# Eag2.0设计

## 一、关键类关系图



## 二、对象说明

Blkmem：内存管理，主要用于管理类对象的分配，尽可能的防止因大量小内存申请释放出现的内存碎片。

Eag\_blkmem\_create:创建一个正对某种固定大小类型的对象分配管理器。需要指定类型的大小以及相关的参数。

Eag\_blkmem\_destroy:释放一个管理器对象。

Eag\_blkmem\_mallo\_item:分配一个对象。

Eag\_blkmem\_free\_item:释放内对象的内存空间至管理器。

Thread：任务调度，确切的讲是事件调度。不同于系统的thread，操作系统的thread是根据时间片以及认为的优先级来调度的，这个调度是按照时间的发生来调度的，比如socket上有数据，或者timer定时器的时间到了。这个是整个系统最为基础的模块，所有的流程都由它来进行管理。熟悉它的使用对了解整个系统有很重要的帮助。添加任务时，指定的回调函数有固定的参数，也就是thread实例自身。

Eag\_thread\_master\_new:创建一个新的任务调度器。

Eag\_thread\_master\_free:释放一个任务调度器。

Eag\_thread\_add\_read:添加一个socket的读取任务，当某个socket上有数据可以接收时，调用指定的函数进行处理。

Eag\_thread\_add\_write:添加一个socket的写任务，当socket的写缓冲准备就绪时，调用指定函数进行后续的写操作。尤其要注意tcp流的写法，需要在写的时候判定已经写了多少了，还有多少未写完。

Eag\_thread\_add\_timer:添加一个定时器，时间到后调用指定的函数。因为不是硬件定时器，如果调度程序因为其他任务受到影响的时候，这个时间可能不会十分的准确。指定的时间是秒。

Eag\_thread\_add\_timer\_msec:添加一个定时器，指定的时间时毫秒。

Eag\_thread\_get\_arg:获取任务参数，在添加任务的时候，除了要指定任务的回调函数，有时还需要指定回调函数需要用到的参数。这个函数可以得到当时指定的参数。一般都是在任务的回调函数种调用。

Eag\_thread\_cancel:取消一个任务，在任务的回调处理中，如果后续不需要执行这个任务，通过这个函数取消这个任务的后续执行。也可能在一个任务处理中，取消其他任务。比如：在添加一个socket的读任务同时，为其添加一个超时任务，如果超时后还是没有收到报文，就需要在超时的任务处理中，将socket的读任务以及超时任务自己删除。如果socket读取任务接收到了报文，也需要将其对应的超时任务以及自己(如果逻辑需要的话)取消。

Appdb：记录用户的数据。

App\_db\_create:创建一个用户管理的数据库。

App\_db\_destroy:销毁一个用户管理的数据库。

Newconn:创建一个新的用户session。

Freeconn:释放一个用户session。

Appconn\_find\_by\_mac:按照mac地址查找用户。

Appconn\_find\_by\_user:按照用户名查找用户。

Appconn\_find\_by\_ip:按照ip地址查找用户。

Get\_appconn\_by\_ip:同上，需要整理成一个，这个删除掉吧。

Appconn\_add\_to\_db:将一个用户session添加到appdb中。一个重要的动作就是添加用户的mac以及ip的hash处理，所以一定要在用户session创建后，初始化用户的ip和mac之后调用。所以不能在newconn的时候就直接加到db里面，尽管用户session也是从db中分配的。

Appconn\_del\_from\_db:将一个用户的session从db中删除。

Appconn\_traversal\_all:指定一个回调函数，遍历所有用户。

这个模块再命名规范上不复合coding style要求，需要调整。

Portal：完成和portal server的交互工作。

Eag\_portal\_create:创建一个portal处理的实例。后续需要修改成new。

Eag\_portal\_destroy:销毁一个portal处理的实例。后续最好修改成free。

Eag\_portal\_set\_params:设置portal监听的相关参数。

Eag\_portal\_set\_thread\_master:设置portal的master，整个eag系统中需要一个master就可以了，其他相关模块都直接引用他，然后再eag主循环中监控着一个master。

Eag\_portal\_set\_get\_sess\_cb:设置portal中获取用户session的的回调函数。以及回调函数的参数。

Eag\_portal\_start:启动portal的服务。建立监听端口，创建相应的thread。

Eag\_portal\_stop:停止portal处理的服务。取消端口监听，取消相应的thread，主要用户再修改nasip能关键参数的时候。

Radius：完成和radius server的交互工作。

Radius\_new:创建一个radius处理的对象

Radius\_free:释放一个radius处理的对象

Radius\_set\_listen:设置radius监听的相关参数

Radius\_set\_thread\_master:设置radius的thread master

Radius\_set\_retry\_params:设置报文重发等相关参数

Radius\_set\_session\_params:设置session计费间隔和超时下线等和用户相关的默认参数

Radius\_start:开启监听

Radius\_stop:关闭监听，主要用于修改了nasip的情况

Radius\_set\_coa\_cb:设置coa处理的回调函数。

Radius\_auth\_req:发送一个认证请求，和计费请求一起应该是radius中最重要和调用最多的的两个api。

Radius\_acct\_req:发送一个计费请求

Captive：完成用户授权以及流量获取等工作。

Eag\_captive\_new:创建一个captive portal管理对象

Eag\_captive\_free:释放一个captive portal管理对象

Eag\_captive\_set\_redir\_srv:设置重定向服务器的地址，也就是redir监听的地址和端口，一般而言为nasip:3990

Eag\_captive\_add\_intf:添加一个或多个管理的三层接口，多个时，使用“,”分割

Eag\_captive\_del\_intf:删除一个或多个管理的三层接口，多个时，使用“,”分割

上面两个函数需要能够动态的添加接口。也就是能够再captive start的状态下实时添加，删除新的接口。同时还需要注意一个问题，2.0上多实例的实现时在各个进程里面，在添加接口的时候，一个接口只能添加到一个实例中，其他的实例不能添加了。所以各个实例之间需要建立一种通信机制。

Eag\_captive\_get\_intf\_list:获得三层接口的列表

Eag\_captive\_start:开启captive服务

Eag\_captive\_stop:关闭captive服务

Eag\_captive\_authorize:用户授权函数。按照设计，最后调用的时用户自己的授权模块。

Eag\_captive\_deauthorize:用户解除授权的函数。

Eag\_captive\_check\_flux:更新用户的流量

Eag\_captive\_add\_white\_list:添加白名单

Eag\_captive\_del\_white\_list:添加黑名单，和添加接口一样，要求能够实时的添加。可能有一个问题，删除captive的三层接口时，对应的白名单也需要删除。

Eag\_captive\_get\_white\_list:获得白名单列表。

Eag\_captive\_add\_black\_list:添加黑名单。

Eag\_captive\_del\_black\_list:删除黑名单。

Eag\_captive\_get\_black\_list:获得黑名单列表。

Redir：完成用户重定向的处理。

Eag\_redir\_new:创建一个重定向管理对象

Eag\_redir\_free:释放一个重定向管理对象

Eag\_redir\_set\_params:设置重定向相关的参数，监听的ip，端口

Eag\_redir\_set\_thread\_master:设置thread调度器

Eag\_redir\_set\_portalserver\_list:设置重定向使用的多portal配置的列表。

Eag\_redir\_start:开启服务

Eag\_redir\_stop:关闭服务。

Eag\_redir\_set\_get\_appsession\_cb:设置获取用户session的相关回调函数。

## 三、处理流程

1、url重定向



2、登陆



3、下线流程

