

Checkliste für sichere Programmierung

Thema	Prüfung	Ja/Nein
Filterung und Validierung von Benutzereingaben	Validieren Sie alle Eingaben?	
	Behandeln Sie invalide Eingaben als Fehler?	
	Verwenden Sie (wenn möglich und angebracht) Whitelist-Filter anstelle von Blacklist-Filtern?	
	Verzichten Sie auf komplexe reguläre Ausdrücke in Filterfunktionen?	
	Wurden Ihre Filterfunktionen von Sicherheitsexperten begutachtet?	
Encodierung von Ausgaben	Werden alle Datenausgaben entsprechend dem Zielkontext encodiert?	
	Verwenden Sie Encodierungsfunktionen aus dem Standard?	
	Wurden Ihre eigenen Encodierungsfunktionen von Sicherheitsexperten auditiert?	
Indirektion	Verhindern Sie den direkten Zugriff auf interne Ressourcen (zum Beispiel über Dateinamen)?	
	Vermeiden Sie die Ausgabe von internen Daten an Benutzer?	

Checkliste für sichere ABAP-Programme

Thema	Prüfung	Ja/Nein
Fehlende Berechtigungsprüfungen bei Transaktionen	Findet bei jedem Zugriff auf kritische Funktionen, Ressourcen und Daten eine explizite und hinreichende Prüfung der/des relevanten Berechtigungsobjekte(s) in ABAP statt?	
	Findet bei jedem Aufruf des Befehls CALL TRANSACTION eine explizite und hinreichende Prüfung der/des Berechtigungsobjekte(s) in ABAP statt?	
	Verwenden Sie den Funktionsbaustein AUTHORITY_CHECK_TCODE, um Startberechtigungen für Transaktionen zu prüfen?	
	Wird nach jedem Aufruf von AUTHORITY-CHECK OBJECT auch der korrekte Returncode (0) in sy-subrc geprüft?	
Hintertüren – hart codierte Berechtigungen	Haben Sie alle Vorkommen von SY(ST)-UNAME auf proprietäre Berechtigungskonzepte hin überprüft?	
	Sind hart codierte Benutzernamen, die beabsichtigt vorkommen, als Abweichung vom Sicherheitsstandard dokumentiert?	
Fehlende Berechtigungsprüfungen in RFC-fähigen Funktionen	Sind all Ihre ABAP-Funktionen entsprechend den benötigten Berechtigungen in Funktionsgruppen zusammengefasst?	
	Führen Sie in allen RFC-Bausteinen explizite programmatische Berechtigungsprüfungen durch?	
Debug-Code in Assert Statements	Haben Sie überprüft, dass in logischen Ausdrücken von ASSERT-Befehlen keine kritischen Methodenaufrufe stattfinden?	
	Haben Sie sämtlichen Code entfernt, der hinter ASSERT-Befehlen mit einer unerfüllbaren Bedingung steht?	
Generischer und dynamischer ABAP-Code	Haben Sie bei jedem Vorkommen von ASSIGN und WRITE geprüft, ob die Variable von Benutzereingaben abhängt und diese Stellen in statische Zugriffe umgewandelt?	
Generische Funktionsaufrufe	Sind alle Stellen mit generischen Funktionsaufrufen identifiziert und geprüft?	
	Haben Sie geprüft, dass Benutzereingaben nicht in generischen Funktionsaufrufen verwendet werden?	
Generische Reports (ABAP Command Injection)	Verzichten Sie auf die Verwendung von INSERT REPORT und GENERATE SUBROUTINE-POOL in Ihrem ABAP-Coding?	
	Unterbinden Sie Benutzereingaben in dynamischem Code?	

Thema	Prüfung	Ja/Nein
SQL-Injection	Haben Sie geprüft, dass in Ihrem ABAP-Programm keine dynamischen SQL-Anweisungen verwendet werden bzw. dass in solchen Anweisungen nur Konstanten verarbeitet werden?	
Directory Traversal	Haben Sie ausgeschlossen, dass Benutzereingaben als Parameter von OPEN DATASET verwendet werden?	
	Haben Sie ausgeschlossen, dass Benutzereingaben als Parameter von DELETE DATASET verwendet werden?	
Aufrufe in den Kernel	Haben Sie geprüft, dass Ihr Programm keine Kernel-Funktionen mittels CALL cfunc aufruft?	
	Haben Sie geprüft, dass Ihr Programm keine Kernel-Funktionen mittels SYSTEM-CALL aufruft?	
System Command Injection und System Command Execution	Haben Sie geprüft, dass Ihr Programm keine Aufrufe mittels CALL FUNCTION 'SXPG_CALL_SYSTEM' absetzt bzw. diese Aufrufe nur Konstanten enthalten?	
	Haben Sie geprüft, dass Ihr Programm keine Aufrufe mittels CALL FUNCTION 'SXPG_COMMAND_EXECUTE' absetzt bzw. diese Aufrufe nur Konstanten enthalten?	
	Haben Sie geprüft, dass Ihr Programm keine Aufrufe mittels CALL TRANSACTION 'sm49' enthält bzw. diese Aufrufe nur konstante Werte verarbeiten?	
	Haben Sie geprüft, dass Ihr Programm keine Aufrufe mittels CALL TRANSACTION 'sm69' enthält bzw. diese Aufrufe nur konstante Werte verarbeiten?	
	Haben Sie geprüft, dass kein OPEN DATASET-Befehl mit der Option FILTER verwendet wird bzw. in FILTER nur Konstanten verarbeitet werden?	
	Haben Sie geprüft, dass Ihr Programm keine Aufrufe mittels CALL 'SYSTEM' enthält bzw. an solche Aufrufe nur Konstanten übergeben werden?	
	Haben Sie geprüft, dass Ihr Programm keine Aufrufe an CL_GUI_FRONTEND_SERVICES =>EXECUTE absetzt bzw. diese Aufrufe nur Konstanten enthalten?	
	Haben Sie geprüft, dass Ihr Programm keine Aufrufe an WS_EXECUTE absetzt bzw. diese Aufrufe nur Konstanten enthalten?	

Checkliste für UI-Programmierung

Thema	Prüfung	Ja/Nein
Cross-Site Scripting	Haben Sie für alle Daten, die Ihr ABAP-Code in eine HTML-Oberfläche ausgibt, geprüft, ob Sie die richtige Encoding-Funktion verwenden?	
Cross-Site Request Forgery	Haben Sie ausschließlich statische Verweise auf Webseiten in Ihrer Webapplikation?	
	Akzeptieren Sie Webformulare ausschließlich über POST-Anfragen und nicht über GET?	
	Nutzen Sie spezielle Formularfelder mit zufälligen Werten für den XSRF-Schutz?	
Forceful Browsing	Haben Sie überprüft, ob alle Seiten und Dokumente Ihrer Anwendung durch eine Berechtigungsprüfung auf Benutzerebene geschützt sind? Hierzu ist auch eine Prüfung aller URL-Parameter erforderlich, die (indirekt) auf solche Ressourcen verweisen.	
	Haben Sie überprüft, dass Ihre Anwendung keine (kritischen) Daten in Hidden Fields speichert?	
	Haben Sie überprüft, dass Ihre Anwendung keine (kritischen) Daten in Cookies speichert?	
	Haben Sie überprüft, ob die Werte sämtlicher Wertauswahlhilfen auf dem Server erneut auf Gültigkeit getestet werden?	
	Haben Sie für alle deaktivierten UI-Elemente geprüft, dass das zugehörige Ereignis nur dann vom Server verarbeitet wird, wenn dem Benutzer das Element auch im aktiven Zustand angezeigt wird?	
Phishing	Verwenden Sie eine Whitelist zur Validierung aller Verweise, die ein Benutzer eingeben kann?	
	Verwenden Sie eine Whitelist zur Validierung aller Rücksprung-URLs in Login-Seiten?	
	Verwenden Sie eine Whitelist zur Validierung aller Exit-URLs?	
HTTP Response Tampering	Vermeiden Sie Benutzerdaten in HTTP-Headern, zum Beispiel in Cookies oder in Redirects über Location?	
	Wenn Sie Benutzerdaten in HTTP-Headern verwenden: Filtern Sie alle Benutzerdaten in HTTP-Headern?	

Checkliste für die Dateiverarbeitung in ABAP

Prüfung	Ja/Nein
Schützen Sie Dateinamen vor Manipulation mittels einer Whitelist bzw. Indirektion?	
Vermeiden Sie Directory-Traversal-Schwachstellen?	
Schränken Sie den Zugriff auf Dateien und Pfade auch durch die Pflege von Berechtigungsobjekten ein?	
Achten Sie bei OPEN DATASET darauf, dass die Option FILTER nicht mit Benutzereingaben gefüllt wird?	
Inhalte aus Dateien sind nicht vertrauenswürdig. Validieren Sie alle Daten und die Struktur der Datensätze bereits sorgfältig?	
Achten Sie bei der Ausgabe von Daten in eine Datei darauf, dass die Struktur der Datensätze in der Datei nicht durch Sonderzeichen in den Daten manipuliert werden kann?	
Externe Dateien können Schadcode enthalten. Verwenden Sie einen Virens Scanner, zum Beispiel mittels der Virusscanner-Schnittstelle?	
Verhindern Sie den Upload von HTML-Dateien?	
Das Berechtigungsobjekt S_GUI kontrolliert nur den generellen Upload bzw. Download von Dateien und macht dabei keinen Unterschied zwischen SAP GUI (Intranet) und Browsern (Internet). Führen Sie bei kritischen Dateien deshalb bereits eine zusätzliche Berechtigungsprüfung in ABAP durch?	

Checkliste für Datenbankzugriffe in ABAP

Prüfung	Ja/Nein
Verwenden Sie Open SQL und nur in dokumentierten Ausnahmefällen Native SQL?	
Prüfen Sie in allen Native-SQL-Anfragen den anfragenden Mandanten ab?	
Loggen Sie alle Native-SQL-Anfragen, sodass die Auditierbarkeit der Daten gewährleistet ist?	
Vermeiden Sie alle datenbankspezifischen, potenziell gefährlichen SQL-Kommandos in Native SQL?	
Verwenden Sie ausschließlich Native-SQL-Aufrufe, die Sie nicht in Open-SQL-Aufrufe ändern können?	
Verzichten Sie auf generische SQL-Anfragen?	
Verzichten Sie auf Benutzerdaten in dynamischen Open-SQL-Anfragen, auch in selbst gebauten Encodierungsfunktionen?	
Filtern und validieren Sie die Daten, bevor Sie sie in der Datenbank speichern?	
Geben Sie nirgendwo interne Datenbank-IDs an den Benutzer weiter bzw. lesen Sie nirgendwo solche internen Datenbank-IDs von Benutzern ein?	
Haben Sie alle Datenbankabfragen in Wrapper-Funktionen gekapselt und diese Funktionen durch Berechtigungsprüfungen abgesichert?	

Checkliste für die Verwendung des SAP GUI

Prüfung	Ja/Nein
Validieren Sie alle Benutzereingaben?	
Verwenden Sie Indirektion, wenn auf interne Ressourcen zugegriffen wird?	
Vermeiden Sie im Backend-Coding alle Schwachstellen?	
Achten Sie auf die Verwendung von Benutzereingaben, die zu SQL-Injection, Directory-Traversal-Schwachstellen sowie zu System Command Injection und System Command Execution führen könnten?	
Prüfen Sie bei kritischen Anwendungen, ob die Events im Ablauf der Logik plausibel sind?	

Checkliste für den AS ABAP

Prüfung	Ja/Nein
Verzichten Sie auf eigene HTTP-Handler?	
Nutzen Sie die Standardsframeworks des AS ABAP?	
Validieren Sie URLs über den Whitelist-Filter des AS ABAP?	
Encodieren Sie HTML-Daten vor der Ausgabe in einem anderen Kontext?	

Checkliste für Business Server Pages

Prüfung	Ja/Nein
Haben Sie die Maßnahmen aus Kapitel 5 zur Entwicklung von sicherem ABAP-Coding beachtet?	
Verwenden Sie, wann immer möglich, HTMLB-Controls, und setzen Sie das forceEncode-Attribut der Controls immer auf ENABLED?	
Encodieren Sie Ausgaben in HTML entsprechend dem Kontext der Ausgabe (HTML, URL, JavaScript etc.)?	
Filtern und validieren Sie alle Benutzereingaben?	
Verwenden Sie Indirektion, um interne Informationen nicht an den Benutzer geben zu müssen?	
Verwenden Sie eine externe Bibliothek, um XSRF-Angriffe gegen Ihre BSP-Anwendungen zu verhindern?	
Verwenden Sie bei Verweisen auf externe Seiten den Whitelist-Filter des AS ABAP?	
Vermeiden Sie, dass vertrauliche Daten an den Browser weitergegeben werden, zum Beispiel über HTML-Hidden-Fields oder Cookies? Verwalten Sie vertrauliche Daten stattdessen in serverseitigen Cookies?	
Unterbinden Sie es, dass Benutzer auf externe Seiten verweisen bzw. externe Seiten in Ihre BSP-Anwendung einbinden können?	
Vermeiden Sie es, Benutzereingaben in HTTP-Headern zu verwenden?	

Checkliste für den Internet Transaction Server

Prüfung	Ja/Nein
Entwickeln Sie die Backend-Logik von Dynpros nach den Sicherheitsrichtlinien, die hier vorgestellt wurden?	
Überprüfen Sie, ob Annahmen, die Sie bei der Entwicklung für das SAP GUI getroffen haben, auch für das WebGUI gelten?	
Nutzen Sie die XSS-Schutzfunktionen?	
Achten Sie auf webrelevante Schwachstellen, wenn Sie eigene IACs oder Flow-Logic-Anwendungen entwickeln?	

Checkliste für Web Dynpro ABAP

Prüfung	Ja/Nein
Setzen Sie Web Dynpro ABAP ein, wenn Sie möglichst wenig Aufwand für sichere Webanwendungen betreiben möchten?	
Schränken Sie die erlaubten Stylesheets mit dem Whitelist-Filter des AS ABAP ein?	
Verzichten Sie beim Aufruf von Web-Dynpro-ABAP-Anwendungen auf die Ausführung von Business-Logik, die auf Start-Parametern basiert, um XSRF-Angriffe zu unterbinden?	

Checkliste für indirekte User Interfaces und externe Systeme

Prüfung	Ja/Nein
Validieren Sie alle eingehenden Daten nach den Prinzipien aus Kapitel 4, »Sichere Programmierung«?	
Schützen Sie alle extern aufrufbaren Funktionen (zum Beispiel RFC, Webservices und SAP NetWeaver PI) durch individuelle Berechtigungsprüfungen?	
Wenn mehrere separate Funktionen Teil eines Gesamtprozesses sind: Prüfen Sie auch die korrekte Reihenfolge der Aufrufe und ob alle notwendigen Funktionen auch tatsächlich aufgerufen werden?	
Verlassen Sie sich nicht darauf, dass eine vorgelagerte Instanz bereits Sicherheitsprüfungen durchgeführt hat! Sorgen Sie selbst für den Schutz Ihrer Geschäftsdaten und -prozesse?	
Statistisch gesehen sind insbesondere Angriffe von innen zu erwarten. Setzen Sie Ihre Prioritäten richtig?	