## 项目名称

## 项目内容

项目主要实现的功能有哪些：

测试profile:xxxxx

测试环境： desktop only

## 测试范围(API list)

本项目所涉及的新增或修改的API有：

## 测试用例：

截图能体现测试项是否符合要求即可

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 检测领域 | 检测要求 | 检测方法 | 是否符合 | 证明截图 |
| 身份认证 | 不能使用Authorization Basic 认证 | 检测接口是否使用Basic Auth作为接口的认证方式，若有，则可以通过base64 decode对认证的账号密码进行破解； | N/A |  |
| 接口开启登录失败锁定功能 | 检测接口是否开启登录失败锁定功能 | N/A |  |
| 账号密码是否明文 | 检测账号密码是否明文 | N/A |  |
| JWT Token 的使用 | 不能使用对称加密算法进行签名生成jwt token | 把jwt token拿到<https://jwt.io>进行解密，检查是否使用了对称密钥，如HS256 | N/A |  |
| token 的过期时间尽可能合理 | 解密token后，查看'iat'和'exp'这两个时间戳间隔是否合理； | N/A |  |
| 不要在 JWT 的请求体中存放敏感数据，如账号密码 | 解密token后，查看里面是否存在敏感信息如用户明文密码、token的解密密码等 | N/A |  |
| 检测服务器是否支持Token加密算法为None | 具体测试方法步骤参考  <https://www.cnblogs.com/xiaozi/p/12031111.html> | N/A |  |
| 密钥混淆攻击 | N/A |  |
| 无效签名攻击 | N/A |  |
| 操纵KID |  |  |
| 访问控制 | 水平越权测试 | 交换cookie测试水平越权 |  |  |
| 垂直越权测试 | 若有多种角色使用该API，交换cookie测试 |  |  |
| 测试是否限制流量来防止 DDoS 攻击和暴力攻击. | 1：扫描API的时候使用其它profile来检测是否能正常访问这条API；一般API做到这就行，若扫描过程中，API响应缓慢，则要求project team提供具体的API rate limit.  2：对于一些重要的API，使用Turbo Intruder(Burp的一个插件)来进行暴力攻击；  3：若API存在服务器响应大小可控时(如/api/users?page=1&size=100)，可把size变大并使用工具进行快速重放请求，同时检查服务器即时的响应情况； |  |  |
| 测试是否接口是否支持http协议 | 查看是否使用https协议；同时检测服务器是否同时支持使用http协议。 |  |  |
| 使用 HSTS 协议防止 SSLStrip 攻击. | 检测服务器的response中是否含有Strict-Transport-Security字段，建议Strict-Transport-Security: max-age=86400; includeSubDomains，表示为当前域名和所有的子域名启用HSTS。 |  |  |
| 是否有对能够访问API接口的地址范围进行限制 | 若api是公开，则该项不适用；若非公开或内网内有作IP地址范围限制的话，可尝试绕过，如在请求头上增加X-FORWARD-FOR字段模拟合法的IP来源； |  |  |
| 破坏功能级的授权和访问控制 | 如正常的请求是GET /api/v1/users/<id> 将其修改为 DELETE / POST 去尝试删除或创建用户。  如POST /api/reset\_pass 接口调用时需要验证旧密码，而PUT /api/update\_user 接口调用可能就不需要旧密码。 |  |  |
| 授权绕过技巧 | 数组绕过：{“id”:111} --> {“id”:[111]}JSON  对象绕过：{“id”:111} --> {“id”:{“id”:111}}  两次传值绕过：/api?id=非法参数值&id=恶意参数值  正则匹配绕过：{"user\_id":"\*"} |  |  |
| 不安全url跳转 | 修改跳转的路径，如修改为百度的地址，修改后回车或输入账号密码后按回车，查看页面是否跳转到百度的页面，若是，则该项不符合。 |  |  |
| 输入 | 检测http请求操作的方法是否合理 | 修改请求方式为OPTIONS,查看服务器的返回包中的methods allow字段来确定服务器支持的请求方式，检查是否存在非必要的请求方式； |  |  |
| 检测服务端是否对输入的内容格式进行验证 | 修改request提交的内容格式，如由json格式改为XML格式(使用burp插件Content Type Conerter)，测试服务端是否接受新的格式，服务器返回error才算符合 |  |  |
| 不要在 URL 中使用任何敏感的数据 | 查看敏感数据(credentials, Passwords, security tokens, or API keys)是否使用post方式进行提交，若使用或者支持请求以GET方式提交敏感数据时，则存在安全风险。因为日志把请求的URL记录下来。 |  |  |
| 对于敏感或交易类等请求，应防止请求的重放攻击。 | 若API属于交易类、转账类请求，则需要对请求重放进行限制。 |  |  |
| 处理 | 检查是否所有的API请求处理都在身份认证之后 | 把cookie或token删除后，检查是否还能正常访问接口数据； |  |  |
| 避免使用特有的资源 id. | 如使用 /me/orders 替代 /user/654321/orders，建议不要在URL中使用特定资源 |  |  |
| 具体标识用户的ID需要有一定的随机性 | 检查与用户身份有绑定的ID如applyNumber等是否有足够的随机性来防止被暴力枚举。 |  |  |
| 检查报错是否合理 | 破坏请求中的数据结构，检查报错信息是否合理，是否包含敏感组件信息等。 |  |  |
| 输出 | Response有设置X-Content-Type-Options | X-Content-Type-Options设置为nosniff |  |  |
| Response有设置X-Frame-Options | X-Frame-Options设置为deny或SAMEORIGIN，若无设置，则其它域名下可以通过<iframe> tag进行调用。 |  |  |
| 删除指纹头信息 | 检查服务器的响应头是否含有敏感的服务器组件及版本信息，如X-Powered-By,X-AspNet-Version 等等,一般以泄漏应用组件为主，apache这些不算，但版本信息一定不可以有。 |  |  |
| 不要返回敏感的数据 | 不要返回敏感的数据, 如 credentials, Passwords, security tokens.即检查服务器是否返回非必要且敏感数据；  检查是否返回账号的相关信息，如account number、账户名，若有，检查是否对这些参数进行mask处理。 |  |  |

备注：由于OAuth 授权或认证协议使用并不多，因此不在此表体现；部分安全问题可由Burpsuite Scan完成则不在此表体现。

## 发现的问题用例情况说明：

下面用于说明具体漏洞问题的情况

|  |  |
| --- | --- |
| 漏洞描述 |  |
| 风险等级 |  |
| 所在URL |  |
| 漏洞验证 |  |
| 修复后截图 |  |

|  |  |
| --- | --- |
| 漏洞描述 |  |
| 风险等级 |  |
| 所在URL |  |
| 漏洞验证 |  |
| 修复后截图 |  |

## 附件(扫描报告)

对本项目涉及的API进行active scan, 报告直接从burp导出html格式，无论是否误报，直接导出。有问题项验证后通过截图形式体现到“发现的问题用例情况说明”。