

Sicherheit mit Cookies: Authentifizierung mit Cookies

Application Security



Inhalt

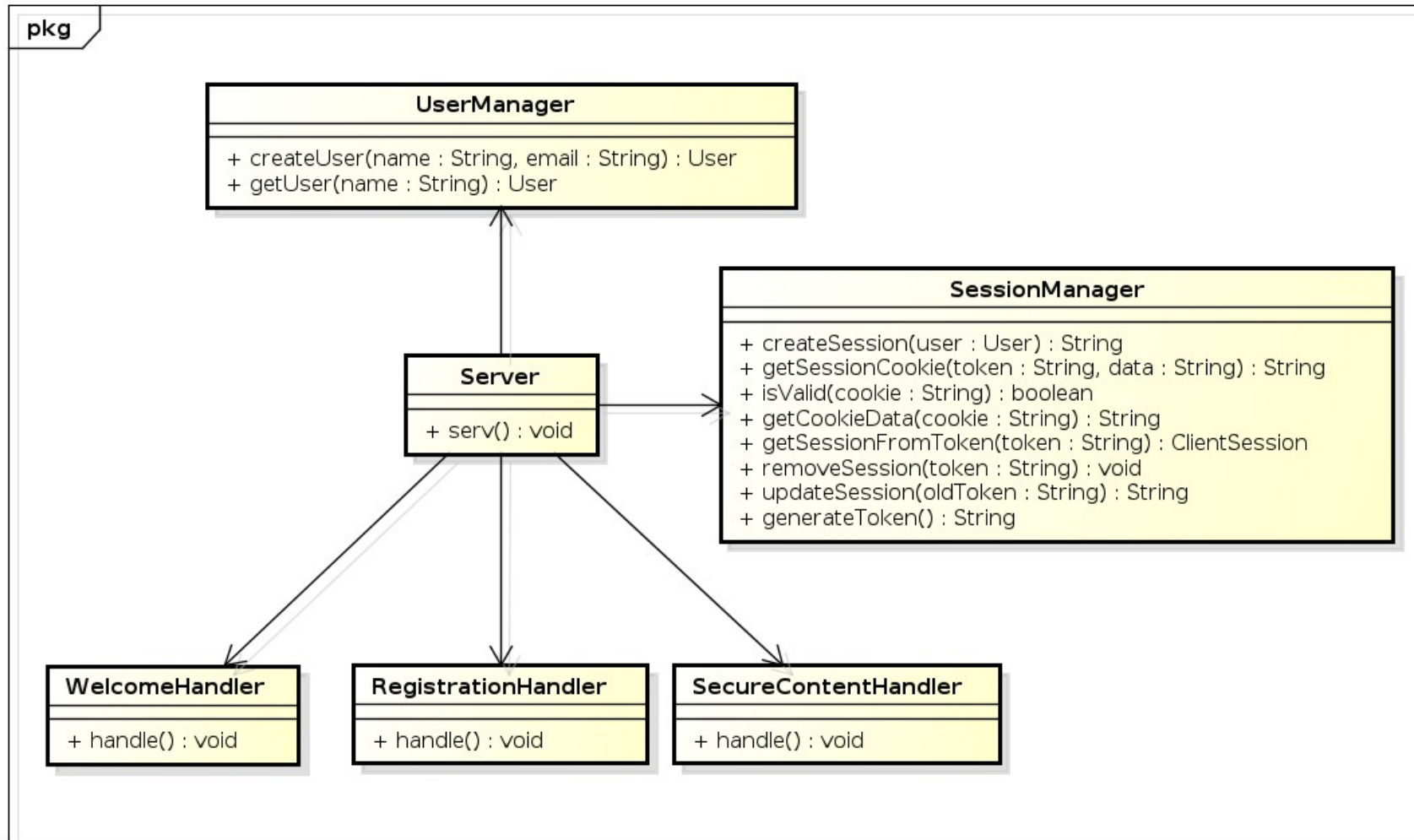
- Architektur
 - Authentifizierung
 - Verwendung kryptographischer Funktionen
 - Demonstration
 - Diskussion
-

Architektur

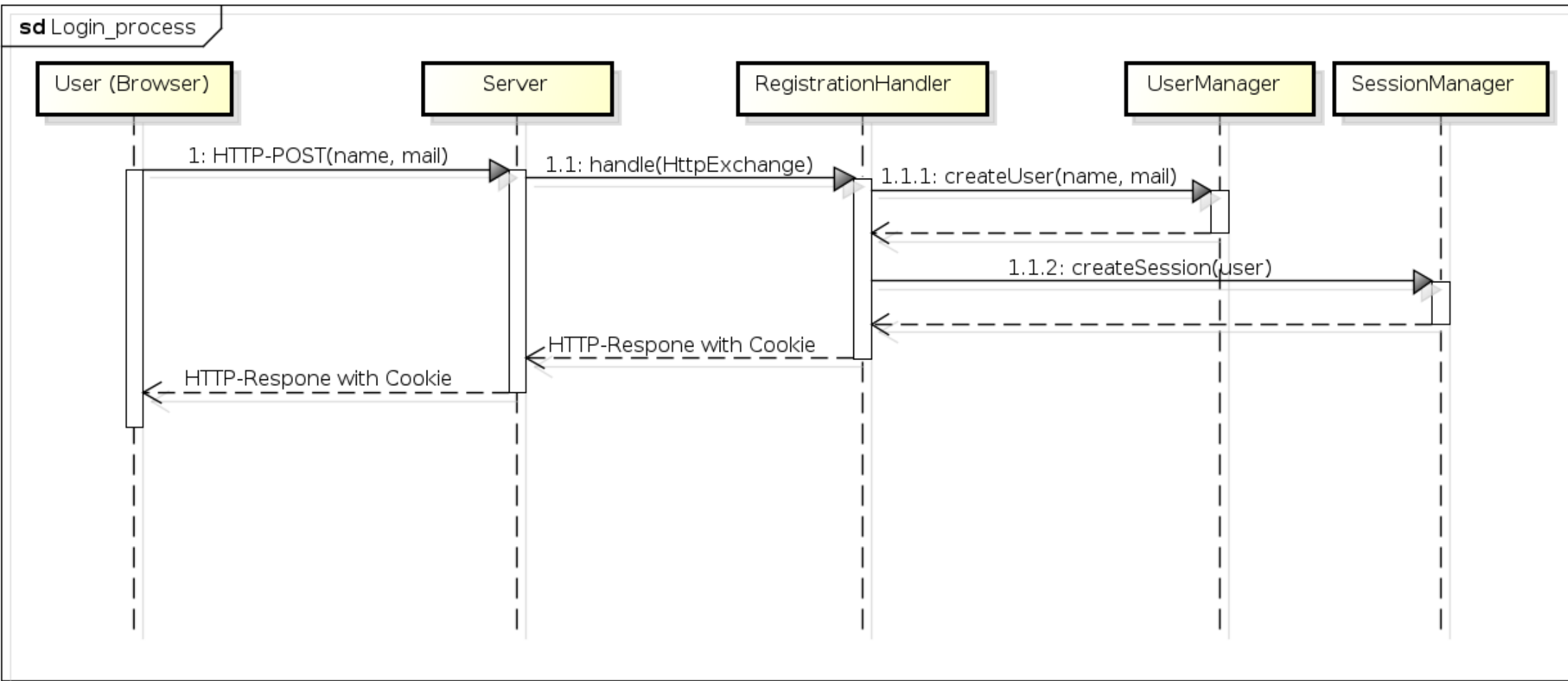
- Embedded Java Http Server
 - `com.sun.net.httpserver`
- Client Webbrowser
 - Keinerlei Logik (nicht vertrauenswürdig)



Architektur



Authentifizierungsprozess



Email Verifikation

- Regex zur Formatprüfung
- Abfrage MX-Record



Aufbau der Session

- Session
 - Owner: User
 - CurrentKey: byte[]
 - Token: String (Identifies the session)
-

Aufbau des Cookies

- Secure Cookie Protocol
- University Texas & South Carolina

(<http://www.cse.msu.edu/~alexliu/publications/Cookie/cookie.pdf>)

```
token | expiration timestamp | AES(data, k)
| HMAC(token | expiration timestamp | data | SSL Session Key, k)

k = HMAC(token | expiration timestamp, sk)
sk = server key
HMAC(d,k) = Hash based message authentication code using data d and key k
```

Aufbau des Cookies

- Daten
 - IP
 - Immutable Header Values

```
token | expiration timestamp | AES(data, k)
| HMAC(token | expiration timestamp | data | SSL Session Key, k)

k = HMAC(token | expiration timestamp, sk)
sk = server key
HMAC(d,k) = Hash based message authentication code using data d and key k
```

Überprüfen des Cookies

```
function checkCookie(cookie, requestHeader) {  
    if(sessionmanager has no token = cookie.token) return FALSE  
    sk = sessionmanager.getServerKeyForClient(cookie.token)  
  
    if(cookie.expirationTime < current type) return FALSE  
  
    k = HMAC(token | expiration timestamp, sk)  
  
    descypted = decrypt(cookie.data, k)  
  
    if(decrypt failed) return FALSE  
  
    expected = HMAC(token | expiration timestamp | data | SSL Session Key, k)  
  
    if(expected != cookie.hash) return FALSE  
  
    if(headerHash(request) != descypted) return FALSE  
  
    return TRUE  
}
```

Session Renewal

```
function renewSession(oldToken) {  
    sess = getSession(oldToken)  
    if(sess == null) return null  
  
    removeSession(oldToken)  
  
    newToken = generateToken()  
  
    sess.expiringTime = current + TIMEOUT_TIME  
    sess.currentKey = generateKey()  
    sess.token = newToken  
  
    putSession(newToken, sess)  
  
    return newToken  
}
```

Demo Time!

Application Security



Diskussion

- Replay Attacks
 - Bedingt
 - So gut wie ohne TLS möglich
 - Mit TLS beinahe unmöglich
 - Einhashen der SSL-Session ID
 - Man-in-the-Middle
 - TLS entscheidend
-

Vorgehen

- Robust Programming
 - FindBugs
 - JSR-305
 - Java Crypto API
 - Auch für generierung von Zufallswerten
-

Sicherheit mit Cookies: Authentifizierung mit Cookies

Application Security

