

Sensitive data exposure



Google Hacking

Google hacking

- Not an attack as such, but the preliminaries: searching for vulnerable systems or vulnerabilities on a site
 - Using a search engine to look for known weaknesses
- Examples:
 - Looking for version numbers (vulnerable versions of software are known; websites running them will be prime subjects!)
 - Looking for "weak" code → "Google Code Search"
 - Search program comments indicating problems
 - Like: `/* TODO: Fix security problems */`
- Note: the subject of the attack has no chance at all of noticing this, as his server is not touched in any way!
 - Attacks come "out of the blue"
 - But not unprepared: only pages existing for a "long" time (typical indexing time: 2-3 weeks!) can be found
 - Usually the vulnerability is older too

Google hacking

■ Requires advanced Google operators:

- ☐ site: search within a single site only (domain name)
- ☐ link: search within hyperlinks
 - With certain words hinting at interesting pages
- ☐ cache: displays the page as it was indexed by Google
 - Turn off image loading and you will not be logged on the server!
- ☐ intitle: within the title tag
 - Directory listings: intitle:index.of
 - Better: intitle:index.of "parent directory"; intitle:index.of name size
- ☐ inurl: within the URL of the web page
 - Webcams: inurl:"ViewerFrame?Mode=" inurl:"/axis-cgi/jpg/image.cgi?"
- ☐ filetype: only files of a specific type (no colon → filetype:doc)
 - MS SQL server error: "A syntax error has occurred" filetype:ihtml

■ Note: such operators exist for most search engines

- ☐ This is **not** a Google-specific problem!

Google hacking

■ Search modifiers:

- ☐ “+”: This term must be present
 - Most commonly used; explicitly require presence
- ☐ “-”: This term is excluded
 - To remove unwanted results (similar but not relevant)
- ☐ “”: Quotes for phrases
 - Not necessarily literal matching!
- ☐ “.”: Single character wildcard
- ☐ “*”: Any word wildcard

■ Examples:

- ☐ `site:microsoft.com -site:www.microsoft.com`
 - Search for all “uncommon” subsites/additional servers

General targets

- Looking for specific vulnerabilities
 - ☐ Version numbers, strings, URLs...
- Error messages with too much information
 - ☐ Before site “lockdown”, which logs the errors and shows a simple and constant message to the user (instead of directly the error)
- Files containing passwords (or their hashes)
 - ☐ For offline breaking
- Logon pages
 - ☐ Where to actually attack
 - ☐ Title/content may give away information about limitations to passwords, method of storage, security precautions...
- Vulnerability information
 - ☐ All kinds of logs (web servers, firewalls...)
 - ☐ May also contain information about the internal network

Examples

- Searching for password lists (examples are **extremely** old!):
 - ☐ inurl:/_vti_pvt/users.pwd
 - ☐ inurl:/_vti_pvt/administrators.pwd
 - ☐ inurl:/_vti_pvt/service.pwd
 - ☐ Still requires to break passwords, but this can be done offline!
- HP JetDirect: printers with an included web server
 - ☐ inurl:hp/device/this.LCDDispatcher
 - Note: these web pages typically cannot be changed at all (firmware update only)!
 - Only access can (and should!) be impossible from the Internet
 - ☐ Searching by title (model numbers) or strings (handbook, questions...) would not be successful here!
- Login portals of routers
 - ☐ intitle:"Cisco Systems, Inc. VPN 3000 Concentrator"
 - ☐ Only shows where to attack; passwords must still be guessed!
 - But try passwords of producer; often the same for all appliances

Examples



HP LaserJet P3005

HP LaserJet P3005-Drucker

Informationen

Gerätestatus

Konfigurationsseite

Verbrauchsmaterialstatus

Ereignisprotokoll

Verbrauchsseite

Geräteinformationen

Bedienfeld

Drucken

Abbildung des Bedienfelds

Dies ist ein inaktives Bild des Bedienfelds des Geräts. Um das Bild zu aktualisieren, klicken Sie auf **Aktualisieren**.

在 3 號紙匣裝入紙張
普通紙 A4

Aktualisieren

Andere Verknüpfungen

[hp instant support](#)

[Verbrauchsmaterial bestellen](#)

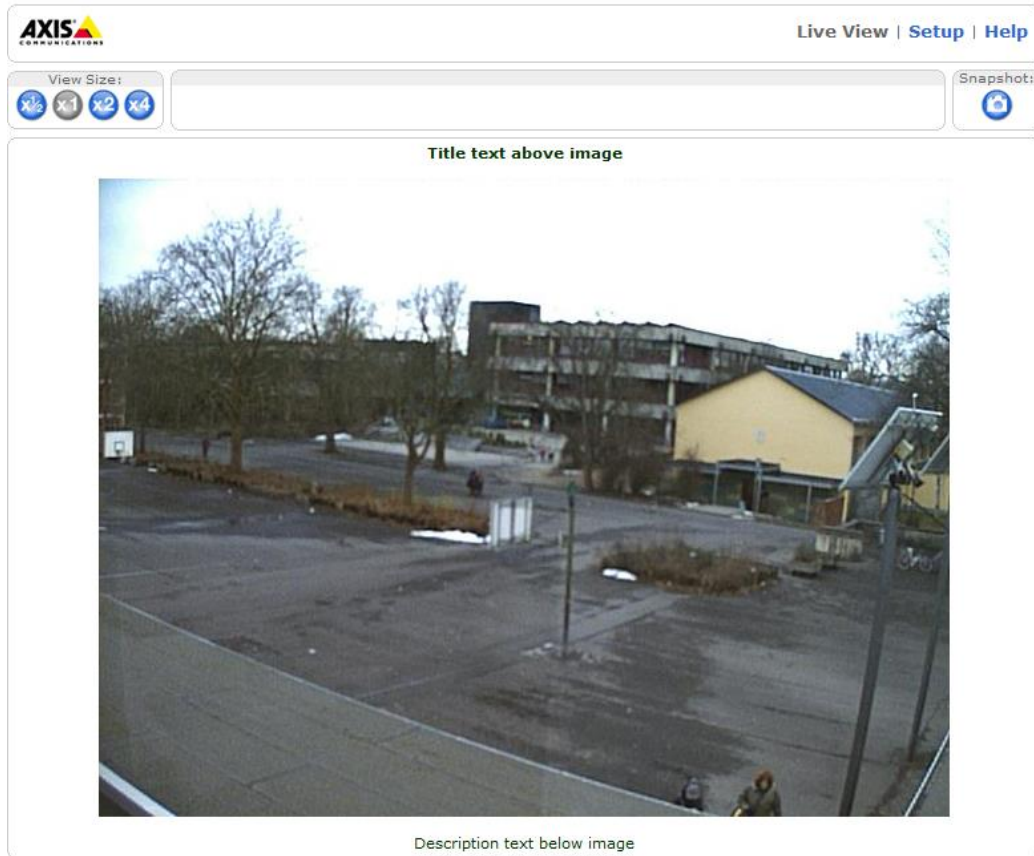
[Produktunterstützung](#)

[Tipp](#)

Examples




- VNC viewers (Java client: Port 5800; server: Port 5900):
 - intitle:VNC inurl:5800
 - Depending on page title the version/product can be distinguished
- Webcams (Axis):
 - intitle:"Live View / - AXIS"
 - Title can be used for further restriction, e.g. the model used
- Server version:
 - intitle:index.of server.at
 - Example result at bottom of page: “Apache/2.2.9 (Debian) mod_ssl/2.2.9 OpenSSL/0.9.8g Server at www.????? Port 80”
 - mod_ssl/OpenSSL version might also be **very** interesting!
 - Also the default test pages (after installation) often remain accessible even after installing the final website
 - intitle:welcome.to intitle:internet IIS
- Looking for known-vulnerable cgi files
 - inurl:/random_banner/index.cgi

Examples



■ Geschwister-Scholl Gesamtschule Göttingen

See for more open cameras:
<http://www.insecam.org/>

	v6.0.9-beta/	2007-02-07 23:17	-
	v6.0.9/	2007-02-07 23:17	-
	KEYS	2013-02-12 12:57	29K

intitle:welcome.to intitle:internet IIS

The screenshot shows the 'Welcome to IIS 5.1' window in Windows XP Professional. The window title is 'Microsoft Windows XP Professional'. The main content area is titled 'Welcome to IIS 5.1' and contains several sections of text and links. Annotations with arrows point to specific parts of the window:

- OS version**: Points to the 'Windows XP' logo.
- Default pages**: Points to the text 'You do not currently have a default Web page established for your users. Any users attempting to connect to your Web site from another machine are currently receiving an Under Construction page. Your Web server lists the following files as possible default Web pages: default.htm, default.asp, index.htm, iisstart.asp. Currently, only iisstart.asp exists.'
- Local path**: Points to the text 'To add documents to your default Web site, save files in c:\inetpub\wwwroot\.'
- IIS version**: Points to the 'Welcome to IIS 5.1' section header.

The 'Welcome to IIS 5.1' section includes the following text:

Internet Information Services (IIS) 5.1 for Microsoft Windows XP Professional brings the power of Web computing to Windows. With IIS, you can easily share files and printers, or you can create applications to securely publish information on the Web to improve the way your organization shares information. IIS is a secure platform for building and deploying e-commerce solutions and mission-critical applications to the Web.

Using Windows XP Professional with IIS installed, provides a personal and development operating system that allows you to:

- Set up a personal Web server
- Share information within your team
- Access databases
- Develop an enterprise intranet
- Develop applications for the Web.

IIS integrates proven Internet standards with Windows, so that using the Web does not mean having to start over and learn new ways to publish, manage,

The 'Integrated Management' section includes the following text:

You can manage IIS through the Windows XP Computer Management console or by using scripting. Using the console, you can also share the contents of your sites and servers that are managed with Internet Information Services to other people via the Web. Accessing the IIS snap-in from the console, you can configure the most common IIS settings and properties. After site and application development, these settings and properties can be used in a production environment running more powerful versions of Windows servers.

The 'Online Documentation' section includes the following text:

The IIS online documentation includes an index, full-text search, and the ability to print by node or individual topic. For programmatic administration and script development, use the samples installed with IIS. Help files are stored as HTML, which allows you to annotate and share them as needed. Using the IIS online documentation, you can:

- Get help with tasks
- Learn about server operation and management
- Consult reference material
- View code samples.

The window has a taskbar at the bottom with the 'Fertig' button and several icons.

Apache test page

Testing 123..

This page is used to test the proper operation of the [Apache HTTP server](#) after it has been installed. If you can read this page it means that this site is working properly. This server is powered by [CentOS](#).

OS

Just visiting?

The website you just visited is either experiencing problems or is undergoing routine maintenance.

If you would like to let the administrators of this website know that you've seen this page instead of the page you expected, you should send them e-mail. In general, mail sent to the name "webmaster" and directed to the website's domain should reach the appropriate person.

For example, if you experienced problems while visiting www.example.com, you should send e-mail to "webmaster@example.com".

Are you the Administrator?

You should add your website content to the directory `/var/www/html/`

To prevent this page from ever being used, follow the instructions in the file `/etc/httpd/conf.d/welcome.conf`.

Local path

Config location

Promoting Apache and CentOS

You are free to use the images below on Apache and CentOS Linux powered HTTP servers. Thanks for using Apache and CentOS!



Important note:

The CentOS Project has nothing to do with this website or its content, it just provides the software that makes the website run.

If you have issues with the content of this site, contact the owner of the domain, not the CentOS project. Unless you intended to visit CentOS.org, the

The CentOS Project

The CentOS Linux distribution is a stable, predictable, manageable and reproduceable platform derived from the sources of Red Hat Enterprise Linux (RHEL).

Additionally to being a popular choice for web hosting, CentOS also provides a rich platform for open source communities to build upon. For more information please visit the [CentOS website](#).

Examples

- MySQL database dumps
 - "# Dumping data for table (username|user|users|password)" - site:mysql.com -cvs
- phpMyAdmin: database administration tools
 - intitle:phpMyAdmin "Welcome to phpMyAdmin ***" "running on * as root@*"
- Registry dumps
 - filetype:reg reg HKEY_CURRENT_USER username
- Looking for code/passwords (often contains cleartext pwds!)
 - filetype:inc intext:mysql_connect
- Printers/Faxes:
 - inurl:webArch/mainFrame.cgi
- UPS:
 - intitle:"ups status page"

Examples

```
--  
-- Table structure for table `users`  
--  
  
CREATE TABLE IF NOT EXISTS `users` (  
  `Uname` varchar(255) CHARACTER SET latin1 NOT NULL,  
  `UID` int(11) NOT NULL AUTO_INCREMENT,  
  `pass` varchar(255) CHARACTER SET utf8 COLLATE utf8_unicode_ci NOT NULL,  
  `lname` varchar(512) CHARACTER SET latin1 NOT NULL DEFAULT 'new',  
  `fname` varchar(512) CHARACTER SET latin1 NOT NULL DEFAULT 'new',  
  `openID` text CHARACTER SET latin1 NOT NULL,  
  `accepted` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  
  `hasAccepted` int(11) DEFAULT '0',  
  `lastActive` timestamp NOT NULL DEFAULT '0000-00-00 00:00:00',  
  PRIMARY KEY (`UID`)  
) ENGINE=MyISAM DEFAULT CHARSET=utf8 COLLATE=utf8_bin AUTO_INCREMENT=265 ;
```

```
--  
-- Dumping data for table `users`  
--
```

```
INSERT INTO `users` (`Uname`, `UID`, `pass`, `lname`, `fname`, `openID`, `accepted`, `hasAccepted`,  
  `lastActive`) VALUES  
( 'admin', 1, '335ded56c9ca54f9fb7aa4cd61455a4bfa0af7c8', 'admin', 'admin', '', '0000-00-00 00:00:00', 0,  
  '2012-05-01 10:21:33');
```

Examples

Project Hosting will be READ-ONLY



[Project Home](#)

[Wiki](#)

Source

[Checkout](#) [Browse](#) [Changes](#)

Source path: [svn/](#) [trunk/](#) [lib/](#) mysql_connect.inc

```
1 <?php
2 $dbc = db_connect ('mysql13.000webhost.com', 'a2801869_db', 'a2801869_dbadmin', 'easy123');
3 ?>
```

APCUPSD UPS Network Monitor								
Wed Feb 27 09:35:08 EST 2013								
System	Model	Status	Battery Chg	Utility	UPS Load	UPS Temp	Batt. Run Time	Data
hector	SMART-UPS 1400	ONLINE NO BATTERY	100.0 %	113.1 VAC	21.8 %	24.3° C	6.0 min.	All data
grinch	SMART-UPS 3000 RM	ONLINE	099.0 %	117.0 VAC	20.8 %	43.2° C	4.0 min.	All data
ghuska	Smart-UPS 2200	ONLINE	100.0 %	114.4 VAC	16.2 %	23.4° C	95.0 min.	All data
nod1	SMART-UPS 3000 RM	ONLINE SLAVE	099.0 %	117.0 VAC	20.8 %	43.2° C	4.0 min.	All data
nod16	Smart-UPS 3000 RM	ONLINE	100.0 %	115.2 VAC	0.0 %	23.4° C	162.0 min.	All data
stat27	Smart-UPS 3000 RM	ONLINE	100.0 %	115.2 VAC	29.9 %	22.9° C	27.0 min.	All data
stat30	Smart-UPS 3000 RM	ONLINE	100.0 %	115.2 VAC	27.9 %	26.5° C	29.0 min.	All data
stat31	Smart-UPS 3000 RM	ONLINE	100.0 %	115.9 VAC	8.4 %	27.0° C	88.0 min.	All data
geo0	SMART-UPS 3000 RM	ONLINE	100.0 %	115.0 VAC	68.6 %	36.0° C	13.0 min.	All data
geo9	SMART-UPS 3000 RM	ONLINE REPLACE BATTERY	100.0 %	114.4 VAC	54.0 %	36.9° C	8.0 min.	All data
geo10	Smart-UPS 3000 RM	ONLINE	100.0 %	114.4 VAC	43.5 %	25.2° C	10.0 min.	All data
geo11	Smart-UPS 3000 RM	ONLINE	100.0 %	114.4 VAC	52.6 %	25.6° C	11.0 min.	All data

Google Cache

- The cache gives you access to old/removed content
 - Which might still be applicable!
- Attention: Surfing the cache will still touch the server
 - E.g. images are loaded from the “source”
 - Preventing this: View the text-only version
 - Add “&strip=1” to the search URL

Google Cache

Dies ist der Cache von Google von <https://ins.jku.at/>. Es handelt sich dabei um ein Abbild der Seite, wie diese am 26. März 2019 05:16:21 GMT angezeigt wurde. Die **aktuelle Seite** sieht mittlerweile eventuell anders aus. [Weitere Informationen.](#)

[Vollständige Version](#) [Nur-Text-Version](#) [Quelle anzeigen](#)

Tipp: Um deinen Suchbegriff schnell auf dieser Seite zu finden, drücke Strg+F bzw. ⌘-F (Mac) und verwende die Suchleiste.



Dies ist der Cache von Google von <https://ins.jku.at/>. Es handelt sich dabei um ein Abbild der Seite, wie diese am 26. März 2019 05:16:21 GMT angezeigt wurde. Die **aktuelle Seite** sieht mittlerweile eventuell anders aus. [Weitere Informationen.](#)

[Vollständige Version](#) [Nur-Text-Version](#) [Quelle anzeigen](#)

Tipp: Um deinen Suchbegriff schnell auf dieser Seite zu finden, drücke Strg+F bzw. ⌘-F (Mac) und verwende die Suchleiste.

[Home](#)

[research](#)

- [Research](#)
- [Teaching](#)
- [News](#)
- [Cooperations](#)
- [About us](#)
- [Contact](#)

[teaching](#)

Who we are:

Our Mission.

The Institute of Networks and Security (INS) is part of the Department of Computer Science in the Faculty of Engineering and Natural Sciences (TNF) at Johannes Kepler University Linz, Austria (...)

[More](#)

Our Vision.

Computer networks and security are fast-moving targets. Research and teaching at the Institute of Networks and Security therefore includes the full range from theoretical to highly practical...

[More](#)

Code of Ethics.

In our research, we are regularly dealing with technologies and techniques that could potentially be misused to cause harm in all sorts of ways. We also demonstrate the application of those...

[More](#)

WHO
ARE:



Welcome to Institute of Networks and Security

No front page content has been created yet.

Recent posts

[Tutor KV Systems Programming](#)

20.09.2018

Bewertung von 5 Programmieraufgaben LVA Systems Programming (Assembler, C) Unterstützungen: • Detaillierte Angabe • Kommentierte Musterlösung • Test-Ein-/Ausgaben • Umfassendes Testing-Framework • Bewertungsschema • Jederzeitiger Kontakt mit LVA-Leitern EUR 932 im Semester pro Gruppe Unter Geringfügigkeitsgrenze, daher steuerfrei € 233 pro Monat...

[More](#)

Google hacking: Prevention

- Ensure “private” computers (e.g. printers) are inaccessible from the “public” internet, e.g. by firewall (packet filter alone might be insuff.)
- Automated tools for Google search: e.g. SiteDigger
 - ☐ Can also be used on your own pages to look for “weaknesses” (verification); seem to not be available anymore!
- Check what Google (and others) know about your site
 - ☐ site:www.mysite.com
 - ☐ Google Webmaster Tools
 - ☐ Is this **only** what **should** be accessible to everyone?
- Use “robots.txt” to limit web crawlers to “relevant” pages
- Captchas/Remove from Google index (→ Desirable?)
 - ☐ Not that easy and/or quick!
 - ☐ Requires often extensive measures (removal of page + notification of Google + wait for reindexing-visit)
 - ☐ Yahoo, Bing, Baidu...?

Google hacking: Legal aspects

- The site is not attacked at all in this stage
 - Just some information is collected
 - The information is gathered from public sources
- Between this action and the attack is a big gap → no “immediate continuation to actually performing the attack”
 - A real and separate decision is necessary
- In contrast to other attacks, this is legal in most countries!
 - Too far away from a concrete attack
 - Only when at least trying the attack on the real server (even if unsuccessful!), this is typically a punishable offence!
 - Note: UK and USA are notable exceptions!
 - “Unauthorized access” (=not allowed, but perfectly normal access to public web pages) may be an offence

Google hacking: Legal aspects

- BUT: Should something happen, this can be used as evidence
 - Also, it is very good evidence to prove intentionality
 - When explicitly looking for weaknesses, you can later hardly claim that you sent a special/problematic request “accidentally”...
 - Note: finding evidence of Google hacking is difficult. It requires access to
 - your computer,
 - log files of intermediaries (like proxies, wiretapping at the ISP...), or
 - the Google server logs

Another search engine: Shodan

- “Search engine for Internet-connected devices”
 - But often used for nefarious (or research!) purposes
 - “Freelancer (\$ 59/month) → Scan up to 5120 IPs per month”
- Not explicitly web-related, but also covers this
 - Tries to connect to ports, saves which are open and what the banner (= immediate response) is
- You can search e.g. for all devices with some required properties
 - listening on a specific port
 - using some software (version); defined through response header
 - running a certain OS
- Login required for more complex queries, e.g. country or network

Shodan – Example (“vnc”)

81.10.28.54 host-81.10.28.54.tedata.net

Country	Egypt
Organization	TE Data
ISP	TE Data
Last Update	2019-03-27T15:24:40.486413
Hostnames	host-81.10.28.54.tedata.net
ASN	AS8452

Ports

445 1434 5357 5800 5900 10243

Services

445
tcp
smb

SMB Status
Authentication: disabled
SMB Version: 2
Capabilities: raw-mode

Share Name	Type	Comments
ADMIN\$	Disk	Remote Admin
C\$	Disk	Default share
D\$	Disk	Default share
E\$	Disk	Default share
HP LaserJet 1022	Printer	HP LaserJet 1022
IPC\$	IPC	Remote IPC
print\$	Disk	Printer Drivers
Users	Disk	

1434
udp
ms-sql-monitor

Microsoft SQL Server Version: 8.00.194
nServerName;HP-PC;InstanceName;MSSQLSERVER;IsClustered;No;Version;8.00.194;tcp;1433;np;\HP-PC\pipe\sql\query;;

5357
tcp
http-simple-new

HTTP/1.1 503 Service Unavailable
Content-Type: text/html; charset=us-ascii
Server: Microsoft-HTTPAPI/2.0
Date: Wed, 20 Mar 2019 02:47:03 GMT
Connection: close
Content-Length: 326

5800
tcp
http-simple-new

HTTP/1.1 200 OK
Server: VNC Server Enterprise Edition/E4.5 (r21561)
Date: Wed, 27 Mar 2019 15:28:21 GMT
Last-Modified: Wed, 27 Mar 2019 15:28:21 GMT
Content-Length: 240
Content-Type: text/html
Connection: close

5900
tcp
http-simple-new

RealVNC Enterprise
RFB 004.001

10243
tcp
http-simple-new

HTTP/1.1 404 Not Found
Content-Type: text/html; charset=us-ascii
Server: Microsoft-HTTPAPI/2.0
Date: Sat, 23 Mar 2019 10:48:15 GMT
Connection: close
Content-Length: 315

VNC password: „Only the first eight characters are significant“

HTML5 specialties

HTML5 – Geolocation

- Geolocation API allows websites to trace users
 - They have to ask the user before, but it is unclear how long this permission is granted, and how “remembering” works
 - See mobile phones: changing to another app does not necessarily “close” the browser completely, so the permission can survive for a long time!
- This potentially allows tracing users across several hours, as long as they remain on the page (and the page is not “frozen” in background)
 - Or on any later visit if they grant this permission permanently
 - Note: there is (currently) no indication that a webpage is doing this apart from asking for permission (if necessary!)
- Also: correlating users on different domains
 - If both are sitting on the very same chair, they are probably the same person...

HTML5 – Local storage

- HTML 5 introduced local storage in databases
 - Interesting e.g. for „offline“ applications, or to store data too large for cookies
- Problem: it is stored on the client, but it is not encrypted
 - So make sure that only such data ends up in there that the user may see (you are sending it to his computer anyway...)
 - Or anybody getting access to the device and the user account
 - Could this be a shared computer? You would not (cannot) know as the server, so be careful!
 - Example: API keys, security tokens, passwords, encryption keys etc
 - Local Storage is potentially a full transactional databases, so injection attacks become a local issue too
 - Encryption: mostly useless if done on the client, as the key is then on the client as well (at least temporarily)
 - Encrypted on server → why store on client?

HTML5 – Local storage

- Local storage **is** under the SOP (Same Origin Policy), so access is (hopefully) impossible from “foreign” websites – but what about XSS?
- But: there exists **no** “HTTP-Only” flag, like for cookies
 - If data is in there, every page from that site can access all the data - and modify it
- Also: there exists no “Secure” flag, like for cookies
 - Data must be encrypted manually: (before being stored) or transmitted somewhere else
 - Access is not limited to pages retrieved via https
 - You can send out the data unencrypted easily & inadvertently
- Make sure to have some integrity check – anybody with access to the filesystem might be able to modify the database outside of the browser
- Can be used as “eternal cookie” to track users

HTML5 – Local storage

■ Recommendations:

- ☐ Do not store session identifiers
 - Closing the browser does not delete the content, so if the user did not logout, the next person opening the browser is still logged in
 - This can be a feature too, so this might be used if the user explicitly asks for it (and is explicitly informed about the danger)
- ☐ Validate and escape user input before storing it
- ☐ Validate and escape data before inserting it into a page
- ☐ XSS: **know** whether data in there has been escaped or not!
- ☐ Do not store anything security-related in there – a single XSS flaw allows full access to the whole content
- ☐ Different applications on one domain should **never** use local storage if the data needs to be separated
 - Access control is based on the domain only, not on the path!
- ☐ All data in local storage is untrusted – it can be modified from outside the browser and anyone with access to the computer

Error messages

Error messages

- Web applications usually report detailed information on errors encountered during their execution
 - This is a significant information leak
 - No vulnerability itself, but allows deducing/exploiting others!
 - Attackers may gain lots of information
 - Disk layout (paths), database layout (tables, queries), stack traces, "File not found" vs. "Access denied"
- Similar to Google hacking:
 - This is not a security problem in itself
 - But it gives away information:
 - What security problems exist
 - How to exploit them, if one is known
 - Information for phishing (e.g. admin name and E-Mail)
- But this information is often indispensable for finding problems (bug-fixing by programmers, but also helplines!)

Examples of leaked information

- Local file/path names: allows predicting where a file would be physically (important for “blind” attacks!), OS...
 - Backups, temporary files, configuration files, unlinked files...
- Server configuration
 - Example: phpinfo() → shows detailed information on what modules are installed, version numbers, paths...
- Environment values: path, security settings, OS...
 - Previously less important, but very often used in containers/cloud!
- Exact time: can be important regarding cryptography
 - General time (minutes) is no problem
 - But avoid seconds precision, if possible (and definitely anything more!)
- (SQL) query structure: table/column names, exploitable query structure, missing quotes, etc.
- Comments left in the public part
 - “<!-- TODO: Fix security issue here -->” → Bad idea!

Really bad examples

- Error page with information in URL
 - <http://www.insecure.com/errors.asp?Error=Login%20failed>
 - Page then displays the string “Login failed” somewhere
 - XSS attack (reflected) becomes possible!
 - You can also provide a script and send the link to someone
- Stack traces: internal program structure
 - Buffer overflows, XSS opportunities, line number information, library information, (version number/path - if source referenced)
- Crossover with social engineering: search support forums!
 - There a lot of (internal!) information might be posted when looking for help, e.g. full stack traces not visible from the outside, IP addresses, parts of program code etc.
 - Make sure to anonymise them properly – including yourself!
 - And be sure to mask/remove all usernames & passwords, IP addresses, E-Mails...

Good error messages

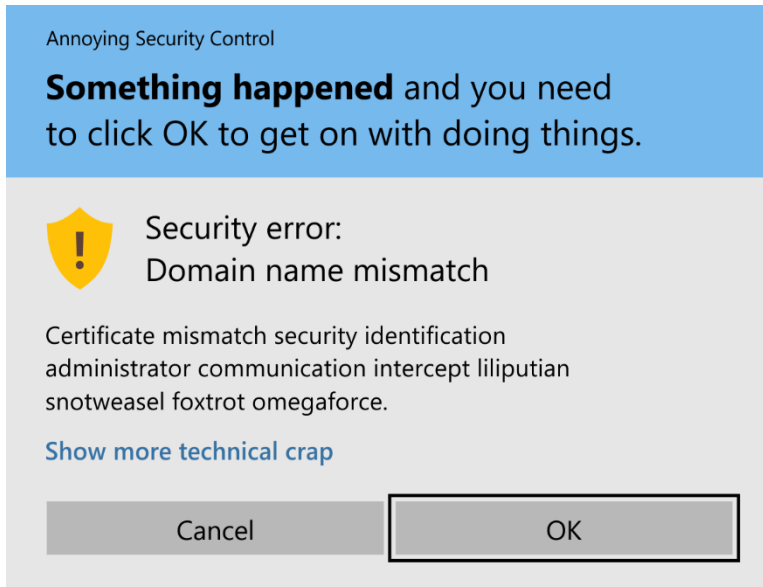
- They should include the following information:
 - ☐ That a problem occurred
 - ☐ Why the problem occurred
 - ☐ How to fix the problem
 - ☐ Security issue: Explain the danger (e.g. with example)
 - I.e. **why** should I fix the problem (or should I not?)
- BUT: in terms of the **user**, not of the **developer**!

Good error messages

■ Therefore:

- No technical internals (why, how)
 - Users won't understand them anyway
 - But give the option to access them (→ good for helplines, debugging)
- Security: better too little information than too much
 - Example: don't tell that the password was wrong, say that "username/password could not be validated"
 - Careful with the technical details, e.g. reference to a log line/date
 - Access the server and investigate the detailed message there – which can be easily found through this information
- Try to do away with the message
 - Program for automatic recovery (or a standard response)
 - Take explicit care of the difficulty, don't depend on a generic error page, unless constructed specifically
 - It might show inappropriate things!
- Safe way out should be the default
 - Otherwise: why is the message shown at all?

Bad error messages...



Obviously intended to be funny,
but a good start for discussion!

■ Why is this problematic?

- ☐ No details on what happened
 - No technical things, but at least in user terms!
- ☐ “Domain name mismatch”
 - What is a domain name?
 - Which A does not match which B?
- ☐ “OK” is the default selection
 - If it is problematic, the “safe” version should be preselected
- ☐ Based on what information would the user decide where to click?
- ☐ No danger information
 - “Why should I **not** click OK?”

Good error handling

- But how to keep the information for the developers?
 - Provide two versions of error message display
 - For debugging → turn all output options on
 - Or use a development environment with auto-break on errors...
 - Show as much information as you need/want
 - For release → turn **all** output options **off**!
 - Make sure to use a framework and a generic solution
 - Individual solutions → some settings will be forgotten
 - Ensure that public versions **always** use the release version
 - E.g. big message on home page “Development version”
 - Automatically shown based on location/domain name/flag/...
 - Use a logging framework
 - Allows centralized logging in various details
- Show an individual page with only the necessary information
 - Pre-created to explain the problem to the user
 - See previous slide!

Good error handling

- As (only very last!) **fallback** return default page: "An error occurred"
 - Detailed information should be logged
 - As extensive as possible, perhaps even creating new log files
 - But beware of DoS attacks through this!
 - An alert should be sent to the admin
 - E.g. by E-Mail (beware of security! → encryption?)
 - This is definitely something important, as it was never considered!
 - The output page/E-Mail may not include **any** "offending" user input or **any** internal data
 - XSS reflection vulnerability, resp. information leak!
 - Should always look exactly the same!
 - Small differences → this is again information disclosure!
 - Password recovery page example: showing "password was sent" or "Username/E-Mail was invalid" allows testing for valid account names or E-Mail addresses
 - Access problem example: "access denied" vs. "file doesn't exist" allows finding presence/absence of files and directory structure

Log injection

- Logging is very important, but also needs security
 - E.g. can an attacker inject false log lines?
 - Creating false traces or evidence against third persons!
 - Add “\r\n” into the offending input and add text looking like a real log line (needs additional information how it should look like)
 - Can an attacker hide dangerous data?
 - Inserting a null-byte to terminate the logging there
 - Note: might be useless in Java, but eventually a C program might do the actual log file writing, and there it **is** effective!
 - Injecting scripts: when the administrator reviews the log through a browser (web-based administration is common!), these scripts will be executed within the browser
 - This occurred e.g. with server management SW (access to BIOS, restarting computer etc)!
 - Automatic log analysis tools can be misused too
 - E.g. causing too many false login attempts results in a ban
 - Repeatedly (try) login with false source IP or with “root” account!

Log handling

■ General rules for good logging

- ☐ What to log? Errors, problems, high-value transactions
 - Everything which might be needed as (technical or legal) evidence
- ☐ How to log? So that the problem can be readily identified
 - But make sure the logging does not produce problems, e.g. log injection or DoS attack through filling the log
- ☐ How to log? In a common format like everything else
 - Allowing easier understanding , but also correlation
- ☐ How to log? With some integrity assurance
 - Checksums, signatures etc.
- ☐ Log inspection? Logging without looking is useless
 - This means at least responding to alerts, but also regular manual inspection is recommended
 - Alert thresholds and escalation procedures should be defined
- ☐ Where to log? Somewhere else, ideally centrally
 - If a system is compromised, its logs should already have been sent to a different place, which needs at least one more successful hack

Error messages: How to handle them

- Provide error handlers
 - Good approach, but typically does not cover all problems
- Use specific exception handlers
 - Allows individually coping with problems
- At the outermost possible place put an all-encompassing default exception handler for everything
 - For whatever slips through → this should catch it!
- Do not put the exception (the text/content/...) into the error page
 - You don't know what's in there (→ XSS!); see previous slide
 - Class, line number etc may be in there (but ...)!
- Use web server plugins for filtering out such information
 - Attention: good, but not perfect!
 - May work for suppressing such pages or filtering out content

Error messages: How to handle them

- Take care of resource exhaustion → Denial of Service
 - Use “finally” clauses if available
- Beware of default pages of web servers
 - They typically show much too many details!
- Ensure that all similar paths return exactly the same error
 - If it is the same, of course
- Make sure that all paths return the result in the same time
 - Or: impose random delays for all paths
 - Except perhaps the successful one
 - More theoretical advice than practically useful...
- Investigate the difference between errors in the code, the framework, and the web server
 - All should be handled - and handled in the same way
 - Add a default error handler for framework and server

Error messages: How to handle them

- Override default error pages
 - Don't return "naked" 404s (page doesn't exist), but a 200 (OK) with normal HTML telling the user that the page doesn't exist
 - Otherwise this can be used for trivial page enumeration/site scraping
- Don't provide **internal** contact information in messages
 - Or any information usable for social engineering, like names
 - Better just redirect to a page with "help desk contact information"

Other information leaks

- Information leakage is also possible through other ways:
 - Old versions located in “temp”, “tmp”, or “backup” subdirectories (permissions often not changed!)
 - Versioning control systems
 - E.g. CVS or SVN create a special subdirectory with versioning data – copying the whole application to the webserver copies this too. And in there is a file which lists all files of the directory!
 - Similar for GIT: .git may contain databases with passwords, hashes, third-party API keys etc
 - Log files, e.g. from FTP clients
 - WS-FTP leaves “WS_FTP.LOG” in all directories where files were uploaded to, which lists all these files
 - Wayback machine/Google cache: old versions, when the security was not so tight, or accessible only through strange/external links

Detecting information leakage

- Fuzzing tools: sending incorrect/arbitrary data
 - Will often produce error messages
 - Automatic search for dangerous elements (input, error codes, stack traces...)
 - Manual review for other information
- Static analysis tools: looking for API uses, which are known to be problematic
 - E.g. `System.err.println(exception.toString());`
- Manual code review and testing
 - Coverage is a problem here
 - Create problems and execute the “important things”
 - Then make sure the appropriate information was logged

Data Exposure during Transport

Insufficient transport layer protection

- Passwords may be secure and stored securely, but they are sent from the client to the server in cleartext
 - Monitoring the network traffic can be very difficult - or not
 - You never know how your clients will access the server: they could be using an unencrypted WLAN, broadcast network, proxies...!
 - If monitoring is possible, modifications might also be an option
 - Injection, man-in-the-middle...
- Typical problem: TLS is used for the login, but not afterwards
 - Result: the password is secure, but the session-ID/-cookie can be stolen easily → Impersonation of this user is possible
- “Big” problem: SSL/TLS may cause performance issues, as it requires more CPU power
 - Special hardware for acceleration, “better” servers ...
 - For sites with many visitors this can be (was) a real problem!

Today this should NOT be an excuse anymore!

Insufficient transport layer protection

■ This applies to the frontend: Client/Browser – Server

☐ But check the backend too!

- Is it a dedicated single cable to the DB server? Or who/how would it be possible to listen in on this traffic? See also cloud systems!

☐ Internal attacks by employees are always possible

- Even if you fully trust them: what about an internal PC infected with malware, acting as a network sniffer?

☐ Unencrypted acceptable: 127.0.0.1/::1, but nothing else

- Note: <IP of this host> **should** be encrypted → it might easily change in the future to another host!

■ Check and secure all connections:

☐ Front end

☐ Back end to database

☐ Connections to web services

☐ Mirroring/Obtaining content from 3rd sites (screen scraping, Ajax...)

- This is a security problem in itself...

Detection

- Use tools to check which algorithms are accepted
 - E.g. openssl s_client -connect www.site.org:443 -ssl2
 - Should fail: SSLv2 is insecure → Use TLS 1.2/1.3!
 - Or check with Qualys SSL Labs: <https://www.ssllabs.com/ssltest/>
- Spider the whole site: check where you are redirected to a TLS version and check whether later on a “downgrade” to HTTP is possible (or check webserver configuration to ensure “HTTPS only”!)
- Check whether the first http://... request is being redirected to https://
- Use checklists
 - https://cheatsheetseries.owasp.org/cheatsheets/Transport_Layer_Protection_Cheat_Sheet.html
 - With links to lists from the BSI:
 - <http://www.it-tuv.com/news/singleview/datum/2010/09/20/sicherheit-von-webapplikationen-unterbewertet/>

Prevention

- Today: all page traffic **must** use TLS
 - Starting from the homepage and all subpages
 - No exceptions
- All resources **should** use TLS
 - Images perhaps not (check!), but other files (e.g. PDFs, videos, documents, **JavaScript, CSS**) do!
 - Note: when requesting images from authenticated pages without TLS, cookies (→ session-ID) are sent too, so special pre-cautions (different domain, HTTPS-only cookies...) are necessary!
 - Mixed content (TLS and normal) on single page may cause browser warnings and is a security problem

Prevention

■ Best solution: use TLS everywhere

- Certificates are less expensive today (+ free ones are available!)
 - Recommendation: use Let's encrypt and set up automatic renewal
- CPUs have special instructions for encryption → small overhead
- All end-user devices & libraries should support **at least** TLS 1.0
 - Example: if a library can only do SSL but not even TLS – you should not use the library at all, as
 - security is completely ignored, or
 - it has been no longer in development for at least 10 years
 - So even if it is “internal-use only” without external access, this is too dangerous!
- Practical recommendation: set TLS 1.1 as minimum (=requires justification why), TLS 1.2 as normal, and TLS 1.3 as future target (might not yet be implem./supported by many libraries/devices)

Prevention

- Session cookies must have the “secure” flag set
 - So they are sent only over encrypted connections
 - Check that the application still works (see above, e.g. images!)
- Accept only strong algorithms (“downgrading attacks”)
 - In old times the “null-cipher” was enabled by default...
 - Also: don’t use RSA 768 Bit (1024 Bit is already “dangerous”)
- The server has an appropriate and valid certificate
 - Authorized issuer, not expired, not revoked
 - Matches all domain names of the site
 - May contain other domain names too, but everything you use should be included
- HTTP requests should be declined, not redirected to HTTPS
 - Common practice, but would allow modifying the unencrypted page and “getting rid” of the redirection → user would probably not notice that he had **not** been redirected **this** time (see HSTS)!

HTTP Protocol

XST: Cross Site Tracing

- HTTP “TRACE” command: intended for debugging
 - Send back the whole request (body + **header**) the webserver received as the response **body**
- How can this be used as an attack?
 - Convert “invisible” data to “visible” one!
 - No access to cookie? Send trace request to server (→includes cookie!) and receive it back as normal text content (→ accessible by JavaScript)
 - Then the “HTTP-Only” flag for cookies is useless
 - More problematically: proxies on the way might answer the trace request, so it never reaches the actual server
 - Even if the server is secure, clients might have problems!
- Requirements:
 - XSS or something else to get JavaScript onto the target page
 - Vulnerable server: the one where cookies will be automatically sent to which we are interested in

XST: Cross Site Tracing

■ Variation of attack:

- ☐ Get user to visit a site under control of an attacker
- ☐ Send JavaScript to client
- ☐ JavaScript sends TRACE request to “victim” server
- ☐ If the user has a valid cookie from the “victim” server, the attacker will get it and can send it back to his own server

■ Prevention:

☐ Server:

- Turn off the “TRACE” command on the webserver
- Prevent XSS so no malicious code gets into your website
 - Will not help against variation of attack above!

☐ Client:

- Use browser → SOP is used
 - No connection by JavaScript to server other than the page came from (will prevent variation of attack)

■ Seems to be a small problem today!

ETag

- ETag: used for efficient caching
 - When sending a resource, an ETag is sent with it as HTTP header
 - Later, this value is sent back to the server as a header to ask whether the resource is still unchanged from that “state”
 - Usually this is a hash of the resource content or the last change time of the file
- But what if we create unique ETags for each visitor (standard: “opaque string”) ?
 - Depending on the value we receive on “check if recent” we can recognize the client again
- From the privacy viewpoint this is identical to cookies
 - Server sends us data, we store it, we send it back to the server...

Privacy issues

- Remember the privacy issues of HSTS?
- The same is possible with **everything** remembered across visits
- Newest idea:
 - ☐ Favicons: Create numerous subsites with their own Favicon
 - ☐ Load them in subframes
 - ☐ Notice where the Favicon is already present and where not
 - Note: Specific timing & coordination necessary!
 - Setting the ID: notify server → request Favicons → get very long caching
 - Checking the ID: notify server → request Favicons → get very short caching (So the one's not set before will not be stored now! Might need re-setting the ID afterwards.)
 - ◆ Alternative: when checking → never send a Favicon, only 404s
 - Problem: separately cached, not the general web-content cache
 - Clearing the cache therefore does not help!

THANK YOU FOR YOUR ATTENTION!



JOHANNES KEPLER
UNIVERSITÄT LINZ



INSTITUTE
OF NETWORKS
AND SECURITY

<http://www.ins.jku.at>

Michael Sonntag

michael.sonntag@ins.jku.at

+43 (732) 2468 - 4137

S3 235 (Science park 3, 2nd floor)

**JOHANNES KEPLER
UNIVERSITÄT LINZ**

Altenberger Straße 69
4040 Linz, Österreich
www.jku.at