链滴

App开放接口api安全性—Token签名sign的设计与实现

作者：justdoit

原文链接：https://ld246.com/article/1465287920592

来源网站：链滴

<h2>前言</h2>
在app开放接口api的设计中，避免不了的就是安全性问题，因为大多数接口涉及到用户个人信息以及一些敏感的数据，所以对这些接口需要进行身份的认证，那么这就需要用户提供一些信，比如用户名密码等，但是为了安全起见让用户暴露的明文密码次数越少越好，我们一般在web项目，大多数采用保存的session中，然后在存一份到cookie中，来保持用户的回话有效性。但是在app供的开放接口中，后端服务器在用户登录后如何去验证和维护用户的登陆有效性呢，以下是参考项目设计的解决方案，其原理和大多数开放接口安全验证一样，如淘宝的开放接口token验证，微信开发台token验证都是同理。

<h2>签名设计</h2>
<h2>对于敏感的api接口，需使用https协议</h2>
https是在http超文本传输协议加入SSL层，它在网络间通信是加密的，所以需要加密证。

https协议需要ca证书，一般需要交费。

</
>
<h2>签名的设计</h2>
原理：用户登录后向服务器提供用户认证信息（如账户和密码），服务器认证完后给客端返回一个Token令牌，用户再次获取信息时，带上此令牌，如果令牌正取，则返回数据。对于获取Tken信息后，访问用户相关接口，客户端请求的url需要带上如下参数：

<span时间戳：timestamp

Token令牌：token

<spn>然后将所有用户请求的参数按照字母排序（包括timestamp，token），然后更具MD5加密（可加点盐），全部大写，生成sign签名，这就是所说的url签名算法。然后登陆后每次调用用户信息时，上sign，timestamp，token参数。

例如：原请求https://www.andycn/api/user/update/info.shtml?city=北京 （post和get都一样，对所有参数排序加密）<b />
加上时间戳和token

https://www.andy.cn/api/user/update/info.shml?city=北京×tamp=12445323134&token=wefkfjdskfjewfjkjfdfnc&nbp;
然后更具url参数生成sign

最终的请求如</pan>

https://www.andy.cn/api/user/update/info.shtmlcity=北京×tamp=12445323134&token=wefkfjdskfjewfjkjfdfnc&sign=FDK243JKJFD334FDF2
其最终的原理是减小明文的暴露次数保证数据安全的访问。

具体实现如下：

<spa>1. api请求客户端想服务器端一次发送用用户认证信息（用户名和密码），服务器端请求到改请求后验证用户信息是否正确。

如果正确：则返回一个唯一不重复的字符（一般为UUID），然后在Redis（任意缓存服务器）中维护Token----Uid的用户信息关系，以便其他pi对token的校验。

如果错误：则返回错误码。

2.服务器设计一个url请求拦截规则

（1）判是否包含timestamp，token，sign参数，如果不含有返回错误码。

2）判断服务器接到请求的时间和参数中的时间戳是否相差很长一段时间（时间自定义如半个小时）如果超过则说明该 url已经过期（如果url被盗，他改变了时间戳，但是会导致sign签名不相等）。</san>

（3）判断token是否有效，根据请求过来的token，查询redis缓存中的ui，如果获取不到这说明该token已过期。

（4）根据用户请求的url参，服务器端按照同样的规则生成sign签名，对比签名看是否相等，相等则放行。（自然url签名 也无法00%保证其安全，也可以通过公钥AES对数据和url加密，但这样如果无法确保公钥丢失，所以签名只很大程 度上保证安全）。

（5）此url拦截只需对获取身份认证的url行（如登陆url），剩余所有的url都需拦截。

3.Token和Uid关系维护/span>

对于用户登录我们需要创建token--uid的关系，用户退出时需要需删除oken--uid的关系。

<h2>签名实现</h2>
获取全部请求参数

[java] <a href="http://www.lai18.com/content/

44366.html" class="ViewSource" title="view plain">view plaincopy

<ol class="dp-j" start="1">
<li class="alt">String sign = request.getParameter(</spa>"sign");
   Enumeration<?>nbsp;pNames = request.getParameterNames(); </l>
<li class="alt">   Map<Strig, Object> params = new<spn> HashMap<String, Object>();
   <span class="keyword>while (pNames.hasMoreElements()) { </pan>
<li class="alt">   String pName = (String) pNames.nextElement();&bsp;
   if("sign".equals(pName))continue; &bsp;
<li class="alt">   Object pValue = request.getParameter(pName); &bsp;
   params.put(pName, pValue);
<li class="alt">   } &np;

生成签名

[java] view plaincopy

<ol class="dp-j" start="1">
<li class="alt">public static String createSign(Map<String, tring> params, boolean&nsp;encode)
   throws UnsupportedEncodingException nbsp;{
<li class="alt">   Set<Strin> keysSet = params.keySet();
   Object[] keys&nsp;= keysSet.toArray();
<li class="alt">   Arrays.sort(k

ys);
   StringBuffer tem
 = new StringBuffer(); &n
sp;
<li class="alt">   <span class
"keyword">boolean first = <span class="keywo
d">true;
   <span class="keyword
>for (Object key : keys) { </sp
n>
<li class="alt">   &nb
p; if (first) {

   &
bsp;   first = false<
pan>;
<li class="alt">   &nb
p; } else { <
span>
   &
bsp;   temp.append("&"<span
);
<li class="alt">   &nb
p; }
   &
bsp;temp.append(key).append("="); </s
an>
<li class="alt">   &nb
p; Object value = params.get(key); </l
>
   &
bsp;String valueString = "";
nbsp;
<li class="alt">   &nb
p; if (<span class="ke
word">null != value) { </l
>
   &
bsp;   valueString = String.valueOf(value); &nb
p;
<li class="alt">   &nb
p; }
   &
bsp;if (encode) { </span

<li class="alt">   &nb
p;   temp.append(URLEncoder.encode(valueString,
nbsp;"UTF-8"));
   &
bsp;} else { </sp
n>
<li class="alt">   &nb
p;   temp.append(valueString); </spa
>

   }
<li class="alt">   }

<li class="alt">   <span class "keyword">return MD5Utils.getMD5(temp.toString()).toUpperCase(); nbsp;
   }
