

Handout zur Vorlesung
DBSP – Sessions, Cookies, Login-Systeme – Unit 8

Vorlesung



University of Applied Sciences

DBSP

Unit 8

Sessions, Cookies, Formulare und Login-System

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

1



University of Applied Sciences



**Prof. Dr. rer. nat.
Nane Kratzke**

*Praktische Informatik und
betriebliche Informationssysteme*

- Raum: 17-0.10
- Tel.: 0451 300 5549
- Email: kratzke@fh-luebeck.de



@NaneKratzke

Updates der Handouts auch über Twitter #dbsp

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

2

Handout zur Vorlesung
DBSP – Sessions, Cookies, Login-Systeme – Unit 8

Übergreifende Ziele der Lehrveranstaltung



University of Applied Sciences

Client- und Serverseitige Entwicklung

PHP (Serverseitig)

JavaScript (Clientseitig)

„Hosten“ von Apps

Framework Erfahrungen

CMS (Drupal)

WebServices (Google-Maps)

jQuery

Datenbank-Integration

Berücksichtigung von Sicherheitsaspekten

HTML-Injections

SQL-Injections

Session Hijacking

Login-Systeme

Um sich weitere Web-Technologien autodidaktisch erarbeiten zu können.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

3

Units



University of Applied Sciences

Unit 1
Cloud Computing IaaS

Unit 2
CMS Drupal

Unit 3
HTML und CSS

Unit 4 - 7
PHP I - IV

Unit 8
Sessions, Cookies, Formulare und Login-System

Unit 9
JavaScript

Unit 10
Drupal Module Development

Unit 11
Datenmodellierung

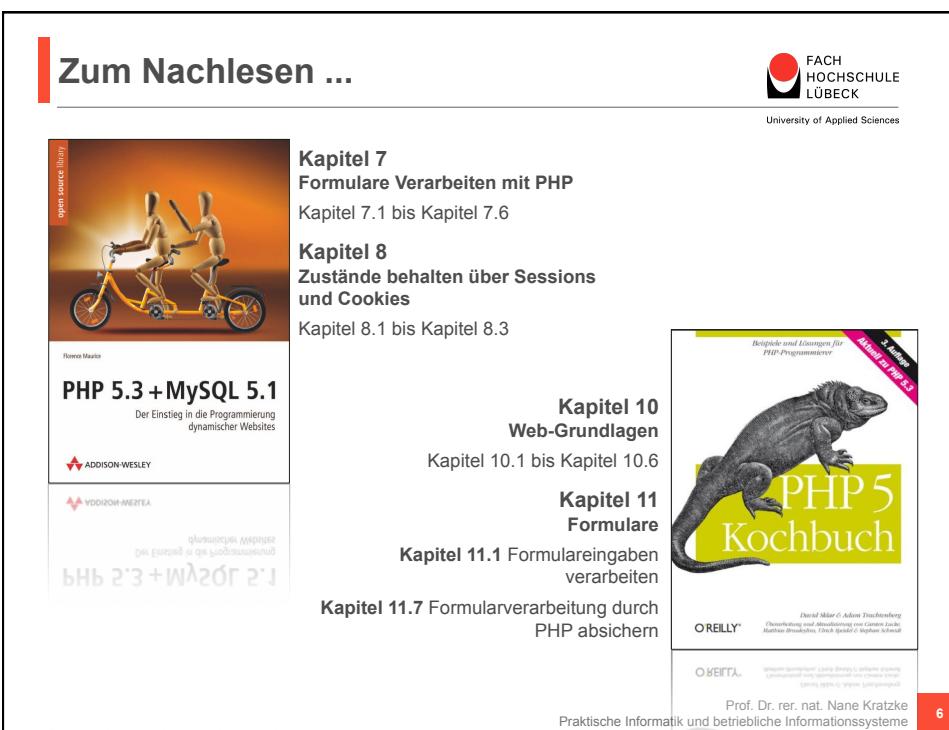
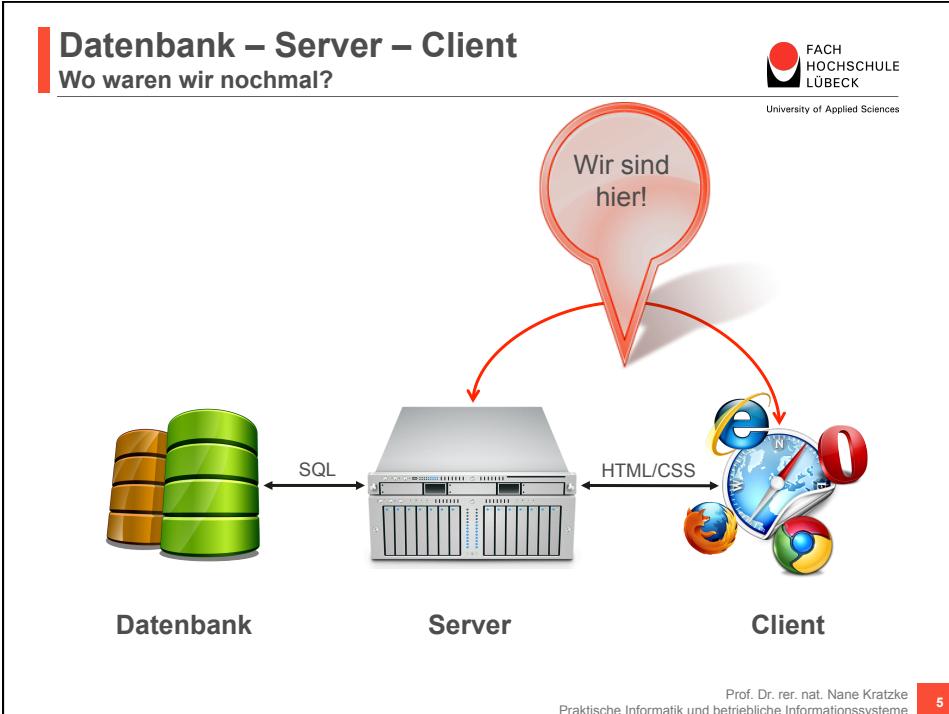
Unit 12 - 13
Datenbanken und SQL
Vom Datenmodell zur Datenbank

Unit 14
Datenbank-gestützte Web-Anwendungen

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

4

Handout zur Vorlesung
DBSP – Sessions, Cookies, Login-Systeme – Unit 8



Zum Nachlesen ...



University of Applied Sciences



Bereitgestellte Skripte:

PHP Programmierung

- **Kapitel 15:** Formularhandling mit PHP
- **Kapitel 16:** Cookies und Sessionhandling mit PHP

<http://praktische-informatik.fh-luebeck.de/node/39>

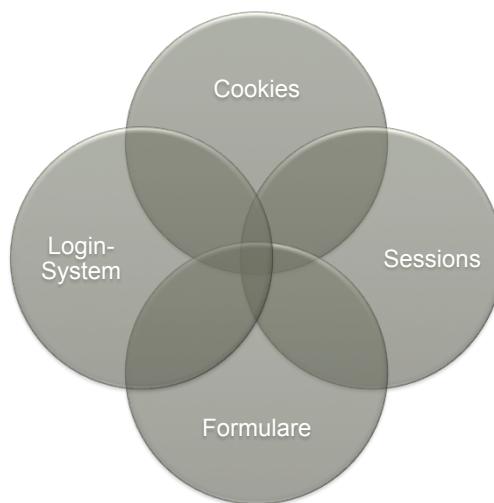
Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

7

Inhalte dieser Unit



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

8

Datenaustausch zwischen Client und Web-Server



University of Applied Sciences

Anfrage über HTTP Protokoll

- z.B. Anfrage einer HTML-Seite



Antwort über HTTP Protokoll

- z.B. Auslieferung einer HTML-Seite

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

9

Informationsaustausch über HTTP ist eigentlich statuslos



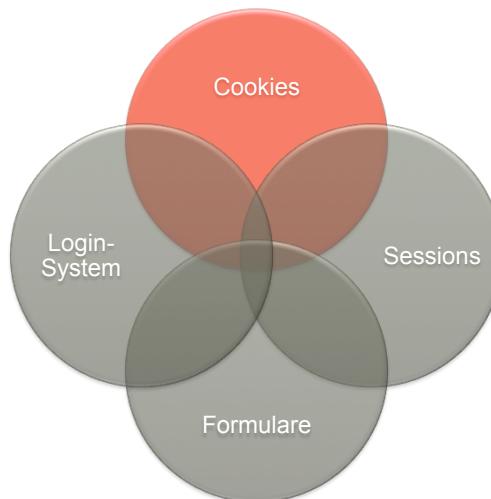
University of Applied Sciences

- HTTP ist grundsätzlich statuslos
- Im Hypertext Transfer Protokoll werden also keine Zustände verwaltet.
- Hieraus ergeben sich bspw. diese Probleme:
 - Wie kann festgestellt werden, dass ein Nutzer sich eingeloggt hat?
 - Wie kann festgestellt werden, welche Anteile eines mehrstufigen Formulars ein Nutzer bereits ausgefüllt hat?
- Es müssen also entweder Server-seitig oder Client-seitig Stati gespeichert und ausgetauscht werden können.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

10

Inhalte dieser Unit



Sessions und Cookies



Client-seitig

- **Cookies**
- Speichern
- von Zuständen
- als Textrepräsentationen
- auf einem Client

Server-seitig

- „Verfolgen“ eines Nutzers mittels **Sessions**
- um zusammengehörige Nutzerinteraktionen in Sitzungen zu verwalten
- Zustände werden auf dem Server gespeichert.

Antwort: Cookies



University of Applied Sciences

- Cookie (Keks)
- Kurzes Stück Text
- Das der Browser auf Veranlassung des Servers mit seinen Anfragen versendet
- Über einen Cookie können also mehrere Anfragen miteinander verknüpft werden (z.B. Login-ID).



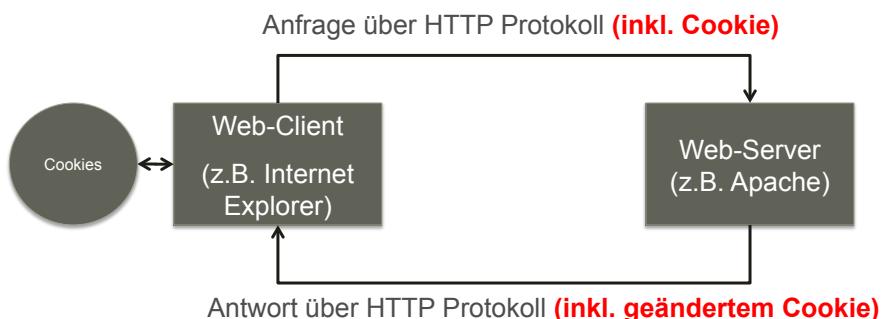
Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

13

Datenaustausch zwischen Client und Web-Server mit Cookies



University of Applied Sciences



Ein **Cookie** (kurzes Stück Text) ist also eine textuelle Repräsentation eines Zustands (z.B. Login-ID, Suchhistorie, Elemente eines Einkaufswagens), der zwischen Client und Server ausgetauscht und durch den Server gesetzt und verändert werden kann.

Es können beliebig viele Cookies genutzt werden, z.B. einer für das Login, einer für die Suchhistorie, einer für den Warenkorb, und viele weiter beliebig definierbare mehr.

Cookies werden Client-seitig **in unverschlüsselten** Textdateien gespeichert.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

14

Setzen und Löschen von Cookies auf dem Client durch den Server



University of Applied Sciences

```
bool setcookie (string $name, string $value,  
                int $expire = 0,  
                string $path, string $domain,  
                bool $secure = false);
```

- Setzt ein Cookie auf einem Client
- Muss aufgerufen werden, bevor irgendwelche Ausgaben generiert werden,
- also bevor ein Header durch den Webserver generiert wurde.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

15

Setzen von Cookies auf dem Client durch den Server



University of Applied Sciences

```
bool setcookie (string $name, string $value,  
                int $expire = 0,  
                string $path, string $domain,  
                bool $secure = false);
```

Parameter	Bedeutung	Bemerkung
\$name	Name des Cookies	Nicht optional
\$value	Der Wert des Cookies (Zeichenkette)	Nicht optional
\$expire	Der Zeitpunkt, an dem das Cookie ungültig wird. (Unix Timestamp) also die Anzahl Sekunden seit Beginn der Epoche	Optional, Bsp. ein Wert von 30 Tagen Gültigkeit: time() + 60 * 60 * 24 * 30
\$path	Der Pfad auf dem Server, für welchen das Cookie verfügbar sein wird.	Optional, Bsp. „/“ Gesamter Server „/foo“ nur im Verz. foo.
\$domain	Die Domain, der das Cookie zur Verfügung steht.	Optional, Bsp: '.example.com'. Alle Subdomains von example.com
\$secure	Bei True nur Transfer über verschlüsseltes HTTPS	Optional

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

16

setcookie Beispiele



University of Applied Sciences

Beispiel: Setzen eines Cookies (und impliziter Gültigkeit bis der Browser geschlossen wird)

```
setcookie("Geschmack", "Chocolate-Chip");
```

Beispiel: Setzen eines Cookies mit 24 Stunden Gültigkeit

```
setcookie("Geschmack", "Chocolate-Chip", time() + 60 * 60 * 24);
```

Beispiel: Setzen eines Cookies (welches nur im Unterverzeichnis foo des Servers zu verarbeiten ist)

```
setcookie("Geschmack", "Chocolate-Chip", "", "/foo");
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

17

Wie löscht man Cookies auf dem Client durch den Server?



University of Applied Sciences

Es gibt keinen deletecookie Befehl

Lösung: Setzen einer Gültigkeit deutlich vor dem aktuellen Zeitstempel)

```
setcookie("Geschmack", "Chocolate-Chip", time() - 60 * 60 * 24);
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

18

Auslesen von an den Server übertragenen Cookies



University of Applied Sciences

- Auf dem Server (und Unterverzeichnissen) vorliegende Cookies werden in PHP in dem globalen Array
- **\$_COOKIE**
- für Applikationslogiken in einer Key Value Struktur bereitgestellt.
- Auf die Werte von kann mittels normalen Array Operationen zugegriffen werden.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

19

Beispiel: Auslesen von Cookies



University of Applied Sciences

Wurde vorher auf dem Client folgendes Cookie hinterlegt.

```
setcookie("Geschmack", "Chocolate-Chip");
```



Erzeugt diese Zeile, welche Ausgabe?

```
echo $_COOKIE['Geschmack'];
```

Chocolate-Chip

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

20

Auslesen und Ausgeben aller Cookies



University of Applied Sciences

Alle an den Server übertragenen Cookies lassen sich einfach mit dieser Schleife ermitteln und auswerten:

```
foreach ($_COOKIE as $key => $value) {  
    echo "$key = $value";  
}
```

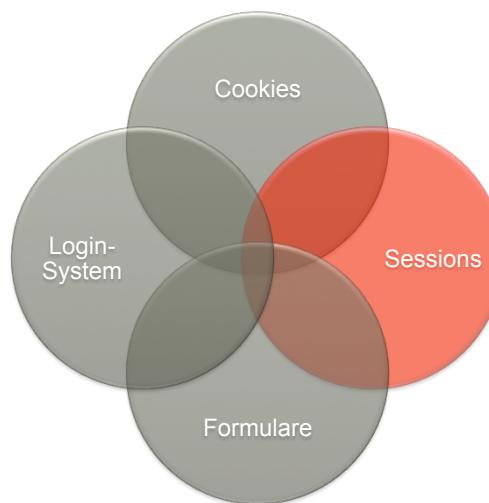
Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

21

Inhalte dieser Unit



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

22

Sessions und Cookies



University of Applied Sciences



Client-seitig

- **Cookies**
- Speichern von Zuständen als Textrepräsentationen auf einem Client

Server-seitig

- „Verfolgen“ eines Nutzers mittels **Sessions**
- um zusammengehörige Nutzerinteraktionen in Sitzungen zu verwalten
- Zustände werden auf dem Server gespeichert.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

23

Sessions



University of Applied Sciences

- Daten werden nicht auf dem Client
- sondern unter einer ID (Session-ID) auf dem Webserver gespeichert.
- Nur die Session-ID wird auf dem Client gespeichert, nicht die damit verknüpften Inhalte.
- Die Session dauert üblicherweise die Dauer einer Browser-Sitzung.

session_start()

Schließen des Browsers

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

24

Setzen und Lesen von Session-Informationen



University of Applied Sciences

- Auf dem Server gesammelte Informationen werden in PHP in dem globalen Array
- **`$_SESSION`**
- für Applikationslogiken in einer Key Value Struktur bereitgestellt.
- Auf die Werte von kann mittels normalen Array Operationen zugegriffen werden.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

25

Beispielhafter Einsatz von Sessions auf einem Web-Server



University of Applied Sciences

Beispiel: Realisierung eines simplen Nutzer-bezogenen Seitenaufrufzählers:

```
start_session();  
$_SESSION['Besuche']++;  
  
echo "Dies ist Ihr $_SESSION['Besuche']ter Besuch.";
```

Beispiel: Auswertung aller Informationen, die während einer Benutzer-Session zu einem Benutzer gesammelt wurden.

```
foreach ($_SESSION as $key => $value) {  
    echo "$key = $value <br/>";  
}
```

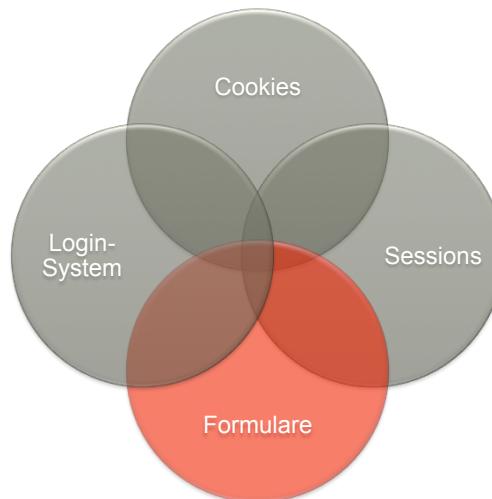
Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

26

Inhalte dieser Unit



University of Applied Sciences



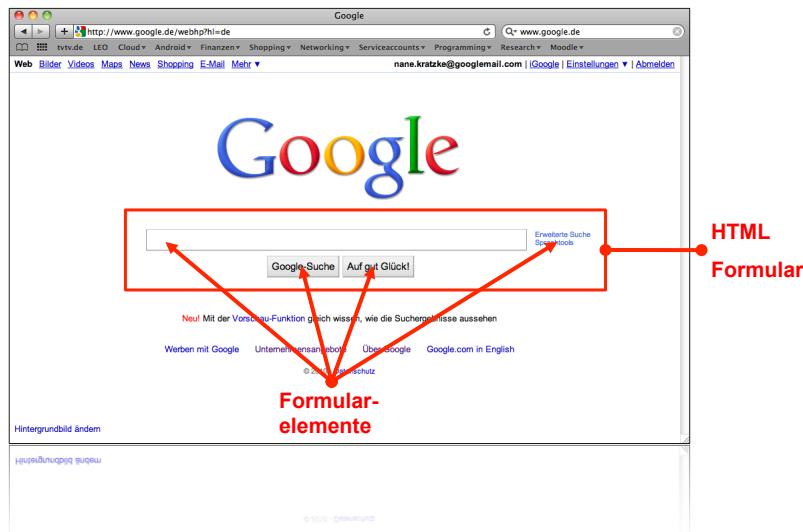
Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

27

Formulare



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

28

Formulare dienen der Nutzerinteraktion



University of Applied Sciences

- Ermöglichen die Interaktion mit dem Nutzer, z.B.
 - Kontaktformulare
 - Suchfeld
 - Formular für Forenbeitrag
- Formulare werden mittels HTML-Tags definiert
- und mittels einer Serverseitigen Programmiersprache ausgewertet und verarbeitet

Formular

—Persönliche Daten

Vorname	Nachname
Straße Hausnr.	
PLZ	Wohnort
Telefonnummer	E-Mail-Adresse

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

29

Gebräuchliche Formularelemente



University of Applied Sciences

Anzeige innerhalb der Seite

* Eingabe erforderlich

Name:	* Ihr Name
E-Mail:	* Ihre E-Mailadresse
Telefon:	Ihre Telefonnummer
Strasse:	Stadt
Wohnort:	Wohnort
Anreise:	<input type="radio"/> mit dem Auto <input type="radio"/> mit der Bahn <input type="radio"/> mit dem Flugzeug
Informationen:	<input type="checkbox"/> Briefkasten <input type="checkbox"/> Kartenmaterial <input type="checkbox"/> Unterkunftsverzeichnis
Nachricht:	* Ise Text <input type="button" value="Senden"/> <input type="button" value="Reset"/>

Notierungen im HTML-Dokument

```
<form action="" method="">
    <input type="text" />
    <input type="radio" />
    <input type="checkbox" />
    <input type="textfield" />
    <input type="button" />
</form>
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

30

Prinzipieller Aufbau eines HTML Formulars am Bsp. eines Logins



University of Applied Sciences

```
<form action="handler.php"
      method="post">
  ID:
  <input type="text"
         name="user_id"/>
  PWD:
  <input type="password"
         name="pwd"/>

  <input type="submit"
         value="Enter"/>

</form>
```

ID: PWD:

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

31

Prinzipieller Aufbau eines HTML Formulars am Bsp. eines Logins



University of Applied Sciences

Skript auf
Web-Server,
welches das
Formular
verarbeiten
soll

```
<form action="handler.php"
      method="post">
  ID:
  <input type="text"
         name="user_id"/>
  PWD:
  <input type="password"
         name="pwd"/>

  <input type="submit"
         value="Enter"/>

</form>
```

Transfer-
methode,
mit denen
die Inhalte
des
Formulars
an den
Server
gesendet
werden.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

32

POST und GET



University of Applied Sciences

Mittels GET

- wird eine URL generiert,
- in dieser URL sind die Formularinhalte codiert.

Mittels POST

- werden die Formularinhalte im Dokumentenkörper transferiert
- und sind damit nicht unmittelbar einsehbar.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

33

URL-Erzeugung mittels GET



University of Applied Sciences

```
<form action="handler.php"
      method="get">
  ID:
  <input type="text"
        name="userid"/>
  PWD:
  <input type="password"
        name="pwd"/>

  <input type="submit"
        value="Enter"/>

</form>
```

`http://localhost/handler.php?userid=Nutzer&pwd=Geheim`

Die so erzeugten URLs sind im Adressfeld eines Webbrowser im Klartext einsehbar. Hinter dem ? folgen die Formularinhalte durch & separiert.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

34

Vorteile und Nachteile von POST und GET



University of Applied Sciences

POST

Vorteile

- Inhalte nicht unmittelbar einsehbar
- Für Passwörter geeignet
- Menge der Daten ist prinzipiell nicht begrenzt

GET

Vorteile

- URLs können gebookmarked werden.
- Formulareingaben lassen sich so speichern.

Nachteile

- Formularinhalte sind einsehbar
- Nicht für Passwörter geeignet
- Sind einfach durch den Nutzer ohne Browser zu ändern.
- Menge der Daten ist begrenzt (maximale Länge einer URL)

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

35

Verarbeitung von Formularinhalten mit PHP (I)



University of Applied Sciences

PHP stellt alle Formulardaten in zwei assoziativen Arrays zur Verfügung

`$_POST` für mittels POST transferierte Daten

`$_GET` für mittels GET transferierte Daten

`$_POST` und `$_GET` sind „superglobal“, d.h. diese Variablen existieren im globalen Kontext, man kann aber auf diese Variablen zugreifen *ohne* sie über `global $_POST` oder `global $_GET` sichtbar machen zu müssen.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

36

Verarbeitung von Formularinhalten mit PHP (II)



University of Applied Sciences

PHP stellt alle Formulardaten in zwei assoziativen Arrays zur Verfügung

\$_POST für mittels POST transferierte Daten

\$_GET für mittels GET transferierte Daten

Um auf den Inhalt eines bestimmten Formularfeldes zuzugreifen, geben Sie den Namen des Formularfeldes (über das **name**-Attribut im HTML-Dokument definiert) als Schlüssel an.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

37

Beispiel: Loginverarbeitung mit GET (nicht zu empfehlen)



University of Applied Sciences

```
<form action="handler.php" method="get">
    ID: <input type="text" name="userid"/>
    PWD: <input type="password" name="pwd"/>
    <input type="submit" value="Enter"/>
</form>
```

ID:

PWD:

Datei auf Server: **Handler.php**

```
echo "Ihre Kennung lautet: {"$_GET['userid']}";
echo "Ihr Passwort lautet: {"$_GET['pwd']}";
```

Ihre Kennung lautet: nutzer

Ihr Passwort lautet: geheim

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

38

Beispiel: Loginverarbeitung mit POST



University of Applied Sciences

```
<form action="handler.php" method="post">
    ID: <input type="text" name="userid"/>
    PWD: <input type="password" name="pwd"/>
    <input type="submit" value="Enter"/>
</form>
```

ID: nutzer

PWD: geheim

Enter

Datei auf Server: Handler.php

```
echo "Ihre Kennung lautet: {"$_POST['userid']}";
echo "Ihr Passwort lautet: {"$_POST['pwd']}";
```

Ihre Kennung lautet: nutzer

Ihr Passwort lautet: geheim

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

39

Angriffsmöglichkeiten auf Formulare



University of Applied Sciences

ID: nutzer

PWD: Enter

Was würde passieren, wenn
sie in PWD folgende
Zeichenkette eintragen
würden?

```
<a href='http://www.ard.de'>
geheim</a>
```

Datei auf Server: Handler.php

```
echo "Ihre Kennung lautet: {"$_POST['userid']}";
echo "Ihr Passwort lautet: {"$_POST['pwd']}";
```

Ihre Kennung lautet: nutzer

Ihr Passwort lautet: geheim

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

40

Angriffsmöglichkeiten auf Formulare



University of Applied Sciences

Dieser Inhalt wird an den Browser gesandt, was passiert?

Ihre Kennung lautet: nutzer
Ihr Passwort lautet: geheim



Was passiert nun, wenn Sie auf den Link geheim klicken?

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

41

Dieser Effekt wird auch Cross Site Scripting genannt



University of Applied Sciences

- Es lässt sich so über Formulare
 - beliebiger Script Code (z.B. JavaScript Code)
 - Adressbuch durchsuchen?
 - Tastatureingaben protokollieren (um Ihre Passwörter auszuspähen)
 - beliebiger HTML Code
 - z.B. Links auf Phishing Seiten, um Sie unbemerkt von Ihrer Online Banking Site auf eine Seite umzulenken, die nur so aussieht, wie die der vertrauenswürdigen Deutschen Bank.
 - z.B. weitere Formulare zur Abfrage Ihrer Kontodaten mit action Handlern auf ganz anderen Servern (http://all.evil.net/password_fetching.php)
- einschleusen.
- Wird eine Formular-Eingabe gar in einer Datenbank gespeichert, ist sie sogar persistent und kann durch andere aufgerufen werden (z.B. in Forenbeiträgen)

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

42

Schutz vor Cross Site Scripting



University of Applied Sciences

- **Frei nach Dr. House:**

- Alle Nutzer sind böse, lügen, betrügen und wollen Deinem Server etwas Böses.
- Traue keiner Nutzereingabe,
- die nicht auf dem eigenen Server geprüft, gefiltert
- und für unschädlich befunden wurde
- bzw. unschädlich gemacht wurde.



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

43

Schutz vor HTML Injections mittels `htmlspecialchars`



University of Applied Sciences

- Sonderzeichen, die es in HTML gibt, mindestens jedoch die Zeichen < > maskieren
- Z.B. mittels der Funktion `htmlspecialchars`

```
echo htmlspecialchars("<tag>");           &lt;tag&gt;
```

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

44

Schutz vor HTML Injections



University of Applied Sciences

ID:

PWD:

Enter

geheim

Datei auf Server: **Handler.php**

```
$userid = htmlspecialchars($_POST['userid']);  
$pwd = htmlspecialchars($_POST['pwd']);  
  
echo "Ihre Kennung lautet: $userid";  
echo "Ihr Passwort lautet: $pwd;
```



Ihre Kennung lautet: nutzer
Ihr Passwort lautet: geheim;

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

45

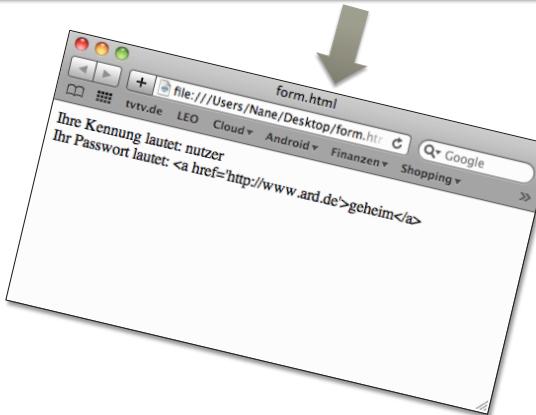
Schutz vor HTML Injections



University of Applied Sciences

Dieser Inhalt wird an den Browser gesandt, was passiert?

Ihre Kennung lautet: nutzer
Ihr Passwort lautet: geheim;

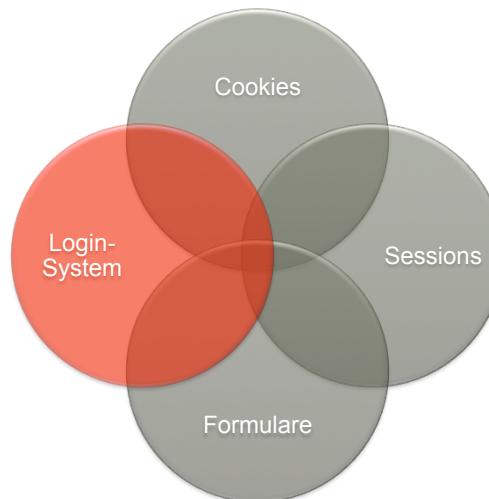


Nicht schön, aber schadlos und ungefährlich!

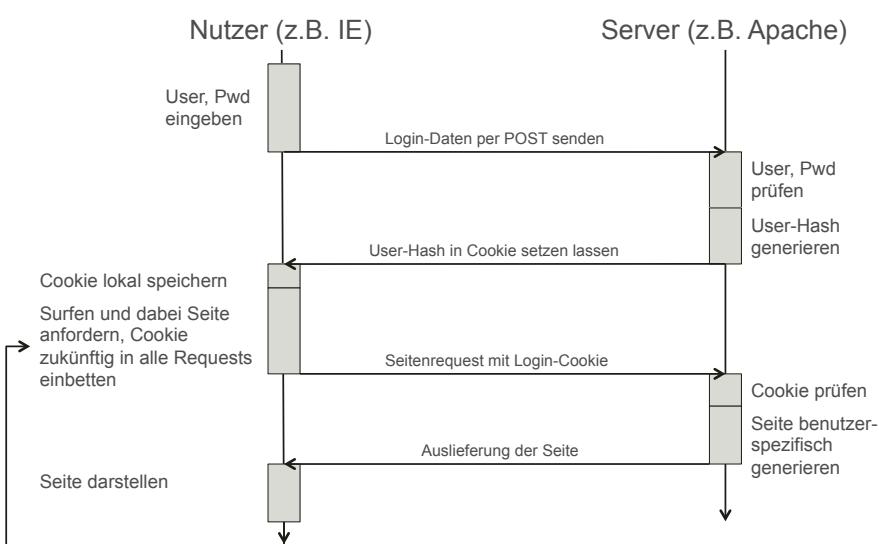
Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

46

Inhalte dieser Unit



Und nun alles zusammen Beispiel: Login Funktionalität



Und nun alles zusammen Beispiel: Login Funktionalität



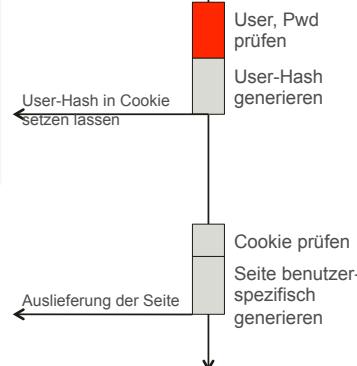
University of Applied Sciences

```
$userid = $_POST['userid'];
$pwd = $_POST['pwd'];

if ($pwd == getPasswordForUser($userid)) {
    // Valid user access data
    // Generate user hash and
    // set login cookie
} else {
    // Non valid user access data
}
```

Die Funktion `getPasswordForUser()` muss applikationsspezifisch entwickelt werden. Sie ermittelt das auf dem Server hinterlegte Passwort eines Nutzers. Üblicherweise werden die Passwortdaten z.B. in einer DB hinterlegt, könnten aber auch von einem LDAP-Server bezogen werden...

Server (z.B. Apache)



Prof. Dr. rer. nat. Nane Kratzke

49

Und nun alles zusammen Beispiel: Login Funktionalität



University of Applied Sciences

```
$userid = $_POST['userid'];
$pwd = $_POST['pwd'];

if ($pwd == getPasswordForUser($userid)) {
    $hash = md5("$userid$pwd");
    setcookie('login', "$userid,$hash");
}
```

Da das Login-Cookie beim Nutzer gespeichert wird, sollte das Passwort nicht im Login-Cookie landen. Gespeichert wird stattdessen eine Zeichenkette, die nur mittels des Passworts generierbar ist, aber nicht auf das Passwort zurücksließen lässt. Hierfür eignet sich z.B. der MD5 Hashing Algorithmus aus der Kryptologie. Es wird die Userid mit dem Passwort verkettet und darüber der Hashwert (z.B. mittels MD5) errechnet. In das Cookie wird die Userid und der Hashwert durch Komma separiert geschrieben. Beide Informationen werden benötigt, um im nächsten Schritt den Nutzer zu identifizieren und die Unversehrtheit des Cookies zu prüfen.

Server (z.B. Apache)



Prof. Dr. rer. nat. Nane Kratzke

50

Und nun alles zusammen Beispiel: Login Funktionalität



University of Applied Sciences

```

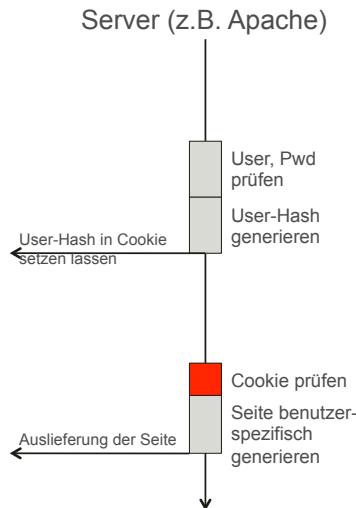
if isset($_COOKIE['login']) {
    list($usr, $hash_remote) = explode('',
        $_COOKIE['login']);

    $hash_local =
        md5($usr . getPwdForUsr($usr));

    if ($hash_remote == $hash_local) {
        // korrekt angemeldeter Nutzer
    } else {
        // nicht korrekt angemeldeter Nutzer
        // An der Hashsumme wurde manipuliert
    }
}

```

Auf dem Server wird aus dem Cookie der Benutzername und der aus Benutzername und Passwort gebildete Hashwert (hash_remote) gelesen. Der Hashwert wird auch lokal (d.h. nicht manipulierbar) berechnet. Beide Hashwerte werden miteinander verglichen. Sind sie identisch ist der Nutzer authentifiziert, andernfalls wurde ein falscher Hashwert gebildet, dies deutet auf einen Manipulationsversuch hin.



Prof. Dr. rer. nat. Nane Kratzke

51

Und nun alles zusammen Beispiel: Login Funktionalität



University of Applied Sciences

```

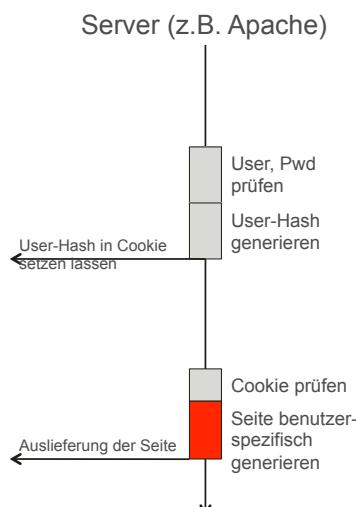
if isset($_COOKIE['login']) {
    list($usr, $hash_remote) = explode('',
        $_COOKIE['login']);

    $hash_local =
        md5($usr . getPwdForUsr($usr));

    if ($hash_remote == $hash_local) {
        // korrekt angemeldeter Nutzer
        // hier können nun beliebige $usr
        // spezifische Inhalte generiert
        // werden.
    } else {
        // nicht korrekt angemeldeter Nutzer
        // An der Hashsumme wurde manipuliert
    }
}

```

Wurde der Nutzer mittels Cookie identifiziert, können nun für \$usr beliebige spezifische Inhalte generiert werden. Bspw. nur Kurse ausgegeben werden, die \$usr belegt hat.



Prof. Dr. rer. nat. Nane Kratzke

52

>Login Prüfung in einer Komfortroutine getUsrByLogin()



University of Applied Sciences

Die gerade beschriebene Funktionalität lässt sich in folgender Komfortroutine kapseln. Diese wertet einen ggf. vorliegenden Login Cookie aus, prüft die Unversehrtheit des Userhashs und liefert die User Kennung zurück. Andernfalls null

```
function getUsrByLogin() {
    if (not isset($_COOKIE['login'])) return null;

    list($usr, $hash_remote) = explode(' ', $_COOKIE['login']);
    $hash_local = md5($usr . getPwdForUsr($usr));

    return $hash_remote == $hash_local ? $usr : null;
}
```

Diese Login Routine kann dann in folgendem Stil an beliebiger Stelle in einem PHP Skript aufgerufen werden und prüft die Korrektheit eines Login Cookies.

```
if ($usr = getUsrByLogin()) {
    // Do something $usr specific
}
```

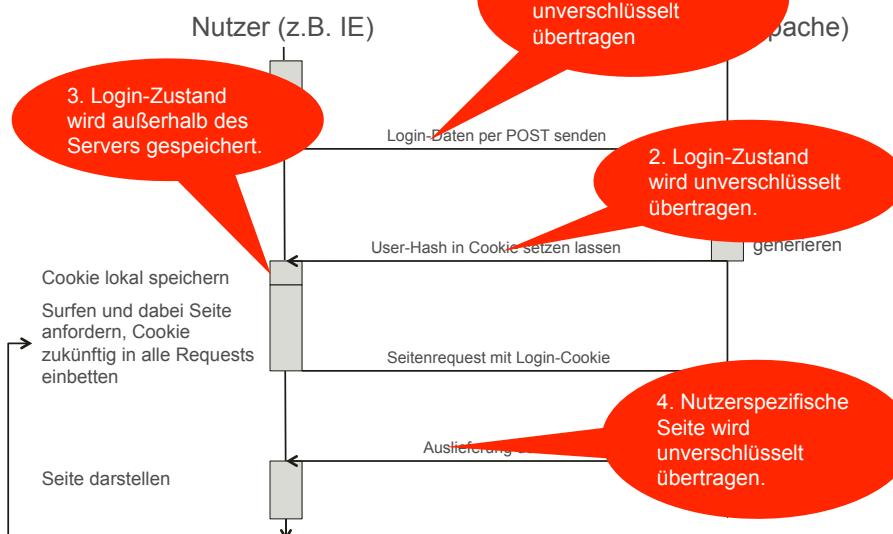
Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

53

Schwachstellenanalyse der bisherigen Login-Funktionalität



University of Applied Sciences



Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

54

Strategien zur Handhabung der Schwachstellen



University of Applied Sciences

Session-based Login Handling

Login-Zustand wird außerhalb des Servers gespeichert.

Verschlüsselte Datenübertragung

Login Daten (Passwörter) werden unverschlüsselt übertragen.

Login-Zustand wird unverschlüsselt übertragen.

Nutzerspezifische Seite wird unverschlüsselt übertragen.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

55

Session based Login Handling



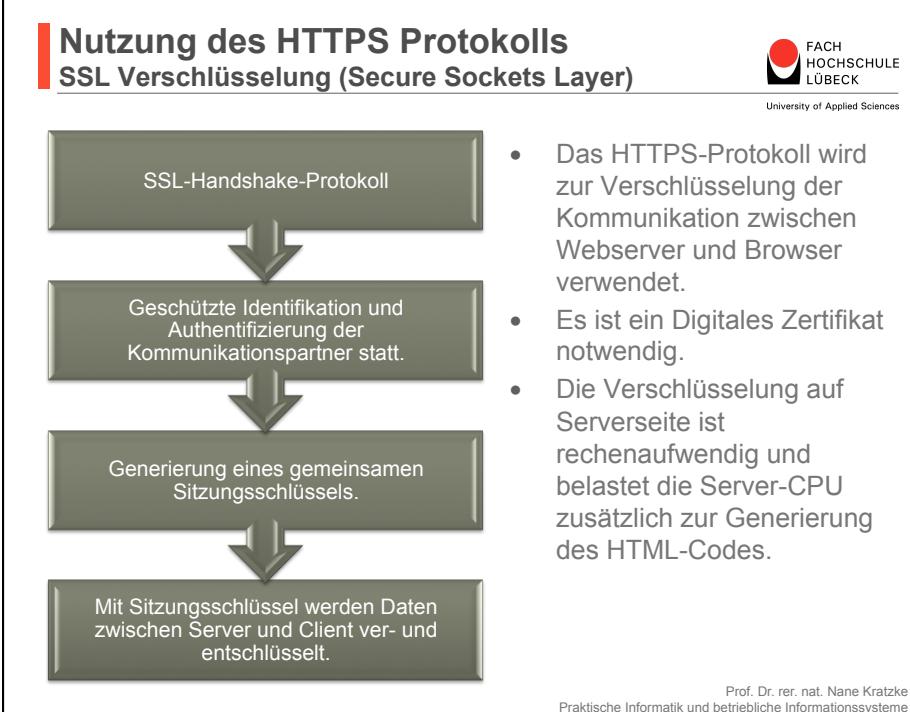
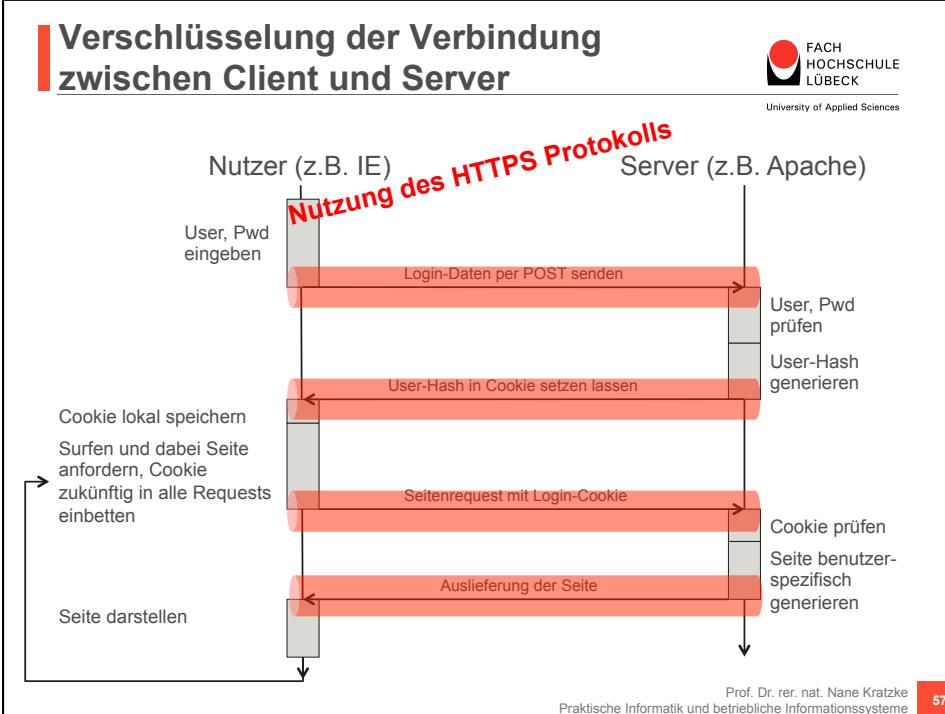
University of Applied Sciences



- Der Nutzer und sein Hash
- wird nicht in einem Cookie außerhalb des Servers
- sondern in einer Session innerhalb des Servers gespeichert.
- In der programmierten Umsetzung ist im Kern nur die superglobale Variable `$_COOKIE` durch `$_SESSION` zu ersetzen (vgl. auch Praktikum) und das Sessionhandling zu aktivieren (`session_start()`).
- Dennoch kann ein Angreifer theoretisch eine Session ID erraten oder **bei unverschlüsselter Datenübertragung mitlesen**.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

56



Strategie



University of Applied Sciences

Feststellen ob eine Seite per HTTPS angefordert wurde



Ggf. dieselbe Seite erneut über HTTPS anfordern

```
function redirectToHTTPS() {  
    if ('on' != $_SERVER['HTTPS']) {  
        $server = $_SERVER['SERVER_NAME'];  
        $uri = $_SERVER['REQUEST_URI'];  
        $url = "$server$uri";  
  
        header("Location: https://$url");  
        exit();  
    }  
}
```

Diese Komfort Funktion kann als erstes Statement einer jeden mit PHP generierten Seite aufgerufen werden und stellt so sicher, dass deren Datentransfer verschlüsselt erfolgt.

Auf dem Server muss ein SSL-Zertifikat installiert sein.

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

59

Beispiel eines PHP basierten HTTPS Redirects



University of Applied Sciences

```
<?php redirectToHTTPS() ?>  
  
<html>  
<head></head>  
<body>  
  
<h1>Hello secure World</h1>  
  
</body>  
</html>
```

Der Einsatz einer solchen Routine empfiehlt sich bei allen Seiten die sensible Daten abfragen (z.B. Login Formular) oder darstellen (sensible Nutzerdaten, z.B. Noten).

Die Verschlüsselung erfordert aber mehr Rechenleistung vom Server ab.

Daher „sparsam“ einsetzen.

Redirect to HTTP funktioniert analog

GET **http://**my.server.de/index.html



GET **https://**my.server.de/index.html

Prof. Dr. rer. nat. Nane Kratzke
Praktische Informatik und betriebliche Informationssysteme

60

Zusammenfassung



University of Applied Sciences

- **Sessions und Cookies**
 - Client-seitige Speicherung von Zuständen
 - Server-seitige Speicherung von Zuständen
- **Formulare**
 - Formularelemente
 - Angriffsmöglichkeiten auf Server durch die Verarbeitung von Formularinhalten
 - HTML Injections
 - Cross Site Scripting
- **Login-System**
 - Kennung und Passwort zur Bestimmung eines User-Hash (local und remote)
 - HTTPS für verschlüsselte Kanäle
 - Automatischer Kanalwechsel (redirection between http ↔ https)