

## Übungsblatt 1

### Aufgabe 2: HTTP Authentisierung

Lesen Sie RFC 2616. Welche Möglichkeiten der Benutzerauthentisierung sieht der HTTP/1.1 Standard vor? Erläutern Sie jeweils deren Funktionsweise.

#### Antwort:

HTTP/1.1 sieht 2 verschiedene Methoden zur Benutzerauthentisierung vor:

#### 1. Basic-Access-Authentisierung (RFC 2617):

Client sendet Benutzernamen und Passwort Base64-kodiert (Kodierung in 7-bit ASCII) im Authorization Header des HTTP-Requests. (Passwort dadurch nicht sofort lesbar, aber ohne Probleme ermittelbar)

→ Authentisierungstyp allenfalls für sehr geringe Vertraulichkeitsanforderungen zu gebrauchen.

#### Funktionsweise:

Der Client sendet mit seiner ersten Request eine Anfrage ohne Authentifizierungsinformationen. Die Antwort des Servers enthält das WWW-Authenticate-Header-Feld, das den Client auffordert, die Zugangsdaten zu übermitteln. In diesem Headerfeld sind zudem Details definiert, wie die Authentifizierung erfolgen muss. Der Client wird zur Übermittlung der Authentifizierungsinformationen aufgefordert. Der Server überträgt in seiner Antwort den so genannten Realm, eine Zeichenfolge, mit der er dem Client mitteilt, wer die Daten fordert. Der Client verwendet für die Kodierung von Kennung und Passwort das Base64 -Verfahren. Diese kodierte Zeichenfolge wird im Authorization-Header-Feld an den Server übermittelt.

#### Ein Beispiel für eine solche Request:

```
GET /verzeichnis_x/seite_y.html HTTP/1.1
User-agent: Mozilla/4.6
Accept: text/html, image/gif, image/jpeg
Authorization: Basic KDWkfowsOm=
```

Der Authentifizierungsvorgang muss allerdings bei jeder Request erfolgen. Der Client legt daher die Daten im Cache ab und greift bei jeder nachfolgenden Request auf diese Informationen zurück.

#### 2. Digest-Authentisierung (ebenfalls RFC 2617)

Auf dem Server liegt das Passwort des Benutzers in Klartext vor. Der Client erhält einen vom Server generierten Zufallsstring (Challenge). Aus dem Challenge und dem Passwort des Benutzers errechnet der Client nach einem standardisierten Verfahren einen sog. Digest, der

als Authentisierung dem Server gesandt wird. Der Server berechnet ebenfalls Digest und vergleicht  
→ geeignet für höheren Schutzbedarf, da das Passwort nicht über das Netz verschickt wird

### **Funktionsweise:**

Analog zu Basic Access, außer:

Server antwortet (3.) zusätzlich mit einem vom Server generiertem String (Nonce), der für jede Autorisierungsanfrage eindeutig neu erzeugt werden sollte

Es gibt einen zusätzlichen Header: Authentication-Info

Die Digest Access Authentication benutzt zudem mehrere Parameter für die Verschlüsselung. Die Verwendung von Zufallswerten ist dabei vorgeschrieben. Für die Verschlüsselung des Passwortes nimmt der Mechanismus neben dem Zufallswert den Benutzernamen, das Passwort, die HTTP-Methode und die angeforderte URL. Die Verschlüsselungsmethode basiert in der Praxis meist auf einem MD5-Algorithmus. Versucht ein Client auf eine gesicherte Site zuzugreifen, antwortet der Server zunächst mit einem Unauthorized-Header, z.B.:

```
HTTP/1.1 401 Unauthorized
www-Authenticate: Digest
realm="testrealm@host.com",
qop="auth,auth-int",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

Der Client übermittelt anschließend Benutzername und Passwort zum Server, z.B.

```
Authorization: Digest username="Benutzername",
realm="testrealm@host.com",
nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
uri="/dir/index.html",
qop=auth,
nc=00000001,
cnonce="0a4f113b",
response="6629fae49393a05397450978507c4ef1",
opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

Nach der Übermittlung des verschlüsselten Passwortes gewährt oder verwehrt der Server den Zutritt. Die zwischen HTTP-Server und Client übertragenen Daten und Kommandos bleiben von diesem Mechanismus unberührt.