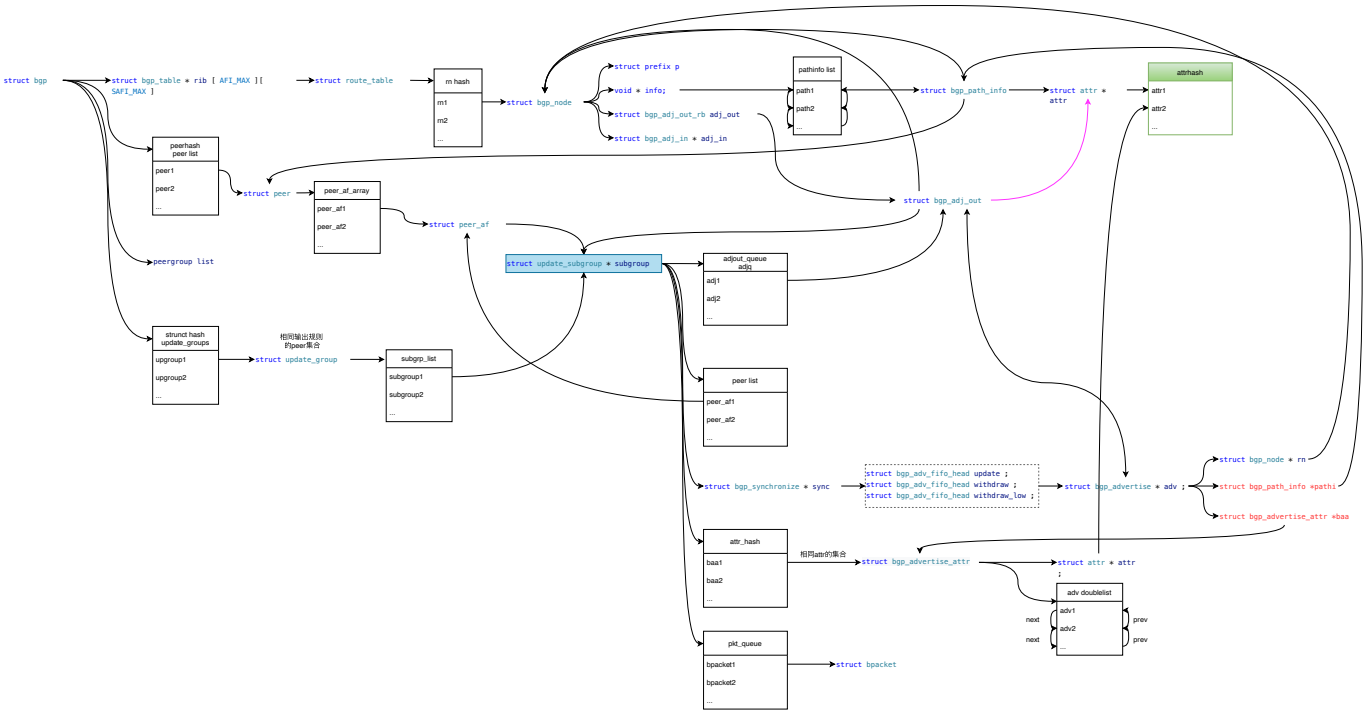
bgpd路由通告分析

核心结构体



- 路由部分
  - 路由表(route_table)
  - 路由节点(bgp_node): 代表一个前缀
    - 路径信息(path_info): 来源属性
    - adj_out: Adj-RIBs-Out 邻接输出
    - adj_int: Adj-RIBs-In 邻接输入
- 邻居部分
  - 邻居(peer)
  - 邻居协议族(peer_af)
- 更新部分
  - 更新组(update_group)
  - 更新子组(udpate_subgroup)
  - 通告(bgp_advertise)
    - update通告
    - withdraw通告
  - 通告属性(bgp_advertise_attr)
  - 字节包(bpacket)

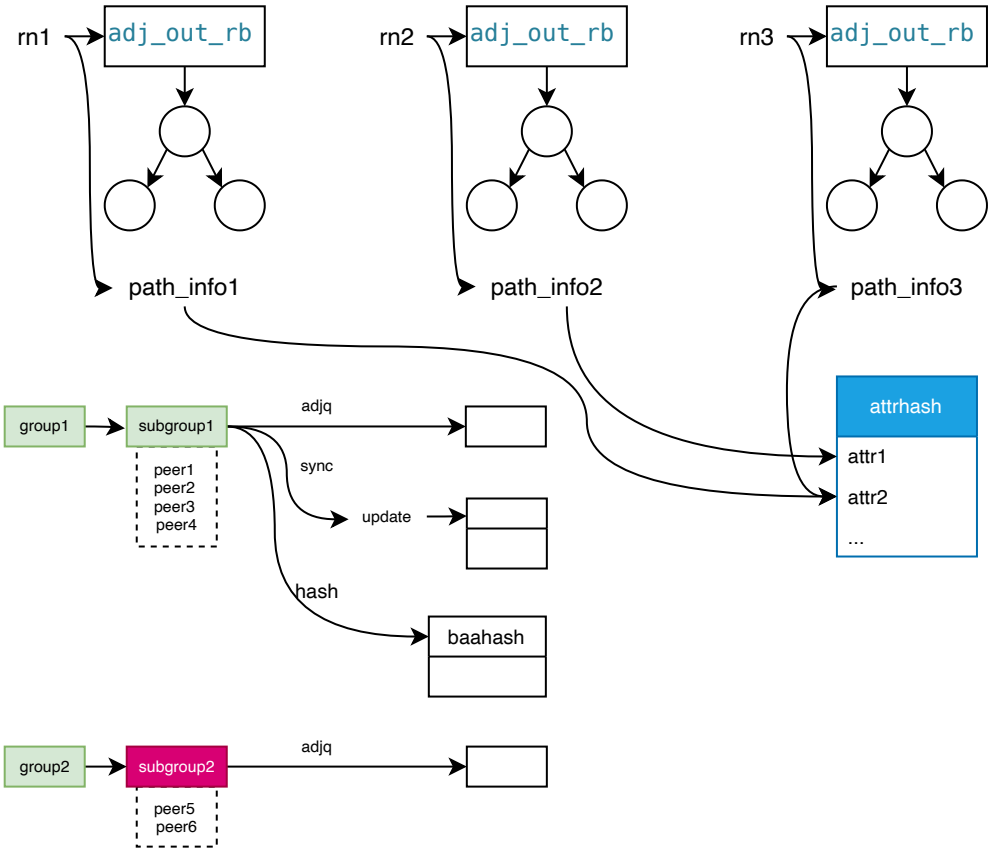
问题

- 什么是更新组?
  - 具有相同输出规则的peer集合
- 为啥引入更新组?
  - 同一个更新组的peer，可共用相同的更新流程，避免重复处理，提升性能
- 什么是更新子组?

    - 更新进度相同的peer集合
- 更新子组有什么用?
    - 虽然同更新组下的peer输出规则是一样的，但可能处于不同的更新进度。因此将相同进度的peer聚合为一个子组
    - subgroup数量是动态变化的，相同进度的subgroup会合并为一个subgroup。类似地，subgroup也可能会拆分
    - 如果所有peer进度相同，则只有一个subgroup
- 什么是通告(adv: bgp_advertise)?
    - 前缀的对外通告信息，后续会根据adv生成通告消息
- 什么是通告属性(baa: bgp_advertise_attr)?
    - 具有相同属性的通告(adv)集合

路由通告可分为3部分: 通告发布、字节包处理、消息收发流程

简单示例

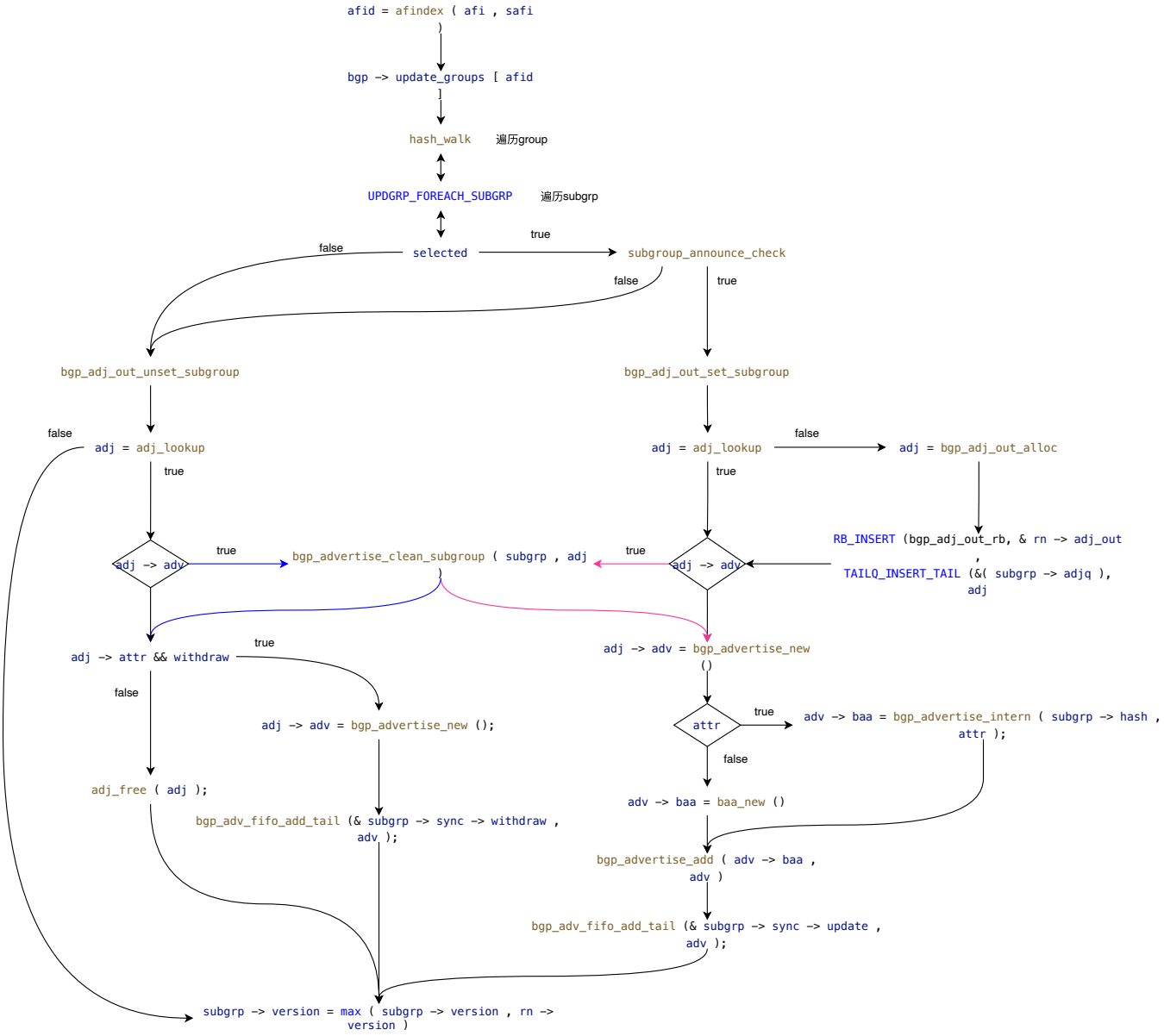


说明

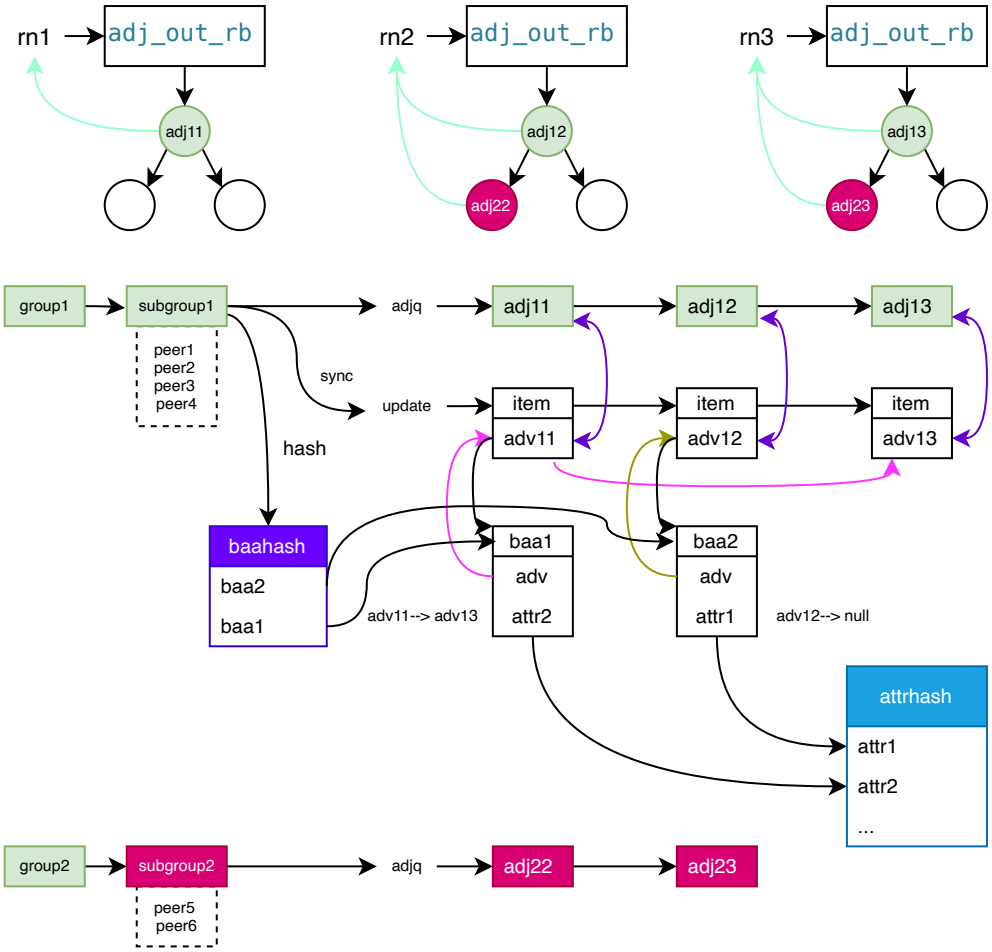
- 三个路由条目
    - adj_out_rb为空
    - path_info1和path_info3具有相同的属性attr2
- 两个更新组及子组
    - subgroup的adjq、sync、baa hash皆为空
    - subgroup1包含peer1~peer4
    - subgroup2包含peer5~peer6

通告发布

通告发布流程



发布后变化

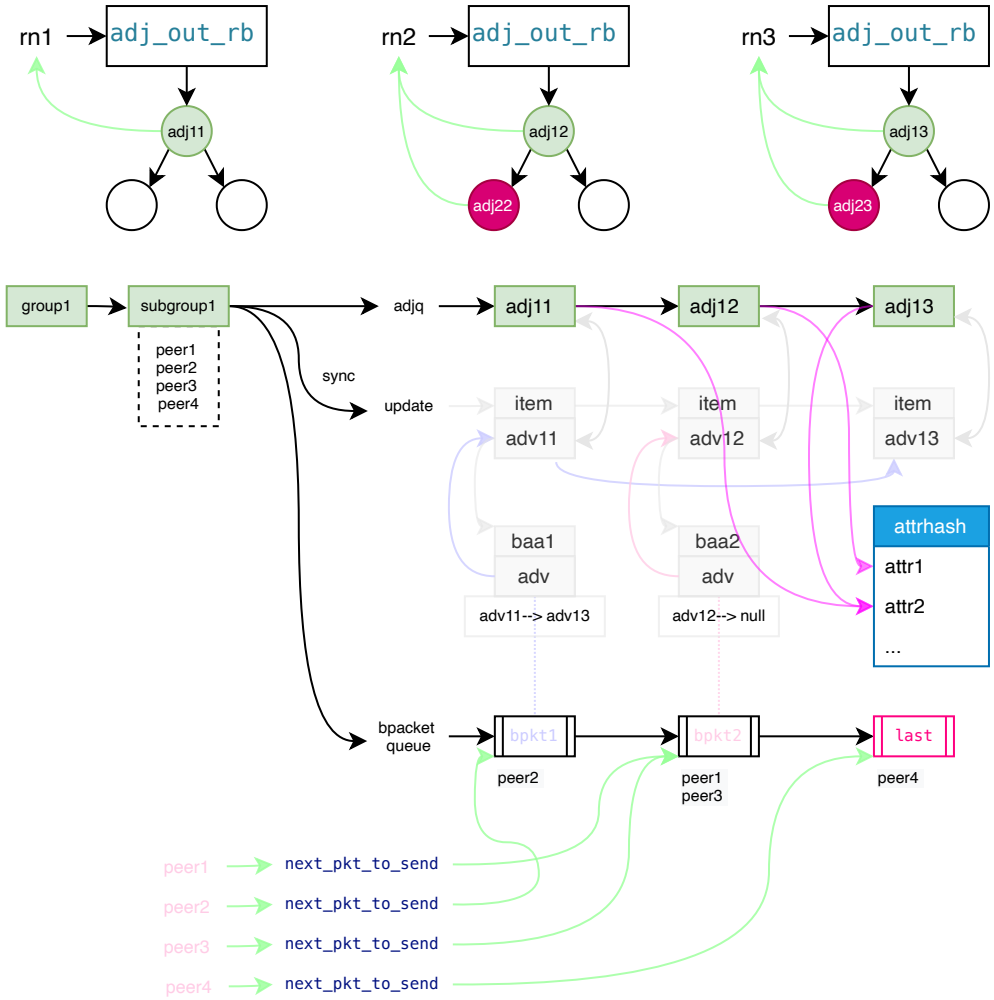


- 不同subgroup有自己的邻接节点(adjxx)
  - 路由节点(rn)中的邻接输出是按subgroup比较的
- 每个通告(adv)与一个邻接节点(adjxx)相互关联
- 相同属性的通告(adv)，通过共享的通告属性(baa)聚合为链表

问题

- subgroup1的邻接输入有3个，而subgroup2只有2个
  - rn1被group2的输出规则过滤掉了
- 为啥需要通告属性(baa)?
  - bgp协议规定相同属性的前缀可封装为一个update消息(未超出包大小限制)
  - baa将具有相同属性的adv连接在一起，以便封装到同一个update消息中

字节包处理



> 对于同一个subgroup，字节包是所有peer共享的

字节包生成

假设peer4优先发送包，当输出buf为空时，则会走字节包生成流程

- 从update的通告fifo队列中，取出第一个通告(adv11)
- 根据其引用的通告属性(baa1)填充update消息的路径属性字段
- 根据其引用的路由节点(rn1)填充update消息的NLRI字段
- 设置adj11的属性为attr2：adj11->attr = adv11->baa1->attr2
- 然后选择adv11的同属性的下一个通告(adv13)，并删除adv11
    - 删除adj11对adv11的引用，adj11->adv = NULL
- 根据adv13填充update消息的NLRI字段
- 类似adv11，设置adj13的attr和adv后，删除adv13
- 由于没有同属性的通告，则删除baa1
- 生成字节包bpkt1，并将其添加到subgroup的字节包队列(bpacket_queue)

字节包输出

每个peer都有一个next_pkt_to_send指针指向下一个将要发送的字节包 字节包队列有个默认的last节点，开始所有peer都指向last节点

入队列

当添加一个bpkt到字节包队列时

- 拷贝bpkt内容到last节点
- 清空bpkt，将其添加到队列尾部，变成新的last节点

> 不需要修改peer的next_pkt_to_send指针

出队列

以peer2为例，其next_pkt指向队列中的第一个包

- 将bpkt1内容输出到自己的发送缓存
- peer2从bpkt1的peers列表中移除，
- 修改peer2的next_pkt，指向bpkt1的下一个包bpkt2，同时加入bpkt2的peers列表中
- 由于bpkt1的peers为空(不被peer引用)，被删除

消息收发