**控制器应用程序访问权限管理系统AccessPermissionMS**

**一 主要需求分析：**

在SDN控制层和应用层的安全交互方面，北向接口的安全性扮演者重要角色。目前，各类控制器并没有为北向应用程序提供标准化的安全接口。攻击者可以利用北向接口的安全漏洞直接向控制器发起攻击，从而使得SDN控制器对应用程序的认证和权限管理变得尤为重要。应用程序通过北向接口与控制器进行通信主要包括两个方面：读取网络状态（Reading Network State）和写入网络策略（Writing Network Policies）。所面临的基本的安全问题是恶意应用程序访问网络状态信息和操纵网络流量，以破坏网络的正常状态。通过对控制器上应用程序的访问权限分析，针对使用北向API与控制器进行通信的应用，设计一种细粒度的访问权限管理系统，主要目的有两方面：一是为了防止未经过身份认证的应用对控制器的访问尝试；二是对经过认证的应用进行权限检查，确保应用只能访问具有访问权限的控制器资源，对权限范围外的控制器资源的试图访问予以拒绝，从而达到对应用程序访问权限进行更细粒度的认证和管理。因此，本系统实现的功能需求如下：

1 网络管理员能够对应用程序进行注册，在注册时除了进行身份信息的记录外，还可以对应用程序的访问权限在初始化时按需求进行分配。

2 当应用程序试图访问控制器资源时，对其进行身份认证，若验证通过则可以继续访问，否则对访问请求予以阻止，并写入系统异常行为监测日志中。

3 当应用程序通过身份认证后，启用权限集合中的访问权限时，对其试图使用的权限进行检查，若满足权限集合的要求则请求可顺利通过，否则将请求信息视为无效请求，并写入系统异常行为监测日志中。在该应用程序的生命周期内，AccessPermissionMS严格按照应用程序访问权限集合对应用程序的每个访问请求进行检验，通过这种细粒度的访问权限管理方法，实现了对应用程序访问行为的监测。

4 可以根据系统需要，动态的调整（增加/移除）某个应用程序的访问权限集合，也可以修改应用程序的某些身份信息，使系统具有较高的灵活性。

5 可以查询应用程序的身份信息和其所具有的访问权限

6 将所有未经授权的应用程序访问请求记录到日志文件以进行审计。

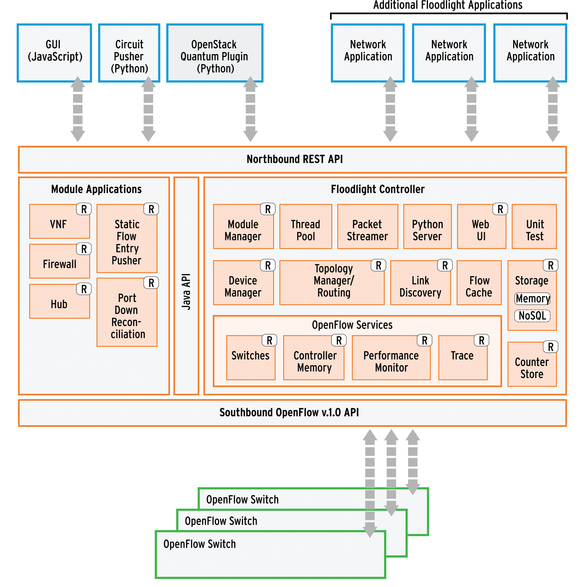


图1：Floodlight架构图 （针对最底层的API进行访问权限管理）

**二 系统总体框架图：**

XACML

访问控制

应用身份

信息管理

身份认证

应用权限管理

权限检查

内存数据库

日志记录

前端页面

**系统具有七个属性：**

1 定义了一组完整的权限集合。

针对控制器上应用程序访问的资源进行权限定义，如图所示，这些权限反映了OpenFlow相关命令，包含了开发人员开发SDN应用时所有与OpenFlow相关的任务，用于应用程序读取网络状态（Reading Network State）和写入网络策略（Writing Network Policies）使用。

table1：权限目录分类表（映射到控制器Floodlight中相对应的API）

|  |  |  |
| --- | --- | --- |
| Category | Permission | Screening method(s) |
| Read | read\_topology | getAllSwitchMap:Controller.java |
| getLinks:LinkDiscoverManager.java |
| read\_all\_flow | getFlows:StaticFlowEntryPusher.java |
| read\_statistics | getSwitchStatistics:SwitchResourceBase.java |
| getCounterValue:SimpleCounter.java |
| read\_pkt\_in\_payload | get:FloodlightContextStore.java |
| read\_controller\_info | retrieve:ControllerMemoryResource.java |
| Notification | pkt\_in\_event | addToMessageListeners: Controller.java  addListener:ListenerDispatcher.java |
| flow\_removed\_event |
| error\_event |
| Write | flow\_mod\_route | insertRow:AbstractStorageSource.java |
| flow\_mod\_drop | deleteRow: AbstractStorageSource.java |
| set\_flow\_priority | insertRow: AbstractStorageSource.java |
| set\_devices\_config | setAttribute:OFSwitchBase.java |
| set\_pkt\_out | write:IOFSwitch.java |
| writeThrottled:IOFSwitch.java |
| flow\_mod\_modify\_hdr | parseActionString:StaticFlowEntries.java |
| modify\_all\_flows | setCommand:OFFlowMod.java |

2 安全的存储结构。

用一种安全的存储结构将应用程序的身份信息和访问权限进行加密和序列化存储，保证信息的安全性。

3 访问控制：对应用程序的注册，身份认证，权限检查。

4 动态调整：网络管理员对应用程序身份信息的修改/注销，增加/移除应用的访问权限

5 权限查询：网络管理员和应用程序都可以查询到该应用被授予了哪些访问权限。

6 应用程序黑名单（application blacklisting）：防止不良程序的对控制器资源的访问

7 历史记录：记录未授权的访问操作到日志文件，提供非法访问的历史依据。

**三 模块设计：**

**1 应用注册（网络管理员）**

注册成功

填写应用身份信息

写入数据库

初始化访问权限

**应用的身份信息（身份证书）：**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| AppID | AppName | AppKey | Registry | RegistrationDate | ExpDate | ATL |

AppID: 应用程序标识，每个应用程序只有唯一一个。

AppName：应用程序名称

AppKey：应用程序密钥

Registry：应用注册厂商

Registration Date：应用注册日期

Exp Date：应用有效期

ATL：Application trust level，应用程序信任级别：分为三种high;medium;low

**应用程序权限信息：**

|  |  |
| --- | --- |
| AppID（key） | permisssions |

将应用程序的AppID和其访问权限集合映射到一个安全的存储结构，存储每个应用的权限集。类似于LinkedHashMap结构，AppID为关键字

**2身份认证**

流程如下：

写入系统异常行为监测日志

从信息交互模块获得策略的相关信息

获得应用的APPID和AppKey

进行认证

Y

N

访问检查

是否通过认证

**3资源访问检查（基于属性的访问控制）**

当应用程序通过身份认证后，访问检查模块会**基于应用的权限集，应用的有效期，是否在黑名单内这三个属性**对应用调用控制器的API进行访问控制，限制不符合条件的应用的非法访问，从而达到保护控制器资源的目的

得到应用试图使用的权限PER0

在内存数据库中查询与该APPID对应的权限列表PER

PER是否含有PER0

Y

N

XACML访问控制

PER0是否满足权限集PER的要求

得到应用试图使用的权限PER0

在安全存储结构中查询与该APPID对应的访问权限集合PER

N

Y

通过检查，继续访问控制器资源

应用是否在

有效期内

应用是否在

黑名单中

N

N

Y

Y

写入系统异常行为监测日志

入口**main()**函数

启动**FloodlightModuleLoader**

前端视图层

**SystemClientModule**

**FloodlightModuleLoader**查看配置文件，寻找需要加载的模块

逻辑控制层

**AppinfoManageModule**

**AppPermissionManageModule**

**AppAuthenticationModule**

**AppPermissionCheckModule**

**XacmlCtrModule**

**LogManagementModule**

**InformationInteractionModule**

关联模块与对应的服务

生成模块加载列表

数据存储层

**DataInfoStorgeModule**

**FloodlightModuleLoader**对模块进行初始化并启动

**4应用身份信息管理**

分为三个子模块：

应用身份信息修改

应用身份信息查询

应用注销

应用身份信息修改：网络管理员可以修改应用的密钥，有效期，注册厂商等。

应用注销：网络管理员将应用的身份信息和所具有的访问权限从系统中移除。

应用身份信息查询：网络管理员可以通过APPID查询应用的信息。

**5应用权限管理**

分为三个子模块：

权限修改

权限查询

黑名单管理

权限修改：网络管理员可以动态增加（set）和移除（unset）应用程序的访问权限。

权限查询：应用程序和网络管理员可以通过应用的AppID查询被授予的访问权限

黑白名单管理：可以根据策略需要，网络管理员将应用程序加入/移除黑名单中，使其所有具有的访问权限失效。

**6日志记录**

当应用程序访问请求被拒绝时调用此模块，将非法的访问请求（即试图获取网络信息或者修改网络状态的未被授权的访问请求）记录到日志。日志文件提供了两个重要的功能：记录在日志文件中的未被授权的操作顺序可以被识别为一种入侵行为模式，除了为黑名单管理提供部分参考依据外，更重要的是为以后恶意应用的检测提供依据。

日志记录的信息采用以下形式：

<date><time><AppID>< exceptionEvent><deniedpermission>

**异常事件ExceptionEvent:**分为四类

（1）NoIdentityAuthenticatio:未经过身份注册的应用的非法访问

（2）BeyondValidity:不在有效期内的非法访问

（3）BlacklistRestrictions:处在黑名单内的限制访问

（4）[ExceedsAuthorizedAccess](https://www.baidu.com/link?url=aHYuB5zVVqXR-PhipxUWzr3PT5ERpLdCxTVypttOpjFC7ge0xB6GonOnANadKD7j2fPs-mebuoEWFS3I6WTKCWbwR9zWNvAE1owOGWUGasGqTCON4NRV9hpeBhTbDGJ8&wd=&eqid=be9582d4002d84b00000000358b24e93):不具备对该资源访问权限的越权访问, deniedpermission为拒绝的具体权限类型

VirtualHost

**7数据库**

数据库主要是记录应用程序的身份信息，权限集合以及黑名单，并且对其进行加密和序列化存储，防止信息被泄露或恶意篡改。在身份认证，资源访问检查进行查询时访问数据库，在应用程序权限管理和身份信息管理时修改数据库相关数据。

Restlet组件

R

HTTP客户端

SystemClientModule

REST应用

PermissionListResource

Router

AppListResource

PolicyCreatResource

REST应用

VirtualHost

HTTP服务器

Restlet组件

Restlet组件