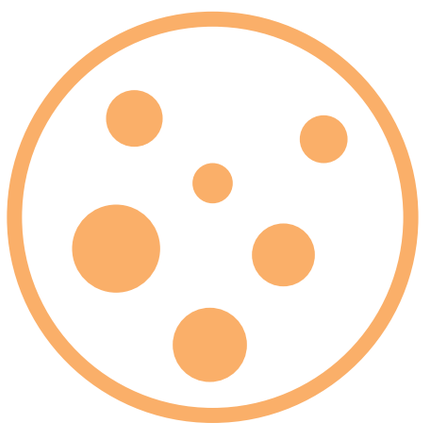
Authentication & Caching

Alex Vollmer
http://alexvollmer.com
@alexvollmer

pluralsight
hardcore dev and IT training

Authentication Mechanisms



Cookies

HTTP
Authentication

Custom
Headers

Cookies


Sign In
localhost:3000/account/new
Reader
Chat Rooms
localhost:3000/rooms
Reader
Bonjour
iTunes Affili...eative Tools
Instapaper
Tools
Reference
Send to OmniFocus
Sign Out
Chat Rooms
Here are the available chat rooms:
Xcode
Create a new chatroom
Display a menu for "http://localhost:3000/rooms/new"
POST /account
Cookie: …
Set-Cookie: …
ChatCave
Server

Cookies

An addition to the HTTP spec.

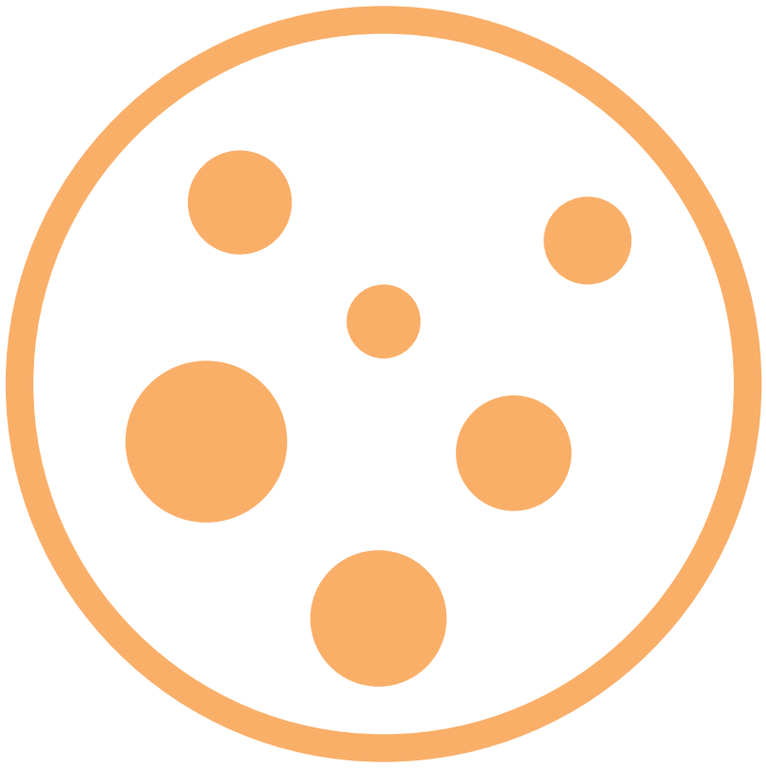
A “hack” for web browsers

Session vs. Persistent

NSHTTPCookie

NSHTTPCookieStorage

“It just works”…mostly…



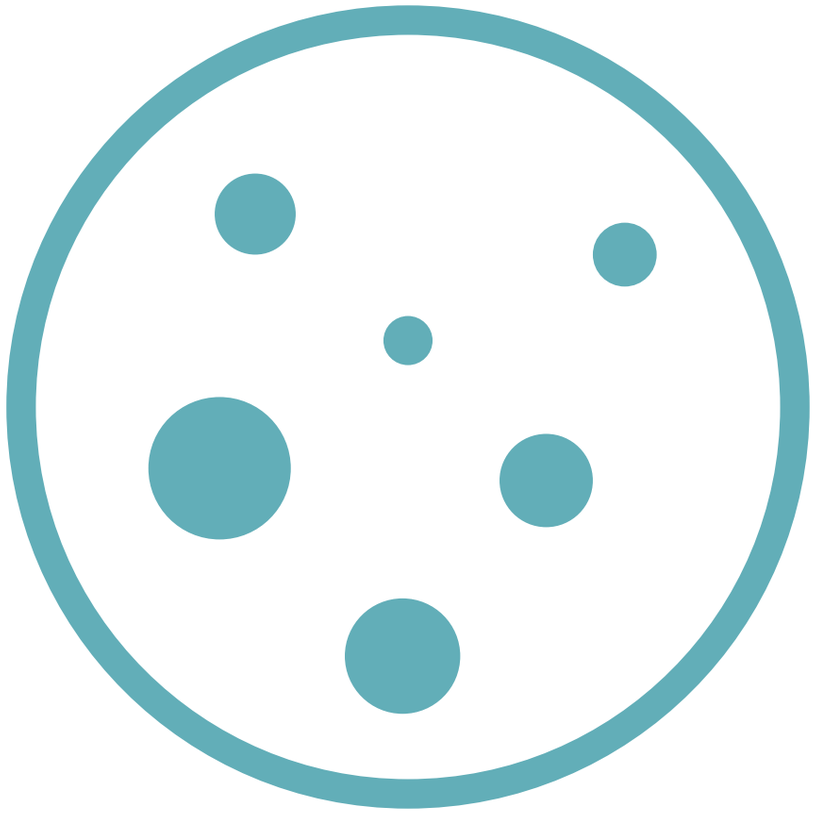
Cookie Acceptance Policy

```
/*!
    @method cookieAcceptPolicy
    @abstract Returns the cookie accept policy preference of the
    receiver.
    @result The cookie accept policy preference of the receiver.
*/
- (NSHTTPCookieAcceptPolicy)cookieAcceptPolicy;

/*!
    @method setCookieAcceptPolicy:
    @abstract Sets the cookie accept policy preference of the
    receiver.
    @param cookieAcceptPolicy The new cookie accept policy for the receiver.
*/
- (void)setCookieAcceptPolicy:(NSHTTPCookieAcceptPolicy)cookieAcceptPolicy;

/*!
    @enum NSHTTPCookieAcceptPolicy
    @abstract Values for the different cookie accept policies
    @constant NSHTTPCookieAcceptPolicyAlways Accept all cookies
    @constant NSHTTPCookieAcceptPolicyNever Reject all cookies
    @constant NSHTTPCookieAcceptPolicyOnlyFromMainDocumentDomain Accept cookies
    only from the main document domain
*/
typedef NS_ENUM(NSUInteger, NSHTTPCookieAcceptPolicy) {
    NSHTTPCookieAcceptPolicyAlways,
    NSHTTPCookieAcceptPolicyNever,
    NSHTTPCookieAcceptPolicyOnlyFromMainDocumentDomain
};
```

Cookies and Headers

```
/*!
    @method cookiesWithResponseHeaderFields:forURL:
    @abstract Return an array of cookies parsed from the specified
    response header fields and URL.
    @param headerFields The response header fields to check for
    cookies.
    @param URL The URL that the cookies came from - relevant to how the
    cookies are interpeted.
    @result An NSArray of NSHTTPCookie objects
    @discussion This method will ignore irrelevant header fields so
    you can pass a dictionary containing data other than cookie data.
*/
+ (NSArray *)cookiesWithResponseHeaderFields:(NSDictionary *)fields
                                      forURL:(NSURL *)URL;

/*!
    @method requestHeaderFieldsWithCookies:
    @abstract Return a dictionary of header fields that can be
    used to add the
    specified cookies to the request.
    @param cookies The cookies to turn into request headers.
    @result An NSDictionary where the keys are header field names,
    and the values
    are the corresponding header field values.
*/
+ (NSDictionary *)requestHeaderFieldsWithCookies:(NSArray *)cookies;
```

Cookie Gotchas



Acceptance policy is “sticky”

Session cookies are temporary

Be careful with custom persistence

HTTP Authentication

Defined in the HTTP spec.

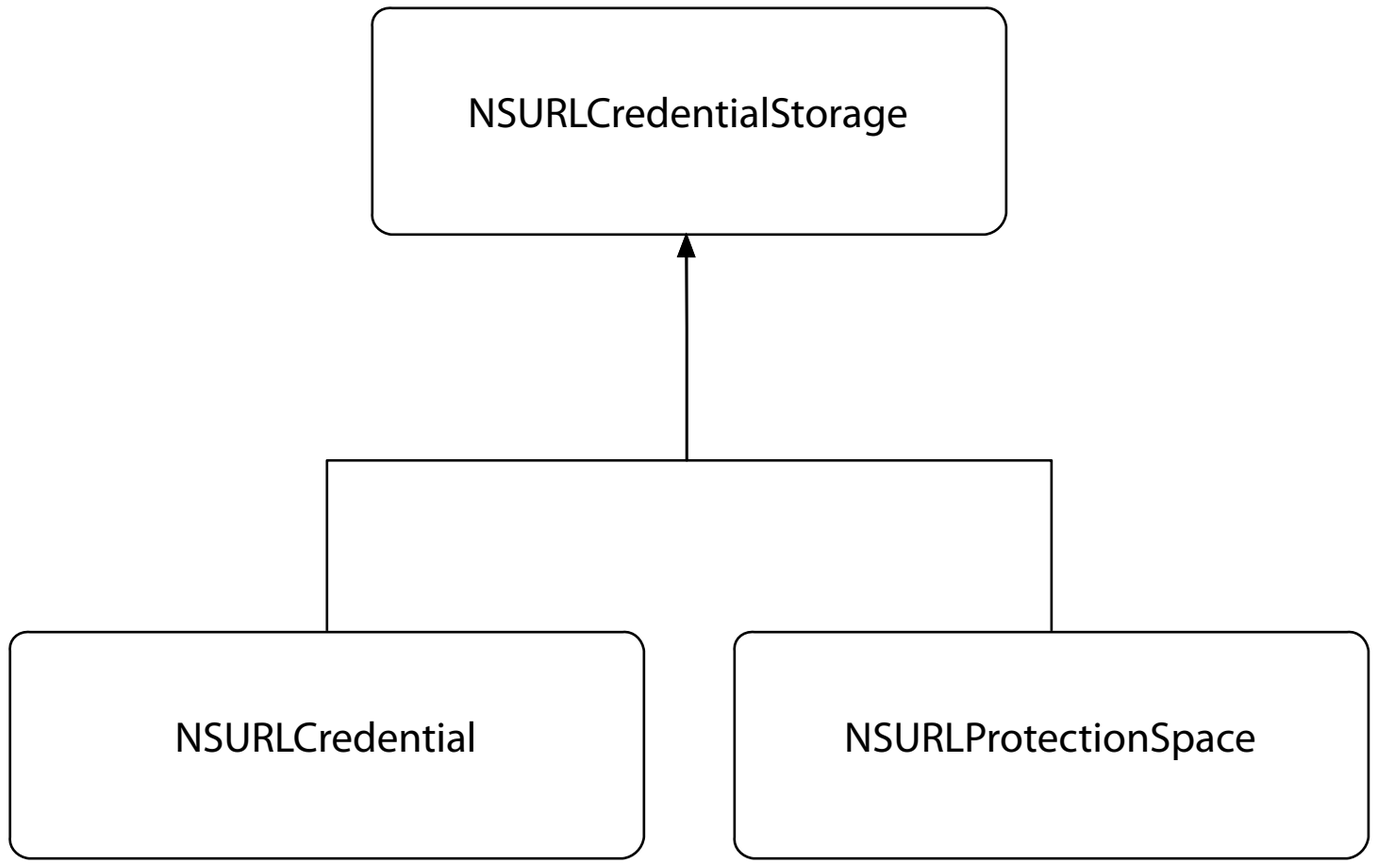
Two flavors: Basic & Digest

Basic is insecure *without* TLS

NSURLCredential

NSURLCredentialStorage

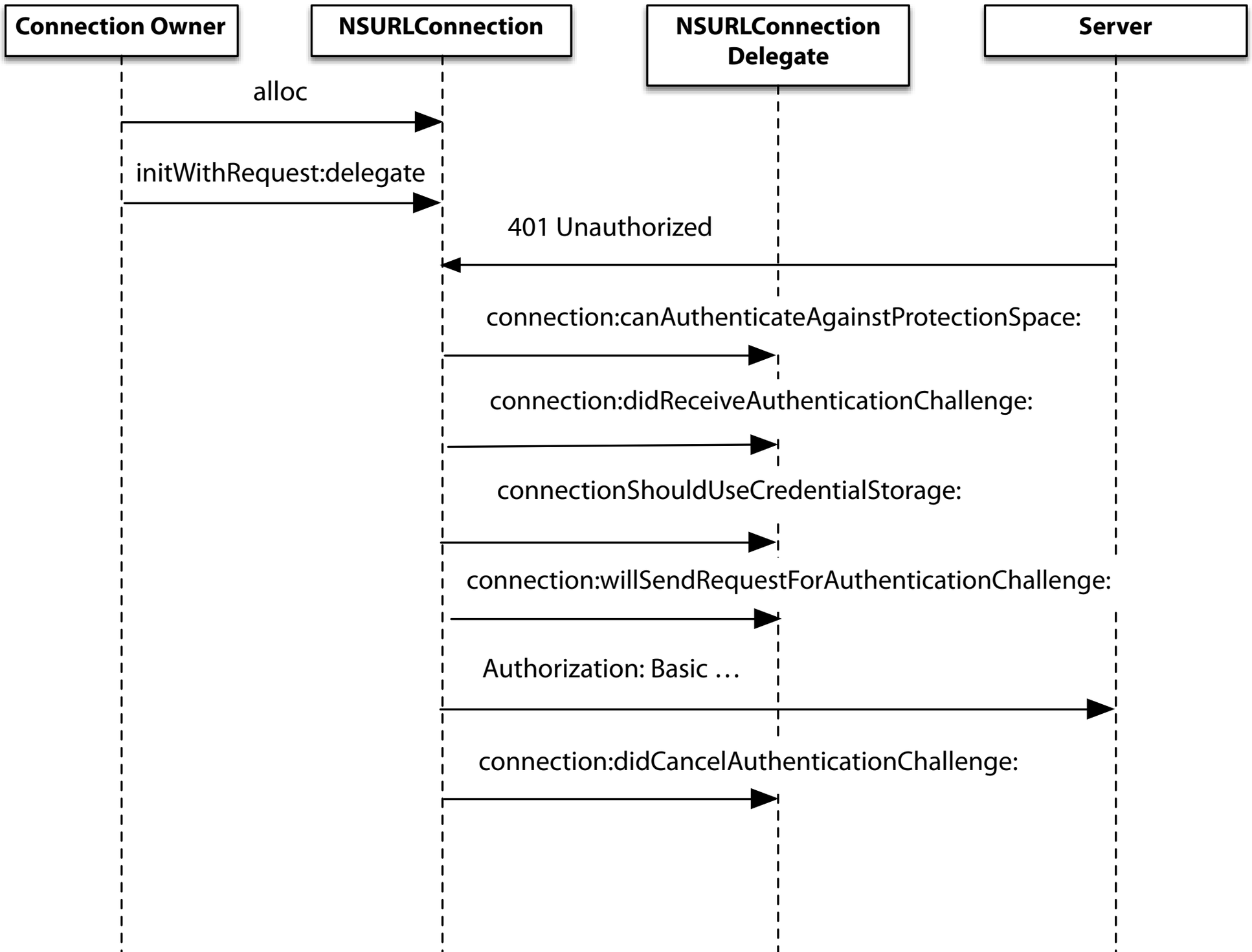
NSURLCredential & Friends



Authentication Methods

```
NSString * const NSURLAuthenticationMethodDefault;

NSString * const NSURLAuthenticationMethodHTTPBasic;

NSString * const NSURLAuthenticationMethodHTTPDigest;

NSString * const NSURLAuthenticationMethodHTMLForm;

NSString * const NSURLAuthenticationMethodNTLM;

NSString * const NSURLAuthenticationMethodNegotiate;

NSString * const NSURLAuthenticationMethodClientCertificate;

NSString * const NSURLAuthenticationMethodServerTrust;
```

Responding to Authentication Challenges



Custom HTTP Headers

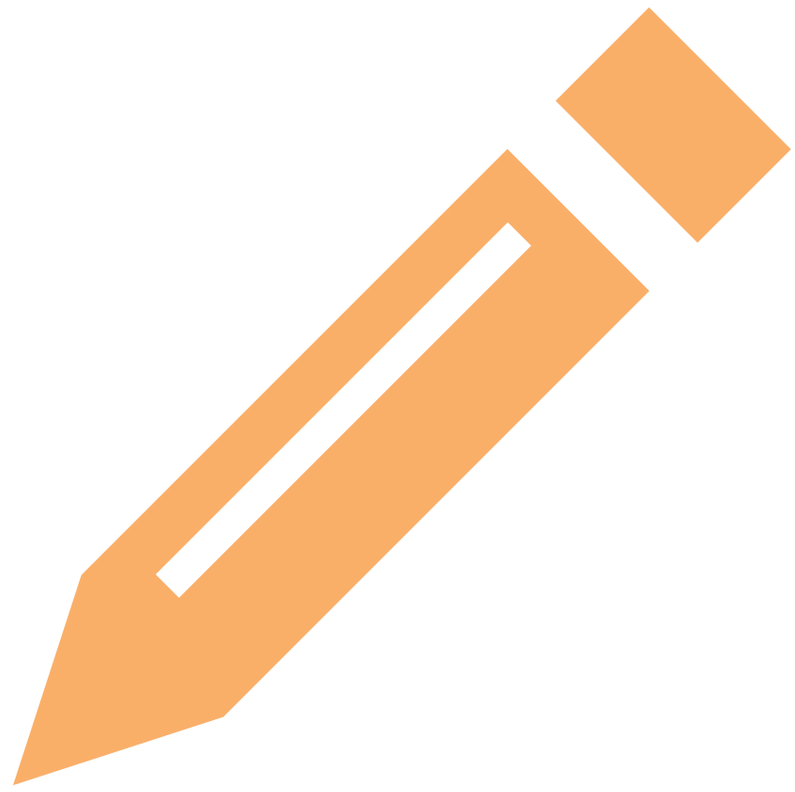
Custom authentication process

OAuth & xAuth are examples

App handles header writing

Use NSMutableURLRequest

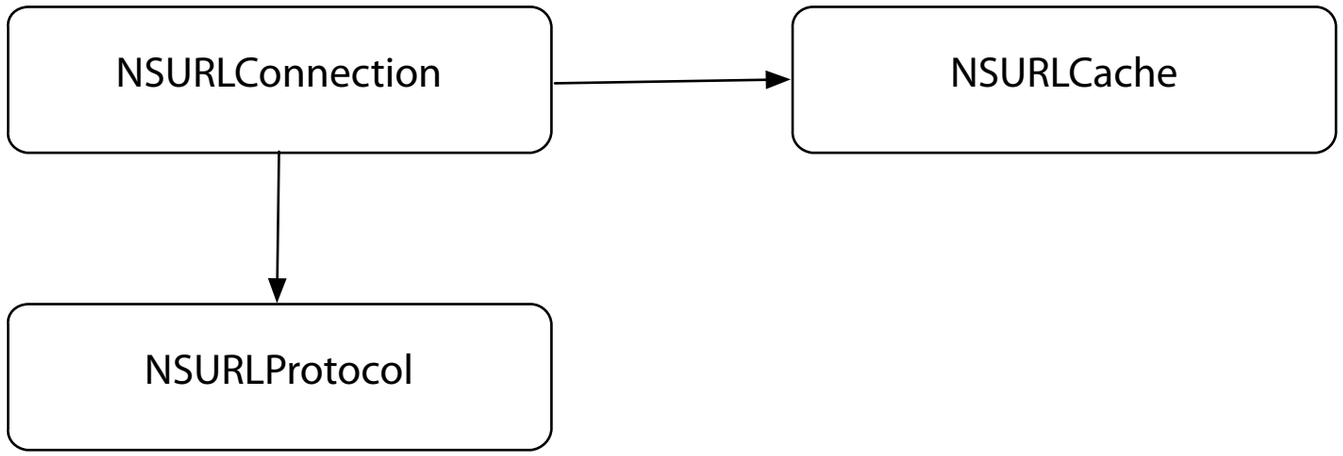
Use keychain to persist values



Caching

-initWithURL:cachePolicy:timeoutInterval:

+requestWithURL:cachePolicy:timeoutInterval:



Caching Goals

Prevent client from making extra requests

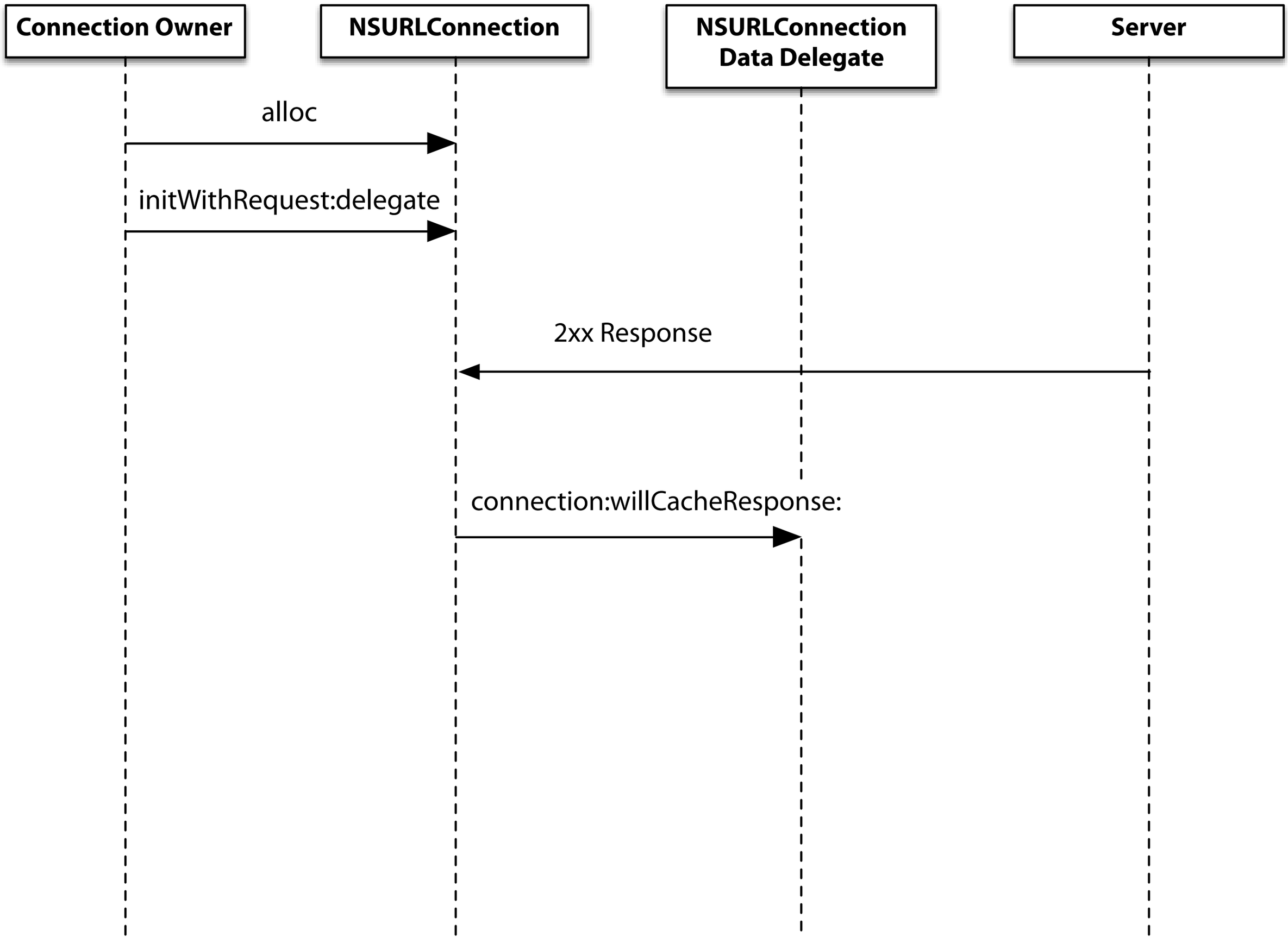
Typically uses date-based expiration

Diminish the payload of server responses

Typically uses ETags

Setting the Cache Policy

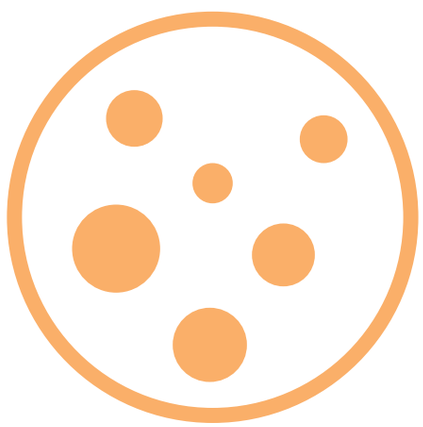
```
enum
{
    NSURLRequestUseProtocolCachePolicy = 0,

    NSURLRequestReloadIgnoringLocalCacheData = 1,
    NSURLRequestReloadIgnoringLocalAndRemoteCacheData = 4,
    NSURLRequestReloadIgnoringCacheData = NSURLRequestReloadIgnoringLocalCacheData,

    NSURLRequestReturnCacheDataElseLoad = 2,
    NSURLRequestReturnCacheDataDontLoad = 3,

    NSURLRequestReloadRevalidatingCacheData = 5,
};
typedef NSUInteger NSURLRequestCachePolicy;
```


Connection Owner
NSURLConnection
NSURLConnection Data Delegate
Server
alloc
initWithRequest:delegate
2xx Response
connection:willCacheResponse:

Accessing the Cache

```
/*!
    @method sharedURLCache
    @abstract Returns the shared NSURLCache instance.
    @discussion Unless set explicitly through a call to
    <tt>+setSharedURLCache:</tt>, this method returns an NSURLCache
    instance created with the following default values:
    <ul>
    <li>Memory capacity: 4 megabytes (4 * 1024 * 1024 bytes)
    <li>Disk capacity: 20 megabytes (20 * 1024 * 1024 bytes)
    <li>Disk path: <nobr>(user home directory)/Library/Caches/(application bundle id)</nobr>
    </ul>
    <p>Users who do not have special caching requirements or
    constraints should find the default shared cache instance
    acceptable. If this default shared cache instance is not
    acceptable, <tt>+setSharedURLCache:</tt> can be called to set a
    different NSURLCache instance to be returned from this method.
    @result the shared NSURLCache instance.
*/
+ (NSURLCache *)sharedURLCache;

/*!
    @method setSharedURLCache:
    @abstract Sets the NSURLCache instance shared by all clients of
    the current process. This will be the new object returned when
    calls to the <tt>sharedURLCache</tt> method are made.
    @discussion Callers should take care to ensure that this method is called
    at a time when no other caller has a reference to the previously-set shared
    URL cache. This is to prevent storing cache data from becoming
    unexpectedly unretrievable.
    @param cache the new shared NSURLCache instance.
*/
+ (void)setSharedURLCache:(NSURLCache *)cache;
```

Authentication Mechanisms



Cookies

HTTP
Authentication

Custom
Headers

Caching

-initWithURL:cachePolicy:timeoutInterval:

+requestWithURL:cachePolicy:timeoutInterval: