干掉状态：从 session 到 token

美好的旧时光

我经常怀念三十年前那美好的旧时光， 工作很轻松， 生活很悠闲。

上班的时候偶尔有些 HTTP 的请求发到我这里， 我简单的看一下， 取出相对应的 html 文档，图片，发回去就可以了， 然后就可以继续喝茶聊天。

我的创造者们对我很好， 他们制定的一个简单 HTTP 协议， 就是请求加响应， 尤其是我不用记住是谁刚刚发了 HTTP 请求， 每个请求对我来说都是全新的！

邮件服务器很羡慕我， 他说：老弟，你的生活太惬意了， 哪像我， 每次有人从客户端访问邮箱， 我都得专门给他建立一个会话， 来处理他发的消息， 你倒好， 完全不用管理会话。

这是由应用的特性决定的， 如果邮件服务器不管理会话， 那多个人之间的邮件消息就会完全混到一起了， 乱作一团了。

而 30 年前的 Web 基本上就是文档的浏览而已， 既然是浏览，我作为一个服务器， 为什么要记住谁在一段时间里都浏览了什么文档呢?

Session

但是好日子没持续多久， 很快大家就不满足于静态的 Html 文档了， 交互式的 Web 应用开始兴起， 尤其是论坛，在线购物等网站。

我马上就遇到了和邮件服务器一样的问题， 那就是必须管理会话，必须记住哪些人登录系统， 哪些人往自己的购物车中放了商品， 也就是说我必须把每个人区分开。

这对我来说是个不小的挑战， 由于 HTTP 协议的无状态特性， 我必须加点小手段，才能完成会话管理。

我想出的办法就是给大家发一个会话标识(session id), 说白了就是一个随机的字符串，每个人收到的都不一样， 每次大家向我发起 HTTP 请求的时候，把这个字符串给一并捎过来， 这样我就能区分开谁是谁了。

沉重的负担

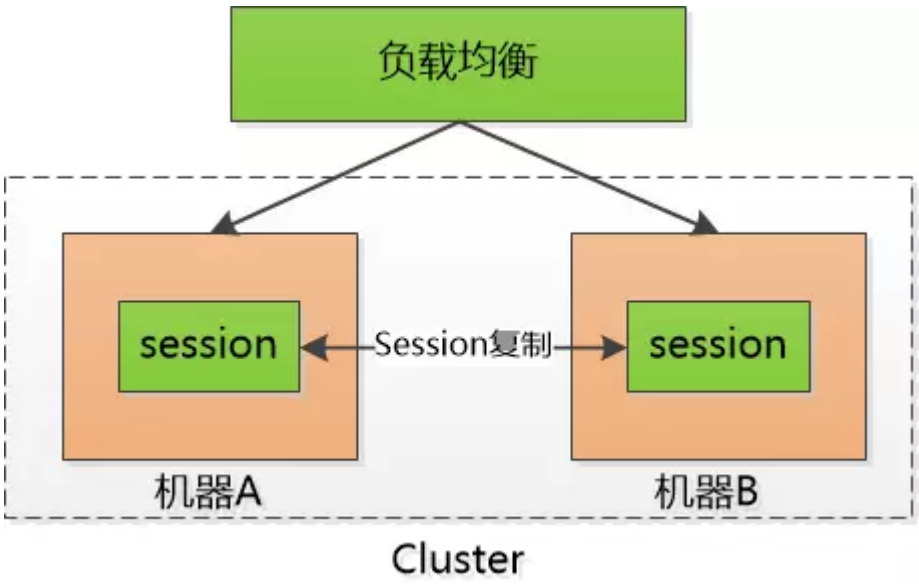
大家都很高兴， 可是我就不爽了。

每个人只需要保存自己的 session id，而我需要保存所有人的 session id ！ 如果访问我的人多了， 就得由成千上万，甚至几十万个。

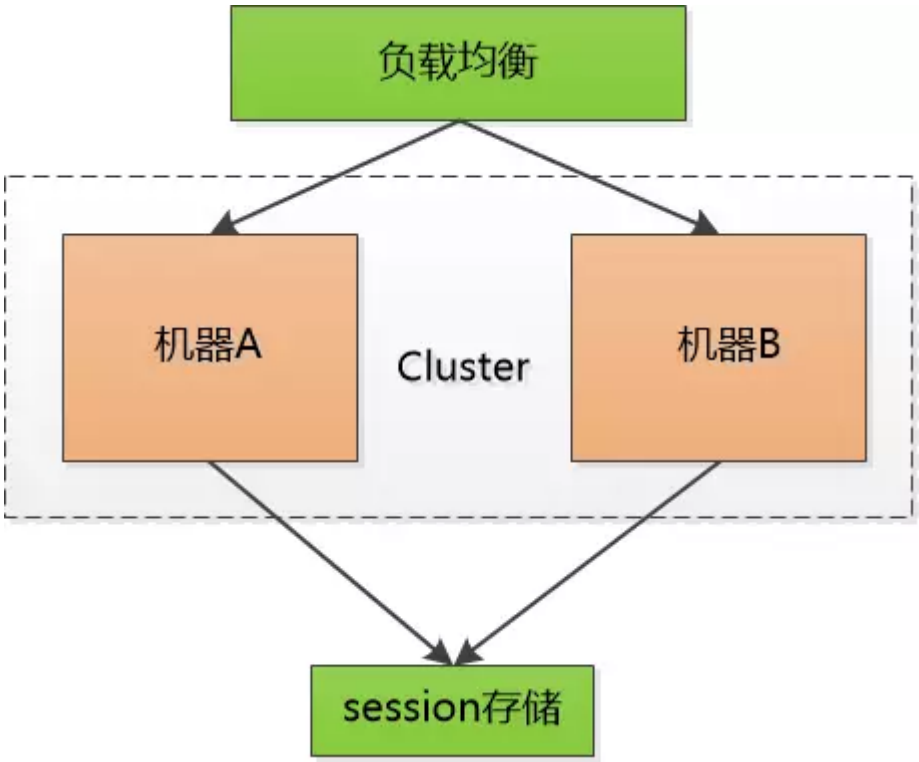
这对我来说是一个巨大的开销 ， 严重的限制了我的扩展能力， 比如说我用两个机器组成了一个集群， 小 F 通过机器 A 登录了系统， 那 session id 会保存在机器 A 上， 假设小 F 的下一次请求被转发到机器 B 怎么办? 机器 B 可没有小 F 的 session id 啊。

有时候我会采用一点小伎俩： session sticky ， 就是让小 F 的请求一直粘连在机器 A 上， 但是这也不管用， 要是机器 A 挂掉了， 还得转到机器 B 去。

那我只好做 session 的复制了， 把 session id 在两个机器之间搬来搬去， 快累死了。



后来有个叫 Memcached 的给我支了招： 把 session id 集中存储到一个地方， 所有的机器都来访问这个地方的数据， 这样一来，就不用复制了， 但是增加了单点失败的可能性， 要是那个负责 session 的机器挂了， 所有人都得重新登录一遍， 估计得被人骂死。



我也尝试把这个单点的机器也搞出集群，增加可靠性， 但不管如何， 这小小的 session 对我来说是一个沉重的负担。

时间换空间

这几天的晚上我一直在思考， 我为什么要保存这可恶的 session 呢， 只让每个客户端去保存该多好?

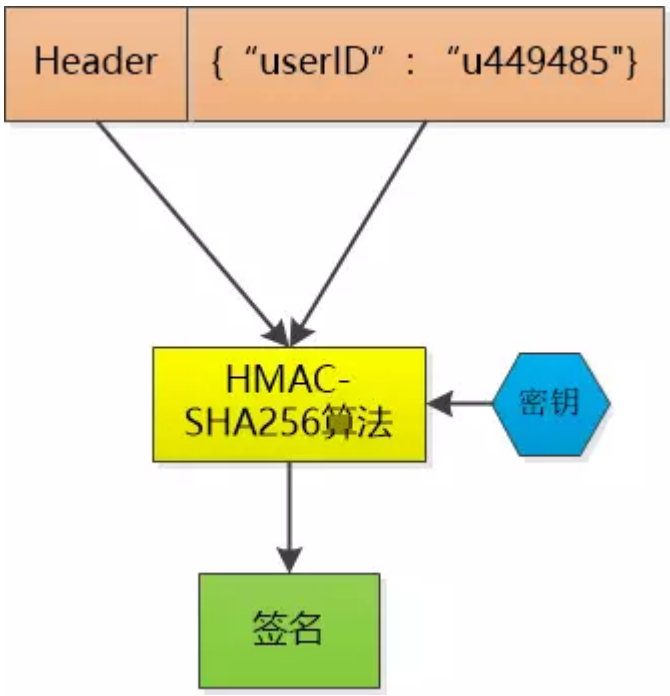
可是如果我不保存这些 session id , 我怎么验证客户端发给我的 session id 的确是我生成的呢？ 如果我不去验证，我都不知道他们是不是合法登录的用户， 那些不怀好意的家伙们就可以伪造 session id , 为所欲为了。

嗯，对了，关键点就是验证！

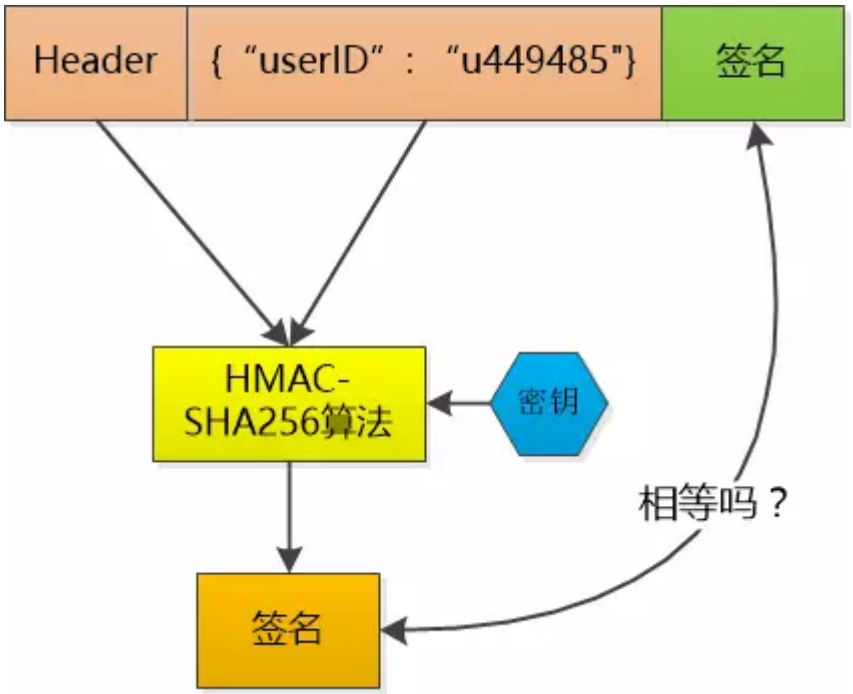
比如说， 小 F 已经登录了系统， 我给他发一个令牌(token)， 里边包含了小 F 的 user id， 下一次小 F 再次通过 Http 请求访问我的时候， 把这个 token 通过 Http header 带过来不就可以了。

不过这和 session id 没有本质区别啊， 任何人都可以可以伪造， 所以我得想点儿办法， 让别人伪造不了。

那就对数据做一个签名吧， 比如说我用 HMAC-SHA256 算法，加上一个只有我才知道的密钥， 对数据做一个签名， 把这个签名和数据一起作为 token ， 由于密钥别人不知道， 就无法伪造 token 了。



这个 token 我不保存， 当小 F 把这个 token 给我发过来的时候，我再用同样的 HMAC-SHA256 算法和同样的密钥，对数据再计算一次签名， 和 token 中的签名做个比较， 如果相同， 我就知道小 F 已经登录过了，并且可以直接取到小 F 的 user id , 如果不相同， 数据部分肯定被人篡改过， 我就告诉发送者： 对不起，没有认证。



Token 中的数据是明文保存的（虽然我会用 Base64 做下编码， 但那不是加密）， 还是可以被别人看到的， 所以我不能在其中保存像密码这样的敏感信息。

当然， 如果一个人的 token 被别人偷走了， 那我也没办法， 我也会认为小偷就是合法用户， 这其实和一个人的 session id 被别人偷走是一样的。

这样一来， 我就不保存 session id 了， 我只是生成 token , 然后验证 token ， 我用我的 CPU 计算时间获取了我的 session 存储空间！

解除了 session id 这个负担， 可以说是无事一身轻， 我的机器集群现在可以轻松地做水平扩展， 用户访问量增大， 直接加机器就行。 这种无状态的感觉实在是太好了！