Лекция 21

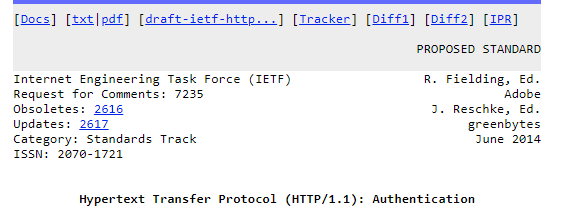
ПСКП

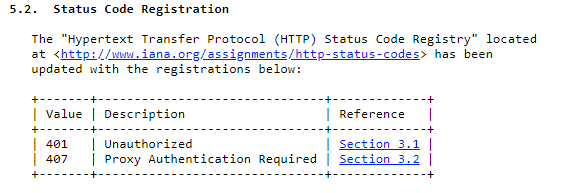
ПОИТ-3

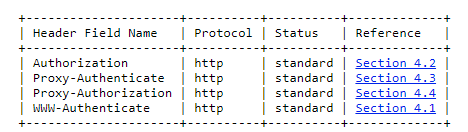
**HTTP Authentication**

1. **Идентификация** – заявление пользователя о себе.
2. **Аутентификация** – процедура проверки подлинности идентификации пользователя.
3. **Авторизация** -процедура проверки прав аутентифицированного пользователя.

1. **HTTP Authentication**:

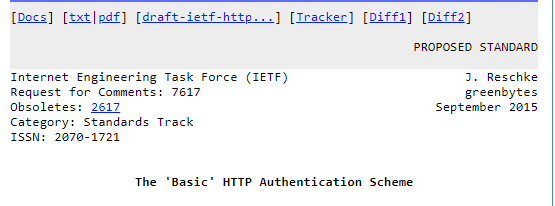






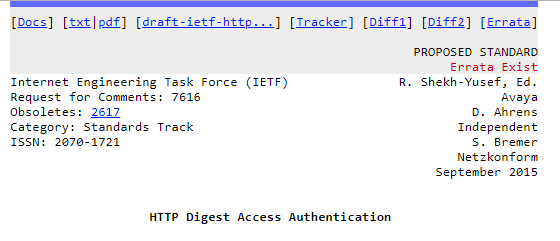


1. **HTTP Authentication**: **BASIC**

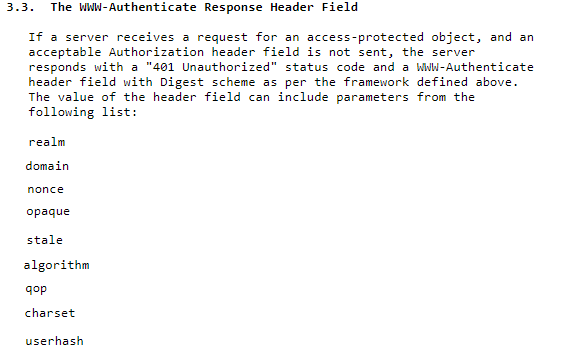


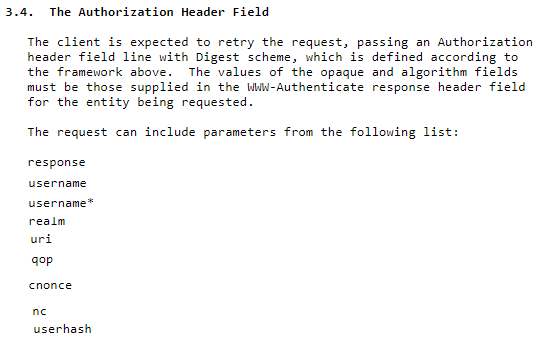


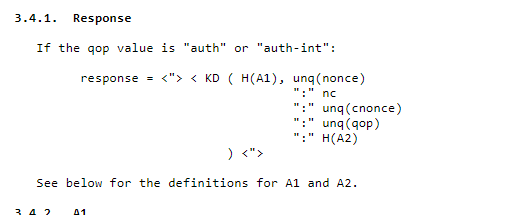
1. **HTTP Authentication**: **Digest**

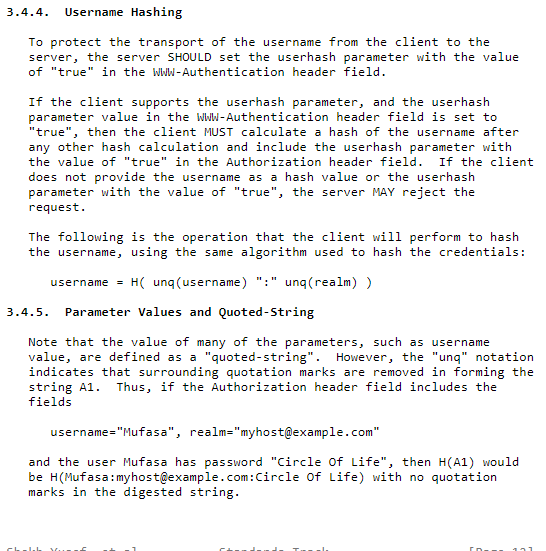








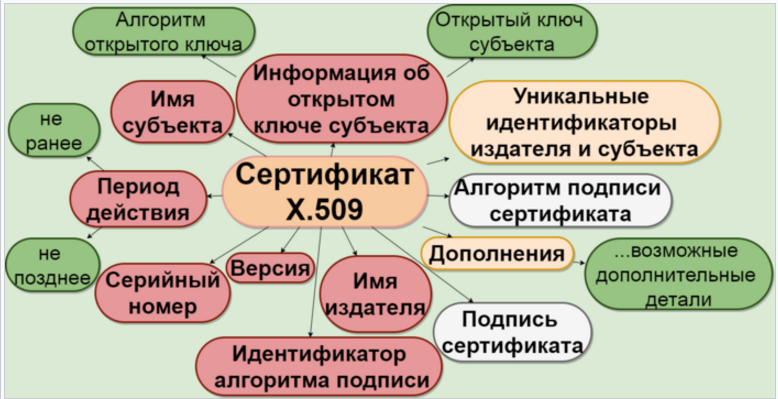




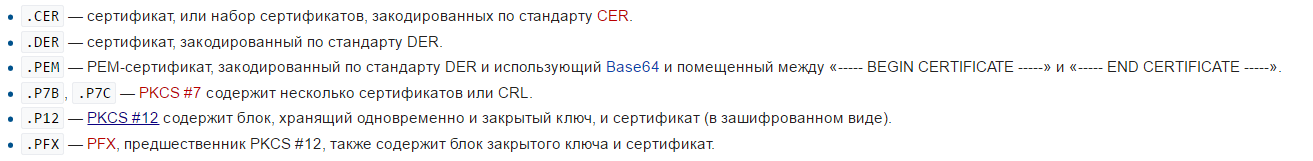
1. **HTTP Authentication**: Forms (нет стандарта, но применяется)



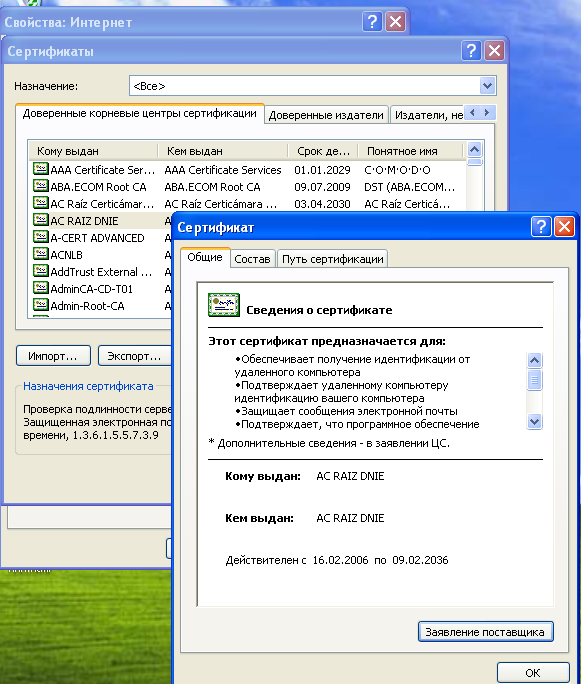
1. **HTTP Authentication**: TSL-аутентификация.
2. **Сертификат X.509:** стандартныйформат хранения и транспортировки атрибутов безопасности; главное – открытый ключ; сертификаты выдают центры сертификации (Certificate Authority, CA). <http://www.thwate.com>, <http://www.Verysign.com>; если это intranet, то можно использовать **Certificate Server Active Directory**; сертификат содержит: имя держателя, адрес, серийный номер сертификата, даты проверки, открытый ключ держателя.

****

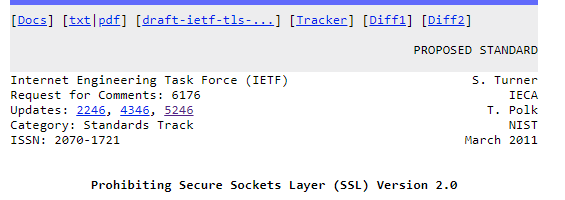
1. **Сертификат X.509:** Выдает центр сертификации (СА). <http://www.thwate.com>, <http://www.Verysign.com>. Если это intranet, то можно использовать **Certificate Server Active Directory**. Сертификат содержит: имя держателя, адрес, серийный номер сертификата, даты проверки, открытый ключ держателя.
2. **Сертификат X.509:** форматы файлов сертификатов X.509

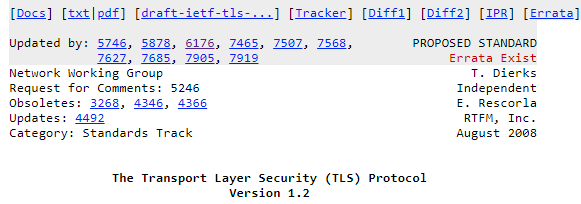


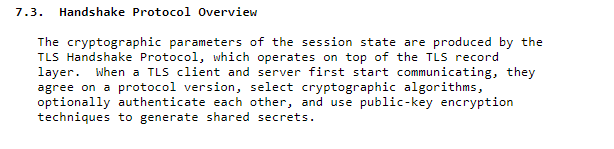
1. **Сертификат X.509:** хранилище сертификатов

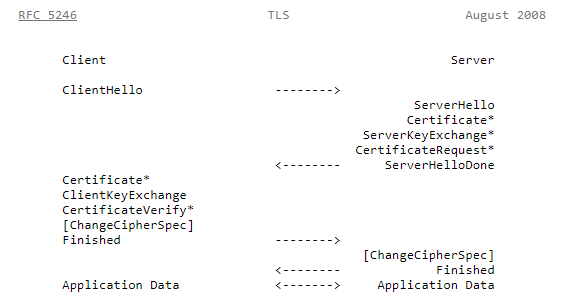


1. **TLS:** 1)обеспечиваетконфиденциальность; 2) обеспечивает целостность (обнаружение подмены); 3) аутентификация узлов (проверка подлинности источника сообщений); последняя версия 1.2, в разработке 1.3 (значительные изменения от 1.2); поверх потокового надежного соединения (для ненадежной передачи есть DTSL);
2. **TLS. RFC 2246,** новое название **Transport Layer Security (TLS).** Последние обновления RFC 5246, 6176**.**





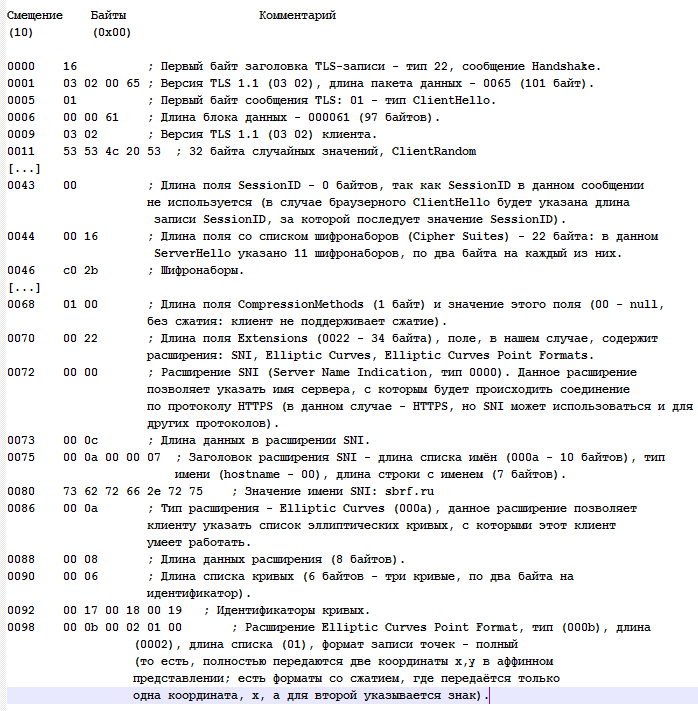




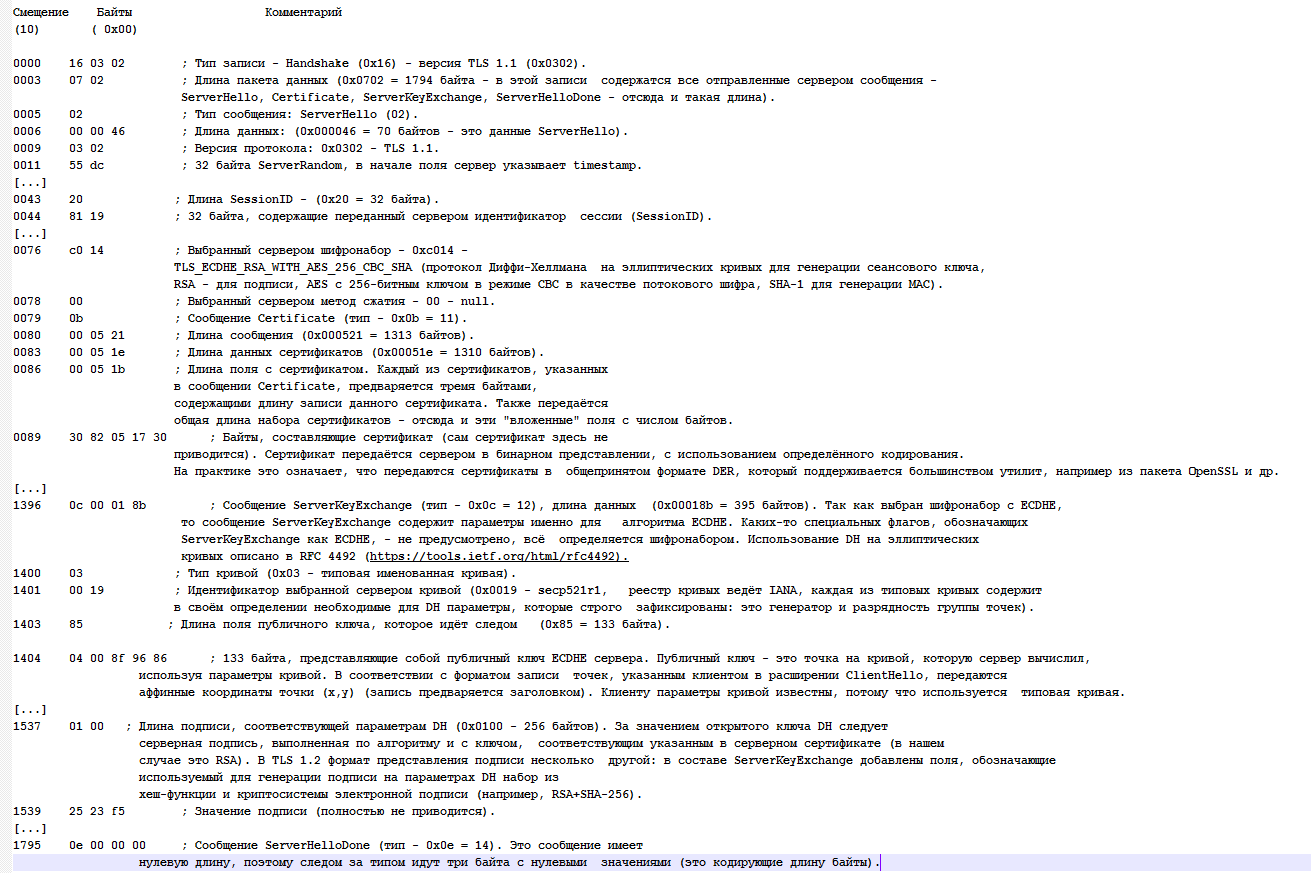
1. **TLS:**  пересылает записи



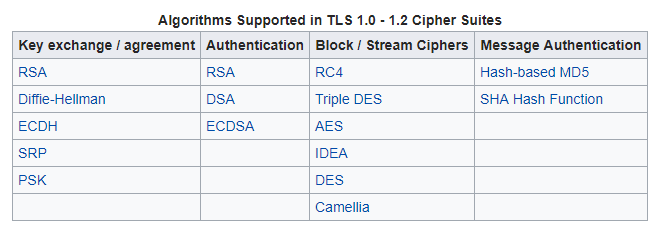
1. **TLS:** Client Hello, пример

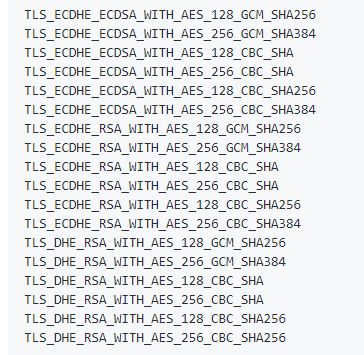


1. **TLS:** Server Hello, пример

****

1. **TLS:**  Cipher Suites – шифронаборы; в шифронабор входит: криптосистема для аутентификации сервера и сеансового секрета; шифр для защиты передаваемых данных; хеш-функция для кода аутентификации HMAC.

****

****

**TLS\_ECDHE\_ECDSA\_WITH\_AES\_128\_GCM\_SHA256**

**ECDHE -** Диффи-Хеллман на эллиптических кривых для вычисления общего секрета;

**ECDSA -** аутентификацияданных на этапе установления соединения на основе цифровой подписи на эллиптических кривых;

**AES\_128** – шифрование полезной нагрузки с помощью алгоритма AES с 128-битным ключом в режиме GCM;

**SHA256** -для хеширования применяется алгоритм SHA с 256-битным ключом.

1. **TLS:** Схема работы
2. **Клиент** выдает запрос серверу (Client Hello).
3. **Сервер** подписывает свой сертификат и высылает клиенту (Server Hello).
4. **Клиент** проверяет сертификат в центре сертификации, которому доверяет.
5. **Клиент** сравнивает данные сертификата с информацией центра сертификации.
6. **Клиент** сообщает серверу, какие ключи шифрования он поддерживает.
7. **Сервер** выбирает подходящую длину ключа.
8. **Клиент** генерирует симметричный ключ, шифрует его открытым ключом.
9. **Сервер** получает симметричный ключ и расшифровывает его.



1. **NTLM** (Windows authentication): пароль не передается в чистом виде. Эта схема не является стандартом HTTP, но поддерживается большинством браузеров и веб-серверов. Преимущественно используется для аутентификации пользователей Windows Active Directory в веб-приложениях.

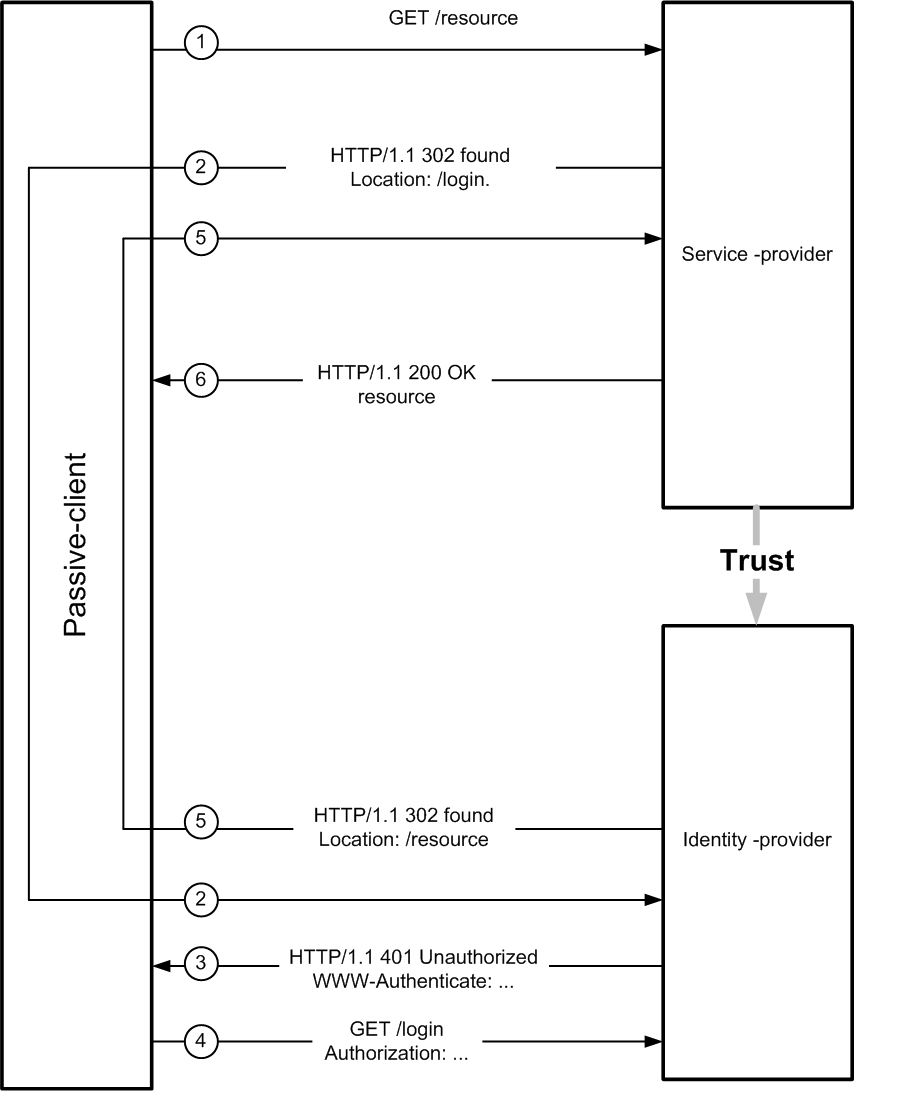
1. **Negotiate** (Windows authentication): клиенту выбрать можно выбрать между NTLM и Kerberos -аутентификацией. Kerberos – протокол, основанный на принципе Single Sign-On, клиент и сервер должны находятся в зоне intranet и являться частью домена Windows.

**Token-аутентификация**

1. **Token:** битовая последовательность, построенная по определенному принципу.
2. **Token-аутентификация:** аутентификация, использующая token для идентификации пользователя или авторизации операции пользователя.
3. **Token-аутентификация:** применяется, как правило, для реализации Single Sign-On в распределенных системах.
4. **Token-аутентификация:** Identity-provider - сервер, генерирующий token.
5. **Token-аутентификация**: Service-provider – сервер, предоставляющий сервис клиенту.
6. **Token-аутентификация**: активный клиент – программный код, который может выполнять любые запросы и обрабатывать любые ответы. Обычно – это пользовательская программная реализация http-клиента. Пассивный клиент – браузер.
7. **Token-аутентификация**: схема работа с активным клиентом.

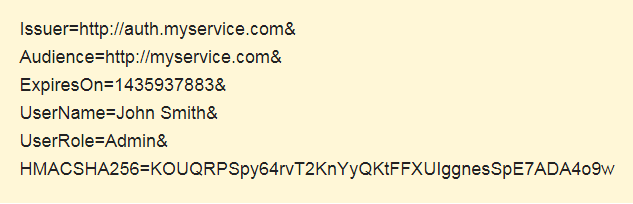


1. **Token-аутентификация**: схема работа с пассивным клиентом (браузером).

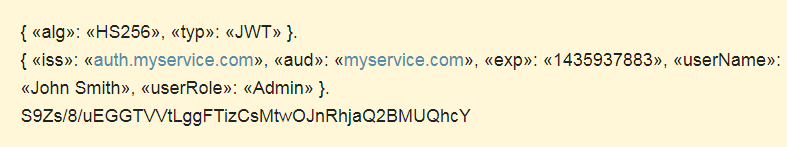
****

1. **Token-аутентификация**: Service-provider и Identity-provider должны иметь общий секретный ключ для шифрования/проверки token’а.
2. **Token-аутентификация**: token обычно строится на базе Issuer(эмитент),Audience (аудитория),Expires On (время жизни), Claim (сведение о пользователе) & Statements (дополнительные утверждения о пользователе), HMAC (хеш).

1. **Token-аутентификация**: Simple Web Token (SWT)

****

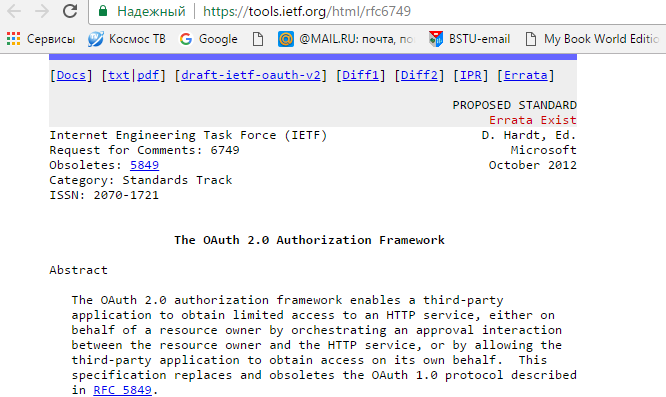
1. **Token-аутентификация**: JSON Web Token (JWT)

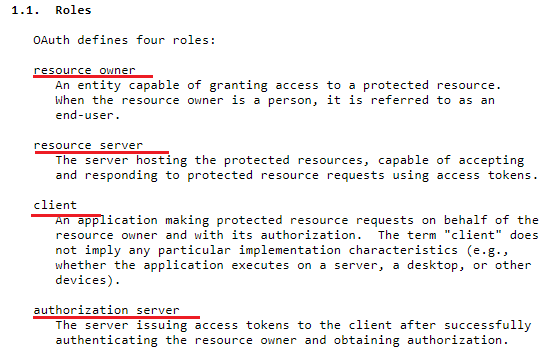
****

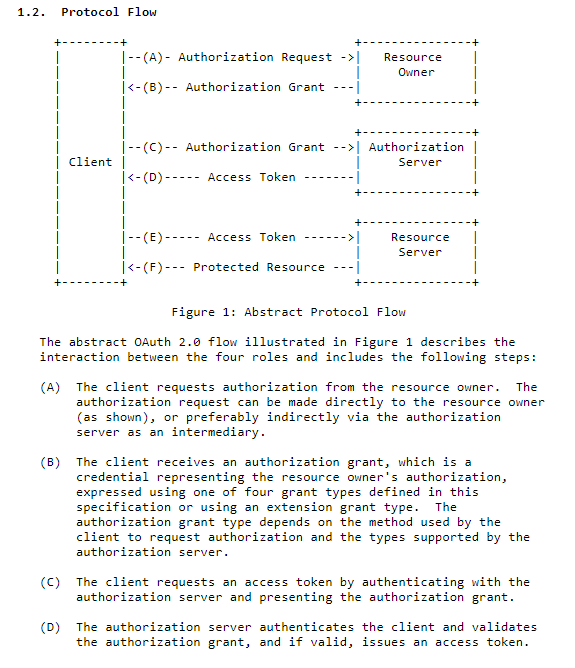
1. **Token-аутентификация**: Security Assertion Markup Language (SAML), разработчик OASIS, XML-формат, механизм подтверждения токенов, подпись ассиметричный ключ.

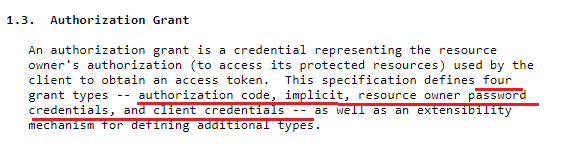
****

1. **OAuth 2.0:** Открытый стандарт аутентификации

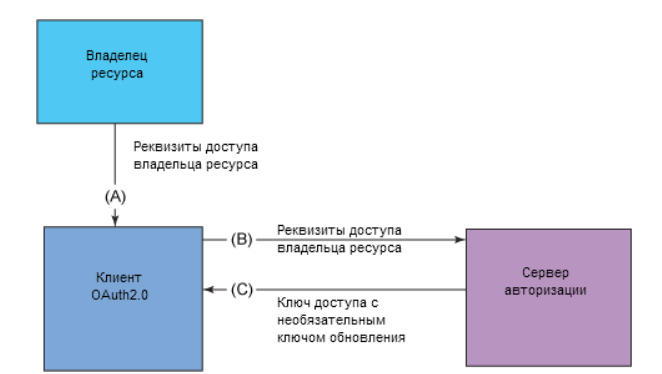






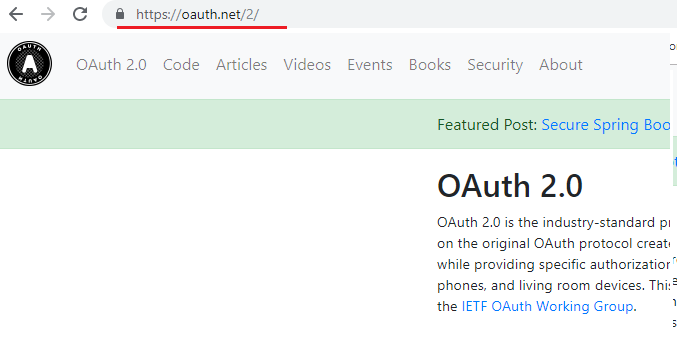


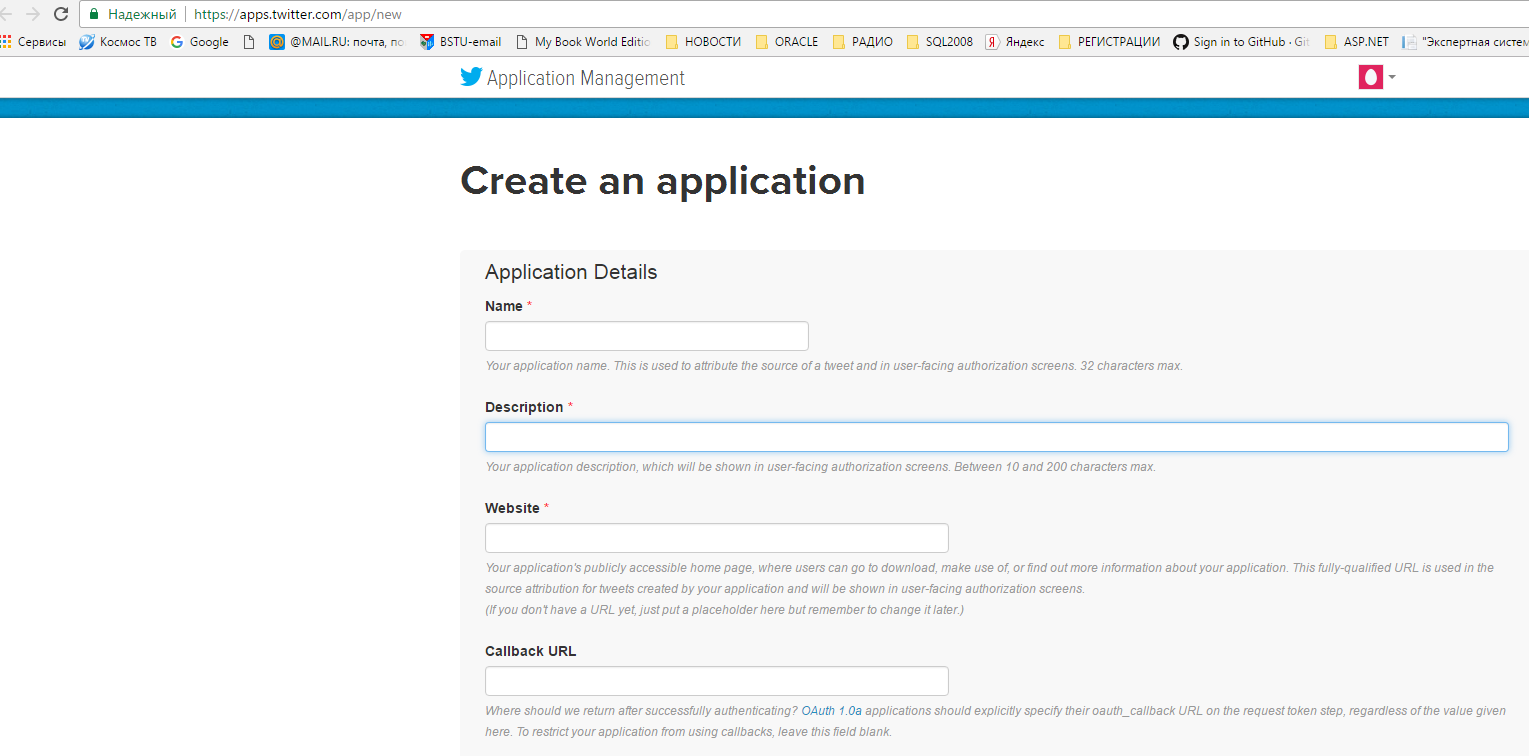
1. **OAuth 2.0:** resource owner password credential

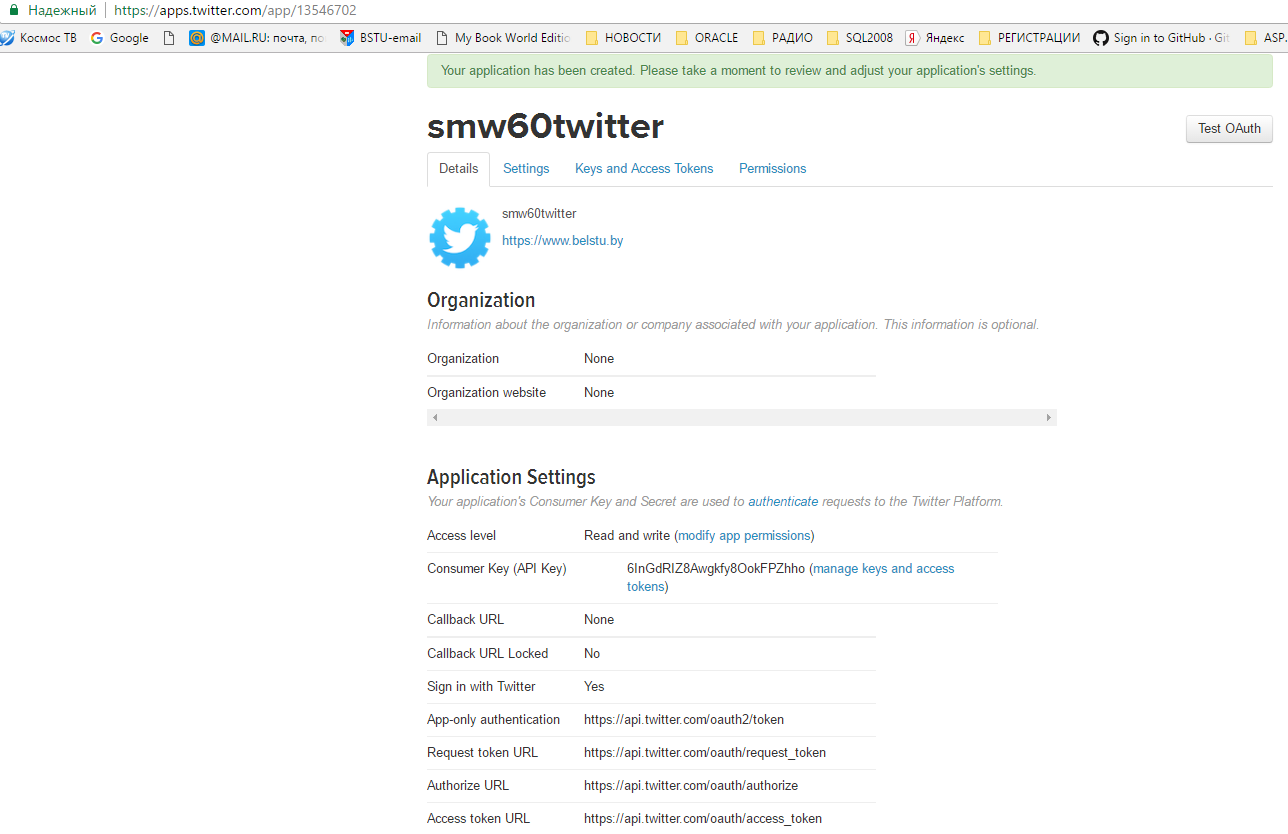


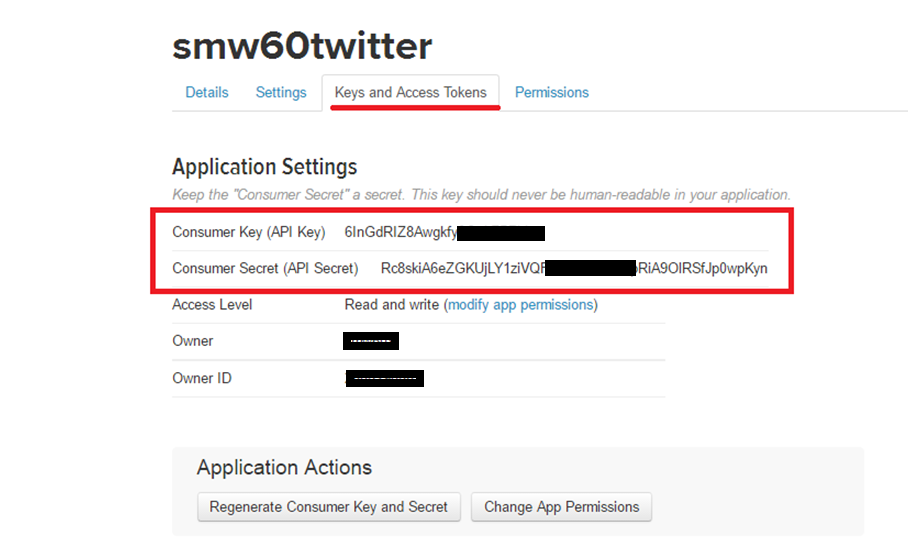
1. **OAuth 2.0:** пример запрос/ответ resource owner password credential ( на рис. шаги В и C)



1. **OAuth 2.0: официальный сайт**
2. **Twitter OAuth 2.0**:



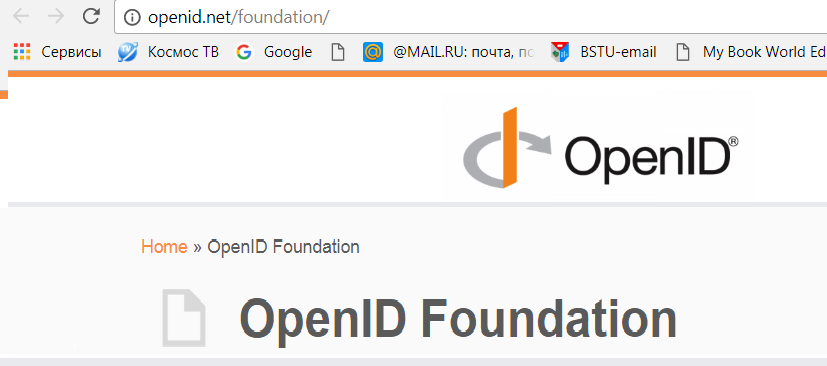




1. **Twitter OAuth 2.0**: ASP.NET MVC 5/Web Forms



1. **OpenID:** автор Брэд Фицпатрик (создатель LifeJournal), 2006.
2. **OpenID:** открытый стандарт аутентификации, позволяющий реализовать Single Sin-On (технология единого входа). Версии: OpenID 1.1, OpenID 2.0, OpenID Connect 1.0 (над протоколом OAuth). Стандарт можно расширять.
3. **OpenID:** генерация цифрового идентификатора для децентрализованного применения.
4. **OpenID:** использует только стандартные HTTP(S)-сообщения; не требуется cookie.
5. **OpenID:** идентификатор в формате HTTP(S) URI или XRI (формат разработанный OASIS).
6. **OpenID:** OpenID-провайдер –сервер, хранящий информацию об идентификаторе, подтверждает подлинность идентификатора.
7. **OpenID:** интернет-сервис - приложение, которое хочет проверить подлинность OpenID-идентифкатора
8. **OpenID Connect:**



1. **OpenID:**
2. OAuth 2.0
3. WS-\*
4. Bearer
5. HOBA
6. Mutual