückantworten und Fehlermeldungen von Webanwendungen keine Informationen enthalten, die einem Angreifer Hinweise darauf geben, wie er Sicherheitsmechanismen umgehen kann.

### **3.2 Standard-Anforderungen**

Gemeinsam mit den Basis-Anforderungen entsprechen die folgenden Anforderungen dem Stand der Technik für den Baustein CON.10 *Entwicklung von Webanwendungen*. Sie SOLLTEN grundsätzlich erfüllt werden.

#### **CON.10.A11 Softwarearchitektur einer Webanwendung (S)**

Die Entwickler SOLLTEN die Softwarearchitektur der Webanwendung mit allen Bestandteilen und Abhängigkeiten dokumentieren. Die Dokumentation SOLLTE bereits während des Entwicklungsverlaufs aktualisiert und angepasst werden. Die Dokumentation SOLLTE so gestaltet sein, dass sie schon in der Entwicklungsphase benutzt werden kann und Entscheidungen nachvollziehbar sind. In der Dokumentation SOLLTEN alle für den Betrieb notwendigen Komponenten gekennzeichnet werden, die nicht Bestandteil der Webanwendung sind. In der Dokumentation SOLLTE beschrieben sein, welche Komponenten welche Sicherheitsmechanismen umsetzen, wie die Webanwendung in eine bestehende Infrastruktur integriert wird und welche kryptografischen Funktionen und Verfahren eingesetzt werden.

#### **CON.10.A12 Verifikation essenzieller Änderungen (S)**

Falls wichtige Einstellungen mit der Anwendung geändert werden sollen, dann SOLLTEN die Entwickler sicherstellen, dass die Änderungen durch die Eingabe eines Passworts erneut verifiziert werden. Falls dies nicht möglich ist, dann SOLLTE die Webanwendung auf andere geeignete Weise sicherstellen, dass sich die Benutzer authentisieren. Die Benutzer SOLLTEN über Änderungen mithilfe von Kommunikationswegen außerhalb der Webanwendung informiert werden.

#### **CON.10.A13 Fehlerbehandlung (S)**

Treten während der Laufzeit einer Webanwendung Fehler auf, SOLLTEN Entwickler diese so behandeln, dass die Webanwendung weiter in einem konsistenten Zustand bleibt.

Die Webanwendung SOLLTE Fehlermeldungen protokollieren. Falls eine veranlasste Aktion einen Fehler verursacht, SOLLTE die Webanwendung diese Aktion abbrechen. Die Webanwendung SOLLTE im Fehlerfall den Zugriff auf eine angeforderte Ressource oder Funktion verweigern.

Zuvor reservierte Ressourcen SOLLTEN im Rahmen der Fehlerbehandlung wieder freigegeben werden. Der Fehler SOLLTE möglichst von der Webanwendung selbst behandelt werden.

#### **CON.10.A14 Sichere HTTP-Konfiguration bei Webanwendungen (S)**

Zum Schutz vor Clickjacking, Cross-Site-Scripting und anderen Angriffen SOLLTEN die Entwickler geeignete HTTP-Response-Header setzen. Es SOLLTEN mindestens die folgenden HTTP-Header verwendet werden: Content-Security-Policy, Strict-Transport-Security, Content-Type, X-Content-Type-Options sowie Cache-Control. Die verwendeten HTTP-Header SOLLTEN auf die Webanwendung abgestimmt werden. Die verwendeten HTTP-Header SOLLTEN so restriktiv wie möglich sein.

Cookies SOLLTEN grundsätzlich mit den Attributen *secure*, *SameSite* und *httponly* gesetzt werden.

#### **CON.10.A15      Verhinderung von Cross-Site-Request-Forgery (S)**

Die Entwickler SOLLTEN die Webanwendung mit solchen Sicherheitsmechanismen ausstatten, die eine Unterscheidung zwischen beabsichtigten Seitenaufrufen des Benutzers von unbeabsichtigt weitergeleiteten Befehlen Dritter ermöglichen. Dabei SOLLTE mindestens geprüft werden, ob neben der Session-ID ein geheimes Token für den Zugriff auf geschützte Ressourcen und Funktionen benötigt wird.

#### **CON.10.A16      Mehr- Faktor- Authentisierung (S)**

Die Entwickler SOLLTEN eine Mehr-Faktor-Authentisierung implementieren.

### **3.3 Anforderungen bei erhöhtem Schutzbedarf**

Im Folgenden sind für den Baustein CON.10 *Entwicklung von Webanwendungen* exemplarische Vorschläge für Anforderungen aufgeführt, die über das dem Stand der Technik entsprechende Schutzniveau hinausgehen und BEI ERHÖHTEM SCHUTZBEDARF in Betracht gezogen werden SOLLTEN. Die konkrete Festlegung erfolgt im Rahmen einer Risikoanalyse.

#### **CON.10.A17      Verhinderung der Blockade von Ressourcen (H)**

Zum Schutz vor Denial-of-Service (DoS)-Angriffen SOLLTEN ressourcenintensive Operationen vermieden werden. Falls ressourcenintensive Operationen notwendig sind, dann SOLLTEN diese besonders abgesichert werden. Bei Webanwendungen SOLLTE ein möglicher Überlauf von Protokollierungsdaten überwacht und verhindert werden.

#### **CON.10.A18      Kryptografische Absicherung vertraulicher Daten (H)**

Entwickler SOLLTEN sicherstellen, dass vertrauliche Daten einer Webanwendung durch sichere, kryptografische Algorithmen abgesichert werden.

## **4 Weiterführende Informationen**

### **4.1 Wissenswertes**

Das Open Web Application Security Projekt stellt auf seiner Webseite Hinweise zur Absicherung von Webanwendungen zur Verfügung.

Das Bundesamt für Sicherheit in der Informationstechnik (BSI) stellt im Dokument „Kryptographische Verfahren: Empfehlungen und Schlüssellängen: BSI TR-02102“ Hinweise zur Anwendung kryptografischer Verfahren zur Verfügung.

Das Bundesamt für Sicherheit in der Informationstechnik (BSI) stellt im Dokument „Entwicklung sicherer Webanwendungen“ Hinweise zur Entwicklung sicherer Webanwendungen zur Verfügung.

Das Bundesamt für Sicherheit in der Informationstechnik (BSI) stellt im Dokument „Leitfaden zur Entwicklung sicherer Webanwendungen“ Hinweise zur Entwicklung sicherer Webanwendungen durch Unternehmen zur Verfügung.

## **5 Anlage: Kreuzreferenztafel zu elementaren Gefährdungen**

Die Kreuzreferenztafel enthält die Zuordnung von elementaren Gefährdungen zu den Anforderungen. Anhand dieser Tabelle lässt sich ermitteln, welche elementaren Gefährdungen durch welche Anforderungen abgedeckt sind. Durch die Umsetzung der aus den Anforderungen abgeleiteten Sicherheitsmaßnahmen wird den entsprechenden elementaren Gefährdungen entgegengewirkt. Die Buchstaben in der zweiten Spalte (C = Vertraulichkeit, I = Integrität, A = Verfügbarkeit) zeigen an, welche Grundwerte der Informationssicherheit durch die Anforderung vorrangig geschützt werden. Die folgenden elementaren Gefährdungen sind für den Baustein CON.10 *Entwicklung von*

*Webanwendungen* von Bedeutung.

G 0.18 Fehlplanung oder fehlende Anpassung

G 0.28 Software-Schwachstellen oder -Fehler