Der Weg ist das Ziel – Kontrollfluss-Integrität in Web-Applikationen sichern

Bastian Braun (gemeinsame Arbeit mit Patrick Gemein, Hans P. Reiser)
Institute of IT-Security and Security Law (ISL), Universität Passau
bb@sec.uni-passau.de

OWASP
The Open Web Application Security Project

About Me: Bastian Braun

OWASP
The Open Web Application Security Project

Studium Informatik (Dipl.) @ RWTH Aachen
Doktorand (WiMi) @ SVS, Uni Hamburg
Doktorand (WiMi) @ ISL, Uni Passau
EU FP 7 Projekt WebSand

> “Server-driven Outbound Web-application Sandboxing”
>
> https://www.websand.eu
>
> Leitung AP “Sichere Web-Interaktion”

Institute of IT-Security and Security Law

WebSand

Kontrollflüsse in Web-Applikationen

OWASP
The Open Web Application Security Project

Web-Applikationen sind zustandsbehaftet
Zustandsübergänge durch verarbeitete HTTP Requests

http://www.example.de/users.php?action=add
&name=doe&firstname=john

Determinanten: Funktion, HTTP Parameter, *letzter Zustand*
Kontrollfluß = Sequenz von Requests im gleichen Session-Kontext

Kontrollflüsse in Web-Applikationen

OWASP
The Open Web Application Security Project

Moderne Web-Applikationen implementieren meist komplexe Anwendungslogik

Buchungs- und Bezahlprozesse

- Bahn, Flüge, E-commerce (ebay, amazon), Banking

Konfigurationsschritte

- Registrierung, Passwort (zurück|neu) setzen

mehrere involvierte Domains

- Bezahlung via PayPal

Anwendung erfordert schrittweises Vorgehen
Annahme: Benutzer starten bei Eingangsseite & klicken nur auf Hyperlinks und Buttons

Kontrollflüsse in Web-Applikationen

OWASP
The Open Web Application Security Project

Problem: Benutzer kann beliebige Requests senden
    z.B. per Kommando- und Adresszeile

Angriffsvektoren
    beliebige Zustandsmanipulation

    Race Conditions

    HTTP Parameter-Manipulation

Real-World-Beispiele

OWASP
The Open Web Application Security Project

Zustandsmanipulation

alle Gegenstände kostenlos einkaufen [Wang et al., 2011]

Account-Zugang ohne Anmeldung mit Hilfe von Session Puzzles [Chen, 2011]

Race conditions

zu viele kostenlose SMS senden [Paleari et al., 2008]

HTTP Parameter-Manipulation

beliebige Ware zum Preis des günstigsten Gegenstands kaufen [Wang et al., 2011]

Zugriff auf 200.000 Bank-Accounts inkl. Kundennamen, Kreditkartennummern, E-Mail-Adresse und Transaktionshistorie [Citigroup, 2011]

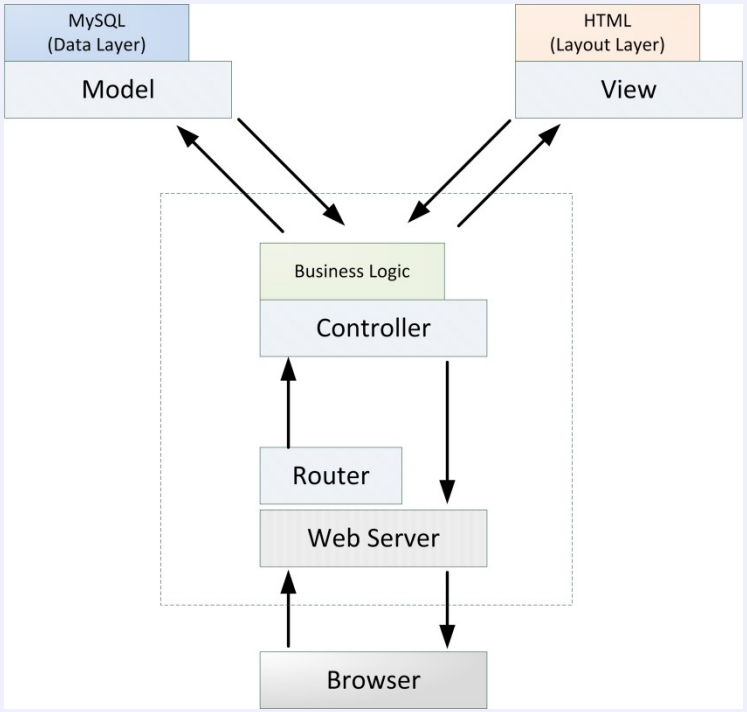
Zugriff auf Bewerber-Daten inkl. aktuellem Gehalt und Arbeitgeber [UNESCO, 2011]

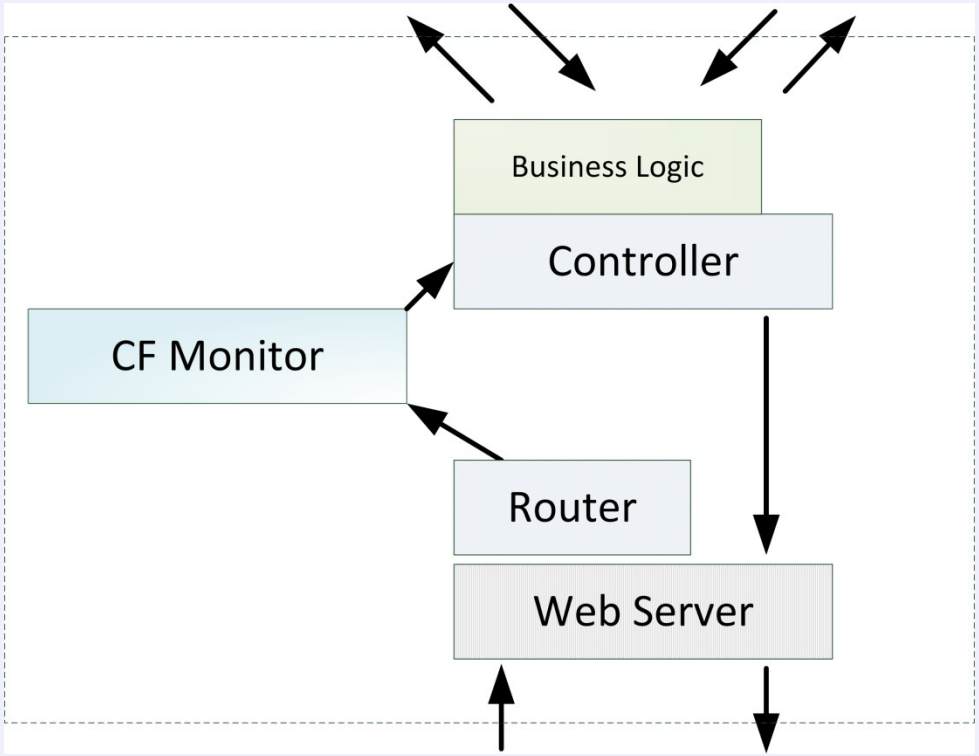
Kontrollfluss-Integrität sichern

OWASP
The Open Web Application Security Project

Schutzansatz: mit CF-Monitor Requests beim Eintreffen zunächst auf Legitimität prüfen

Integration in MVC-Architektur





Kontrollfluss-Integrität sichern

OWASP
The Open Web Application Security Project

Schutzziele

- für jeden Request: akzeptieren/ablehnen basierend auf Kontext
- Schutz vor Race Condition-Exploits
- Unterstützung des Zurück-Knopfes
- Unterstützung für Multi-Tabbing
- Kontrolle von HTTP-Parametern, Datentypen, Werten
- Whitelisting von unkritischen Methoden (z.B. Impressum)
- Integrierbarkeit in existierende Web-Applikationen

Kontrollfluss-Integrität sichern

OWASP
The Open Web Application Security Project

Trennung von Kontrollfluss-Integrität und Anwendungs-Implementierung

CF-Monitor bekommt “Modell” der Web-Applikation (Policy) als Input

dadurch entstehen Garantien für den Entwickler

- Reihenfolge eingehender Requests, angegebener HTTP Parameter + Werte, sequentielle Abarbeitung von kritischen Requests

Kontrollfluss-Policies werden explizit + überprüfbar

Access Control ist keine Lösung!

Implementierung

OWASP
The Open Web Application Security Project

PHP-Framework CodeIgniter
Integration mit Aspect Oriented Programming

vorher:
include (APPPATH. ' c o n t r o l l e r s / ' .$RTR->f e t c h d i r e c t o r y ( ) .
$RTR->f e t c h c l a s s ( ) . ' . php ') ;

nachher:
AOP: : pr o c e s s (APPPATH. ' c o n t r o l l e r s / ' .$RTR->f e t c h d i r e c t o r
y ( ) .
$RTR->f e t c h c l a s s ( ) . ' . php ' ,
$ SESSION [' ' atom parentFramework ' ']->getCacheFolderName ( )) ;

Implementierung

OWASP
The Open Web Application Security Project

Multi-Tabbing

Problem: Identifizierung von Tabs

Ansatz: AJAX-basierter Indikator

- benachrichtige CF-Monitor bei Öffnen, Schliessen, Tab-Wechsel
- jeder Tab hat Session-eindeutige ID
- unterschiedliche Kontrollflüsse in verschiedenen Tabs
- Manipulation erlaubt keinen Vorteil

Implementierung

OWASP
The Open Web Application Security Project

Race Condition-Schutz

Problem: parallelisierte Abarbeitung von Requests

Ansatz: eingehende Requests serialisieren

- alle Requests serialisieren: Performance-Einbußen
- stattdessen: Session-level, User-level, System-level Schutz konfigurierbar via Datei-Locks
- keine Auswirkung auf ungeschützte Resourcen

Implementierung

OWASP
The Open Web Application Security Project

Zurück-Knopf

Problem: Request wird u.U. nochmals ausgeführt

Ansatz: server-seitiges Erkennen + Kontrolle

- Caching verhindern -> Seite muss neu geladen werden
- Policy definiert, wann Schritt zurück erlaubt ist

Implementierung

OWASP
The Open Web Application Security Project

HTTP-Parameter-Kontrolle

Problem: keine Datentypen, beliebige Werte

- Datentypen definierbar: bool, integer, string
- write once read many (WORM): invariant
- Ausschluss von Parameter-Namen

Evaluation

OWASP
The Open Web Application Security Project

Overhead

- abhängig von Policy-Komplexität
- unabhängig von Web-Applikation
- 8,9 – 9,6 ms pro Request

Übertragbarkeit

- PoC in PHP, JEE-Filter möglich

Nicht-MVC-basierte Web-Applikationen schützbar aber schwierig

Zusammenfassung

OWASP
The Open Web Application Security Project

Kontrollfluss-Integrität: WWW-inhärentes Problem
kein Ersatz für die Anwendungslogik
viele zugehörige Aspekte
- Race Conditions, manipulierte HTTP-Parameter, Seiteneffekte des Zurück-Knopfes

Resultat: CF-Monitor
- leicht integrierbar in MVC-Anwendungen
- Trennung von Anwendungssemantik und Kontrollfluss-Integrität

OWASP
The Open Web Application Security Project

Danke für die Aufmerksamkeit!