Maßnahmen im Entwicklungsprozess zur Sicherstellung der Anwendungssicherheit

Dr. Achim D. Brucker

31. Januar 2013

Public

SAP Today

54,500+
SAP employees worldwide

120
countries

25
industries

37
languages

75
country offices

1,200+
services partners worldwide

Agenda

Why is SAP using Static Code Analysis?

Secure Development Lifecycle at SAP

Static Code Analysis at SAP

Challenges and Outlook

Costs of Computer Hacks

Costs of Computer Hacks

TJX Company, Inc. (2007)	$ 250 million
Sony (2011)	$ 170 million
Heartland Payment Systems (2009)	$ 41 million

"A hack not only costs a company money, but also its **reputation** and the **trust** of its customers. It can take years and millions of dollars to repair the damage that a single computer hack inflicts."

(http://financialedge.investopedia.com/financial-edge/0711/Most-Costly-Computer-Hacks-Of-All-Time.aspx)

Has Sony been Hacked this Week? http://hassonybeenhackedthisweek.com/

Time-line of the Sony Hack(s) (excerpt):

- 2011-04-20 Sony PSN goes down
- 2011-05-21 Sony BMG: data of 8300 users leaked (**SQL Injection**)
- 2011-05-23 Sony Japanese database leaked (**SQL Injection**)
- 2011-05-24 Sony Canada: roughly 2,000 leaked (**SQL Injection**)
- 2011-06-05 Sony Pictures Russia (**SQL Injection**)
- 2011-06-06 Sony Portugal: **SQL injection**, iFrame injection and XSS
- 2011-06-20 20th breach within 2 months,
  177k email addresses were grabbed via **a SQL injection**

(http://hassonybeenhackedthisweek.com/history)

A Bluffers Guide to SQL Injection

Assume an SQL Statement for

```
statement="SELECT * FROM `users` WHERE `name` = '" + userName + "';"
```

What happens if we choose the following (weird) `userName`:

```
userName = "' or '1='1"
```

Resulting in the following statement:

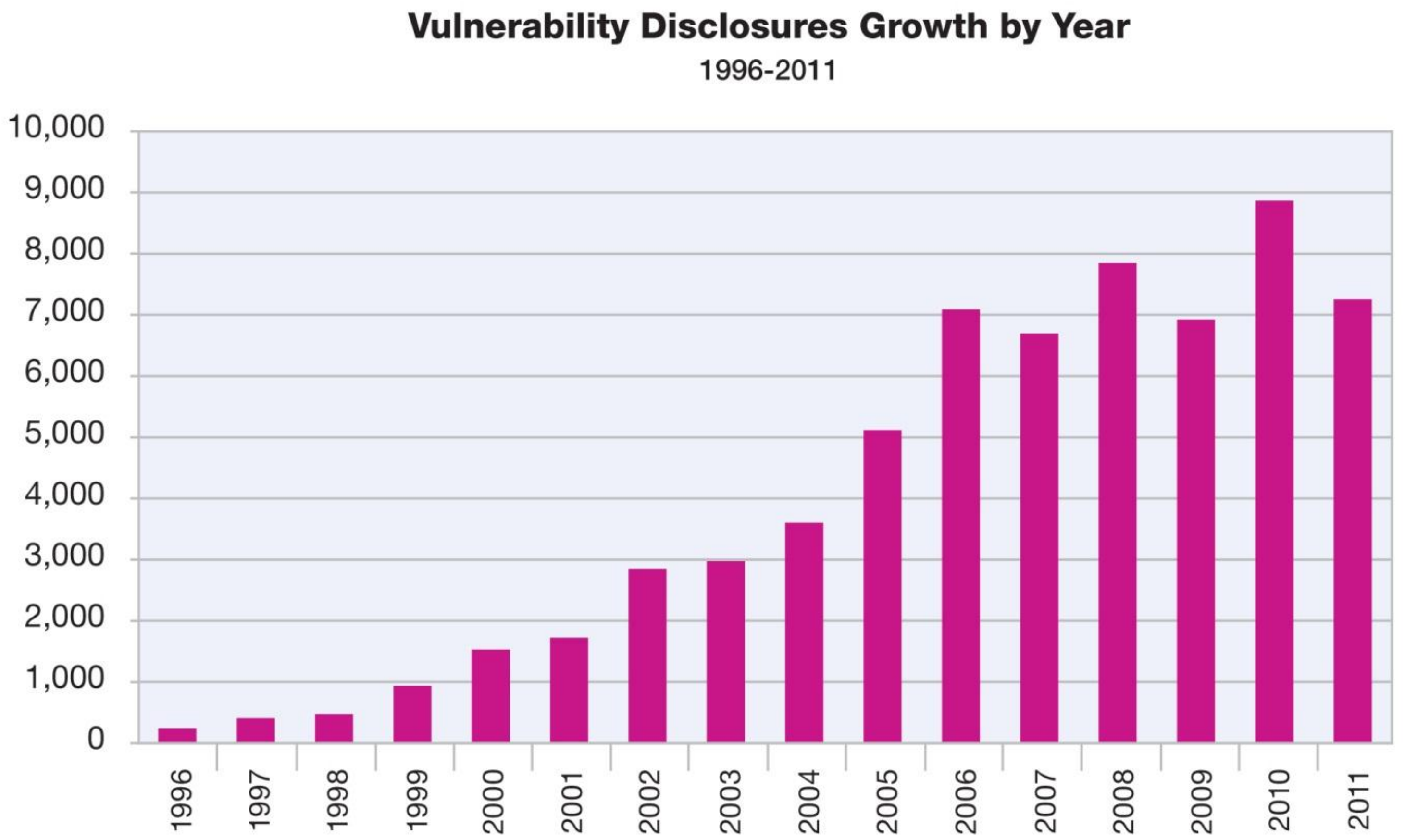
```
statement = "SELECT * FROM `users' WHERE `name' = `' or '1'='1';"
```

Which is equivalent to

```
statement = "SELECT * FROM `users`;"
```

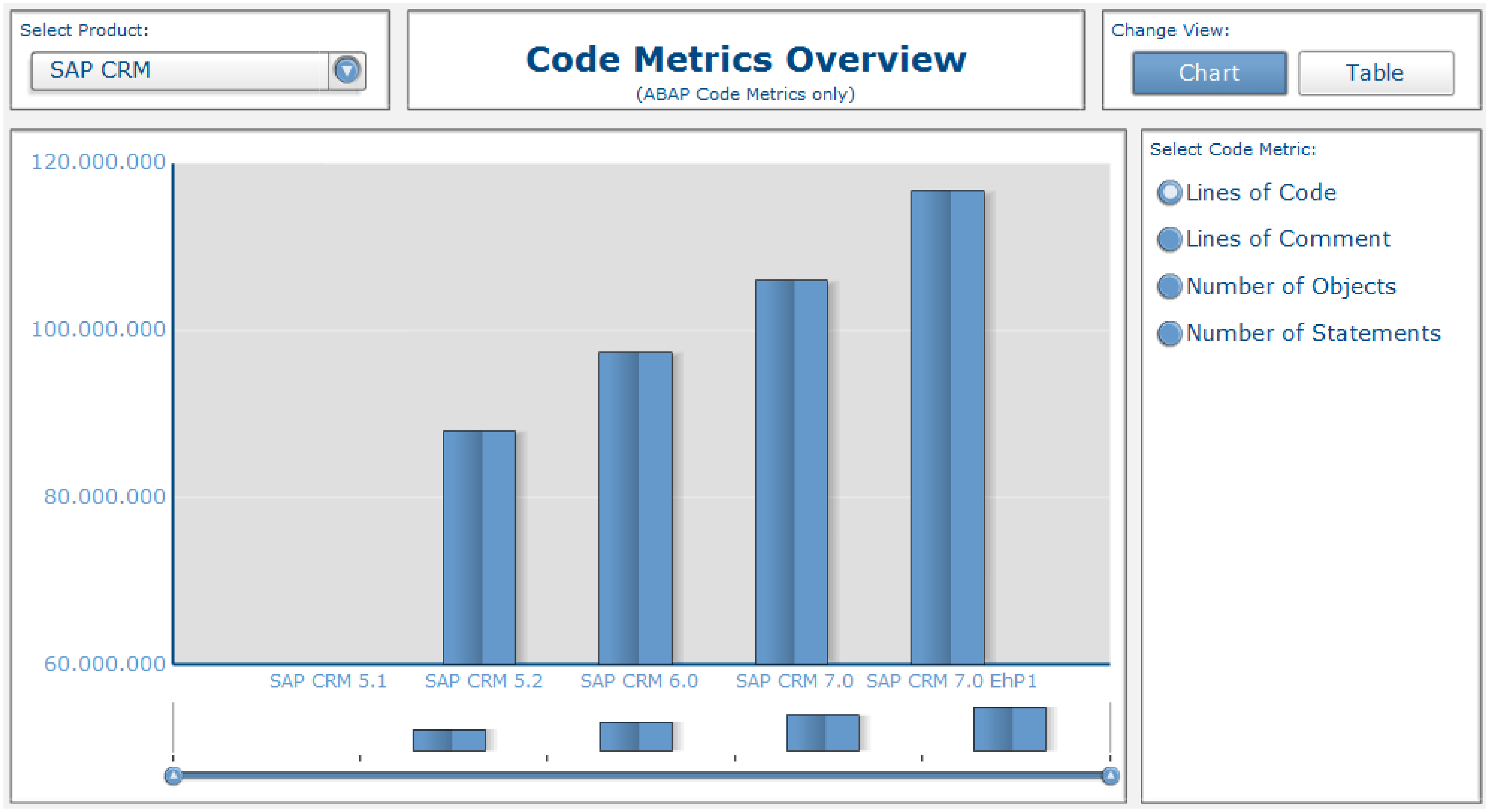
And selects the information about all users stored in the table `users`

Insecure Software



Source: IBM X-Force® Research and Development

Evolution of Code

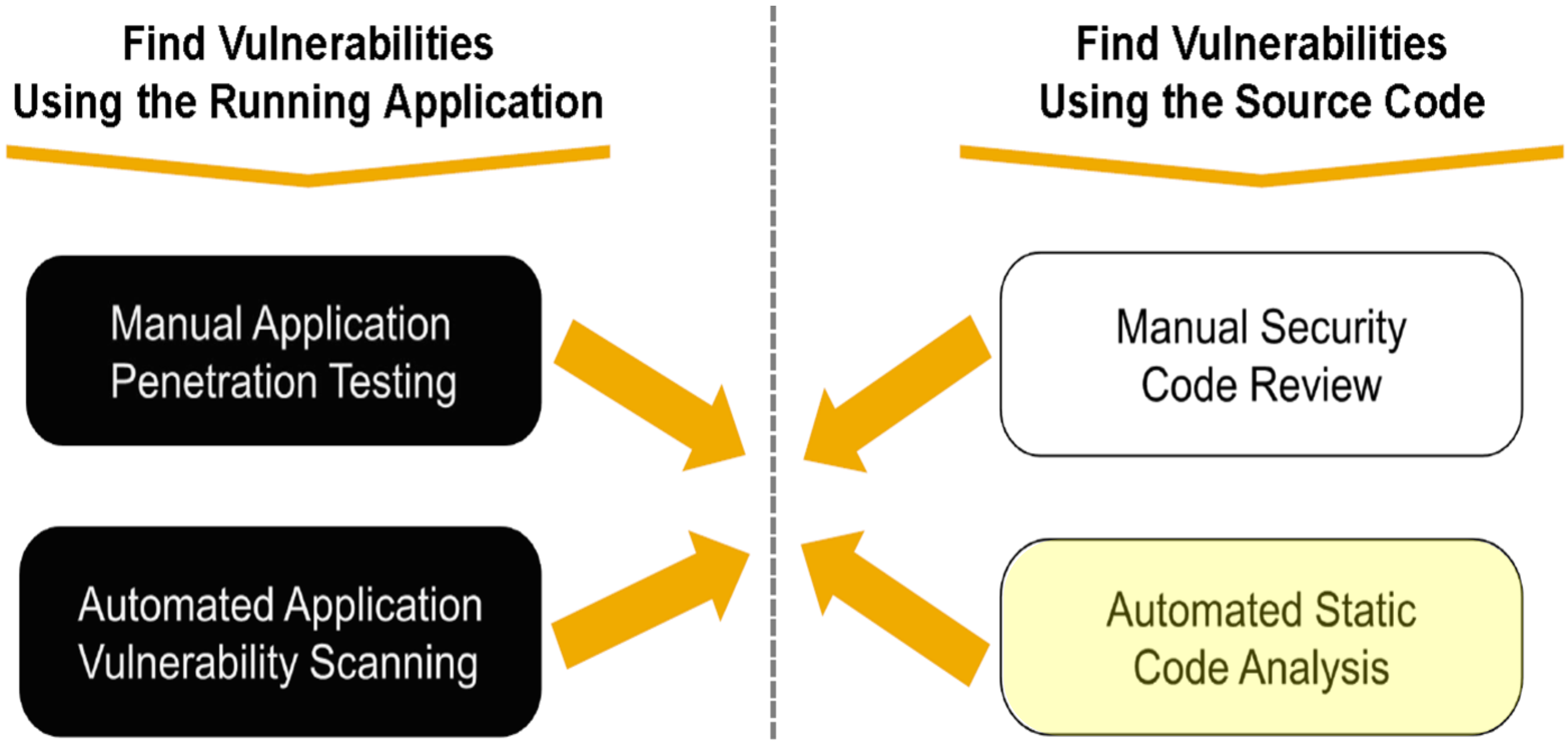

Select Product: SAP CRM

Code Metrics Overview
(ABAP Code Metrics only)

Change View: [✓ Chart] Table

Select Code Metric:
[✓ Lines of Code]
Lines of Comment
Number of Objects
Number of Statements

Security Testing


Find Vulnerabilities Using the Running Application
Find Vulnerabilities Using the Source Code
Manual Application Penetration Testing
Automated Application Vulnerability Scanning
Manual Security Code Review
Automated Static Code Analysis

Dynamic Security Testing

Characteristics

- Black box approach
- Sends input to applications and analyses response

Advantages

- Provides concrete examples (attacks)
- Analyze dataflows across multiple components

Disadvantages

- Coverage unclear
- Requires test system

Static Security Testing

Characteristics

- White box approach
- Analyses abstraction of the source (binary)

Advantages

- Explores all data paths / control flows
- Can analyze single modules (unit test)

Disadvantages

- High false positive rate (not exploitable findings)
- Does not consider application environment

Security Code Scans at SAP: Overview

Started rollout in June 2010

Centrally guided by a project team

- Definition of Security Requirements
- Establishment of Scan Infrastructure

Support of the most important languages

SAP development and third party code

Agenda

Why is SAP using Static Code Analysis?

Secure Development Lifecycle at SAP

Static Code Analysis at SAP

Challenges and Outlook

First Step: Security Training

Education

- The prerequisite for achieving a high security quality

Security awareness

- Reducing the number of “built-in” security problems

Trained persons

- Analyze and fix vulnerabilities much more efficiently

Trainings

- Secure Programming, Build & Scan, Auditing, ….

Secure Development Lifecycle (SDLC) at SAP

Structure the investment of time and resources

- to safeguard a high level of security
- to ensure security standards across all areas

Security requirements

- are taken into account and
- are implemented

in all phases of product development

The Different Roles

Developer

- fixes software security issues

Security Expert

- review scan results, decides on fixes

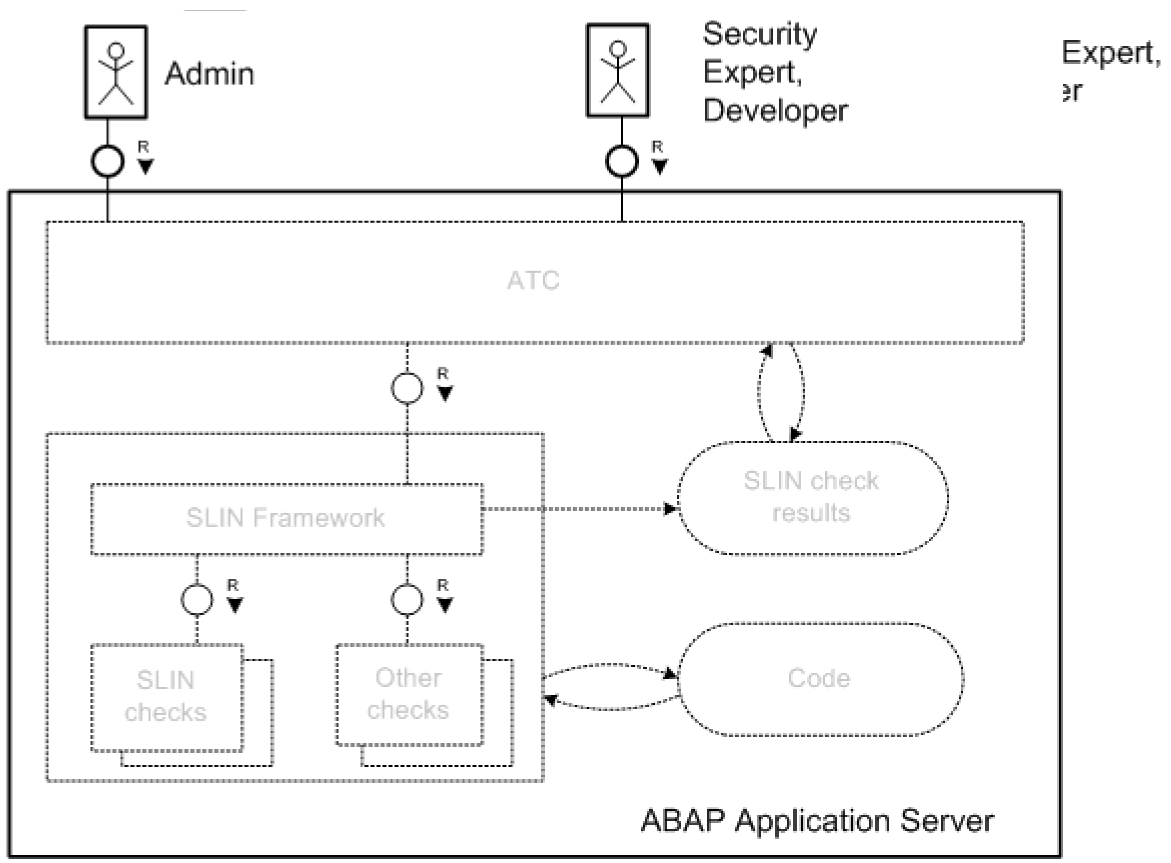
Build Master

- scans the source code, manages results

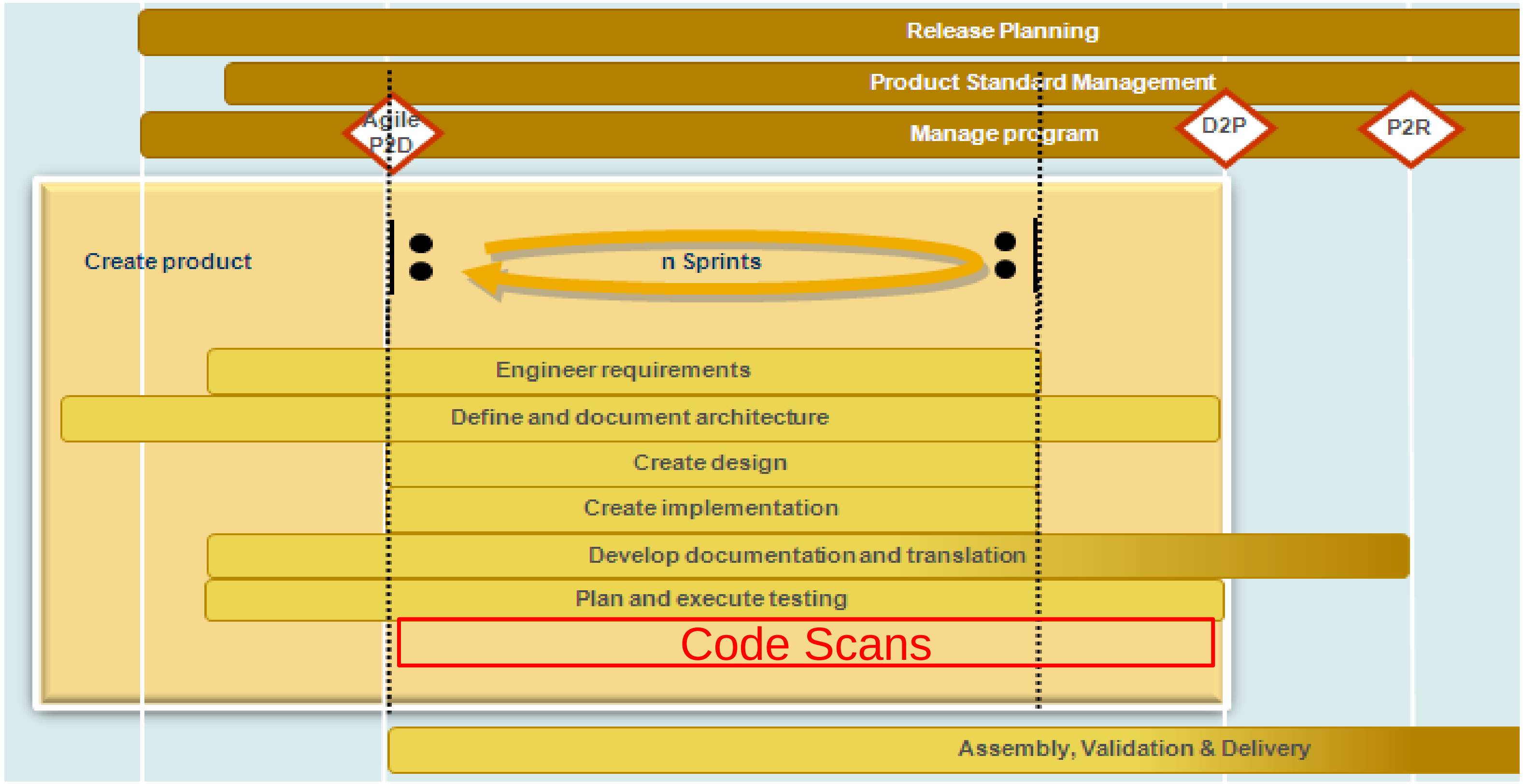
Scrum Master

- requests scan, assigns vulnerabilities to developers

Infrastructure



SAP Secure Software Development Life Cycle



For passing D2P Q-gate, evidence has to be provided that the source code has been scanned and exploitables have been fixed.

P2D: Planning to Development. / D2P: Development to Production. /

P2R: Production to Ramp-up (gradual roll-out to customers).

Third Party Code

Third party code

- Open Source libraries and frameworks
- Freeware
- other third party components

Different approaches

- SAST analysis by SAP
- Trusted (certified) vendors
- Certificate from trusted third party (e.g., based on binary analysis)
- SLA with vendor

Agenda

Why is SAP using Static Code Analysis?

Secure Development Lifecycle at SAP

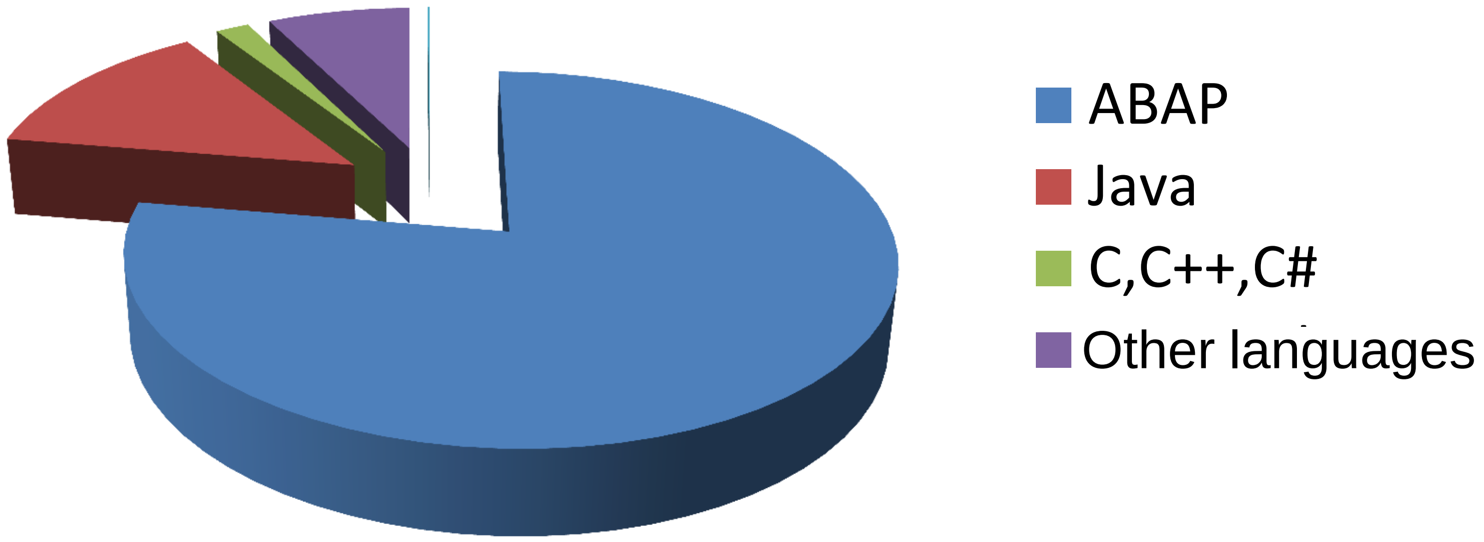
Static Code Analysis at SAP

Challenges and Outlook

Code Scan Facts

Over 2000 developers are using SAST tools

Over 500 MLOC scanned



Statistics Jan 2012

Security Scan Tools used at SAP

Language	Scan Application
ABAP	SAP
C/C++	Coverity
Others	HP/Fortify

Security Requirements

SAP on Corporate Security Requirements

- SAP Applications shall be free of backdoors
- SQL injection vulnerabilities shall be avoided
- Cross-Site Scripting vulnerabilities shall be prevented
- Directory traversal vulnerabilities shall be prevented
- The system shall be protected against buffer overflow vulnerabilities

OWASP Top 10

CWE/SANS Top 25 2011

CVE

Continuous Improvement

Collect feedback from the

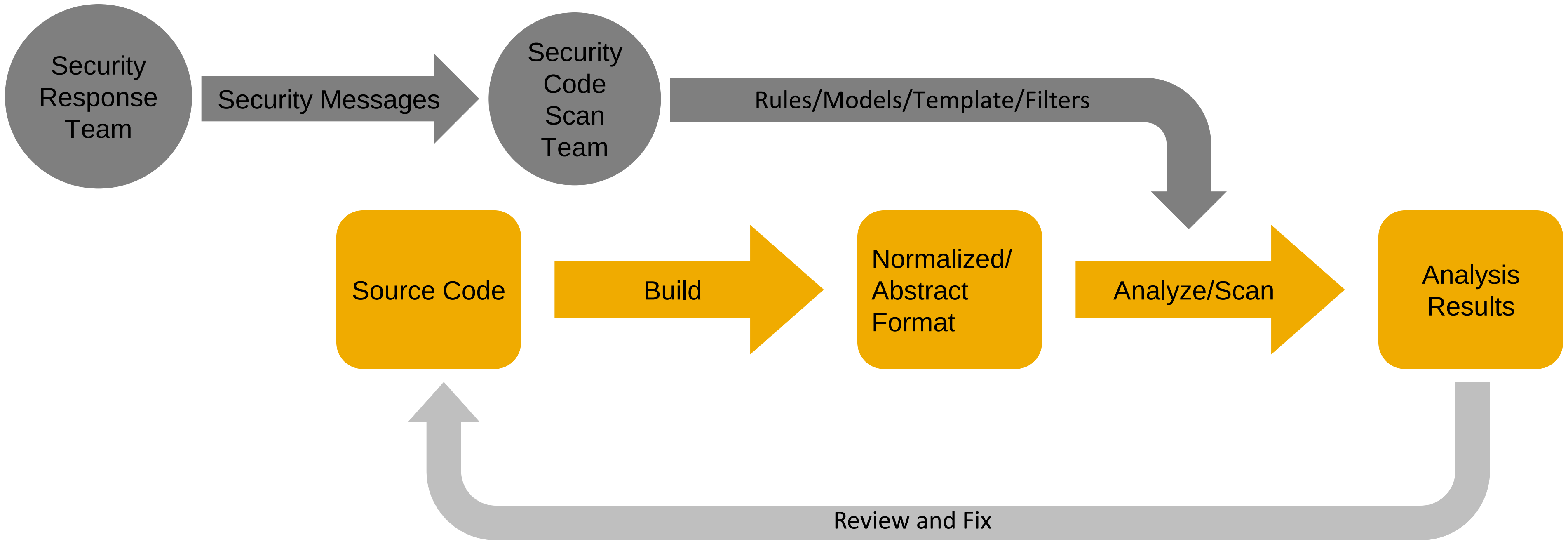
- Product Security Response Team
- Development Teams

Develop rules/models to improve the scans

Continuously improve the infrastructure

Continuously improve the rollout process

Input to Improve Code Scans



Further input channels:
Development teams, internal research, scan reviews, code reviews

Lessons Learned

Scans have to be obligatory

- **but not** introduced ‘brute force’

Establish Secure Development Life Cycle

- make scans a natural part of development

Plan carefully

  - Do not start with scans right before Dev. Close
  - Do it regularly (nightly)
  - Do regression testing of new versions of the used tools
  - Do continuously discuss new threats with the security community

Do not introduce changes during development

Agenda

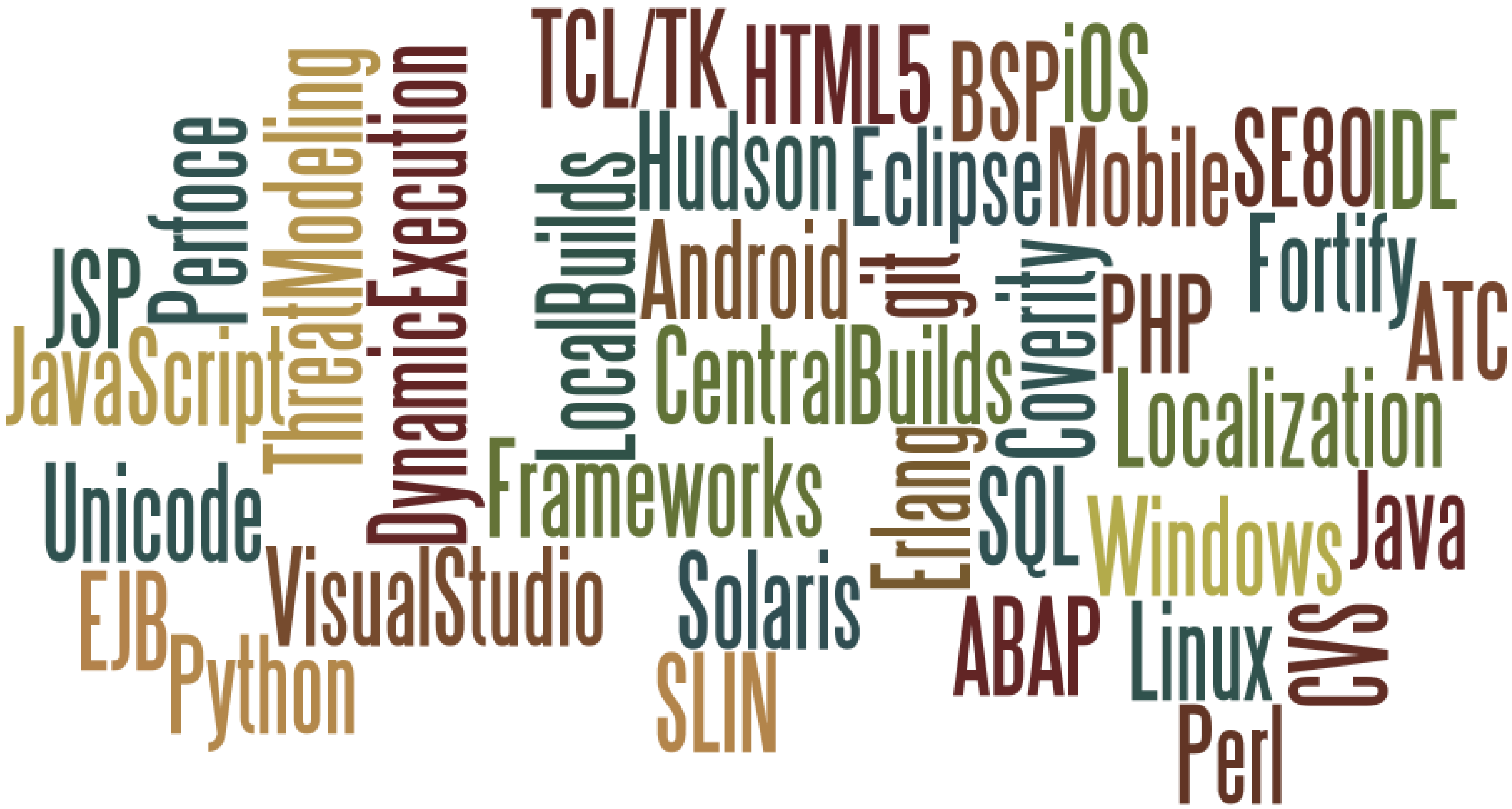
Why is SAP using Static Code Analysis?

Secure Development Lifecycle at SAP

Static Code Analysis at SAP

Challenges and Outlook

Challenges


TCL/TK
HTML5
BSP
iOS
Perfoce
ThreatModeling
DynamicExecution
LocalBuilds
Hudson
Eclipse
Mobile
SE80
IDE
JSP
Android
git
Coverity
PHP
Fortify
ATC
JavaScript
CentralBuilds
Localization
Unicode
Frameworks
Erlang
SQL
Windows
Java
EJB
VisualStudio
Solaris
ABAP
Linux
CVS
Python
SLIN
Perl

JavaScript I
Unerstand the DOM

Assume the following (simplified) index.html:

```
<TITLE>Welcome!</TITLE>
  Hi
  <SCRIPT>
    var pos=document.URL.indexOf("name=")+5;
    document.write(document.URL.substring
                                (pos,document.URL.length));
  </SCRIPT>
  Welcome to our system
```

And a call

```
index.html?name=<script>alert(document.cookie)</script>
```

Resulting in a DOM-based XSS attack

Warning: DOM implementations are Browser specific

JavaScript II
Dynamic Evaluation

A simple script tag:

```
<script language="javascript">
   document.write("<script src='other.js'></script>");
</script>
```

Dynamic creation of script tags

```
var oHead = document.getElementsByTagName('HEAD').item(0);
var oScript= document.createElement("script");
oScript.type = "text/javascript";
oScript.src="other.js";
oHead.appendChild( oScript);
```

Or using eval() directly (not shown here)

JavaScript II
Dynamic Evaluation

A simple script tag:

```
<script language="javascript">
   document.write("<script src='other.js'></script>");
</script>
```

Dynamic creation of script tags

```
var oHead =ndocument.getElementsByTagName('HEAD').item(0);
var oScript= document.createElement("script");
oScript.type = "text/javascript";
oScript.src="other.js";
oHead.appendChild( oScript);
```

Or using eval() directly (not shown here)

JavaScript III
Server-Side JavaScript

Combining the complexity of both worlds:

```
var entry=JSON.parse(data);
query = “insert into \”FOO(“.NAME”)\””;
var conn = $.db.getConnection();
conn.execute(query);
```

Challenges: Current Trends

"You cannot pay people well enough, to do proper code audits. I tried it."

Yaron Minsky, Jane Street Capital

SAST works very well for

- "traditional" programming languages
- Analyzing data paths within one technology

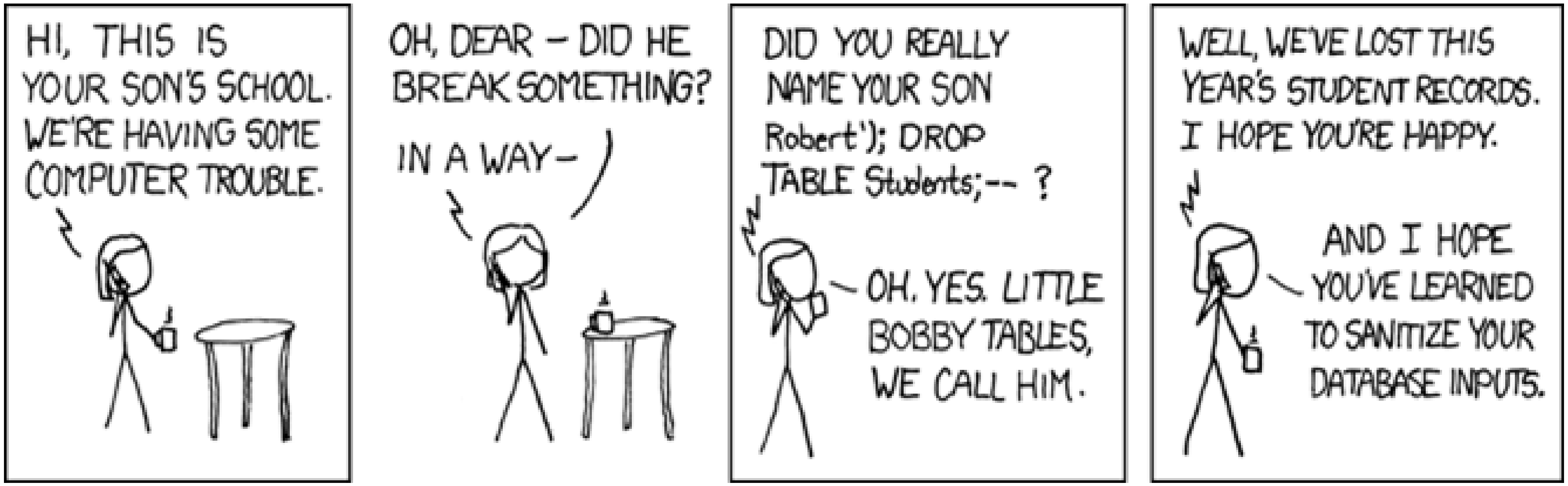
Many new development uses JavaScript

- HTML5 / JavaScript UIs
- Server-side JavaScript

JavaScript

- Untyped / dynamically typed
- Dynamic programming model

Thank you


HI, THIS IS YOUR SON'S SCHOOL. WE'RE HAVING SOME COMPUTER TROUBLE.
OH, DEAR – DID HE BREAK SOMETHING?
IN A WAY–
DID YOU REALLY NAME YOUR SON Robert'); DROP TABLE Students;-- ?
OH. YES. LITTLE BOBBY TABLES, WE CALL HIM.
WELL, WE'VE LOST THIS YEAR'S STUDENT RECORDS. I HOPE YOU'RE HAPPY.
AND I HOPE YOU'VE LEARNED TO SANITIZE YOUR DATABASE INPUTS.

http://xkcd.com/327/

Contact information:

Dr. Achim D. Brucker
Senior Researcher
Vincenz-Priessnitz-Strasse 1, 76131 Karlsruhe
achim.brucker@sap.com