

系统中有 3 个结构体：客户端 Client、管理者 Manager、存储节点 Storer。其中，Client 和 Storer 是可以创建多个运行于不同物理机的独立进程，Manager 只需创建一个。Client 是一切存储服务请求的发起者，Storer 为(keyword,{fids})创建可验证结构，执行存储和查询服务，Manager 维护可验证结构的根哈希，接收来自 Client 的请求，拆解存储和查询任务，根据可扩展一致性哈希将任务分配给不同的 Storer 执行，并最终验证 Storer 的执行结果，将验证结果返回给 Client。

### 1. 初始化

该函数是系统入口，调用三个结构体的 New 函数，创建指定数量的 Client 和 Storer 实体，创建一个 Manager 实体。它们的 New 函数中除了初始化所需的数据结构外，还需建立对给定网络地址的监听，以实现三者之间的通信。**特别注意的是**，Storer 的 New 函数中能够根据给定的 mode 创建指定类型的可验证结构。

参数：包括但不限于 Client 数量，Storer 数量，可验证结构类型 mode，Clients 的网络地址文件路径（可能不需要），Storers 的网络地址文件路径，Manager 的网络地址。

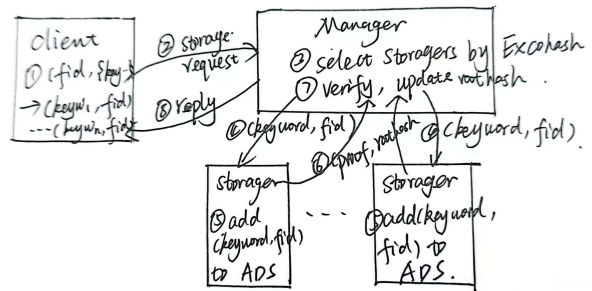
### 2. 文件上传

该函数向系统中写入(fid,{keywords})，通过调用 Client 的 PutFile 函数实现。

Client 的 PutFile 函数具体实现：①将(fid,{keywords})拆分为多个(keyword,fid)对；②向 Manager 发送存储请求（通常是 rpc 调用 Manager 上的 Add 函数），其中带有所有的 (keyword,fid)对（可以压缩为只含一个 fid）；③等待收到 Manager 返回的存储回复消息。

Manager 的 Add 函数具体实现：①利用可扩展一致性哈希将每个 keyword 映射至一个 Storer；②将每个(keyword,fid)对发送给相应的 Storer（通常是 rpc 调用 Storer 上的 Add 函数）；③等待 Storer 完成 Add 后返回的(proof,roothash)，验证 proof，更新相应的根哈希；④返回验证结果给 Client。

Storer 的 Add 函数具体实现：将(keyword,fid)插入到可验证结构中，返回(proof,roothash)。



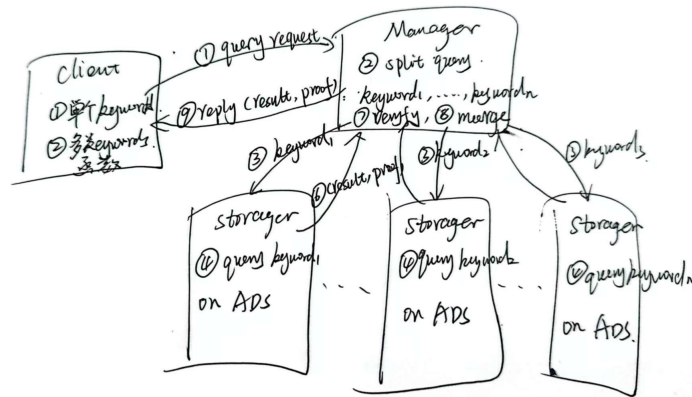
### 3. 文件查询

该函数实现对给定 keyword 的查询，通过调用 Client 的 QueryByKeyword/QueryByFunc 函数实现。

Client 的 Query 函数具体实现：①生成单个 keyword 或由多个 keyword 组成的布尔函数的查询请求包，发送给 Manager（rpc 调用 Manager 的 Query 函数）；②等待 Manager 返回查询结果(result,proof,roothash)，验证查询结果。

Manager 的 Query 函数具体实现：①如果是布尔查询，则拆分查询请求中的多个关键词，通过一致性哈希将所有 keyword 映射至相应 Storer；②将每个 keyword 发送至相应 Storer（rpc 调用 Storer 的 Query 函数）；③等待 Storer 返回查询结果(mresult,proof)，验证 proof；④如果是布尔查询，合并多个 mresult 为最终查询结果 result，生成最终查询结果的 proof，返回(result,proof,roothash)给 Client。

Storager 的 Query 函数具体实现：在 ADS 中查询 keyword，得到查询结果 mresult 并生成 proof，将(mresult,proof)返回给 Manager。



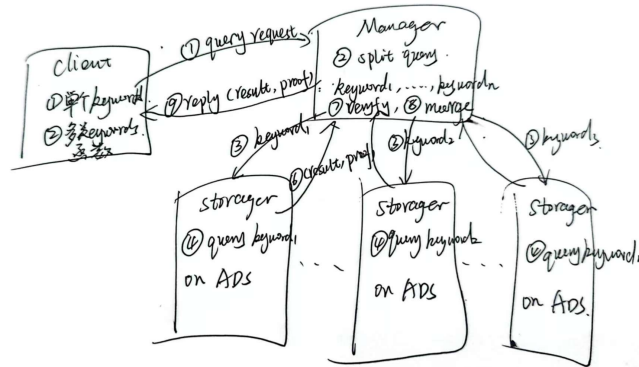
#### 4. 文件删除

该函数在系统中删除(fid,{keywords})，通过调用 Client 的 DeleteFile 函数实现。

Client 的 DeleteFile 函数具体实现：①将(fid,{keywords})拆分为多个(keyword,fid)对；②向 Manager 发送删除请求（通常是 rpc 调用 Manager 上的 Delete 函数），其中带有所有的(keyword,fid)对（可以压缩为只含一个 fid）；③等待收到 Manager 返回的删除回复消息。

Manager 的 Delete 函数具体实现：①利用可扩展一致性哈希将每个 keyword 映射至一个 Storager；②将每个(keyword,fid)对发送给相应的 Storager（通常是 rpc 调用 Storager 上的 Delete 函数）；③等待 Storager 完成 Delete 后返回的(proof,roothash)，验证 proof，更新相应的根哈希；④返回验证结果给 Client。

Storager 的 Delete 函数具体实现：将(keyword,fid)在可验证结构中删除，返回(proof,roothash)。



#### 5. 文件更新

该函数更新系统中的(fid,{keywords})为(fid,{keywords'})，通过调用 Client 的 UpdateFile 函数实现，以(fid,{keywords})和(fid,{keywords'})为参数。

Client 的 UpdateFile 函数具体实现：①将(fid,{keywords})拆分为多个(keyword,fid)对，将(fid,{keywords'})拆分为多个(keyword',fid)对；②向 Manager 发送更新请求（通常是 rpc 调用 Manager 上的 Update 函数），其中带有所有的(keyword,fid)对和(keyword',fid)对（可以压缩为只含一个 fid）；③等待收到 Manager 返回的更新回复消息。

Manager 的 Update 函数具体实现：①利用可扩展一致性哈希将每个 keyword 和 keyword' 映射至一个 Storager；②将每个(keyword,fid)对发送给相应的 Storager（rpc 调用 Storager 上

的 Delete 函数), 同时将每个(keyword',fid)对发送给相应的 Storer (rpc 调用 Storer 上的 Add 函数); ③等待 Storer 完成 Add 或 Delete 后返回的(proof,roothash), 验证 proof, 更新相应的根哈希; ④返回验证结果给 Client。

Storer 的 Add 函数和 Delete 函数同上。

