Sicherheit von SCOPELAND-Web-Anwendungen

Das Thema Sicherheit ist eines der Kernthemen der IT-Branche und zählt zu den notwendigen Schlüsselkompetenzen jedes Software-Anbieters, der langfristig auf dem Markt bestehen will.

SCOPELAND® unterstützt die Entwicklung sicherer Web-Anwendungen durch die „Out-of-The-Box" Verwendung verschiedener Sicherheitsfeatures und das Bereithalten eines Toolsets, um die Sicherheit für die jeweilige Anwendung optimal anzupassen. Die Funktionen werden ständig verbessert, um auf aktuelle Entwicklungen vorbereitet zu sein, und um künftige Trends zu antizipieren. Dabei orientiert sich Scopeland Technology an den gängigen Industriestandards und den Empfehlungen anerkannter Experten auf dem Gebiet der Web-Sicherheit, wie denen des Open Web Application Security Project (OWASP) und des Bundesamts für Sicherheit in der Informationstechnik (BSI).

In diesem White Paper sollen die wichtigsten Sicherheitsfeatures, Lösungen sowie deren Vorteile, die SCOPELAND® unterstützt und anbietet, vorgestellt werden.

Inhalt

1. Schutz vertraulicher Daten

Vertrauliche Daten und Funktionen für einen eingeschränkten Benutzerkreis sind Kernbestandteile der meisten Web-Anwendungen. Dies stellt Anbieter von Web-Anwendungen vor die Aufgabe, einen scheinbaren Widerspruch aufzulösen: Eine Web-Anwendung soll per Definition aus dem öffentlichen und für jeden Zugänglichen „Web“ erreichbar sein, auf der anderen Seite vertrauliche Daten und Funktionen vor unberechtigtem Zugriff schützen.

1.1. Authentifizierung von Benutzern

Typischerweise muss ein Benutzer einer Web-Anwendung sich per Benutzername und Passwort einloggen. SCOPELAND® bietet standardmäßig an, Daten gegen eine Datenbank oder per Lightweight Directory Access Protocol (LDAP) / Active Directory (AD) zu prüfen. Passwörter werden dafür verschlüsselt („gehasht“) abgespeichert. Dies erschwert ein illegales Auslesen der Passwörter, falls sich ein Angreifer lesenden Zugriff auf die Benutzerdaten verschafft.

Alle größeren Webserver bieten ausgereifte Standardlösungen für die Authentifizierung von Usern an. SCOPELAND® unterstützt diese Lösungen und erweitert sie sogar noch: So ist neben den gängigen Hashing-Algorithmen für Passwörter MD5 und SHA auch der wesentlich sicherere PBKDF2 Algorithmus mit zusätzlichem Salt möglich. Damit werden neben einfachen Brute-Force-Angriffen auch fortschrittlichere Rainbow-Table-Angriffe unwirtschaftlich und quasi ausgeschlossen.

Zudem kann per SCOPELAND® eine Password Policy definiert werden, um einem der größten Risiken von Web-Anwendungen entgegen zu wirken: Durch den Nutzer selbst gewählte, schwache und daher unsichere Passwörter.

1.2. Autorisierung des Benutzers für bestimmte Ressourcen

SCOPELAND® bedient sich des Rollenkonzepts, welches von den meisten Webservern angeboten wird. Der Webserver stellt sicher, dass nur Nutzer, die über gewisse Rollen verfügen, Zugriff auf bestimmte Seiten oder Ordner sowie den darin hinterlegten Daten, Ressourcen und Funktionen haben.

Für viele Anwendungsbereiche ist dies jedoch zu grob. Daher bietet SCOPELAND® weiter die Möglichkeit, innerhalb der Anwendungslogik dynamisch zu entscheiden, ob ein Nutzer Zugriff auf bestimmte Daten, Ressourcen oder Funktionen erhält. SCOPELAND® bietet hierfür entsprechende Einstellungsmöglichkeiten und Application-Programming-Interfaces (APIs) an, um dem Anwendungsentwickler die nötigen Andockpunkte für die zu implementierende Geschäftslogik zu bieten. So kann beispielsweise vor besonders sicherheitskritischen Tätigkeiten nochmals eine Authentifizierung erzwungen werden, um sicherzustellen,

dass tatsächlich noch der berechtigte Nutzer die Anwendung bedient. Weiter lassen sich die Berechtigungen der Nutzer für die Verwendung in der Applet-Logik auslesen und Nutzer können zwangsweise abgemeldet werden.

1.3. Schutz der Kommunikation zwischen Benutzer und Server

Eine Verschlüsselung der Kommunikation zwischen dem Browser des Benutzers und des Servers ist notwendig, um Vertraulichkeit zu erreichen. SCOPELAND® empfiehlt und unterstützt hier den Industriestandard **Transport Layer Security** (**TLS)**, der besser bekannt ist unter seiner Vorgängerbezeichnung **Secure Sockets Layer** / **SSL**.

Um **Man-In-The-Middle-Angriffe** zu unterbinden, werden SCOPELAND®-Anwendungen in der Regel so konfiguriert, dass sie strikt auf eine korrekte Signierung der Zertifikate durch eine **Certificate Authority** achten.

1.4. Verschlüsselung von sensiblen Daten

Sensible Daten sollten auch in der Datenbank nicht im Klartext lesbar sein. Neben dem bereits angesprochenen asymmetrischen Hashing-Verfahren, was für Passwörter verwendet wird, brauchen andere Daten eine symmetrische Verschlüsselung. Die großen Datenbanksysteme bieten dafür Funktionen an, um Teile oder auch die ganze Datenbank zu verschlüsseln. Scopeland Technology empfiehlt, diese Funktionen zu nutzen, wenn mit besonders sensiblen Daten umgegangen wird. Hierbei sollte beachtet werden, dass die Ent-/Verschlüsselung zu Lasten der Performance geht und daher nur überlegt eingesetzt werden sollte.

2. Sicherheit bei Ein-/Ausgabe von Daten in Web-Anwendungen

Die Möglichkeit für Benutzer, Daten an die Anwendung zur weiteren Verarbeitung einzugeben ist essentiell für moderne Web-Anwendungen. Hiermit bietet sich aber auch ein mögliches Einfallstor für Angriffe, wenn die Anwendung solche Eingaben nicht mit der notwendigen Sorgfalt prüft und als Gefahrenquelle behandelt. So könnte im schlimmsten Fall Programmcode eines Angreifers zur Ausführung gebracht werden.

2.1. Eingabevalidierung

Der erste Schritt umfasst eine konsequente **Validierung** der Eingaben. SCOPELAND® unterstützt die Standard-Funktionen der jeweiligen Plattform (JSF, .Net), durch die bereits eine zufriedenstellende Qualität der Validierung erreicht wird. Standard-Fälle wären zum Beispiel, dass in Zahlen-Feldern nur Zahlen eingegeben werden können, Texte einer bestimmten Form und Länge entsprechen, oder Werte nach einem Whitelist-Verfahren geprüft werden. Darüber hinaus bietet SCOPELAND® die Möglichkeit, **Regular Expression als Validierungs-Regeln** zu definieren. Damit erhält der Anwendungsentwickler ein mächtiges Werkzeug, um die Validierung auch per Muster und nicht nur per Whitelist-Verfahren implementieren zu können.

2.2. Validierung von Uploads

Für die Validierung von Datei-Uploads steht die Möglichkeit zur Verfügung, das Upload auf bestimmte Dateitypen zu beschränken. Dabei verlässt sich SCOPELAND® nicht allein auf die Datei-Endung, sondern prüft auch den Datei-Inhalt. Damit kann zuverlässig verhindert werden, dass Schadcode über Skripte oder ähnlichem hochgeladen werden kann und danach (in der Regel über Ausnutzung irgendeiner Sicherheitslücke) ausgeführt werden kann.

2.3. Spezialbehandlung für Parameter

Auch nach der Validierung müssen Benutzereingaben immer als potentiell gefährlich angesehen werden. In SCOPELAND®-Anwendungen werden Benutzereingaben als **SQL-Parameter** mit SQL verknüpft, wodurch wirksam **SQL-Injections** verhindert werden.

2.4. Escaping von Ausgabe-Daten

Wenn Daten zum Aufbau einer Webseite benutzt werden, muss immer beachtet werden, dass solche Daten manipuliert sein könnten – mit dem Ziel, der Webseite in unerlaubter Weise zu schaden. Damit dies

nicht passiert, unterstützt SCOPELAND® das **Escaping von Ausgabe-Daten**: Etwaige Steuerungs- oder Sonderzeichen, welche die Darstellung ungewollt beeinflussen könnten, werden in eine harmlose Variante umgewandelt. Dies verhindert unter anderem das weit verbreitete **Cross-Site-Scripting**.

Da in Spezialfällen das Escaping hinderlich sein kann, kann dieses feingranular gesteuert werden, um dem Anwendungsentwickler zu ermöglichen, an den nötigen Stellen die Ausgabe bewusst dynamisch umzubauen. Dieser Fall muss vom Anwendungsentwickler bewusst aktiviert und mit besonderer Sorgfalt behandelt werden.

2.5. URL-Parameter

Nicht nur Eingaben per Web-Formular sind eine Gefahrenquelle. Auch das verbreitete Übergeben von Parametern per URL ist ein möglicher Angriffsvektor. Hierfür besitzt SCOPELAND® ein besonderes Feature: Die **Verschlüsselung der URL-Parameter**. Durch eine symmetrische Verschlüsselung werden die URL-Parameter für einen potentiellen Angreifer nicht mehr lesbar, was eine unerlaubte Manipulation der URL-Parameter erheblich erschwert.

3. Cross-Site-Angriffe

Es existieren verschiedene sogenannte „Cross-Site"-Angriffsmethoden, die darauf basieren, eine gültige Session eines legal angemeldeten Benutzers auszunutzen – durch das Unterschieben von illegalen Requests. Dazu gehören **Cross-Site Request Forgery (XSRF)**, **Session Riding, Clickjacking** und das bereits erwähnte **Cross-Site-Scripting (XSS)**.

SCOPELAND® setzt mehrere Techniken ein, um solche Angriffe zu verhindern. So wird in SCOPELAND®-Anwendungen die vom *World Wide Web Consortium* (W3C) empfohlene **Content Security Police (CSP)** realisiert. Der Zugriff auf Webseiten außerhalb der eigenen Anwendung ist damit standardmäßig nicht erlaubt. Auch per **X-Frame-Options** wird die **Same-Origin-Policy** verwendet, was Cross-Site-Angriffe oder das Laden von Ressourcen anderer Seiten stark erschwert. Vom Anwendungsentwickler kann der Zugriff in eigener Verantwortung pauschal oder gezielt für bestimmte Webseiten frei gegeben werden, falls nötig. Auch die Verwendung eines **Anti-XSRF-Tokens** kann aktiviert werden. Damit wird für jede User-Session ein kryptografisch sicherer Token erzeugt, welcher als Sicherheitsmerkmal in alle Seiten eingefügt und in allen Requests des Users erwartet wird. Illegale, aus anderer Quelle untergeschobene Requests können so erkannt werden.

4. Session-ID-Sicherheit

Um zu verhindern, dass die im Cookie gesetzte Session-ID ausgelesen wird, verwenden SCOPELAND®-Web-Anwendungen standardmäßig das **HttpOnly-Flag**. Die Session-ID wird nach dem ersten Anmelden neu vergeben und nicht über die URL sichtbar gemacht. Die Session-ID wird mit kryptografisch sicheren Algorithmen generiert, was **Brute-Force-Attacken** gegen die Session-ID praktisch unmöglich macht und **Session Hijacking** damit stark erschwert.

Natürlich unterstützten SCOPELAND®-Web-Anwendungen einen **Session-Timeout**. Es wird empfohlen, die Timeout-Zeit minimal zu wählen – im Kontext der Anforderungen.

5. Ausfallsicherheit

Ob durch Hacking-Angriffe oder durch schlichtes technisches Versagen: Server und Hardware können ausfallen. Damit in diesem Fall nicht die ganze Anwendung bis zu einer Reparatur nicht verfügbar ist, empfiehlt Scopeland Technology, die Anwendungen immer auf mehrere Server zu verteilen. SCOPELAND®-Web-Anwendungen sind Cluster-fähig und können daher auf mehrere Rechner verteilt werden. Zudem sind SCOPELAND®-Anwendungen Cloud-fähig und können folglich von geeigneten Cloud-Anbietern gehostet werden.

6. Angriffe durch Überlastung: Distributed-Denial-of-Service-Angriffe (DDoS)

Distributed-Denial-of-Service-Angriffe (DDoS, auch verkürzt **Denial of Service**, **DoS**) sind selbst für große Firmen ein Problem. Da hier die Infrastruktur angegriffen wird, und nicht die Anwendung selbst, kann Scopeland Technology hierfür keine direkte Lösung erarbeiten, bietet jedoch eine umfassende Beratung an, welche Maßnahmen ergriffen werden sollten, um auf DoS-Angriffe vorbereitet zu sein.

Spezielle Filter-Software für Firewalls, Router und Proxys kann eingesetzt werden, um DoS-Attacken zu erkennen und deren Auswirkungen zumindest stark zu mindern. Hierfür kann der Traffic auch über externe Filter-Services darauf spezialisierter Firmen geleitet werden.

Gegen viele DoS-Angriffe reicht aber bereits eine Lastverteilung mit ausreichend dimensionierten Reserven. Wie bereits erwähnt, sind SCOPELAND®-Web-Anwendungen Cluster- und Cloud-fähig und können daher auf mehrere Rechner verteilt werden. Da gerade Cloud-Anbieter über sehr große Reserven an Rechenleistung verfügen, welche dynamisch hinzugeschaltet werden kann, ist dies ein guter Weg, gegen DoS-Attacken gewappnet zu sein.

7. Unerlaubte automatisierte Nutzung von Web-Anwendungen

Bei der Eingabe von ungültigen Anmeldedaten werden keine Informationen zurück geliefert, die darauf schließen lassen, ob nur das Passwort falsch war, oder ob auch das Benutzerkonto nicht existiert. Dadurch wird es automatisierten Brute-Force-Angriffen schwergemacht, Zugangsdaten zu ermitteln – bei ausreichend sicheren Passwörtern ist dies quasi unmöglich. Weiter gibt es, je nach Anmeldetyp SCOPELAND®-intern oder bereitgestellt durch das verwendete Authentifizierungsverfahren (wie beispielsweise Windows-Authentifizierung), die Möglichkeit, bei zu vielen Fehlversuchen den Zugang zu sperren oder die Anmeldeversuche zu verzögern.

Bei anderen Eingaben (außer der Anmeldung) muss der Anwendungsentwickler sicherstellen, dass bei Fehleingaben keine Antworten gegeben werden, die sicherheitskritische Informationen enthalten.

Für die Abwehr von automatisierter Nutzung kommen häufig auch CAPTCHAs (Completely Automated Public Turing Test To Tell Computers and Humans Apart) zum Einsatz. Da auf SCOPELAND®-Web-Anwendungen meist nur authentifizierte Nutzer Zugriff haben und daher nach erfolgter Anmeldung eine unerlaubte automatisierte Nutzung sehr unwahrscheinlich ist, gab es bisher keinen Bedarf für den Einsatz von CAPTCHAs. Mechanismen für die Einbindung von CAPTCHAs könnten jedoch bei Bedarf kurzfristig in das Produkt integriert werden.

Weiter sind Infrastrukturmaßnahmen wie Teergruben denkbar, um eine Anwendung vor automatisierter Nutzung zu schützen – beispielsweise auf der Basis von IP-Adressen. Scopeland Technology bietet hier keine eigene Lösung an, sondern eine umfassende Beratung, welche Maßnahmen ergriffen werden können.

8. Vom Anwendungsentwickler durchzuführende Maßnahmen

Einem SCOPELAND®-Anwendungsentwickler wird viel Arbeit für sicherheitsrelevante Konfiguration und Prüfung abgenommen. Diverse Abläufe funktionieren automatisch, so dass der Entwickler sich möglichst auf die Geschäftslogik konzentrieren kann. Vorgänge, wie das Setzen von Security-Headern, Ausgabe-Escaping, sichere Sessions-IDs erzeugen, Erzeugen von sicheren SQL-Abfragen und anderes, funktionieren ganz einfach im Hintergrund.

Trotzdem gibt es Dinge, die SCOPELAND® dem Anwendungsentwickler nicht abnehmen kann, und die er daher selbst sicherzustellen hat. Daraus ergeben sich folgende Vorgaben für die Anwendungsentwicklung:

Konfiguration

- Session-Timeout immer minimal im Kontext der Anforderungen wählen.
- Darauf achten, vor einem Produktivbetrieb alle Test- und Standardkonfigurationen zu entfernen, inklusive Test-Nutzer.
- Eine Passwortrichtlinie vergeben. Als Minimalanforderung sollte ein Passwort acht Zeichen lang sein, sowie Buchstaben und Zahlen enthalten.
- Sicherheitseinstellungen des Servers überprüfen. Wird durchgängig TLS/SSL verwendet? Sind nur benötigte Ports ansteuerbar? Korrekte Log-Einstellungen? Korrekte Fehler-Anzeigen?
- Management-Zugriff ggfs. nur aus Intranet möglich?
- **Fehlerseiten für Produktiveinsatz konfigurieren. Je nach Anforderungen Anti-XSRF-Token aktivieren und konfigurieren.**
- Neuere SCOPELAND®-Versionen benutzen auch neuere Software-Komponenten von Drittanbietern. Daher empfehlen wir, immer mit neuen SCOPELAND®-Versionen zu arbeiten.

Rechte-Management

- Seiten und andere Ressourcen (Dokumente, Vorlagen, XML-Dateien, etc.) nach Rollen-Zugriffsrechten in Ordnern sammeln und diesen Ordnern dann die Rechte für die Rollen geben.
- Es ist zu vermeiden, in einer Seite Funktionen zu verwenden, die für manche Rollen aktiv und für andere nicht aktiv sind, und dies dann selbst zu prüfen. Daher empfiehlt es sich, eine Seite pro Rolle zu erstellen, und den Server den Zugriff sichern zu lassen.
- Falls User innerhalb einer Seite unterschiedliche Rechte haben können (Herr Meier darf den Datensatz bearbeiten, Herr Müller nicht; Herr Meier darf Datensatz X sehen, Herr Müller nicht), sollte immer vor Ausführung der Aktion/Anzeige des Datensatzes geprüft werden, ob der Nutzer das entsprechende Recht hat.

Geschäftslogik

- Validierungsregeln für Eingabefelder sollten möglichst genau gesetzt werden.
- Wenn möglich, Benutzereingaben per Whitelist-Verfahren prüfen: „Ist diese Eingabe für dieses Feld überhaupt zulässig?“
- Dateiuploads mit fester Dateierweiterung benutzen, um das Hochladen gefährlicher Dateitypen zu verhindern.
- Wenn Daten in einem temporären Verzeichnis gespeichert werden, auf Zugangsbeschränkungen auf dieses Verzeichnis achten, damit keine Downloads von sensiblen Daten möglich sind. Beim IIS muss hierfür ein RequestFiltering auf diesem Verzeichnis eingerichtet werden; für den Tomcat sind keine zusätzlichen Einstellungen nötig, wenn der **Standard-Ordner** für temporäre Dateien benutzt wird.
- Keine Daten aus der Datenbank direkt in Formulare/Links schreiben und für die Navigation oder Logik-Steuerung benutzen. Beispiel: Nicht in Links die Primary-Keys von Datensätzen des Ziels schreiben. Besser: Temporäre IDs vergeben und diese mit den PK verknüpfen. So können keine Ziele angesteuert werden, die nicht erlaubt sind, da diese in den temporären IDs nicht enthalten sind.
- Bei sensiblen Formularfeldern das AutoComplete anwendungsseitig deaktivieren.
- Eine Navigation auf andere Seiten nicht direkt von Nutzereingaben abhängig machen. Beispiel: Keinen „OpenURL“-Aufruf durchführen mit einem Wert, der direkt vom Nutzer kommt. Hier immer prüfen/filtern, dass nur zulässige Werte aufgerufen werden.
- Antworten auf Fehleingaben dürfen keine sicherheitskritischen Informationen enthalten.
- Bei besonders kritischen Operationen den Nutzer nochmal zur Eingabe des Passworts auffordern. Beispiele sind die Änderung des Passworts oder das Löschen von besonders wichtigen Daten.
- Bei Nutzung von XHTML-Fragmenten besonders sorgfältig Nutzer-Daten bei der Eingabe und vor der Ausgabe prüfen. Hier greifen automatisches Escaping und Validierung nicht. Bei hohen Sicherheitsanforderungen sollte besser darauf verzichtet werden.
- Werden HTML-Emails versendet, und wird darin nicht 100% vertrauenswürdiger Text eingefügt, sollte der Text escaped werden. Im Gegensatz zur Anzeige innerhalb von JSF-/ ASP-Webseiten muss der Anwendungsentwickler hier selbst die entsprechenden Textbausteine escapen, wozu in SCOPELAND® Funktionen bereitgestellt sind.
- Bei Nutzung des XHTML-Editor-Controls wird die *Content Security Policy* deaktiviert. Bei hohen Sicherheitsanforderungen sollte am besten darauf verzichten werden.

- Bei Nutzung eines AutoRefresh Controls greift der Timeout nicht, solange noch eine Seite geöffnet ist. Dies ist beim Einsatz zu bedenken und abzuwägen.
- Bei SQL-Statements keinen IN-Operator verwenden. Diese werden ohne SQL-Parameter verwendet und eröffnen daher zumindest theoretisch die Möglichkeit von SQL-Injection.

9. Anhang

9.1. Vergleich mit gebräuchlichen Sicherheitsstandards

Es gibt verschiedene Sicherheitsstandards, an denen die Sicherheit von Web-Anwendungen gemessen werden kann. Um eine bessere Einordnung der Schutzmaßnahmen in SCOPELAND® zu ermöglichen, vergleichen wir diese im Folgenden mit den Standards OWASP 10, OWASP 25 und dem BSI-Grundschutz.

9.1.1. OWASP 10

	OWASP Top 10	Schutzmaßnahme SCOPELAND®
1	Injection	Eingabevalidierung, SQL-Parameter
2	Fehler in Authentifizierung und Session-Management	Passwort-Hashing, kryptografisch sichere Session IDs, nach dem Einloggen neue Session-ID, Session-ID nicht in URL, Session-Timeout, HttpOnly-Flag
3	Cross-Site-Scripting	Eingabevalidierung, Ausgabe-Escaping, Content Security Police, X-Frame-Options mit Same-Origin-Policy
4	Unsichere Direkte Referenzen	Wird durch Anwendungsentwickler sichergestellt. URL-Parameter Verschlüsselung
5	Sicherheitsrelevante Fehlkonfiguration	Wird durch Anwendungsentwickler sichergestellt. Konfiguration Manager unterstützt dabei.
6	Verlust der Vertraulichkeit sensibler Daten	Passwort-Hashing mit PBKDF2, Datenbank Verschlüsselung, TLS/SSL-Nutzen
7	Fehlerhafte Autorisierung auf Anwendungsebene	Benutzung von Standardverfahren zur Autorisierung. Spezielle Fälle durch Anwendungsentwickler sichergestellt.
8	Cross-Site Request Forgery	Anti-XSRF-Tokens auf Session Ebene
9	Nutzung von Komponenten mit bekannten Schwachstellen	Neuere SCOPELAND®-Versionen tauschen regelmäßig veraltete Komponenten aus.
10	Ungeprüfte Um- und Weiterleitungen	Wird durch Anwendungsentwickler sichergestellt.

9.1.2. OWASP 25

Im Folgenden sind die OWASP Top 25 Punkte (Version 3.0, 27.6.2011) und die Schutzmaßnahmen durch SCOPELAND® dokumentiert.

CWE ID	Name	Schutzmaßnahme SCOPELAND®
CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	Eingabevalidierung, SQL-Parameter
CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	Keine Maßnahme im Produkt. Anwendungsentwickler darf entweder keine OS-Commands verwenden

		oder muss eine eigene Eingabevalidierung implementieren.
CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	Eingabevalidierung, Ausgabe-Escaping, Content Security Police, X-Frame-Options mit Same-Origin-Policy
CWE-434	Unrestricted Upload of File with Dangerous Type	Ein Upload wird vom Produkt nicht verhindert, aber der Anwendungsentwickler kann Dateitypen abfragen und eigene Prüfungen vornehmen.
CWE-352	Cross-Site Request Forgery (CSRF)	Anti-XSRF-Tokens auf Session Ebene
CWE-601	URL Redirection to Untrusted Site ('Open Redirect')	Wird durch Anwendungsentwickler sichergestellt.
CWE-120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')	Nicht relevant, da Java/.Net
CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	Hier ist im Moment kein vollständiger Schutz durch das Produkt gewährleistet; die Anwendungsentwickler kennen aber das offene Problem und können es vermeiden. Eine Korrektur ist geplant.
CWE-494	Download of Code Without Integrity Check	Kein Download von SourceCode
CWE-829	Inclusion of Functionality from Untrusted Control Sphere	Nutzung von verbreiteten, als sicher geltenden Standard Libraries. Häufiger Update der Versionen.
CWE-676	Use of Potentially Dangerous Function	Nicht relevant, da Java/.Net
CWE-131	Incorrect Calculation of Buffer Size	Nicht relevant, da Java/.Net
CWE-134	Uncontrolled Format String	Nicht relevant, da Java/.Net
CWE-190	Integer Overflow or Wraparound	Sicherstellung durch Tests mit Grenzwerten
CWE-306	Missing Authentication for Critical Function	Passwort-Hashing, kryptografisch sichere Session-IDs, nach dem Einloggen neue Session-ID, Session-ID nicht in URL, Session-Timeout, HttpOnly-Flag
CWE-862	Missing Authorization	Benutzung von Standardverfahren zur Autorisierung. Spezielle Fälle durch Anwendungsentwickler sichergestellt.
CWE-798	Use of Hard-coded Credentials	Hard-coded Credentials werden mit einer Ausnahme nicht benutzt. Diese Ausnahme betrifft die Parameterverschlüsselung; eine Behebung ist geplant. Da eine Ausnutzung der Lücke schwierig und nur durch einen authentifizierten Nutzer möglich ist, wird dieses Problem als nicht besonders schwerwiegend eingeschätzt.
CWE-311	Missing Encryption of Sensitive Data	Wird durch Anwendungsentwickler sichergestellt.
CWE-807	Reliance on Untrusted Inputs in a Security Decision	Da die Parameterverschlüsselung momentan theoretisch noch umgangen werden kann, nur teilweise erfüllt. Behebung ist in Arbeit.
CWE-250	Execution with Unnecessary Privileges	Es werden nur notwendige Privilegien gefordert.
CWE-863	Incorrect Authorization	Benutzung von Standardverfahren zur Autorisierung. Spezielle Fälle durch Anwendungsentwickler sichergestellt.
CWE-732	Incorrect Permission Assignment for Critical Resource	Wird durch Anwendungsentwickler sichergestellt.

CWE-327	Use of a Broken or Risky Cryptographic Algorithm	Wird durch Anwendungsentwickler sichergestellt.
CWE-307	Improper Restriction of Excessive Authentication Attempts	Kann konfiguriert werden.
CWE-759	Use of a One-Way Hash without a Salt	Einsatz von PBKDF2 Algorithmus mit zusätzlichem Salt.

9.1.3. BSI Grundschutz

Im Folgenden werden nur die Punkte des BSI-Grundschutzes aufgeführt, die auf die Entwicklung von Anwendungen mit SCOPELAND® zutreffen. Dabei wurden auch explizit Punkte ausgeklammert, die vom Betreiber des Rechenzentrums, und nicht vom Hersteller der Anwendung zu verantworten sind. Um eine höhere Stufe als den Grundschutz zu erreichen, sind individuelle Sicherheitsanalysen während der Anwendungsentwicklung nötig.

BSI-ID	Geforderte Schutzmaßnahme	Schutzmaßnahme SCOPELAND®
M 4.392	Authentisierung bei Web-Anwendungen	Nach Standard des jeweiligen Webservers, Passwortrichtlinien
M 4.393	Umfassende Ein- und Ausgabevalidierung bei Web-Anwendungen und Web-Services	Eingabevalidierung (bei HTML-Formularen) nach Standard der Plattform (JSF, .Net), zusätzlich Regular Expressions möglich. Keine Ausgabevalidierung, aber Ausgabe-Escaping
M 4.394	Session-Management bei Web-Anwendungen und Web-Services	Nach Standard des jeweiligen Webservers, kryptografisch sichere Session IDs, nach dem Einloggen neue Session-ID, Session-ID nicht in URL, Session-Timeout, HttpOnly-Flag
M 4.395	Fehlerbehandlung durch Web-Anwendungen und Web-Services	Fehler Protokollierung, Spezielle Fehlerseiten ohne Daten für Produktiv-Betrieb, Fehlerbehandlung für Web-Service-Fehler möglich
M 4.396	Schutz vor unerlaubter automatisierter Nutzung von Web-Anwendungen	Generische Antworten bei zurückgewiesenen Eingaben, Anmeldeverzögerung und Sperrung von Accounts bei wiederholter Fehleingabe
M 4.397	Protokollierung sicherheitsrelevanter Ereignisse von Web-Anwendungen und Web-Services	Liegt beim Anwendungsentwickler
M 4.398	Sichere Konfiguration von Web-Anwendungen	Liegt beim Anwendungsentwickler/Kunden, gängige Verfahren wie TLS/SSL, Unterstützung durch Konfiguration-Manager
M 4.399	Kontrolliertes Einbinden von Daten und Inhalten bei Web-Anwendungen	Keine Maßnahmen im Produkt. Der Anwendungsentwickler muss das durch Prüfung von weiterleitungsrelevanten Eingaben und Inhalten von Uploads sicherstellen.
M 4.400	Restriktive Herausgabe sicherheitsrelevanter Informationen bei Web-Anwendungen und Web-Services	Spezielle Fehlerseiten ohne Daten für Produktiv-Betrieb
M 4.401	Schutz vertraulicher Daten bei Web-Anwendungen	Kryptografisch sichere Session-IDs, TLS/SSL, HttpOnly-Flag, Passwort-Hashing

M 4.402	Zugriffskontrolle bei Web-Anwendungen	Rollenbasiertes Berechtigungssystem, das auf dem jeweiligem Webserver aufsetzt
M 4.403	Verhinderung von Cross-Site Request Forgery (CSRF, XSRF, Session Riding)	Eingabevalidierung, Ausgabe-Escaping, Content Security Police, X-Frame-Options mit Same-Origin-Policy
M 4.404	Sicherer Entwurf der Logik von Web-Anwendungen	Liegt beim Anwendungsentwickler
M 4.405	Verhinderung der Blockade von Ressourcen (DoS) bei Web-Anwendungen und Web-Services	Entschärfung über Clustering/Cloud möglich
M 4.406	Verhinderung von Clickjacking	X-Frame-Options mit Same-Origin-Policy
M 4.450	Absicherung der Kommunikation bei Web-Services	TLS/SSL und WS-Security
M 4.451	Aktuelle Web-Service Standards	Es werden diverse Standards in ihrer aktuellen Form unterstützt.
M 4.452	Überwachung eines Web-Service	Normale Protokollierung und spezielle Protokollierung möglich. Umfassende Überwachung liegt eher auf der Ebene der Infrastruktur.
M 4.453	Einsatz eines Security Token Service (STS)	Nicht relevant, weil kein STS benutzt wird.
M 4.454	Schutz vor unerlaubter Nutzung von Web-Services	Notwendige Authentifizierung, WS-Security
M 4.455	Autorisierung bei Web-Services	Rollenbasiertes Berechtigungssystem des Webservers
M 4.456	Authentisierung bei Web-Services	Sichere Authentisierung per Zertifikat, WS-Security und anderen gängigen Standards möglich
M 4.457	Sichere Mandantentrennung bei Webanwendungen und Web-Services	Üblicherweise kein Problem, da nicht mehrere Mandanten auf einer Anwendung
M 4.495	Sicheres Systemdesign bei der Software-Entwicklung	Liegt beim Anwendungsentwickler
M 4.498	Sicherer Einsatz von Single-Sign-On	Soweit möglich, wird ausgereiftes SSO des Webservers genutzt

Scopeland Technology GmbH
Düsterhauptstraße 39 - 40
D - 13469 Berlin
Tel. +49 (30) 209 670 - 0
Fax +49 (30) 209 670 - 111

info@scopeland.de
www.scopeland.de

Geschäftsführer: Karsten Noack
Amtsgericht Charlottenburg HRB 176787 B
USt.-Nr.: DE 183446349