АУТЕНТИФИКАЦИЯ И АВТОРИЗАЦИЯ В WEB

KEYWORDS: AUTHENTICATION, AUTHORIZATION, OAUTH, FORMS, HTTPS

СОДЕРЖАНИЕ

- **HTTP аутентификация**
  - Basic authentication
  - Digest authentication
  - OAuth
- **Forms аутентификация**
- **HTTP Secure (HTTPS)**

ТЕРМИНОЛОГИЯ

Идентификация – присвоения субъектам меток (ID), а также процесс сравнение этих меток со списком

Аутентификация – процедура проверки подлинности. Может быть **пользователя** (по ID и паролю), **компьютера**, **документа** (по эл. подписи). Бывает **многофакторная**, односторонняя и двусторонняя

Авторизация – предоставление прав на выполнение действий

ЗАДАЧИ HTTP-АУТЕНТИФИКАЦИИ

- **Ограничение доступа к ресурсу механизмами протокола HTTP**

  - Редкое явление для **сайтов**. Большинство известных решений использует forms-based авторизацию

  - Частое явление для **сервисов** и **API** (там где доступ осуществляется не браузером или через ajax)

ОБЩИЕ ПОЛОЖЕНИЯ АУТЕНТИФИКАЦИИ

Если сервер возвращает 401, значит он требует **аутентификации**. Он обязан содержать заголовок WWW-Authenticate

```
HTTP/1.0 401 Unauthorized
Cache-Control: no-cache
Pragma: no-cache
Content-Length: 58
Content-Type: text/html
Expires: -1
Server: Microsoft-IIS/8.0
WWW-Authenticate: Basic realm="area to be accessed"
```

BASIC AUTHENTICATION

Самый простой вариант аутентификации

```
GET /sometail.aspx HTTP/1.1
Host: somehost
Authorization: Basic bG9naW46cGFzc3cwcmQ=
```

где

“bG9naW46cGFzc3cwcmQ=” == base64(“login:passw0rd”)

NB:

- Логин и пароль, фактически не защищены! Использовать только over HTTPS
- Отправка возможна и сразу (без challenge), она должна быть в каждом запросе

DIGEST AUTHENTICATION

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

… … …

Authorization: Digest username="Mufasa",
                  realm="testrealm@host.com",
                  nonce="dcd98b7102dd2f0e8b11d0f600bfb0c093",
                  uri="/dir/index.html",
                  response="e966c932a9242554e42c8ee200cec7f6",
                  opaque="5ccc069c403ebaf9f0171e9517f40e41"
```

RFC 2069:

$$HA1 = MD5\big(A1\big) = MD5\big(username : realm : password\big)$$
$$HA2 = MD5\big(A2\big) = MD5\big(method : digestURI\big)$$
$$response = MD5\big(HA1 : nonce : HA2\big)$$

RFC 2617:

$$HA1 = MD5\big(A1\big) = MD5\big(username : realm : password\big)$$
$$HA2 = MD5\big(A2\big) = MD5\big(method : digestURI\big)$$

Если значение директивы QOP равно «auth-int», то HA2 равняется:

$$HA2 = MD5\big(A2\big) = MD5\big(method : digestURI : MD5(entityBody)\big)$$

Если значение директивы QOP равно «auth» или «auth-int»,

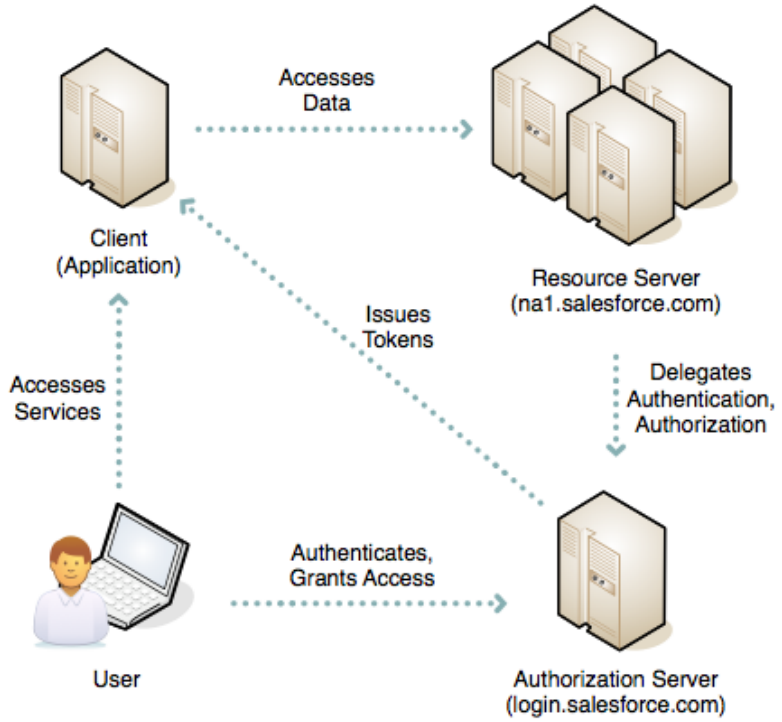
$$response = MD5\big(HA1 : nonce : nonceCount : clientNonce : qop : HA2\big)$$

Если директива QOP не определена, то ответ вычисляется так:

$$response = MD5\big(HA1 : nonce : HA2\big)$$

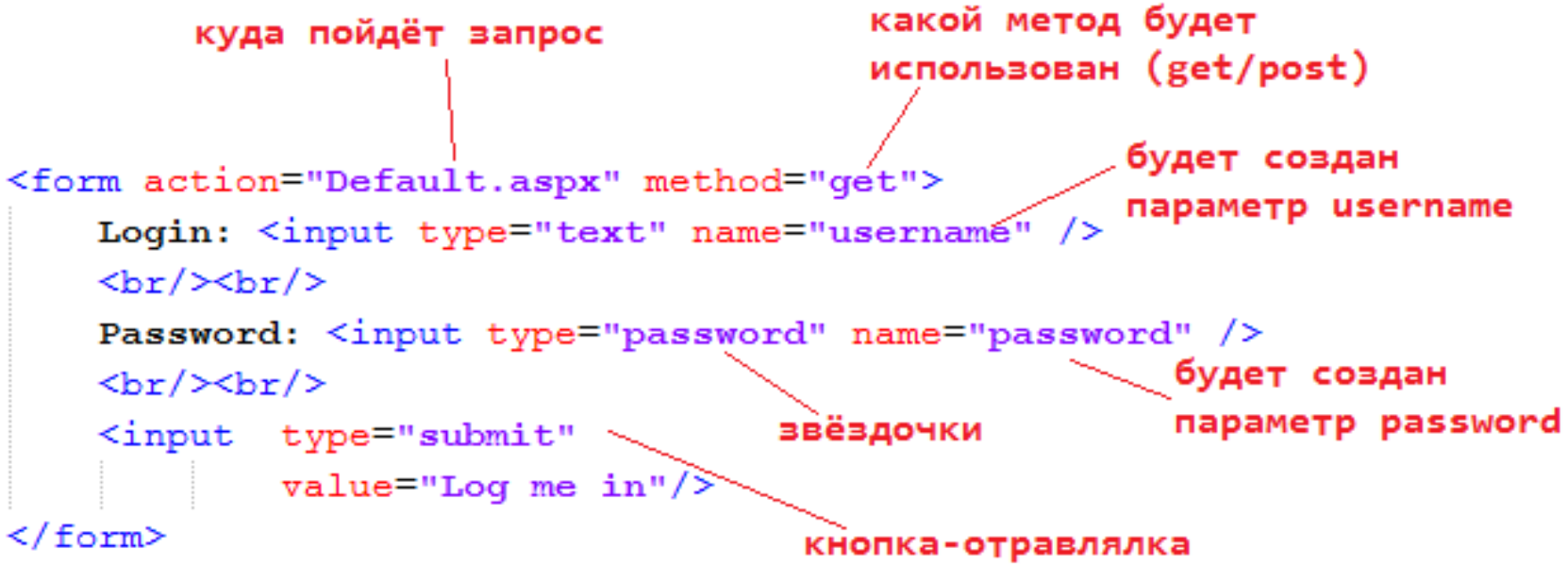
OAUTH

- **Аутенитифицирует приложения на правах пользователя (или анонимно)**
- **Базируется на Access Tokens**



FORMS AUTHENTICATION

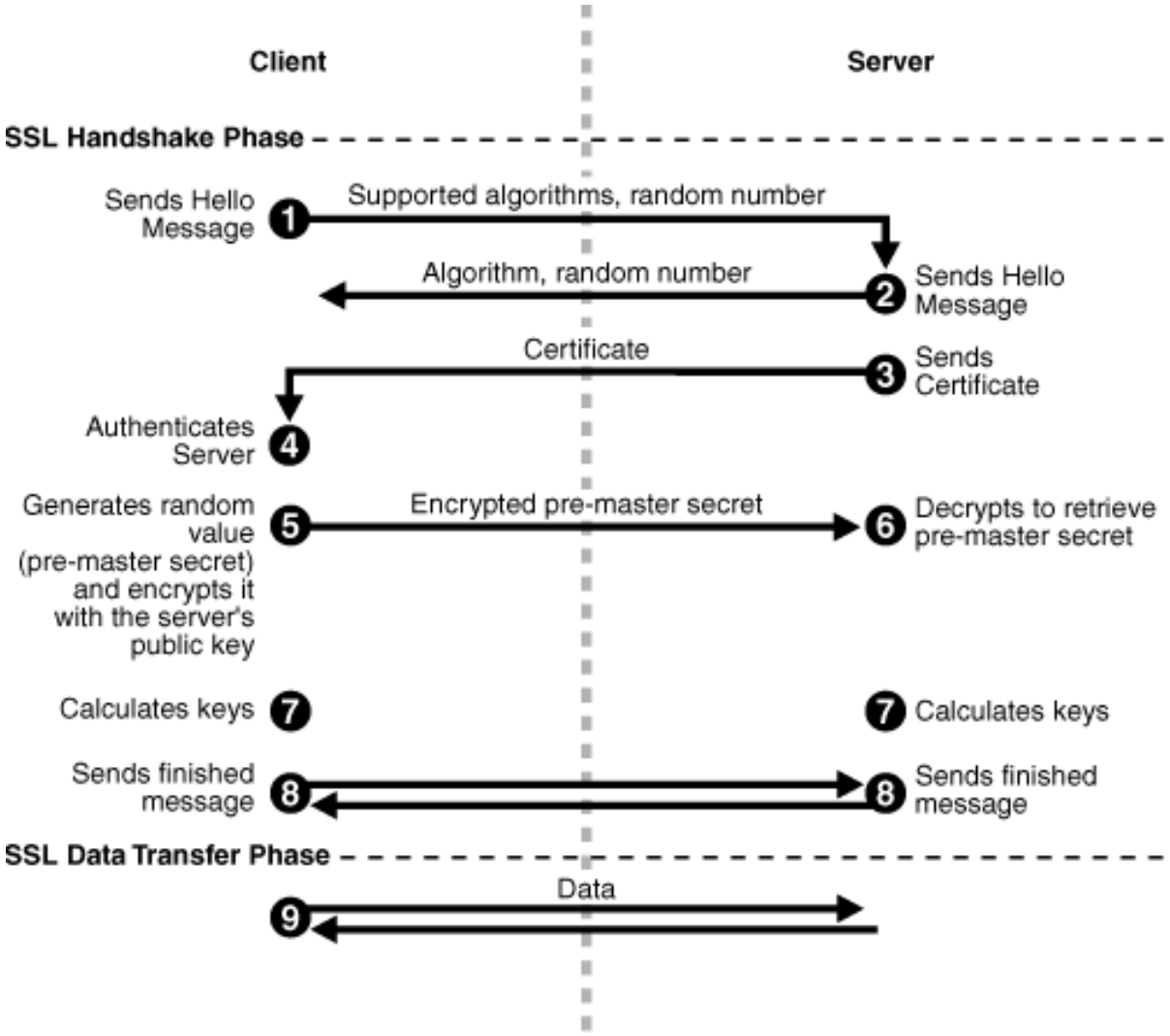
- **Не является часть протокола HTTP**
- **Базируется на HTML-тэге <FORM> и параметрах запроса**



HTTP SECURE

- **HTTPS = HTTP over SSL/TLS**
  - SSL – протокол с ассиметричной криптографией и симметричным шифрованием
  - TLS = SSL v3
- **HTTP (FTP, telnet) прозрачно работает поверх SSL/TLS**
  - Сначала клиентское приложение производит «рукопожатие» (handshake)
  - Затем идёт фаза передачи данных через созданный канал по стандартному протоколу (HTTP)

HTTPS HANDSHAKE



СЕРТИФИКАТЫ

Цифровой сертификат – электронный документ (файл), подтверждающий принадлежность открытого ключа предъявителю. Сертификат должен быть подписан сертификационым центром.

- Обязательные параметры сертификата:
  - resource ID (**Subject**)
  - **public** key
  - certification authority (**Issuer**)
- Необязательные параметры сертификата:
  - private key
  - usage

ПОЛЕЗНЫЕ ССЫЛКИ

OAuth intro

http://www.youtube.com/watch?v=io_r-0e3Qcw

OAuth 2.0

http://www.youtube.com/watch?v=khnmMv4_RCE