云风的 BLOG
思绪来得快去得也快，偶尔会在这里停留

« Skynet 开源 | 返回首页 | Skynet 的一些改进和进展 »

Skynet 集群及 RPC

这几天老在开会，断断续续的拖慢了开发进度。直到今天才把 Skynet 的集群部分，以及 RPC 协议设计实现完。

先谈谈集群的设计。

最终，我们希望整个 skynet 系统可以部署到多台物理机上。这样，单进程的 skynet 节点是不够满足需求的。我希望 skynet 单节点是围绕单进程运作的，这样服务间才可以以接近零成本的交换数据。这样，进程和进程间（通常部署到不同的物理机上）通讯就做成一个比较外围的设置就好了。

为了定位方便，我希望整个系统里，所有服务节点都有唯一 id 。那么最简单的方案就是限制有限的机器数量、同时设置中心服务器来协调。我用 32bit 的 id 来标识 skynet 上的服务节点。其中高 8 位是机器标识，低 24 位是同一台机器上的服务节点 id 。我们用简单的判断算法就可以知道一个 id 是远程 id 还是本地 id （只需要比较高 8 位就可以了）。

我设计了一台 master 中心服务器用来同步机器信息。把每个 skynet 进程上用于和其他机器通讯的部件称为 Harbor 。每个 skynet 进程有一个 harbor id 为 1 到 255 （保留 0 给系统内部用）。在每个 skynet 进程启动时，向 master 机器汇报自己的 harbor id 。一旦冲突，则禁止连入。

master 服务其实就是一个简单的内存 key-value 数据库。数字 key 对应的 value 正是 harbor 的通讯地址。另外，支持了拥有全局名字的服务，也依靠 master 机器同步。比如，你可以从某台 skynet 节点注册一个叫 DATABASE 的服务节点，它只要将 DATABASE 和节点 id 的对应关系通知 master 机器，就可以依靠 master 机器同步给所有注册入网络的 skynet 节点。

master 做的事情很简单，其实就是回应名字的查询，以及在更新名字后，同步给网络中所有的机器。

skynet 节点，通过 master ，认识网络中所有其它 skynet 节点。它们相互一一建立单向通讯通道。也就是说，如果一共有 100 个 skynet 节点，在它们启动完毕后，会建立起 1 万条通讯通道。

为了缩短开发时间，我利用了 zeromq 来做 harbor 间通讯，以及 master 的开发。蜗牛同学觉得更高效的做法是自己用 C 来写，并和原有的 gate 的 epoll 循环合并起来。我觉得有一定道理，但是还是先给出一个快速的实现更好。

我们的早一个 Erlang 版本，把 client 也看成了 skynet 系统中的特殊节点。这次看来，我认为是不必要的设计。

如果在同一个进程内，通讯和包转发足够廉价的话，完全没必要为统一这种特殊性而多做太多工作。所以在这次新实现中，client 被看成是 gate 这个服务才了解的细节。由 gate 收集 client 的数据流，并转发到内部的其它服务上。同时，我为发送数据单独启动了一类服务。为每个接入 gate 的 client 动态生成一个节点。只要向这个节点发送数据，都加上和 client 间约定的打包协议的包头转发给 client 。

把 client 独立出来，不当作是内部节点处理，可以使我们能专心 RPC 的问题。

skynet 的内部节点之间，有很大程度是请求回应模式的消息传递模式。这种请求回应模式，可以是 RPC 请求，也可以是一些更简单的通讯协议。

在前一个版本中，我们认为，skynet 只需要解决后消息包，如何有序的，从一个节点传输到另一个节点就够了。之后的细节是下一个层次考虑的问题。可是做下去我们发现，不同的服务间如果想协同工作，必须约定一些基本的通讯协议。每个服务使用独立不相同的通讯协议几乎是不可能的。这是因为，每个服务节点只有单一的输入消息源。虽然我们可以识别消息的来源地址，但根据来源地址来区分消息协议种类是不可能的事情。

结果，我们采用了 google proto buffer 。消息包必须用 protobuffer 打包，并有统一的一级结构。这反而是整个设计不那么简洁了。

这次，我归纳了这半年来的使用需求。发现，skynet 不应只处理单个包从某点发送到另一点的任务。既然我们不抽象出连接这个概念，那么就至少需要让 skynet 框架了解怎样回应一个特定的包。

所以，最终我把一个 31 位的 session id 放到了底层。每个服务节点内部都维护了一个单调递增的 session id 记数器。一旦它需要时，可以给它发出的消息携带一个唯一的 id ；同时约定，接收到这个包并加以处理的节点，如果想针对这个消息包做回应，它就应该把这个 session id 发送回来。

为了区分请求包和回应包。约定，请求包的 session id 为负数，回应包的 session id 为正数。不需要回应的包，可以用 0 做 session id 。

这样，我们就可以利用收到的 session id 做数据包的有限分发了。这并没有增加太多的协议上的约定，每个服务可以按自己喜爱的方式设计协议。它们可以要求请求它的服务的人按自己的协议发送请求，它可以正确的回应。同时，它需要使用其它远程服务时，则按对方服务的协议来通讯。

接下来，在 lua 的封装层做很少量的工作，就可以让这套机制运转起来了。也就是根据收到的 session id 做一下分发。利用 coroutine ，做远程请求时，记录下产生的 session id ， yield 出来，把线程挂起；待到收到携带有相同 session id 的反馈包，把挂起的线程唤醒即可。

我写了一个简单的 key-value 设置和查询的内存数据库作为范例。起名叫 SIMPLEDB ，可以从 client 发起查询请求：“ GET xxx ”，或是更新请求：“ SET xxx yyy ”。agent 收到请求后，会转发到 SIMPLEDB ，并把结果反馈给 client 。有兴趣的同学可以看看相关的 lua 代码。

当然，lua 在整套系统中并不是必备设施。如果你愿意，也可以写出相同功能的其它动态语言（例如 python ）的对接模块来。

云风 提交于 August 6, 2012 10:09 PM | 固定链接

COMMENTS

我就是用zeromq做得，也可以添加到自己的epoll中，唯一不好就是zeromq的一个实例默认就有3个线程，但对性能没什么影响

Posted by: kenny wong | (22) July 10, 2014 10:15 AM

云风，建议别用erlang~

Posted by: ctemple | (21) August 22, 2012 01:24 PM

会不会开源erlang的，强烈期待

Posted by: hihu | (20) August 14, 2012 09:53 AM

希望有实测性能数据.

Posted by: foyo23 | (19) August 10, 2012 03:22 PM

学习 rpc ing

Posted by: 财 | (18) August 10, 2012 12:01 AM

@David Xu

我没觉得有啥好怀疑的。肯定有Erlang不适合的逻辑，可是考虑到Erlang的Prolog背景，很多逻辑相比用别的语言应该更适合用Erlang来实现。而MMORPG，不论你用啥语言实现，每一个物体都会是一个FSM，恰好对应Erlang里的gen_fsm。

现在在用Erlang开发游戏的不要太多。比如，某游戏就把逻辑从Ruby改用Erlang实现了:

http://www.slideshare.net/martin.rehfeld/2012-05at-scalewithstylemartin

Posted by: 'EXIT' | (17) August 9, 2012 06:35 PM

@EXIT

用单一系统做当然有他的好处，需要语言之间交互的问题就会减少。有些语言之间交互不畅，会影响效率。
但是erlang写逻辑，过去我们总持怀疑态度，不知道有没有现实的例子可以参考。

Posted by: David Xu | (16) August 9, 2012 03:48 PM

是不是可以换个思路，把之前用别的语言实现的逻辑改用Erlang来实现，看看性能有啥差别。

Posted by: 'EXIT' | (15) August 9, 2012 02:20 PM

@Cloud
master的设计是不是有点重，而且没有必要，参考erlang的设计，每个机器上都有一个注册服务器，节点只要将自己的信息注册到这个服务器就行，如果一个节点向另一个节点通信，先访问这个节点的注册服务器就可以了。

Posted by: yaoxinming | (14) August 9, 2012 01:17 PM

@david xu
如果涉及到阻塞IO，用erlang或者自己c写都是挺讨厌的事情，当然erlang帮我们做了很多比如对文件操作erlang底层就用到了线程库，自己也可以写驱动来解决，其实我想说的是集群和rpc在erlang中已经解决的相当好了，当然自己写对项目的把握更好些。不过我是来学习的，看了skynet的代码，我对erlang的实现也更理解了，呵呵。另外erlang中消息传递不不都是拷贝的，有款虚拟机是支持引用的，但据说测试后性能好像不太好

Posted by: yaoxinming | (13) August 9, 2012 11:25 AM

@yaoxinming

给erlang写C代码，有个问题我觉的挺讨厌的，erlang虚拟机是不能被阻塞的，如果你的C接口只是做简单的计算就返回，而没有I/O等阻塞的情况发生，那还好。否则必须要么用port和外部通过进程/线程间通讯，要么用驱动程序的模型把你的代码包装进独立的OS线程或者登记你的file handle进erlang虚拟机，利用erlang的内部使用epoll/kqueue回调你的I/O处理函数？然后再通过erlang的标准接口返回数据（也是一种内部port通讯）？

Posted by: David Xu | (12) August 9, 2012 10:58 AM

to David Xu：
erlang和c混用是比较方便的，只比lua中用c稍微麻烦点

Posted by: yaoxinming | (11) August 9, 2012 10:38 AM

skynet的模型有点象erlang的模型，只是erlang没有中心节点，也就不存在中心节点垮掉导致整个网络出问题。erlang的节点是可以hide，并不是不可控制地被别人识别而产生连接。系统单一使用erlang有点困难，游戏系统各种语言混杂，使用erlang是个难点。

Posted by: David Xu | (10) August 9, 2012 10:19 AM

感觉要把erlang的那套东西，用c全部实现了，其实通信层用erlang来做蛮好的，在elrang中用nif把gate等基础设施启动起来就可以，不打算用erlang了吗？能说说原因吗？

Posted by: yaoxinming | (9) August 9, 2012 10:14 AM

好吧，游戏啥时候能出来！

Posted by: baohuams | (8) August 7, 2012 03:48 PM

根据收到的 session id 做一下分发。利用 coroutine ，做远程请求时，记录下产生的 session id ， yield 出来，把线程挂起；待到收到携带有相同 session id 的反馈包，把挂起的线程唤醒即可。

超时，不解决么？

Posted by: tony | (7) August 7, 2012 03:40 PM

是不是erlang没用好呢？

Posted by: zhxgigi | (6) August 7, 2012 12:00 PM

写点关于面向对象的文章呗

Posted by: ifly | (5) August 7, 2012 11:21 AM

如果不用master 可以试试zeroconf

Posted by: 4nil | (4) August 7, 2012 09:56 AM

阿拉伯数字七？？？坑死我了……我晕…最近设计客户端，觉得要高高效跨平台好难…估计还是抽象没选好的原因…恩，纠结矢量绘图库的选择中…难道非要自己写个纯c的么…

Posted by: Xavier Wang | (3) August 7, 2012 01:32 AM

单点宕机问题...

Posted by: 1 | (2) August 6, 2012 10:54 PM

我会把master看成是DNS root server～ :)

Posted by: Wuvist | (1) August 6, 2012 10:46 PM

POST A COMMENT

非这个主题相关的留言请到：留言本

名字：

Email 地址：

为了验证您是人类，请将六加一的结果（阿拉伯数字七）填写在下面：

URL:

记住我的信息？

留言：
（不欢迎在留言中粘贴程序代码）

提交