**EdgeGallery R1.0渗透测试报告**

**Sprint 1：**

**Story：**[need to add the role of GUEST for all of GET APIs in Developer](https://gitee.com/OSDT/dashboard?issue_id=I24IMF)

**测试用例：EG-TST-SEC-007** 越权

**测试说明**：

为所有的GET型API添加了Guest用户：目前项目仅有2类权限用户：GUEST用户和普通用户，测试内容为判断GUEST用户能否访问普通用户数据。

**测试过程**：

**Developer**：

①新建项目：

尝试新建项目，点击“下一步”按钮时提示“GUEST用户无权操作，请先登录！”，然后“下一步”按钮变为不可点击；通过。

②迁移项目：

尝试迁移项目，点击“下一步”按钮时提示“GUEST用户无权操作，请先登录！”，然后“下一步”按钮变为不可点击；通过。

③上传插件：

尝试上传插件，点击“立即上传”按钮时提示“GUEST用户无权操作，请先登录！”；通过。

④下载插件：

尝试下载已上传的插件，点击“详情”，可以看到详情；点击“下载”，可以正常下载；下载完毕后进行评分，提示“GUEST用户无权操作，请先登录！”；通过。

④工具链：

尝试上传xxx.tar.gz，点击“分析”按钮后，生成历史报告，并可下载；不通过。

⑤测试应用：

尝试上传.csar文件，点击“立即上传”按钮时提示“GUEST用户无权操作，请先登录！”；通过。

**测试结论**：**Not PASS**

**不通过原因：GUEST用户可以通过“工具链”功能中POST接口上传文件。**

**Story：**[need to add the role of GUEST for all of GET APIs in AppStore](https://gitee.com/OSDT/dashboard?issue_id=I24IKL)

**测试用例：EG-TST-SEC-007** 越权

**测试说明**：

为所有的GET型API添加了Guest用户：目前项目仅有2类权限用户：GUEST用户和普通用户，测试内容为判断GUEST用户能否访问普通用户数据。

**测试过程**：

**AppStore**：

①App评论区：

尝试提交评论：提示“GUEST用户无权操作！请登录”；

②上传文件：

尝试上传csar文件，“检测”按钮禁止点击；点击“确认”按钮后，提示“GUEST用户无权操作！请登录”；

**测试结论**：**PASS**

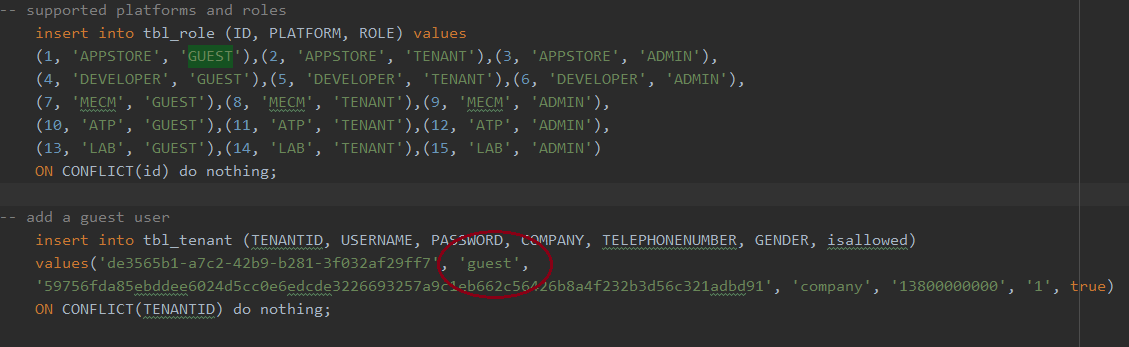
Story：[用户管理模块中需要支持用户的角色定义](https://gitee.com/OSDT/dashboard?issue_id=I23FRE)

**测试用例：EG-TST-SEC-007** 越权

**测试说明**：

为所有的GET型API添加了Guest用户：目前项目仅有2类权限用户：GUEST用户和普通用户，测试内容为判断GUEST用户能否访问普通用户数据。

**测试过程**：



可以看到，已添加guest用户。

**测试结论**：**PASS**

Story：[用户管理支持guest访客用户，在界面上只有只读权限](https://gitee.com/OSDT/dashboard?issue_id=I23FRC)

测试说明：

此条story已分解为3条story：

[need to add the role of GUEST for all of GET APIs in Developer](https://gitee.com/OSDT/dashboard?issue_id=I24IMF)

[need to add the role of GUEST for all of GET APIs in AppStore](https://gitee.com/OSDT/dashboard?issue_id=I24IKL)

[need to add the role of GUEST for all of GET APIs in MECM](https://gitee.com/OSDT/dashboard?issue_id=I24IKE)

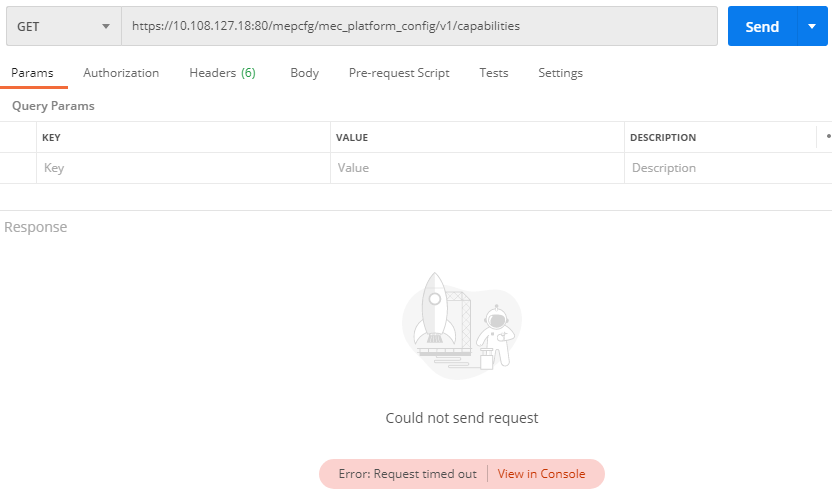
Story：[【MEP】MEP Support to query the capabilities.](https://gitee.com/OSDT/dashboard?issue_id=I1W7UU)

**测试用例：EG-TST-SEC-005 SQL注入**

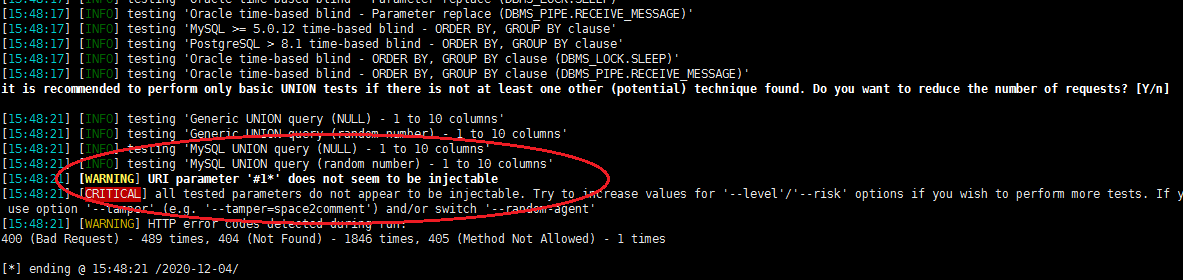
**测试过程：**

Mep的查询功能信息由Mecm使用，以在其门户上显示。Mecm通过mm5接口将功能查询发送到mep。经过确认，mep的查询接口为内部接口，不能由用户直接调用或访问，所以外部攻击者也无法访问该接口，不会受到外部攻击的威胁。同时使用sqlmap进行辅助测试，未发现可利用的注入点。

通过外部无法访问mep查询接口。



使用sqlmap尝试对该接口进行注入，未发现注入点。



**测试结论**：**PASS**

**Sprint 2：**

**Story：**[【MECM 】MECM Support to query the capabilities](https://gitee.com/OSDT/dashboard?issue_id=I1Z3CN)

**测试用例：EG-TST-SEC-007** 越权 **EG-TST-SEC-005 SQL注入**

**测试说明**：

对查询时是否进行了权限鉴别进行判断，防止非授权查询；对查询参数进行SQL注入测试。

**测试过程**：

1.越权测试

1.1调用/lcmcontroller/v1/tenants/{tenantId}/hosts/{hostIp}/mep\_capabilities

查询接口

返回正常

删除身份验证信息access\_token，再次发送，返回401错误，显示"Authorization failed"。

1.2调用/appo/v1/tenants/{tenant\_id}/hosts/{host\_ip}/mep\_capabilities查询接口

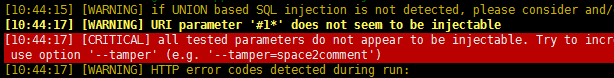
删除身份验证信息access\_token，再次发送，返回401错误，显示"error": "Unauthorized"

。

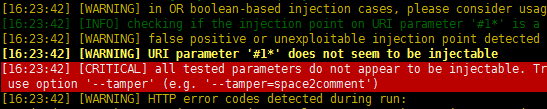
1. SQL注入测试

使用sqlmap进行SQL注入测试，未发现可利用的注入点。

2.1针对/lcmcontroller/v1/tenants/{tenantId}/hosts/{hostIp}/mep\_capabilities接口：

****

**2.2 针对**/appo/v1/tenants/{tenant\_id}/hosts/{host\_ip}/mep\_capabilities接口：



**测试结论**：**PASS**

**Story：**[【MECM】MECM Support Batch Distribution and Batch Deployment](https://gitee.com/OSDT/dashboard?issue_id=I249AF)

**测试用例：EG-TST-SEC-007** 越权

**测试说明**：

对查询时是否进行了权限鉴别进行判断，防止非授权调用接口；

**测试过程**：

①正常的批量分发部署过程，OK

②尝试修改UserID，观察能否以其他用户身份进行批量分发部署操作。

均提示401错误，权限鉴别失败。

**测试结论**：**PASS**

**Story：**[【MECM】提供ak/sk分配功能](https://gitee.com/OSDT/dashboard?issue_id=I1R5LY)

**测试用例：EG-TST-SEC-007** 越权

**测试说明**：

对接口调用时是否进行了权限鉴别进行判断，防止非授权调用接口；

**测试过程**：

①正常的获取access\_token过程，OK

②修改MECMSESSIONID为另一个用户的MECMSESSIONID，观察能否成功获取access\_token。结果无法获取access\_token。

③观察正常获取access\_token时所用的MECMSESSIONID失效后，还能否成功获取access\_token。结果无法获取access\_token。

**测试结论**：**PASS**

**Story：**[mepagent call mepserver service liveness api to keep heartbeat for app service](https://gitee.com/OSDT/dashboard?issue_id=I1R0VX)

[mepserver add app heartbeat API, referring to etsi mec 011 latest stable draft](https://gitee.com/OSDT/dashboard?issue_id=I1QT6E)

**测试用例：EG-TST-SEC-007**越权

**测试过程**：

该接口为MEP内部接口，没有面对公网的ip和端口，无法从外部直接调用，所以不会遭受越权类攻击。

**测试结论**：**PASS**

**Story：**【ATP】创建任务、查询任务、测试报告下载、获取依赖应用、查询测试用例接口开发

**测试用例：EG-TST-SEC-007**越权 **EG-TST-SEC-004文件上传**

**EG-TST-SEC-005 SQL注入**

**测试说明**：

对创建任务、查询任务、测试报告下载、获取依赖应用、查询测试等功能是否进行了权限鉴别进行判断，防止非授权调用接口；

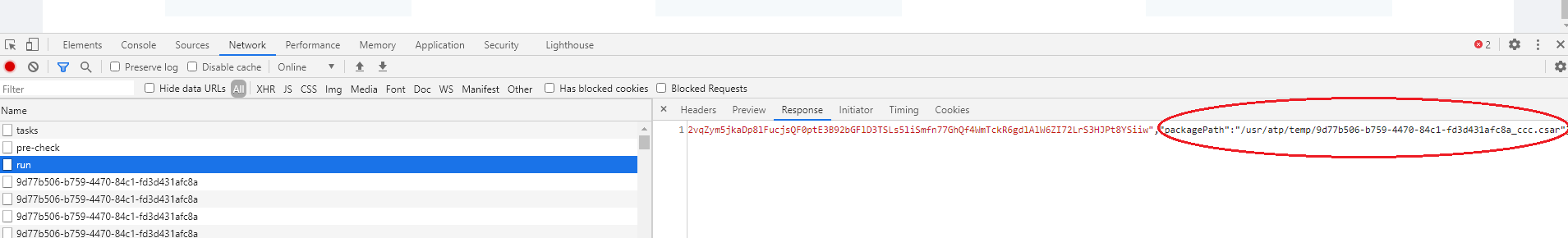
对文件上传功能进行测试，防止服务器被上传木马。

**测试过程**：

**1.EG-TST-SEC-004文件上传测试：**

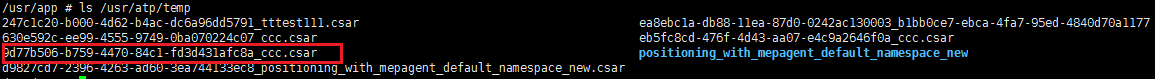
①正常的批量分发部署过程，OK

②进行正常文件上传操作，查看文件上传响应。



看到服务器返回了所上传文件的绝对存储路径，以及随机重命名后的文件名。

登录ATP后台，kubectl exec -it atp-0 -- sh



可以看到刚才所上传的文件路径和文件名无误。

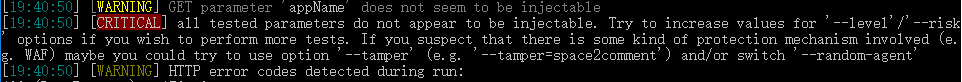
另外抓取上传请求，放入Burp-Repeater模块，修改文件后缀格式后重新发送，可以看到400错误，说明后端进行了文件格式校验，ok。

**2.EG-TST-SEC-005 SQL注入测试**

**查询功能url格式:30094/mec-atp/edgegallery/atp/v1/tasks?appName=&status=**

**使用sqlmap进行测试：**

**未发现可利用注入点**

****

**3.EG-TST-SEC-007越权测试：**

①正常的文件过程，请求包中无类似userID参数，据分析应该是通过cookie鉴别用户身份。尝试将ATPSESSIONID删除，观察能否成功执行。

测试过程：

删除ATPSESSIONID后，无法继续进行应用测试。Ok

**测试结论**：**Not PASS**

**不通过原因：上传.csar文件时，服务器返回了包含文件名的完整的存储绝对路径。**

**Sprint 3：**

**Story：**[【AppStore】appstore增加对atp模块的适配](https://gitee.com/OSDT/dashboard?issue_id=I27X1V)

**测试用例：**EG-TST-SEC-004文件上传

**测试过程**：

[https://x.x.x.x](https://159.138.139.166:30091" \t "https://gitee.com/OSDT/_blank)

①登录账户，上传.csar文件



进入AppStore后台，找到刚才上传的文件，可以看到.csar文件和.jpg文件均未进行随机重命名

②/mec-appstore/mec/appstore/v1/apps/164e37d257564016977a51f3c9e0b79a/packages?userId=364ec623-ec89-481c-92c4-22837b3d4151  
该接口返回了上传文件存储的绝对路径

抓取上传文件的http请求，可以看到返回了绝对路径。

**测试结论**：**Not PASS**

**不通过原因：上传.csar文件时，应用并未对上传文件进行随机重命名，并返回了绝对路径。**

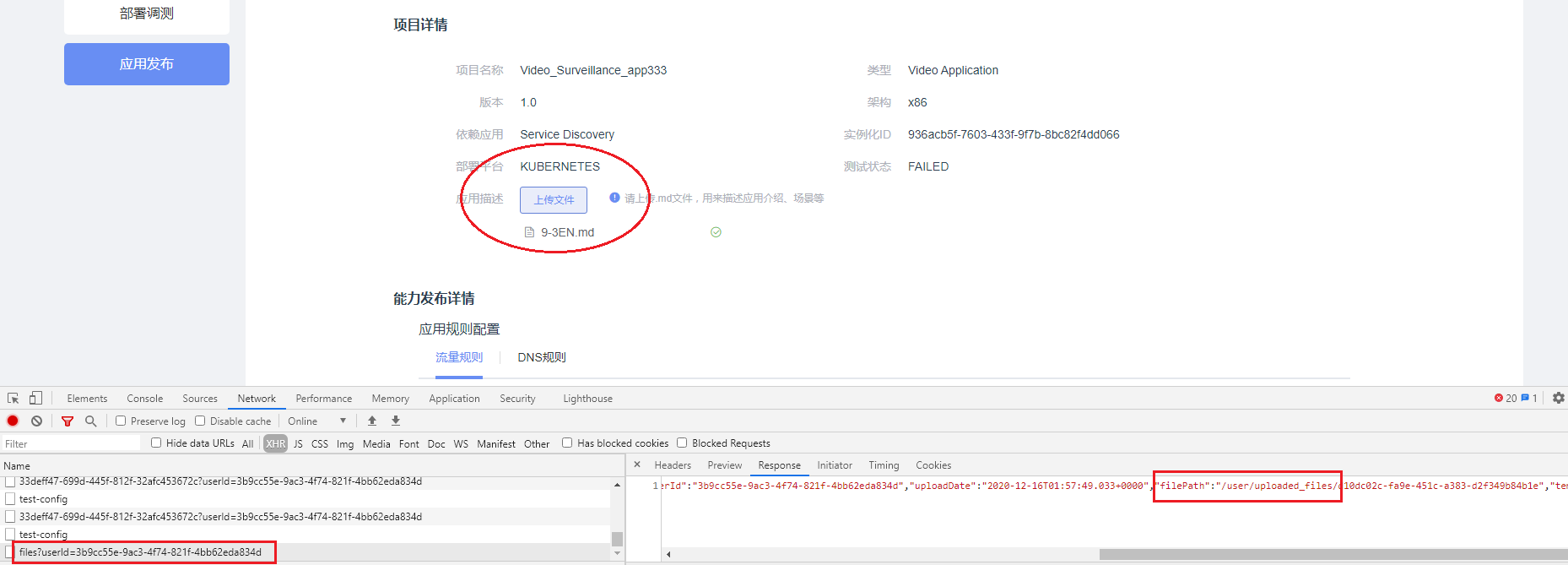
Story：[【develoepr】应用发布界面，发布到appstore接口实现](https://gitee.com/OSDT/dashboard?issue_id=I27ETO)

**测试用例：**EG-TST-SEC-004文件上传 EG-TST-SEC-007越权

**测试过程**：

###问题描述  
/mec-developer/mec/developer/v1/files?userId=3b9cc55e-9ac3-4f74-821f-4bb62eda834d  
该接口不应返回上传的绝对路径：

登录账户，正常创建项目、部署调测，来到应用发布-项目详情界面，上传.md文件，可以看到返回信息中包含了上传的绝对路径。



**测试结论**：**Not PASS**

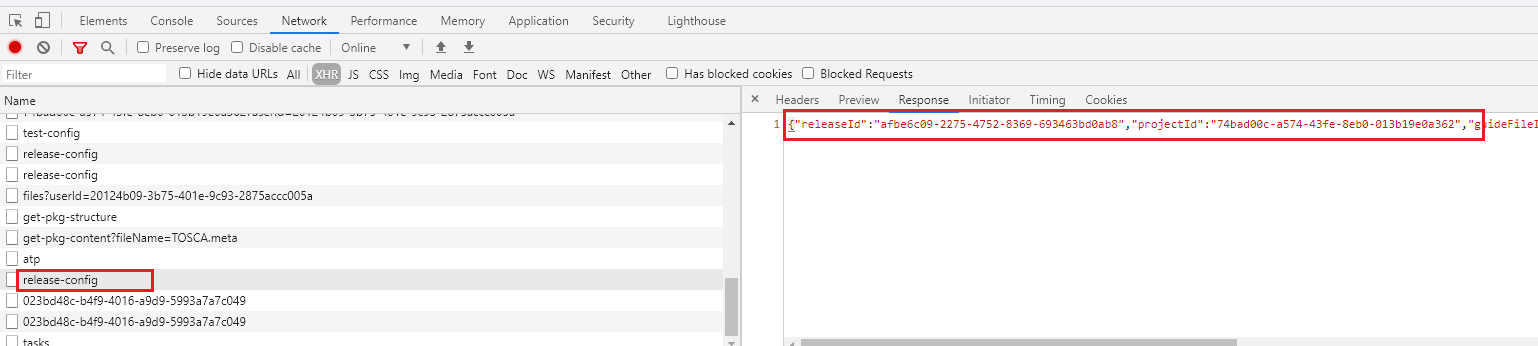
**不通过原因：上传.md文件时，应用返回了绝对路径。**

Story：[【developer】应用发布界面，release-config接口的实现](https://gitee.com/OSDT/dashboard?issue_id=I27ET4)

**测试用例：** EG-TST-SEC-007越权

**测试过程**：

###漏洞详情  
Developer release-config接口存在横向越权漏洞  
如果将url中的 {projectId} 从A用户替换为B用户，可以成功获取B用户的项目信息。  
###操作步骤  
①登录A用户，正常调用release-config接口，如图，可以看到如下项目信息。  
"releaseId":"afbe6c09-2275-4752-8369-693463bd0ab8",  
"projectId":"74bad00c-a574-43fe-8eb0-013b19e0a362".  
"appName":"videosurveillanceapp222"



②登录B用户，也正常调用release-config接口，如图

③此时如果将B用户的release-config接口url中projectId替换为A用户的projectId，可以看到应用做出了正常响应，并获取到了A用户的项目信息。以此推断应用在执行查询时可能未判断projectId是否属于该用户。

**测试结论**：**Not PASS**

**不通过原因：该接口存在横向越权漏洞。**

Story：[【MECM】MECM Support APP Rule Config During APP Orchestration](https://gitee.com/OSDT/dashboard?issue_id=I246ZD)

**测试用例：** EG-TST-SEC-007越权

**测试过程**：

尝试将app-rule相关接口中userID从A替换为B，查看能否正确响应

返回401，ok

**测试结论**：**PASS**

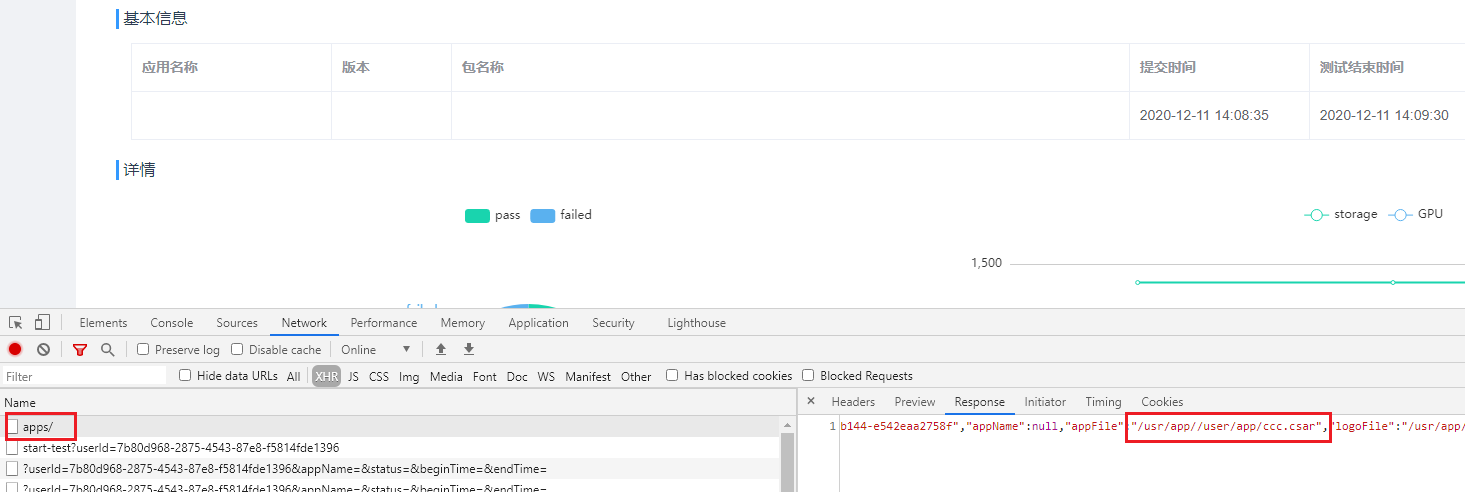
Story：[【Developer】应用发布界面，ATP测试接口实现](https://gitee.com/OSDT/dashboard?issue_id=I27ERS)

**测试用例：** EG-TST-SEC-007越权 EG-TST-SEC-004文件上传

**测试过程**：

/mec-developer/mec/developer/v1/apps/  
应用测试功能返回了所上传文件的绝对存储路径。

①登录账户，在“测试应用”功能中上传.csar文件。可以看到http响应信息中出现文件存储的绝对路径。



②登录Dev后台，可以看到该路径无误。

Developer应用测试功能，没有对上传的.csar文件进行随机重命名，这可能造成以下2方面问题：  
①可能帮助恶意攻击者成功解析木马文件；  
②如果2次上传的.csar文件名相同，后面的文件会将之前上传的文件覆盖掉，这可能造成一些业务逻辑上的混乱。  
###重现步骤  
①登录账户，上传.csar文件。  
②进入Dev后台，找到刚才上传的文件，可以看到图标文件进行了随机重命名，但是.csar文件未进行随机重命名。

③先上传.csar文件，间隔一段时间再上传文件名相同的.csar文件，可以看到先上传的文件被覆盖。

**测试结论**：**Not PASS**

**不通过原因：测试功能中，不应返回上传的绝对路径，并应该对上传文件进行随机重命名。**

Story：[【MECM】MECM Support APP Rule Config During APP Orchestration](https://gitee.com/OSDT/dashboard?issue_id=I246ZD)

**测试用例：** EG-TST-SEC-007越权

**测试过程**：

尝试将userID从A用户修改为B用户，观察能否正确响应。

响应401权限错误。Ok

**测试结论**：**PASS**

**Story：**[【MECM】MECM支持限制活动的并发会话数](https://gitee.com/OSDT/dashboard?issue_id=I1TH8K)

**测试用例：**

**使用slowhttptest建立200个以上的活动连接，查看活动连接数是否存在上限。**

**测试过程**：

命令：slowhttptest -u “https://https://x.x.x.x:30093” -B -c 500

可以看到，连接数增加到200时，应用拒绝服务，说明应用限制的活动的并发连接数。

**测试结论**：**PASS**

Story：[【MECM】MECM支持权限处理，避免横向越权](https://gitee.com/OSDT/dashboard?issue_id=I1TH3T)

**测试用例：** EG-TST-SEC-007越权

**测试过程**：

访问mecm，登录账户，对注册applcm、注册边缘node、分发、上传配置文件、部署等接口分别进行测试，尝试将用户A的userID替换为用户B的userID，观察能否得到正确响应。最终可以看到，应用各个接口均进行了权限鉴别，横向越权尝试失败。

均报401权限错误，ok

**测试结论**：**PASS**