



Sichere Entwicklung und gängige Schwachstellen in eigenentwickelten SAP-Web-Anwendungen

Sebastian Schinzel

Virtual Forge GmbH

sebastian.schinzel@virtualforge.de

OWASP

Nürnberg, 13.10.09

Copyright © The OWASP Foundation
Permission is granted to copy, distribute and/or modify this document
under the terms of the OWASP License.

The OWASP Foundation

<http://www.owasp.org>

SAP in a Nutshell

- Weltweit führendes Unternehmen für Geschäftssoftware
- Sitz in Walldorf
- Praktisch alle SAP-Installationen einzigartig
- Geschäftsanwendungen zunehmend über Web Frontends erreichbar
- Geschäftsanwendungen in Java oder ABAP geschrieben (plus C-API, PHP, ...)

ABAP in a Nutshell

- Existiert seit ~30 Jahren
- COBOL-ähnliche Syntax
- “gewachsene Sprache” → mehrere Programmierparadigmen vermengt
- DB-unabhängiger SQL-Dialekt ist integriert
- Code liegt selbst in Datenbank
- Entwicklungsumgebung ebenfalls in ABAP entwickelt

ABAP und Open Source

- Quellcode liegt vollständig vor
- Kunden können Code ändern
 - ▶ Code kopieren, umbenennen und abändern
 - ▶ Direkt SAP-Standard-Coding ändern
- ABAP enthält diverse Entwicklungsframeworks
 - ▶ Kunden erweitern Code (Customizing)
 - ▶ Eigenentwicklungen für Geschäftsprozesse
 - ▶ Entwicklungen von Drittanbietern

Frontend-Technologien

■ Dynpro:

- ▶ In ABAP geschrieben
- ▶ Benötigt eigenen Viewer (SAP GUI)
- ▶ Funktionsweise ähnlich wie X11

■ Internet Transaction Server (ITS)

- ▶ Erste Web-Technologie von SAP
- ▶ Noch zahlreich in Betrieb, kaum noch aktive Entwicklung

Frontend-Technologien

■ Business Server Pages (BSP)

- ▶ In HTML eingebettetes ABAP (ähnlich zu JSP)
- ▶ Mehrere Programmierparadigmen
- ▶ Breite Installationsbasis, auch weiterhin Neuentwicklungen von Kunden

■ Web Dynpro (ABAP/Java)

- ▶ UI-unabhängiges Framework
- ▶ Entwickler kann kein eigenes HTML/JavaScript einbetten
- ▶ Entwickler kann typische Web-Schwachstellen weder verhindern noch verursachen

Frontend-Technologien

■ Web GUI

- ▶ HTML-Version von normalen Dynpros (SAP GUI)
- ▶ Früher im Internet Transaction Server, heute als Plugin zum SAP Web Application Server

... externe Systeme, Flash, Silverlight.

SAP Web Server

SAP Web Application Server (Web AS):

- Unterstützt Single Sign On (SSO)
- SSO-Ticket in Cookie (`MYSAPSSO2`)
 - ▶ standardmäßig für Pfad `/ und domain.tld` ausgestellt
 - ▶ standardmäßig weder `httpOnly`, noch `secure`
- Entwicklung eigener HTTP-Handler möglich
 - ▶ BSP, Web Dynpro, WebGUI sind HTTP-Handler
- Konfiguration über Profilparameter (Programm `RZ11`) und Transaktion `SICF`
- Blacklist-Filter filtert z.B. `<script` und `alert(:-)`

Agenda

- ABAP/BSP vs. OWASP Top 10
- Business Server Pages
 - ▶ Inline ABAP in HTML
 - ▶ Die HTMLB-Tag-Bibliothek
- Open SQL
 - ▶ Dynamisches Open SQL
 - ▶ SQL-Injections
- Summary

Business Server Pages (BSP)

BSP im Internet erkennen:

<http://www.google.de/search?q=inurl:/sap/bc/bsp/>

[Erweiterte Suche](#)
[Einstellungen](#)

Suche: Das Web Seiten auf Deutsch Seiten aus Deutschland

Ergebnisse 1 - 10 von ungefähr **57.600** für inurl:/sap/bc/bsp/. (0,85 Sekunden)

•Business Server Pages (BSP)

mentor.com

erco.org

sap-ag.de

beiersdorfgroup.com

mybayerjob.de

heraeus.com

wacker.com

heidelberg.com

knorr-bremse.com

ottopersonalsysteme.de

skyguide.ch

eads.com

bsr.de

kuka.de

kpmg.de

daad.de

euhreka.com

vodafone.com

iom.int

wlw.de

erecruiting-randstad.de

lieferantensuchmaschine.com

audi.de

blanco.de

festo.com

vhv.de

otto.de

abb.de

ruv.de

holcim.com

mannheim.cde

gesobau.de

softsurvey.de

umdasch.com

celesio.com

pflegedienst-navigator.de

oebb.at

salzburg-ag.at

whirlpool.com

volkswagen.de

pharma.com

wa.gov

brucepower.com

jetblue.com

suzukiautoco.com

singaporepower.com

kaufland.de

clavis-bonn.de

albatha.ae

•Business Server Pages (BSP)

OWASP Top 10	Potentiell anfällig?
A1 - Cross Site Scripting (XSS)	Ja
A2 - Injection Flaws	Ja
A3 - Malicious File Execution	Ja
A4 - Insecure Direct Object Reference	Ja
A5 - Cross Site Request Forgery (CSRF)	Ja
A6 - Information Leakage and Improper Error Handling	Ja
A7 - Broken Authentication and Session Management	-
A8 - Insecure Cryptographic Storage	Ja
A9 - Insecure Communications	Ja
A10 - Failure to Restrict URL Access	Ja

Business Server Pages

Verhindern von Cross Site Scripting durch Encoding/Escaping

■ in Plain-HTML-Seiten

- ▶ ABAP-Encoding-Funktionen (`CL_HTTP_UTILITY`)
- ▶ BSP-Seitenattribut (`forceEncode`)

■ in Seiten mit HTMLB-Taglib

- ▶ Tag-Attribut (`forceEncode`)

Business Server Pages – Plain HTML

```
1 <%@page language="abap" %>
2 <% DATA: name TYPE string.
3     name = request->get_form_field( 'name' ).
4 %>
5 <html>
6 <head><title>Beispiel owasp.htm</title></head>
7 <p>Hello <%= name %> </p>
8 <body>
9 </body></html>
```

Business Server Pages – Plain HTML

```
1 <html>
2   <head><title>Beispiel owasp.htm</title></head>
3   <body>
4     <p>Hello Open Web Application Security Project!</p>
5   </body>
6 </html>
```



Business Server Pages – Plain HTML

Cross Site Scripting Schwachstelle:

```
http://.../owasp.htm?name=<img src= onerror="alert(document.cookie);">
```

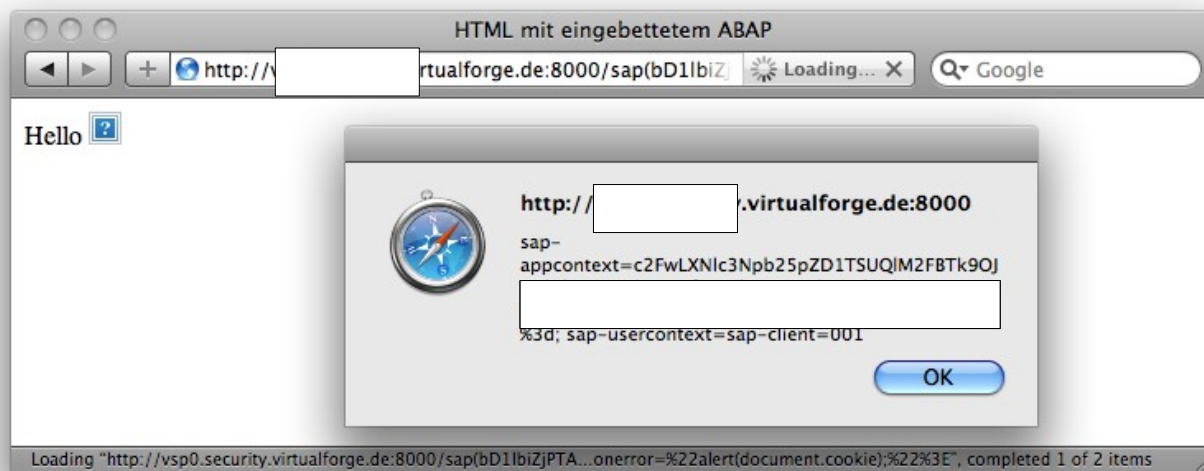
```
1 <html>
```

```
2 <head><title>Beispiel owasp.htm</title></head>
```

```
3 <p>Hello <img src= onerror="alert(document.cookie);"> </p>
```

```
4 <body>
```

```
5 </body></html>
```



Business Server Pages – Plain HTML

```
1 <%@page language="abap" %>
2 <% DATA: name TYPE string.
3     name = request->get_form_field( 'name' ).
4     name = CL_HTTP_UTILITY=>escape_html( name ).
4 %>
5 <html>
6 <head><title>Beispiel owasp.htm</title></head>
7 <p>Hello <%= name %> </p>
8 <body>
9 </body></html>
```

Business Server Pages – Plain HTML

Verhinderte Cross Site Scripting Schwachstelle:

```
http://.../owasp.htm?name=<img src= onerror="alert(document.cookie);">
```

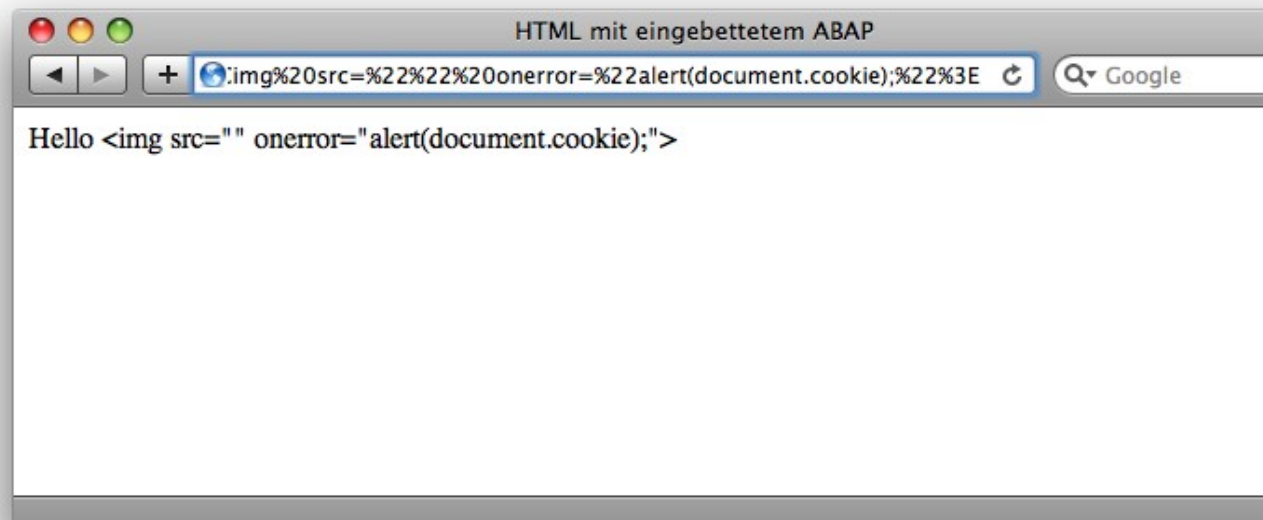
```
1 <html>
```

```
2 <head><title>Beispiel owasp.htm</title></head>
```

```
3 <p>Hello &lt;img src= onerror="alert(document.cookie);"> </p>
```

```
4 <body>
```

```
5 </body></html>
```



Business Server Pages – Plain HTML

■ Enkodierung von Daten über

- ▶ `CL_HTTP_UTILITY=>escape_html()`
- ▶ `CL_HTTP_UTILITY=>escape_javascript()`
- ▶ `CL_HTTP_UTILITY=>escape_url()`

■ Pro:

- ▶ Verhindert XSS-Schwachstellen

■ Contra:

- ▶ Jede einzelne Ausgabe muss entsprechend dem HTML-Kontext enkodiert werden
- ▶ Aufwändig, fehleranfällig

Business Server Pages – Plain HTML

```
1 <%@page language="abap" forceEncode="html"
2 DATA: name TYPE string.
3 name = request->get_form_field( 'name' ).
4 %>
5 <html>
6 <head><title>Beispiel owasp.htm</title></head>
7 <p>Hello <%= name %> </p>
8 <body>
9 </body></html>
```

Business Server Pages – Plain HTML

Verhindern von Cross Site Scripting durch

```
<%page forceEncode="{html|url|javascript}">
```

- Globales Encoding über Seitenattribut
- Alle Ausgaben werden gleich enkodiert, keine Unterscheidung zwischen HTML-Kontexten (JavaScript, URL, ...)

Gegenbeispiel:

```
<a href="<%= request->get_form_field( 'user' ). %>">
```

```
Link</a>
```

```
http://.../test.htm?user=javascript:document.write( ...
```

Business Server Pages – HTMLB

```
1  <%@page language="abap" %>
2  <%@extension name="htmlb" prefix="htmlb" %>
3  <% DATA: name TYPE string.
4     name = request->get_form_field( 'name' ). %>
5  <htmlb:content design="design2003">
6     <htmlb:page title = "HTMLB mit forceEncode">
7         <htmlb:form>
8             <htmlb:textView      text          = "Hello <%= name %>"
9                                 design        = "EMPHASIZED" />
10        </htmlb:form>
11    </htmlb:page>
12 </htmlb:content>
```

Business Server Pages – HTMLB

```
name=<img src="" onerror="alert(document.cookie);">
```

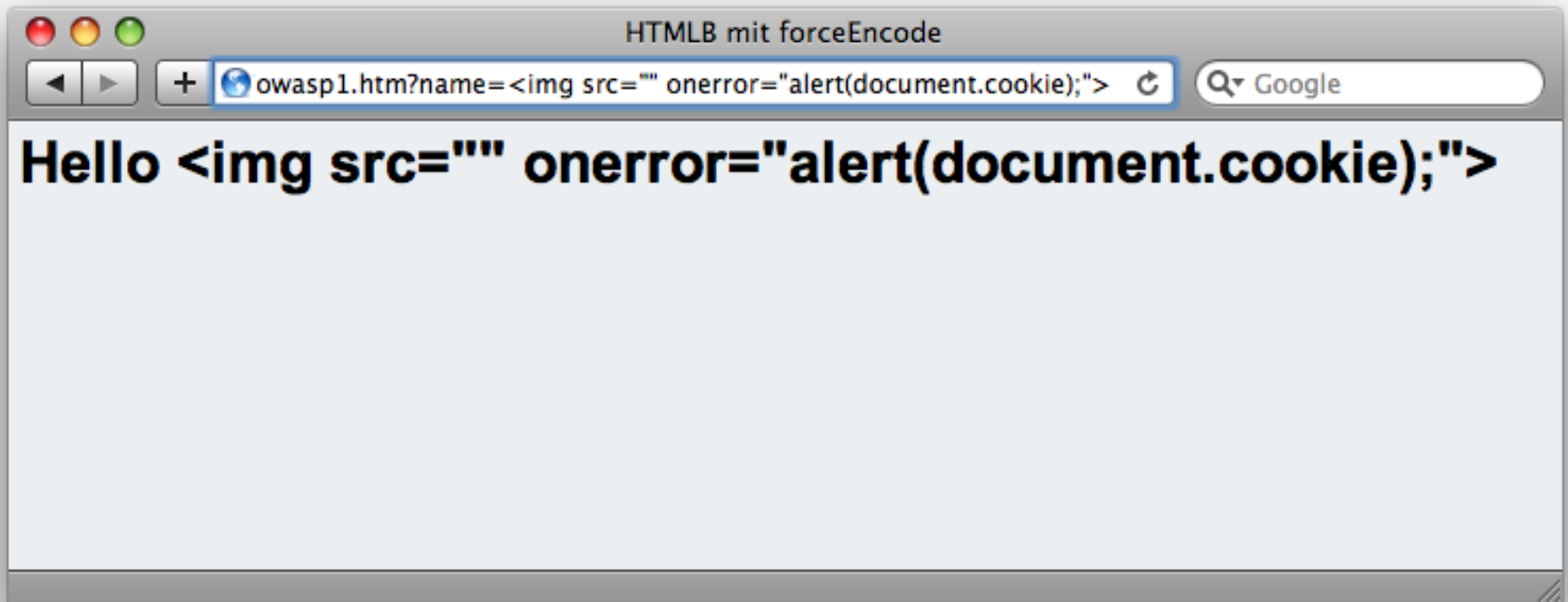


Business Server Pages – HTMLB

```
1  <%@page language="abap" %>
2  <%@extension name="htmlb" prefix="htmlb" %>
3  <% DATA: name TYPE string.
4     name = request->get_form_field( 'name' ). %>
5  <htmlb:content design="design2003" forceEncode="ENABLED">
6     <htmlb:page title = "HTMLB mit forceEncode">
7         <htmlb:form>
8             <htmlb:textView      text          = "Hello <%= name %>"
9                                 design        = "EMPHASIZED" />
10        </htmlb:form>
11    </htmlb:page>
12 </htmlb:content>
```


Business Server Pages – HTMLB

```
name=<img src="" onerror="alert(document.cookie);">
```



Business Server Pages

Verhindern von Cross Site Scripting in Plain-HTML

- **Enkoding über Methoden** (`CL_HTTP_UTILITY`)

- ▶ Aufwändig, Fehleranfällig

- **Enkoding über Seitenattribut** (`forceEncode`)

- ▶ Nicht HTML-Kontext-spezifisch, daher fehleranfällig

Verhindern von Cross Site Scripting in HTMLB

- **Tag-Attribut** `forceEncode` standardmäßig ausgeschaltet, muss explit aktiviert werden

Agenda

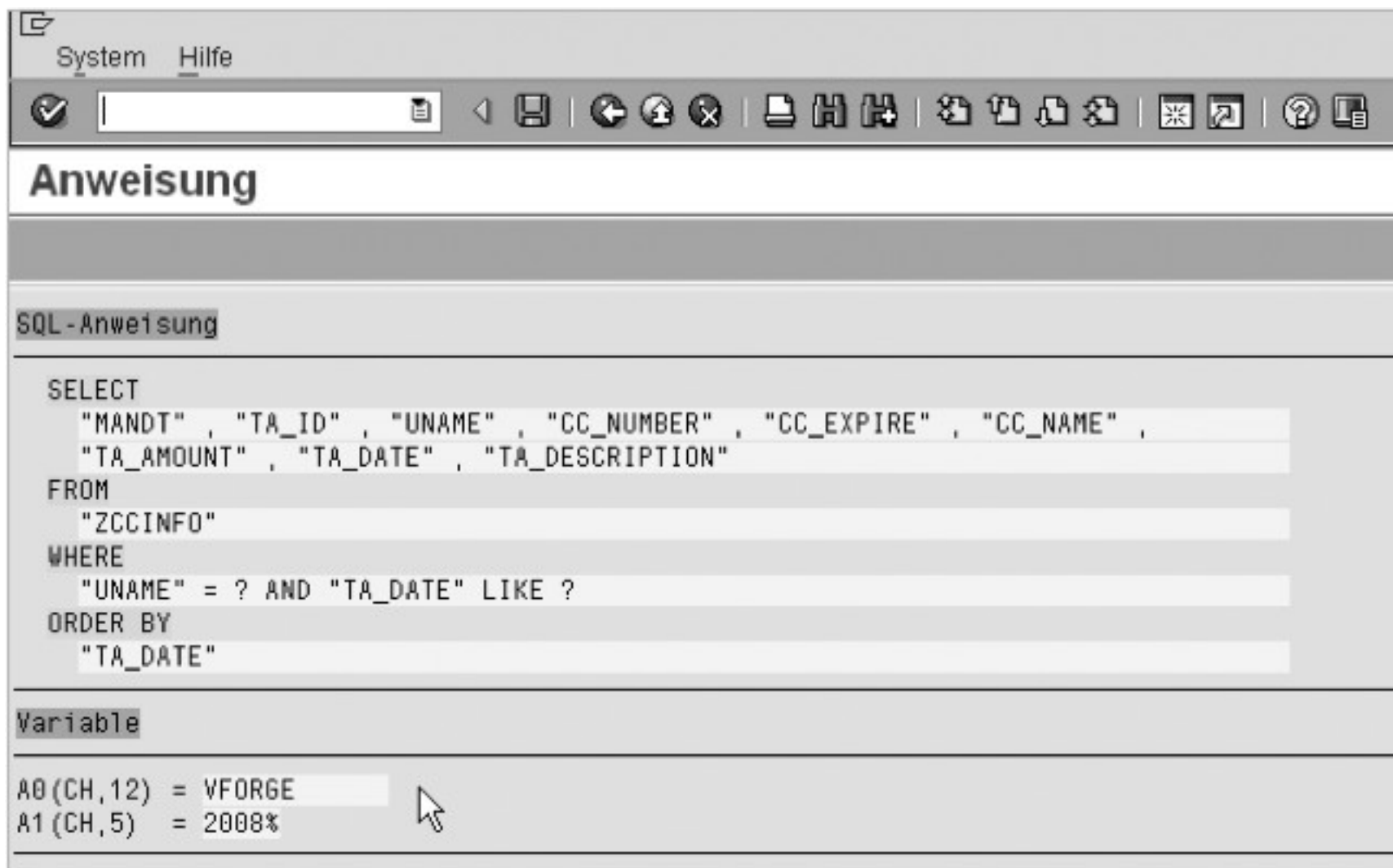
- ABAP/BSP vs. OWASP Top 10
- Business Server Pages
 - ▶ Inline ABAP in HTML
 - ▶ Die HTMLB-Tag-Bibliothek
- Open SQL
 - ▶ Dynamisches Open SQL
 - ▶ SQL-Injections
- Summary

Open SQL

- Open SQL eingebettet in ABAP
- Wird intern in Prepared Statements umgewandelt
- SQL-Statement und Benutzerdaten sauber getrennt, keine SQL-Injection möglich

```
1  SELECT * FROM ZCCINFO
2  INTO l_zccinfo
3  WHERE uname = l_uname
4  AND ta_date = l_date.
```

Open SQL



The screenshot shows a SAP system window with a menu bar (System, Hilfe) and a toolbar. Below the toolbar is a section titled "Anweisung" (Instruction). Underneath, there is a section titled "SQL-Anweisung" (SQL Instruction) containing the following SQL query:

```
SELECT
  "MANDT" , "TA_ID" , "UNAME" , "CC_NUMBER" , "CC_EXPIRE" , "CC_NAME" ,
  "TA_AMOUNT" , "TA_DATE" , "TA_DESCRIPTION"
FROM
  "ZCCINFO"
WHERE
  "UNAME" = ? AND "TA_DATE" LIKE ?
ORDER BY
  "TA_DATE"
```

Below the SQL instruction is a section titled "Variable" (Variable) containing the following definitions:

```
A0(CH,12) = VFORGE
A1(CH,5) = 2008%
```

Dynamisches Open SQL - Beispiel

Show your recent transactions

Please select year and month to filter your transactions:
Year Month

Results:

Your recent credit card transactions

CC_NUMBER	CC_EXPIRE	CC_NAME	TA_DESCRIPTION	TA_AMOUNT	TA_DATE
4149253627344523	11/09	PETER JACKSON	DRUG STORE	65,00	15.01.2008
4149253627344523	11/09	PETER JACKSON	PLANE TICKET	80,00	19.01.2008
4149253627344523	11/09	PETER JACKSON	THANK YOU FOR SHOPPING AT WALMART	180,00	08.02.2008
4149253627344523	11/09	PETER JACKSON	BMW OSTER SAYS THANK YOU FOR YOUR VISIT	2870,00	12.02.2008
4149253627344523	11/09	PETER JACKSON	CRAFTSMAN BILL	420,00	14.03.2008
4149253627344523	11/09	PETER JACKSON	LACOSTE STORE SAYS THANK YOU	360,00	27.03.2008
4149253627344523	11/09	PETER JACKSON	BAKERY	78,00	07.04.2008
4149253627344523	11/09	PETER JACKSON	FLEUROP FLOWER ORDERING	90,00	10.04.2008
4149253627344523	11/09	PETER JACKSON	AMAZON SAYS THANK YOU FOR PURCHASING A DVD PLAYER	1234,00	18.04.2008
4149253627344523	11/09	PETER JACKSON	OPTICIAN	230,00	04.05.2008

.. Page of 1 ..

Dynamisches Open SQL

- Interpretiert String-Literal als SQL-Statement
- Keine Enkodierungsfunktionen vorhanden
 - ▶ Benutzerdaten können nicht sauber von SQL-Kommandos getrennt werden
 - ▶ SQL-Injection wahrscheinlich, wenn Benutzerdaten in dynamisches SQL-Statement gelangen

```
1  SELECT (l_felder) FROM (l_table)
2  INTO l_zccinfo
3  WHERE (l_where) .
```

Dynamisches Open SQL - Beispiel

Business Server Page (BSP) error

Business Server Page (BSP) error

What happened?
Calling the BSP page was terminated due to an error.

SAP Note

- The following error text was processed in the system:
An exception with the type CX_SY_DYNAMIC_OSQ_L_SYNTAX occurred, but was neither handled locally, nor declared in a RAISING clause

Exception Class	CX_SY_DYNAMIC_OSQ_L_SYNTAX
Error Name	SAPSQL_WHERE_PARENTHESES
Program	CL_O27VCP9ZMBO80HYLVYZQJUJ9EFLCP
Include	CL_O27VCP9ZMBO80HYLVYZQJUJ9EFLCM008
ABAP Class	CL_O27VCP9ZMBO80HYLVYZQJUJ9EFL
Method	_ONREQUEST
BSP Application	ZVF_CCDATA_DEMO
BSP Page	DETAILS.HTM
Line	48
Long text	-

Error type: Exception
Your SAP Business Server Pages Team

Dynamisches Open SQL - Beispiel

Show your recent transactions

bc/bsp/sap/zvf_ccdata_demo/details.htm?input_year=2008&input_month=00' OR mandt LIKE '%

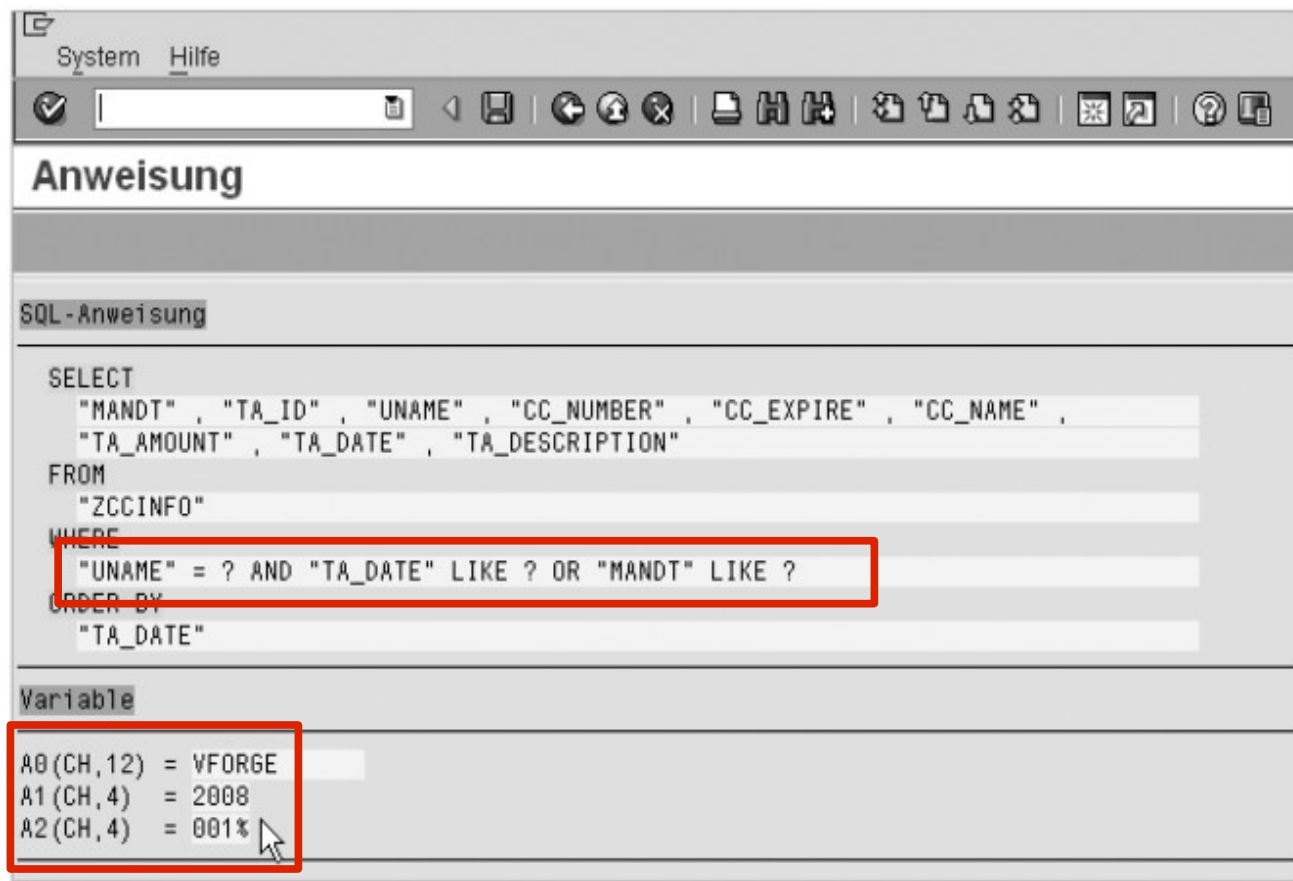
Please select year and month to filter your transactions:
 Year Month

Results:

Your recent credit card transactions

CC_NUMBER	CC_EXPIRE	CC_NAME	TA_DESCRIPTION	TA_AMOUNT	TA_DATE
4130865326190251	11/10	SARAH BLANKS	THE DRUGSTORE SAYS THANK YOU	24,00	03.01.2008
2810491793517364	02/08	LINDA OKE	IMAGE MODE SAYS THANK YOU	45,00	04.01.2008
2810418361048230	10/08	MAI LING	THAI STORE	120,00	06.01.2008
5110282427041850	12/12	ABRAHAM SMITH	HOSPITAL ACCOUNT	490,00	07.01.2008
5067392017603901	08/10	PETER MARSHALL	FODYS SAYS THANK YOU	89,00	07.01.2008
3104827308174926	06/09	LISA IVKIC	THE FITNESS COMPANY SAYS THANK YOU FOR YOUR VISIT	45,00	07.01.2008
3106916205915295	05/10	BEN LANDS	HAIRDRESSER	120,00	07.01.2008
4182018462916318	07/12	TONI STANSFIELD	DELL SAYS THANK YOU FOR YOUR ORDER	1400,00	11.01.2008
6206194916490153	04/12	PIERRE CARDIN	CRAFTSMAN BILL	4980,00	12.01.2008
5194028301639618	10/09	DAVID GARNER	FLEUROP FLOWER ORDERING	50,00	12.01.2008
5104784134789216	10/11	JOHN DOE	DENTAL BILL	1750,00	12.01.2008
4182018462916318	07/12	TONI STANSFIELD	STARBUCKS FRANKFURT	17,00	13.01.2008
4130865326190251	11/10	SARAH BLANKS	H&M ACCOUNT	74,00	15.01.2008
4149253627344523	11/09	PETER JACKSON	DRUG STORE	65,00	15.01.2008
2810418361048230	10/08	MAI LING	LUFTHANSA SAYS THANK YOU	1200,00	17.01.2008
6206194916490153	04/12	PIERRE CARDIN	LACOSTE STORE SAYS THANK YOU	7820,00	18.01.2008
3106916205915295	05/10	BEN LANDS	BOOK STORE	450,00	19.01.2008

Dynamisches Open SQL



The screenshot shows a SAP system window with a menu bar (System, Hilfe) and a toolbar. The main area is titled "Anweisung" and contains a section for "SQL-Anweisung". The SQL query is as follows:

```
SELECT
  "MANDT" , "TA_ID" , "UNAME" , "CC_NUMBER" , "CC_EXPIRE" , "CC_NAME" ,
  "TA_AMOUNT" , "TA_DATE" , "TA_DESCRIPTION"
FROM
  "ZCCINFO"
WHERE
  "UNAME" = ? AND "TA_DATE" LIKE ? OR "MANDT" LIKE ?
ORDER BY
  "TA_DATE"
```

Below the SQL query is a section for "Variable" with the following definitions:

```
A0(CH,12) = VFORGE
A1(CH,4) = 2008
A2(CH,4) = 001%
```

Red boxes highlight the WHERE clause in the SQL query and the variable definitions in the Variable section.

Zusammenfassung Open SQL

- Dynamisches Open SQL führt leicht zu SQL-Injection-Schwachstellen
 - ▶ Keine Enkodierungsfunktion vorhanden
 - ▶ Prepared-Statement-Injection
- Vermeiden Sie dynamisches Open SQL in ABAP wenn immer möglich!

Zusammenfassung

OWASP Top 10	TODO
A1 - Cross Site Scripting (XSS)	✓
A2 - Injection Flaws	✓
A3 - Malicious File Execution	X
A4 - Insecure Direct Object Reference	X
A5 - Cross Site Request Forgery (CSRF)	X
A6 - Information Leakage and Improper Error Handling	X
A7 - Broken Authentication and Session Management	-
A8 - Insecure Cryptographic Storage	X
A9 - Insecure Communications	X
A10 - Failure to Restrict URL Access	X

Zusammenfassung

- SAP-Web-Frontends weit verbreitet und prozessieren meist unternehmenskritische Daten
- Weitere SAP-Web-Frontend-Technologien:
 - ▶ ✓ **Business Server Pages (BSP)**
 - ▶ ✗ Web Dynpro
 - ▶ ✗ Internet Transaction Server
 - ▶ ✗ Eigene HTTP-Handler
 - ▶ ...
- BSP verlangt viel Eigenleistung für sichere Entwicklung

Literatur

- *“Sichere ABAP-Programmierung”* - SAP Press, Wiegenstein, Schumacher, Schinzel, Weidemann
<http://sap-press.de/2037>
- *“SAP Documentation”* <http://help.sap.com/>
- *“Secure Programming – ABAP”* - SAP AG
<http://www.sdn.sap.com/irj/scn/index?rid=/library/uuid/17a4f828-0b01-0010-8da>
- *“Security Scanner für ABAP”* - <http://codeprofilers.com/>

- *“Vmovie: Security Knowledge on Stage”*
<http://secure-abap.de/media>
- *“OWASP Top 10”*