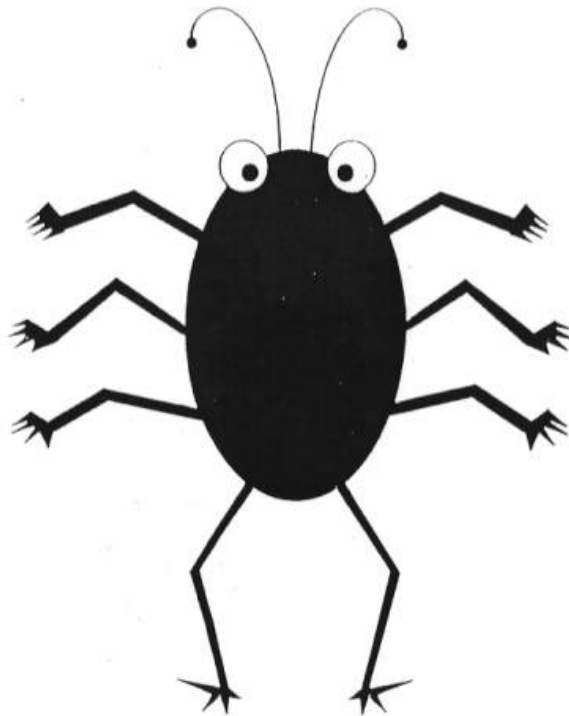


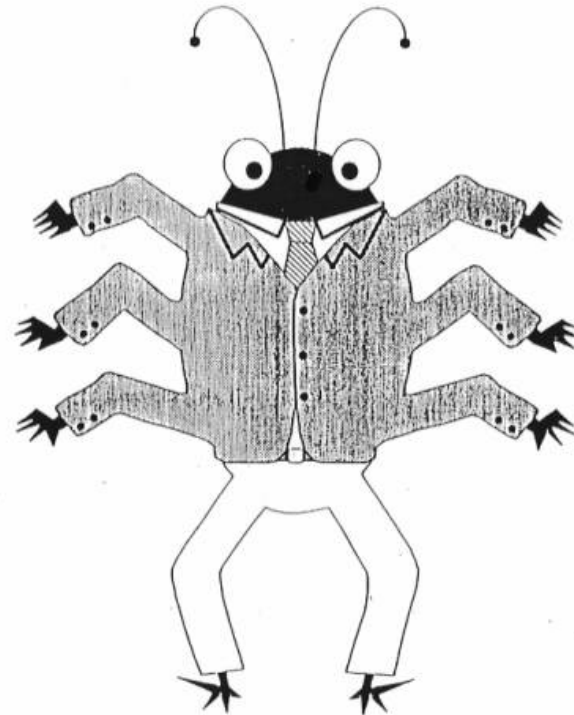
Softwarefehler Top 25

Das Kabinett des Grauens

Was ist ein Bug?



BUG



FEATURE

Reaktionen auf Bugs/Sicherheitsprobleme

- Studenten wollen sofort die „Silver Bullet“
 - „The one fix to fix them all“
- Entwickler weisen Problem von sich
 - „Das kann hier bei uns nicht passieren“
- Firmen „passt das nicht ins Budget“
- Gut reagieren in drei Schritten:
 - Verstehen
 - Untersuchen
 - Beheben

Die Top 25 seit 2009

- Aufgestellt vom CWE (Common Weakness Enumeration)
 - <http://cwe.mitre.org/top25>
- Jährlich aktualisiert (zuletzt 2011)
- Liste der häufigsten Schwachstellen
 - Nicht abhängig von Programmiersprache
- Minimal-Wissensstand für sicherheitsbewusste Entwickler



Drei Hauptkategorien

- Unsichere Zusammenarbeit von Komponenten
 - Insecure Interaction Between Components
- Riskante Verwendung von Ressourcen
 - Risky Resource Management
- Schwache Sicherheitsmaßnahmen
 - Porous Defenses



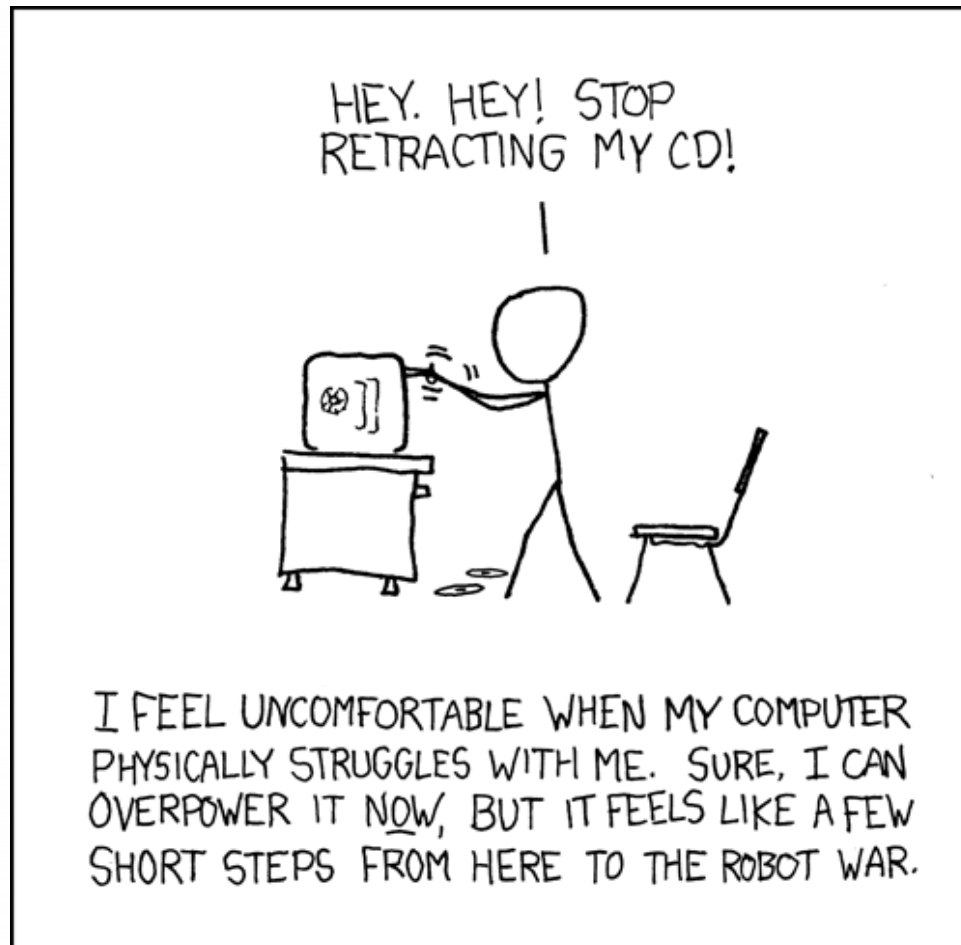
Unsichere Zusammenarbeit zwischen Komponenten

Schwachstellen in Bezug auf unsichere Datenweitergabe u. -empfang zwischen

- Komponenten/Modulen
- Programmen
- Prozessen/Threads
- Systemen
- Webseiten

Komponente 1: Mensch

Komponente 2: CD-Laufwerk

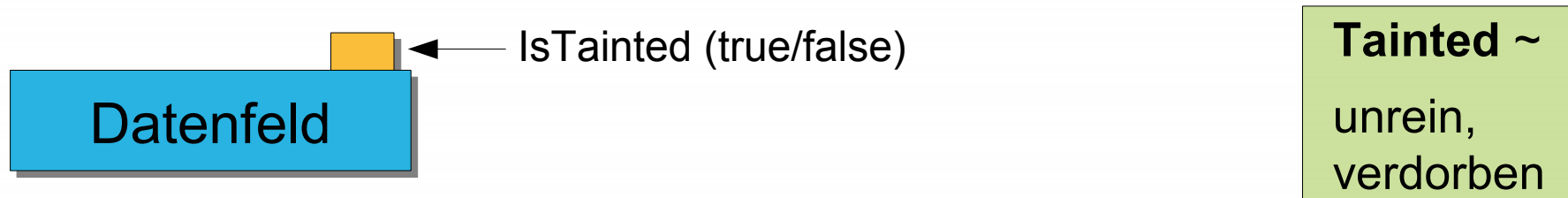


Unsachgemäße Überprüfung von Eingaben

- Improper Input Validation
- Beispiele:
 - Eingabe von Buchstaben bei Zahlenwerten
 - Minuswerte bei Preisen oder Liefermengen
 - Ungeprüfte (sehr hohe) Zahlen
 - Unmögliche Datumswerte (30.02.2009)
- Abhilfe:
 - Taint mode (Perl)

Taint Mode

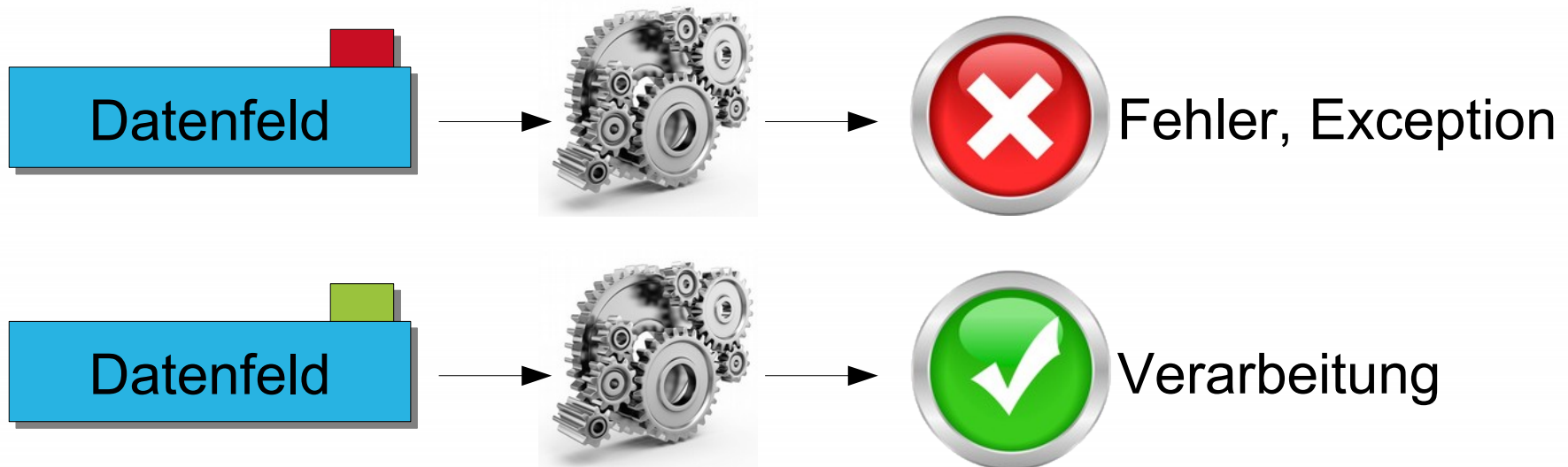
- Jedes Datenfeld hat Zusatzinformation



- Ist immer gesetzt, wenn Datum nicht aus dem Code stammt (also Eingabe ist)
- Berechnungen mit einem tainted Datum ergibt ein tainted Ergebnis
- Explizite Untaint(Datenfeld)-Operation bei der Eingabe-Validierung

Taint Mode

- An vielen Stellen im Programm werden keine tainted Daten angenommen
 - Schreiben in Datenbank, Datei, Socket, etc.
 - Anzeige in Konsole, Monitor, etc.



Unsachgemäße Codierung oder Auszeichnung von Ausgaben

- Improper Encoding or Escaping of Output
- Steigbügel vieler Injektions-Attacken
- Beispiele:
 - HTML-Code-Beispiel in Web-Eingabefeld
 - Verarbeitung von URLs
 - Textdateien (UTF-8, ISO-LATIN, ASCII)
- Erster Schritt zur Abhilfe:
 - Explizite Definition aller Encodings

Unvermögen, die Struktur einer SQL-Abfrage zu wahren

- Failure to Preserve SQL Query Structure
- SQL-Injection
- Ein Bild sagt mehr als tausend Worte:



The image shows a red-themed web interface for finding jobs. At the top, it says "Find great jobs." Below this is an SQL query template with input fields for "JOB_TYPE", "JOB_CITY", and "JOB_SATISFACTION". The query is: `SELECT * FROM JOBS WHERE JOB_TYPE = " " AND JOB_CITY = " " AND JOB_SATISFACTION = "HIGH";`. At the bottom left is the "Dice" logo with the tagline "The Career Hub for Tech Insiders". At the bottom right is a yellow button labeled "RUN QUERY".

Find great jobs.

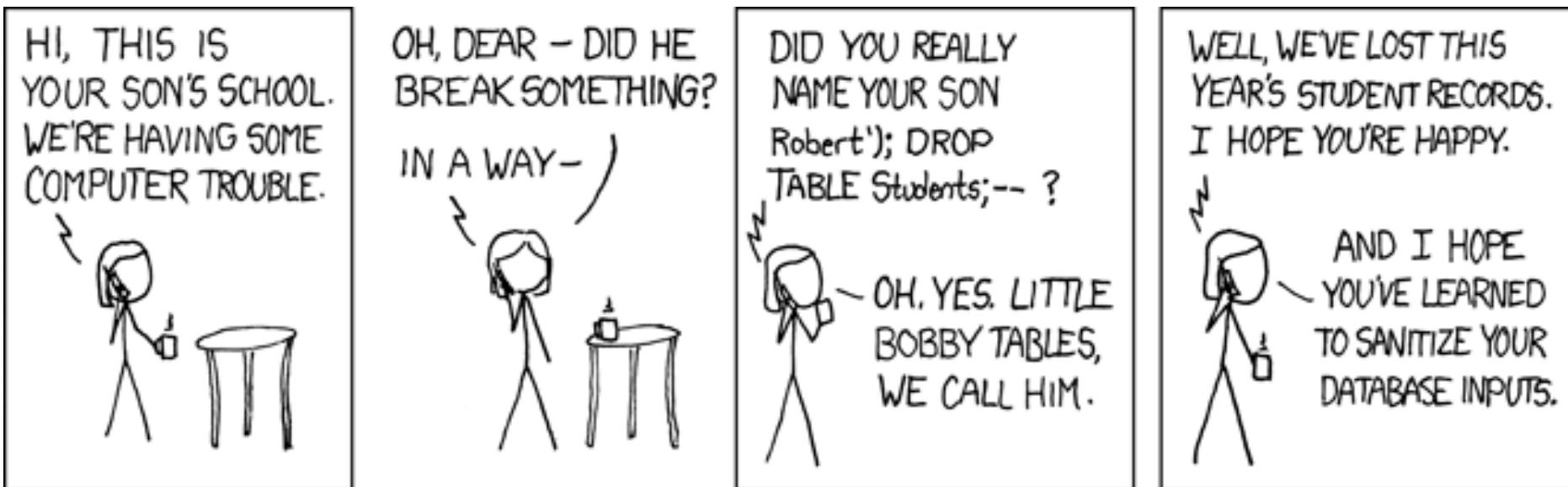
```
SELECT * FROM JOBS
WHERE
JOB_TYPE = " "
AND
JOB_CITY = " "
AND
JOB_SATISFACTION = "HIGH";
```

Dice™
The Career Hub for Tech Insiders™

RUN QUERY

SQL-Injection im Bild

- <http://xkcd.com/327/>



SQL-Injection: Abhilfen?

- Prepared Statements
 - Beispiele für alle Programmiersprachen auf Wikipedia
 - [#Gegenma.C3.9Fnahmen](https://de.wikipedia.org/wiki/SQL_Injection)

Anstatt

```
Statement stmt = con.createStatement();
ResultSet rset = stmt.executeQuery("SELECT spalte1 FROM tabelle WHERE spalte2 = '"
    + spalte2Wert + "';");
```

sollte Folgendes verwendet werden:

```
PreparedStatement pstmt = con.prepareStatement("SELECT spalte1 FROM tabelle WHERE spalte2 = ?");
pstmt.setString(1, spalte2Wert);
ResultSet rset = pstmt.executeQuery();
```

Sicherheitskonzept-Stufen



- Ignorance

- Alle Eingaben werden verarbeitet



- Blacklist

- Filtert auf alle bekannt gefährlichen Inhalte



- Whitelist

- Akzeptiert nur bekannt ungefährliche Inhalte



- Separation

- Verarbeitung getrennt nach Datenursprung

Separation: Trennen der Alphabete

```
SELECT spalte1 FROM tabelle WHERE spalte2 = 'EINGABE';
```

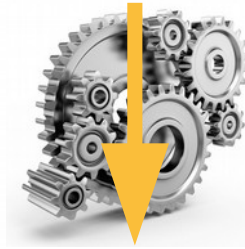
Statement-Code

Eingabe-Daten

- Parser unterscheidet zwischen Statement und Eingabe nur anhand der „Wechsel-Zeichen“ im Statement-Code
 - Können auch Bestandteil der Eingabe sein
- Getrennt, wenn das Statement in einem disjunkten Alphabet formuliert ist
 - Keine zusätzlichen Wechsel-Zeichen durch die Eingabe einfügbar → Keine Injektion

Separation: Trennen der Alphabete

```
SELECT spalte1 FROM tabelle WHERE spalte2 = ?;
```



Prepared
Statement

```
2624864642694768247248942973579538931985839 ?;
```

- Prepared Statements kompilieren den Statement Code getrennt von der Eingabe
 - Kompilieren → Wechsel in anderes Alphabet
- Zum Ausführungszeitpunkt liegen Statement und Eingabe in getrennten Alphabeten vor → keine Injektion möglich

Unvermögen, Webseiten-Struktur zu wahren

- Failure to Preserve Web Page Structure
- Cross-site Scripting („XSS“)
- Die eigene Webseite liefert Skript-Code aus, den andere dort hinterlegt haben
- Ein Benutzer der Webseite führt diesen Skript-Code mit aus
- XSS ausführlich erklärt:
 - www.slideshare.net/Virtual_Forge/xss-cross-site-scripting-explained

Cross-Site Scripting (XSS)

Boodle



Funny cat picture

Suchen

\$SUCHANFRAGE

Sie suchten nach: Funny cat picture

Ergebnisse: 132

Ergebnis 0:
www.funnycats.com

Ergebnis 1:
www.catfacts.com

Ergebnis 2:
www.catordog.com

Cross-Site Scripting (XSS)

Boodle

```
<script type="text/javascript">alert("XSS");</script>
```

Suchen

\$SUCHANFRAGE

Sie suchten nach:

XSS

OK

HTML-Injektion

Drei Stufen des Hacking

- Sich selbst hacken
 - Proof of Concept
 - Tauglichkeit des Angriffs feststellen
- Zufällig andere hacken
 - Vandalism phase
 - Ziellos, diverse Motivationen (Skript-Kiddie)
- Gezielt hacken
 - Profit phase
 - Angriffs zum persönlichen Vorteil (Black hat hacker)



Cross-Site Scripting (XSS)

Boodle

Suchen

Andere beobachten

Andere Besucher suchten nach:

ERP Karlsruhe
Black hat hacking
Unsubscribe from ca
Funny cat picture

XSS

OK

Cross-Site Scripting (XSS)

Boodle

```
<script type="text/javascript">
  var xmlhttp = null;
  xmlhttp = new XMLHttpRequest();
  xmlhttp.open("GET",
    "http://www.ebay.com?bid_auction=12345678",
    false);
  xmlhttp.send(null);
</script>
```

Suchen

Pseudo-Code!

Cross-Site Scripting: Abhilfen?

- Cross-Site Scripting ist eine Form von HTML-Injection
- Gleiche Vorgehensweise wie bei jeder Injektions-Schwachstelle, hier:
 - Escaping von Daten (Separieren)
 - Sperren des Escaping-Zeichens
- Aufpassen! Wie landen die Daten in der Datenbank?
 - Siehe dazu „Persistentes XSS“

Cross-site Scripting im Bild

- Internet Explorer führt(e) Code in Bildern aus
 - www.heise.de/newsticker/Internet-Explorer-fuehrt-Code-in-Bildern-aus--/meldung/132289
- Daher:
 - Auch alle Dokumente säubern bzw. konvertieren
- Auch Binärdaten sind Eingaben
 - Nur schwerer zu validieren

Unvermögen, Befehlsstruktur zu wahren

- Failure to Preserve OS Command Structure
- OS Command Injection
- Beispiel:
 - Einfacher Ping-Service im Netz:
 - Eingabemaske für Rechnername
 - Unsere Eingabe: `localhost; rm -rf /`
 - Der Server führt aus: `ping localhost; rm -rf /`

OS Command Injection: Abhilfen?

- Wie bei allen Injection-Attacken
 - Genaue Eingaben-Validierung
 - Whitelists für gültige Eingaben
 - Keine Blacklists
- Größte Sicherheitslücke:
 - `eval(String)`
 - Code-Interpretation zur Laufzeit
 - Sollte in `evil(String)` umbenannt werden

Klartext-Übertragung heikler Informationen

- Cleartext Transmission of Sensitive Information
- Passwörter oder z.B. Kreditkartennummern werden direkt in eine Verbindung geschrieben
- Problem:
 - Zwischenstellen können Inhalt mitlesen
 - Verschleierung erhöht Sicherheit nicht
- Sniffer: z.B. wireshark (früher ethereal)



Seitenübergreifende Aufruf-Manipulation

- Cross-Site Request Forgery (CSRF)
- Mißbraucht das Vertrauen der Webseite in den Klienten
 - Verletzte Annahme: „Alle Requests des Klienten sind gewollt“
- Auch als „manuelle Version“ beliebt
 - Email-Links („Klick mal hier!“)
 - URL-Spoofing (Verschleiern effektiver URL)
 - Gewolltes URL-Hiding (TinyURL, etc.)

Cross-Site Request Forgery (CSRF)



Neue Auktion

Titel

Text

Startgebot

<http://www.okay.com/auction?id=12345>

Schrottkarre

Dieser Unfallwagen ist wirklich schrottreif. Er hat noch nicht mal Luft in den Reifen. Sieh selbst:



Aktuelles Gebot: 100.000 EUR

<http://www.okay.com/placeBid?id=12345>

Cross-Site Request Forgery (CSRF)



Neue Auktion

Titel

Text

Startgebot

<http://www.okay.com/auction?id=98765>

Scheidungs-Porsche ab 1 EUR

Der bananengelbe Porsche 911 meines Ex-Mannes steht noch in der Garage und ich habe die Ersatzschlüssel.

Mache gleich Fotos, ihr dürft schonmal bieten!

Aktuelles Gebot: 1 EUR

Cross-Site Request Forgery (CSRF)



Auktion bearbeiten

Titel

Text

Startgebot

<http://www.okay.com/auction?id=98765>

Scheidungs-Porsche ab 1 EUR

Der bananengelbe Porsche 911 meines Ex-Mannes steht noch in der Garage und ich habe die Ersatzschlüssel. Hier ist das Foto:



Aktuelles Gebot: 1 EUR

Cross-Site Request Forgery (CSRF)

- Benutzer ist angemeldet
- Benutzer surft auf der Seite
- Request kommt vom Benutzer

Schrottkarre

Dieser Unfallwagen ist wirklich sehr
noch nicht mal Luft in den Reifen. S



Aktuelles Gebot: 100.000 EUR

www.okay.com/auction?id=98765

gs-Porsche ab 1 EUR

engelbe Porsche 911 meines Ex-
Mannes steht noch in der Garage und ich habe
die Ersatzschlüssel. Hier ist das Foto:



<http://www.okay.com/placeBid?id=12345>

Aktuelles Gebot: 1 EUR

Jetzt bieten!

Jetzt bieten!

- Gültige Aktion
- Hier: (teures) Gebot

Schutz vor CSRF: Nur im Verbund möglich

- Entwickler:
 - Kennt die Mechanik und Abhilfen
 - www.secure-abap.de/wiki/Movies
- Webseite:
 - Frei von XSS-Lücken
 - Jede Response erhält Single Use Shared Secret (Page Token)
 - Serverseitig in Session gespeichert
- Klient:
 - Frei von Schadsoftware (v.a. Trojaner)
 - Aufpassen beim Anklicken von Links

Wettlaufsituation

- Race Condition
- Parallele Abläufe, von denen einer unter fremder Kontrolle steht, sind ungenügend synchronisiert
- Granularität der Abläufe beliebig:
 - Threads
 - Prozesse
 - Systeme (z.B. Netzwerk-Multiplayer-Spiele)

Information-Preisgabe durch Fehlermeldung

- Error Message Information Leak
- Fehlermeldungen in Produktiv-Systemen enthalten heikle Informationen
 - Vollständige Installationspfade
 - Interne Identifizierer
 - Zusätzliche Daten des Datensatzes
 - z.B. durch Verwenden eines Dumpers
- Abhilfe:
 - Daten nur in die Logdatei

Information-Preisgabe durch Fehlermeldung

HTTP Status 500 -

type Exception report

message

description The server encountered an internal error () that prevented it from fulfilling this request.

exception

```
javax.servlet.ServletException: java.io.IOException
    org.apache.struts.chain.ComposableRequestProcessor.process (ComposableRequestProcessor.java:286)
    org.apache.struts.action.ActionServlet.process (ActionServlet.java:1913)
    org.apache.struts.action.ActionServlet.doGet (ActionServlet.java:449)
    javax.servlet.http.HttpServlet.service (HttpServlet.java:617)
    javax.servlet.http.HttpServlet.service (HttpServlet.java:717)
```

root cause

```
java.io.IOException
    com.mkyong.common.action.UserAction.execute (UserAction.java:34)
    org.apache.struts.chain.commands.servlet.ExecuteAction.execute (ExecuteAction.java:58)
    org.apache.struts.chain.commands.AbstractExecuteAction.execute (AbstractExecuteAction.java:67)
    org.apache.struts.chain.commands.ActionCommandBase.execute (ActionCommandBase.java:51)
    org.apache.commons.chain.impl.ChainBase.execute (ChainBase.java:191)
    org.apache.commons.chain.generic.LookupCommand.execute (LookupCommand.java:305)
    org.apache.commons.chain.impl.ChainBase.execute (ChainBase.java:191)
    org.apache.struts.chain.ComposableRequestProcessor.process (ComposableRequestProcessor.java:283)
    org.apache.struts.action.ActionServlet.process (ActionServlet.java:1913)
    org.apache.struts.action.ActionServlet.doGet (ActionServlet.java:449)
    javax.servlet.http.HttpServlet.service (HttpServlet.java:617)
    javax.servlet.http.HttpServlet.service (HttpServlet.java:717)
```

note The full stack trace of the root cause is available in the Apache Tomcat/6.0.26 logs.

Apache Tomcat/6.0.26



Riskante Verwendung von Ressourcen

Schwachstellen in Bezug auf die unsaubere Verwendung von wichtigen System-Ressourcen

- Erzeugung
- Verwendung
- Übertragung
- Freigabe

Feels like a Segmentation Fault



Verlassen des reservierten Speicherbereichs

- Failure to Constrain Operations within the Bounds of a Memory Buffer
- Zahlreiche Untertypen
 - Buffer Overflow
 - Buffer Underflow
- Klassischer Fehler in C-Programmen
- Bei moderneren Sprachen subtiler
 - Teilweise indirekte Buffer Xflows

Bedeutung des Buffer Overflows für IT

- Häufigstes Werkzeug für Code Injection
 - Vergleiche mit XSS bei Webanwendungen
- Einfalltor für Würmer
 - Siehe u.a. den Sasser Wurm
- Ein einziger Buffer Overflow und das System ist vollständig unter fremder Kontrolle
- Wissen um die Angriffsvektoren ist erster Schritt zur Absicherung

Typischer Buffer Overflow in C

```
int main(void) {  
    char buffer[8];  
    int isAdmin = 0;  
  
    printf("Enter the password: \n");  
    gets(buffer);  
  
    if (strcmp(buffer, "password")) {  
        printf("Wrong Password \n");  
    } else {  
        printf("Correct Password \n");  
        isAdmin = 1;  
    }  
  
    if (isAdmin) {  
        // Now give admin rights to user  
        printf("Admin rights granted \n");  
    }  
    return 0;  
}
```

Enter the password:

password

Correct Password
Admin rights granted

Enter the password:

dontknow

Wrong Password

Enter the password:

ABCDEFGHI

Wrong Password
Admin rights granted

|----- buffer -----||-- isAdmin --|

65	66	67	68	69	70	71	72	73	0	0	0
----	----	----	----	----	----	----	----	----	---	---	---



Overflow

Typische unsichere Funktionen in C

Unsichere Funktion

`gets()` - read characters

`strcpy()` - copy content of the buffer

`strcat()` - buffer concatenation

`sprintf()` - fill buffer with data of different types

`scanf/fscanf()` - read from STDIN

`getwd()` - return working directory

`realpath()` - return absolute (full) path

`syslog()` - write to system log

`setproctitle()` - change the process title

Sichere Variante

`fgets()`

`strncpy()`

`strncat()`

`snprintf()`

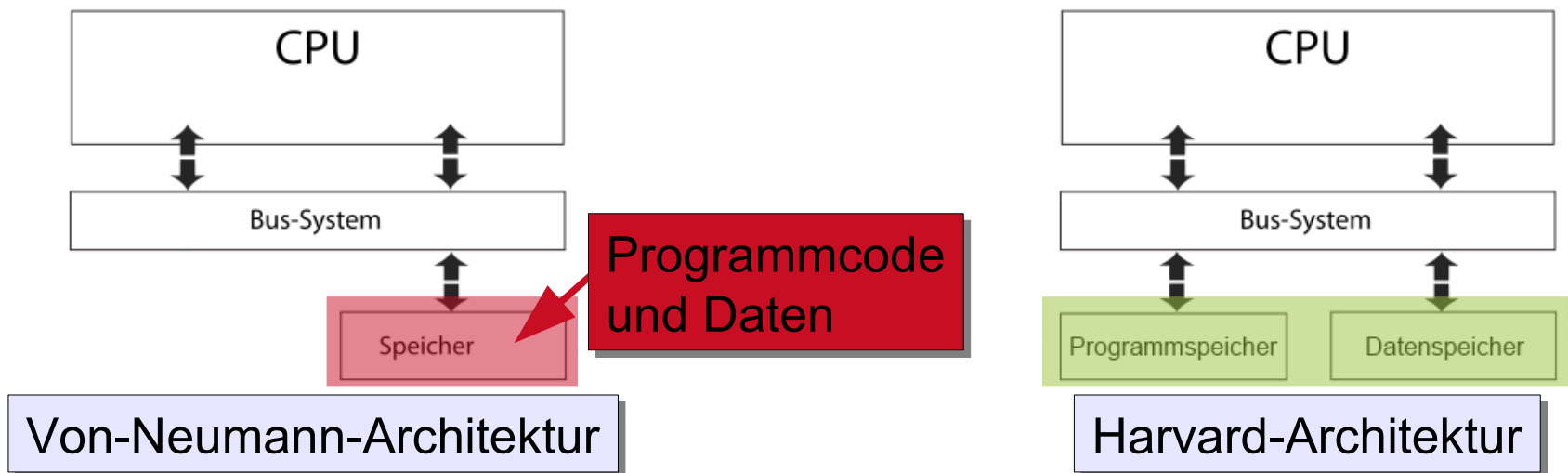
Informationen zu Buffer Overflows

- Funktionierende Code Injection (Beispiel)
 - www-rn.informatik.uni-bremen.de/lehre/itsec/itsec05-2u.pdf
- Ausführliche Beschreibung
 - Mit Anleitung für Review und Vermeidung
 - www.owasp.org/index.php/Buffer_Overflow
- Erstaunlich gut: Englische Wikipedia
 - Sogar mit Hinweisen zur NOP-Rutsche
 - en.wikipedia.org/wiki/Buffer_overflow



Analyse des Buffer Overflow

- Dient der Code Injection
- Problem: Code und Daten sind im gleichen Alphabet gespeichert (Bytes)
- Abhilfe: Trennen der Alphabete in Hardware



Der kleine Bruder des Buffer Overflows

- Buffer Overrun (auch Buffer Over-read)
 - Auslesen über Puffergrenzen hinweg
- Schönes Beispiel: Heartbleed-Bug
- Fehler in OpenSSL
 - Präparierte Heartbeat-Pakete können beliebigen Speicher des Servers auslesen
 - ohne Authentifizierung
- Kompromittiert alle SSH-Keys, Zertifikate, ...



heartbleed.com

Veröffentlicht
am 07.04.2014

Besteht seit
14.03.2012

Externe Kontrolle über heikle Zustandsdaten

- External Control of Critical State Data
- Beispiele:
 - Anwendungszustand in URL-Parametern
 - Authentifizierungsdaten in Cookie
 - Abhängigkeit von Systemvariablen
- Extremes Beispiel:
 - PHP `register_globals`

Externe Kontrolle bei dynamischen Laden

- Moderne Sprachen haben das „Feature“, Code von einer URL nachzuladen
 - PHP: import
 - Java: Classloader
- Den aktuellen Inhalt der URL hat man meistens nicht unter Kontrolle
- Perfektes Einfalltor für Trojaner
 - Die Anwendung selbst kommt aus vertrauenswürdiger Quelle

Externe Kontrolle über Dateiname oder -pfad

- External Control of File Name or Path
- Beispiele:
 - Benutzer darf Dateinamen angeben
 - Hochladen von Bildern oder Dokumenten
- Szenarien:
 - Ausbruch aus Anwendungsverzeichnis („../“)
 - Ungültiger Dateiname (z.B. Sonderzeichen)
- Äquivalent:
 - Externe Kontrolle über URL

Ungesicherte Arbeitsumgebung

- Untrusted Search Path
- Die Umgebungsvariable PATH enthält Verzeichnisse, deren Dateien direkt zugreifbar sind
 - Viele Skripte verlassen sich darauf, dass PATH korrekt gesetzt ist
- Ist PATH von Dritten änderbar, können beliebige Programme untergeschoben werden

Ein Problem auch für Java

- Beim Start eines Java-Programms bestimmt der CLASSPATH sowohl Reihenfolge als auch Umfang des Ladens
- Abhilfe schafft beispielsweise die Verwendung von signierten JARs
 - JAR = Java Archive, dynamische Bibliothek

Fehlende Kontrolle über Code-Generierung

- Failure to Control Generation of Code
- Code Injection
- Tritt immer auf, wenn Teile des Codes zur Ladezeit noch unbekannt sind
- Beispiele:
 - `eval(String)`
 - `include(String)`
- Abhilfe: z.B. Monolithische Anwendung

Code ohne Integritätsprüfung herunterladen

- Download of Code Without Integrity Check
- Beispiel: Automatischer Updater
- Auch sicherste Webserver können „gehackt“ werden
 - DNS-Spoofing
 - Man-in-the-Middle Attack
- Abhilfe: Signierung, Hash, Checksumme

Ungenügende Freigabe einer Ressource

- Improper Resource Shutdown or Release
- Beispiele:
 - Fehlende Freigabe eines Datei-Handles
 - Fehlendes Schließen eines Streams
 - Fehlendes Schließen eines Socket
- Gefahr:
 - Denial-of-Service-Attacke mittels der Ressource (z.B. Socket)

Ungenügende Initialisierung

- Improper Initialization
- Variablen erhalten in (fast) jeder Sprache einen Default-Wert
- Wenn eine Initialisierung z.B. aufgrund eines Fehlers unterbleibt, wird der Default-Wert belassen, aber verwendet
- Siehe auch `register_globals` in PHP

Falsche Berechnungen

- Incorrect Calculation
- Beispiele:
 - Anwender kann Werte direkt bestimmen
 - Werte kommen aus Fremd-Systemen
 - Werte werden aus unsicheren Quellen geladen
- Typischer Fall: Division durch 0
- Eine beliebte Schwachstelle für DOS-Attacken (Denial-of-Service)



Schwache Sicherheitsmaßnahmen

Schwachstellen in Bezug auf Sicherheitsmaßnahmen

- Falsch verwendet
- Mißbraucht
- Weggelassen
- Ignoriert

Der eigenerfundene Zufallsgenerator

```
int getRandomNumber()  
{  
    return 4; // chosen by fair dice roll.  
              // guaranteed to be random.  
}
```

Unvollständiger Zugriffsschutz

- Improper Access Control (Authorization)
- Nicht alle Ausführungspfade werden mit Zugriffskontrollen gesichert
 - Angreifer könnte Fehler verursachen und dann z.B. mittels „Ignorieren“ höhere Rechte erlangen
- Web-Anwendungen: Direkte URL-Eingabe darf die Prüfung nicht umgehen

Verwenden einer veralteten Verschlüsselung

- Use of a Broken or Risky Cryptographic Algorithm
- Eigene Verschlüsselungen taugen nichts
- Viele Verschlüsselungen sind veraltet
 - Geknackt (z.B. mathematische Lücke)
 - Zu schwach (Moore's Gesetz!)
- MD5- und SHA-1-Hashes sind unsicher
 - „Regenbogentabellen“ machen Brute Force unnötig

Festverdrahtetes Passwort

- Hard-Coded Password
- Auch: Default-Passwort
- Erstaunlich weit verbreitet
- Beispiele:
 - BIOS-Passwörter
 - (WLAN-)Router, Drucker, Handies
- Hartverdrahtete Passwörter werden immer gefunden
 - Content Scramble System (CSS) bei DVDs

Heikle Ressource mit unsicheren Rechten

- Insecure Permission Assignment for Critical Resource
- Beispiel:
 - Konfigurationsdatei mit Lese-/Schreib-Recht für Allgemeinheit
 - Lese-Recht für private (Zertifikat-)Schlüssel
 - Passwort als Parameter im Prozess-Aufruf
- Viele Systeme verlangen kurze Phasen der Unsicherheit während Installation

Ungenügend zufällige Zufallswerte

- Use of Insufficiently Random Values
- Beispiele:
 - UID (unique ID) im Bereich von 1E6-9
 - Verwendung von „nicht-initialisiertem“ Speicher als Zufallsquelle
 - Siehe auch Debian OpenSSH Disaster
 - Eigenentwickelter Zufallszahlengenerator
- Computer sind keine Zufallsquellen
 - Einsatz von Zufall ist eine Kunst

Ausführung mit unnötig hohen Rechten

- Execution with Unnecessary Privileges
- Beispiele:
 - Webanwendung läuft als root-Benutzer
 - z.B. Apache (und damit CGI-/PHP-Skripte)
 - Werkzeug-Benutzeraccounts mit Shell-Login
- Injizierter Code läuft mindestens mit den Rechten der Anwendung
- Minimierung der Rechte minimiert Angriffsfläche

Nur klientenseitige Sicherheitsüberprüfung

- Client-Side Enforcement of Server-Side Security
- Beispiele:
 - Netzwerk-Code fast aller (frühen) Spiele
 - Teilweise moderne AJAX-/RIA-Anwendungen
- Alle Eingaben und Informationen vom Klienten sind unsicher
- Sicherheits-Prüfung zur Not duplizieren

Der größte Bug: ISO/OSI Schicht 8

USER FRIENDLY by Illiad

