Skynet 消息系统功能

1、skynet 最核心解决的问题就是 service 的维护和 service 间的通信，而 service 间通信基础是消息包。
2、service 的活动也是被消息包所驱动，service 和 skynet 框架间的互动则是通过消息 callback 进行。
3、消息是服务之间通信的数据单元
4、消息机制所解决的是 skynet 框架下，服务之间数据的交互需求。

消息系统的构成
1、消息体
2、消息队列（消息容器）
    1、全局队列
    2、服务队列
3、队列操作机制
    1、全局队列操作
    2、服务队列操作
4、消息发送机制
5、消息分发机制

消息体
1、source：消息源，来自哪个服务
2、session：消息标示，用于区分自身和上下文
3、data：消息数据（指针）
4、sz：data 长度（高 8 位是消息类型）
消息类型 type 为消息自带了复合类型标示，包括功能类型、session 分配方式，生命周期等；type 通过为消息设置类型，从而为消息选择适当的协议处理方式，这是因为 skynet 选择不统一的消息编码。
skynet_send 发送一个消息时，可根据消息类型中的 session 分配类型进行 session 的再分配，以确保该消息在一次完整的发送过程中是唯一的，开发人员可通过 session 精确区别每一个具体消息。
向服务发送 msg(skynet_message)就是想服务的私有队列压入 msg 的复制体（->data 不复制）

服务的消息队列
1、lock：消息进出队列锁
2、handle：所属 ctx handle
3、queue：动态消息数组
    1、cap：容量
    2、head\tail： 取/存位置
    3、overload：满载标识
    4、overload_threshold:满载上限

4、in_global：队列级别，skynet_mq_pop 失败会设为 0，skynet_mq_push 会根据这个标记来决定是否压入全局队列。
5、release：释放标记
6、next：关联其他消息队列。数据是以数组的形式存在 queue 中，在全局队列满载的情况下，push 进来的 queue 会挤占全局队列 tail queue 到自己的 next 下排队，实现模拟动态队列，同理，next=NULL 就尅进行移除队列操作。

全局消息队列
1、head/tail：取/存队列
2、lock：进出队列过程加锁
3、全局队列是链表结构

队列操作机制--服务队列
1、skynet_mq_push
2、skynet_mq_pop
3、通过维持一个消息指针队列，以及队列的索引操作，来对本质上各自独立 msg 进行队列管理。
4、queue 是可扩展容量，每次扩展都是 cap 的 2 倍。
5、新 push 的 msg 可带动 queue 进入全局队列。

对于 skynet_mq_pop
1、根据自动扩容的设计，pop 后除了调整 head，还需调整 overhead；根据 queue 余量等状况决定是否离开全局队列。

消息发送机制
```
skynet_send(
    struct skynet_context * context,
    uint32_t source,
    uint32 destination,
    int type, int session,
    void * data, size_t sz)
)
```

1、skynet_send
    1、Harbor 节点区分 skynet_harbor_message_isremote(destination)；
    2、skynet_context_push（推送对象是本地）
        1、skynet_mq_push
    3、skynet_harbor_send（推送对象是 harbor 节点）
        1、skynet_context_send
            1、skynet_mq_push

skynet_mq_push 推送目的地是基于 msg 的 handle 所获取的 ctx->queue，所以通过 destination(handle）即可区分 ctx 对象类型；

消息分发机制

1、空闲 worker 从全局队列取 mq；

2、skynet_context_message_dispatch

1、pop mq from global queue

2、pop msg from mq to ctx

3、push mq back to global queue

3、Monitor（skynet_monitor_trigger）

1、Worker 会按照一个权重值(version)每次处理 mq 里的部分包，以免其他 worker 陷入饥饿状态，monitor 则负责更新 worker 的处理权重状态；

4、worker 线程继续其他工作。

spinlock 相关

1、自旋锁_sync_lock_test_and_set

1、持续空转不 sleep，上锁不用切状态

2、锁操作开销小，平时耗 cpu

2、互斥锁 pthread_mutex_lock

1、有锁唤醒，无锁 sleep

2、锁操作开销大，无锁不耗 cpu

3、大操作建议互斥，高频小操作建议自旋