xBFT(Fast Byzantine fault-tolerant Consensus Protocol)

功能介绍:

xBFT 是一种Chain-based BFT 协议实现，其基础理论来自HotStuff的Paper . TOP在其基础上做深度性能优化和安全优化，并抽象成通用的xBFT 共识模块。

基本术语

BFT ： a leader-based Byzantine fault-tolerant protocol

BFT Round ： 一轮BFT的过程，是leader 发起Proposal ,收集到超过2/3+1 验证节点结果的过程。

Proposal ：共识的标的（比如交易转账）

High-QC Block: 完成了第一轮BFT认证的块，这个块可能会被fork 或被丢弃

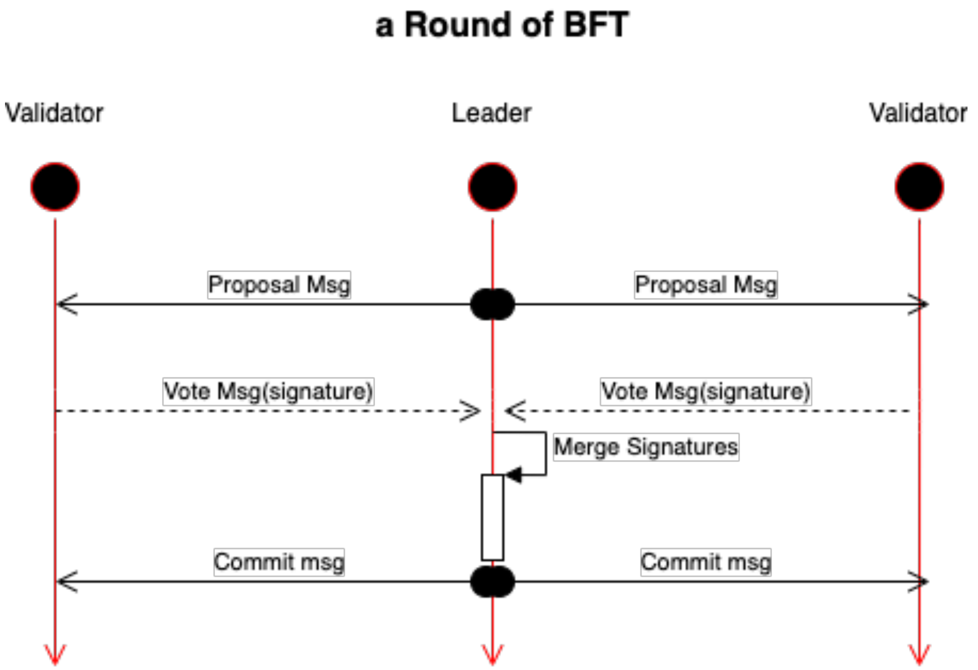
Lock Block ： 完成了2轮BFT 认证的块。这个块已经锁定不允许被fork，但还需再一轮BFT过程

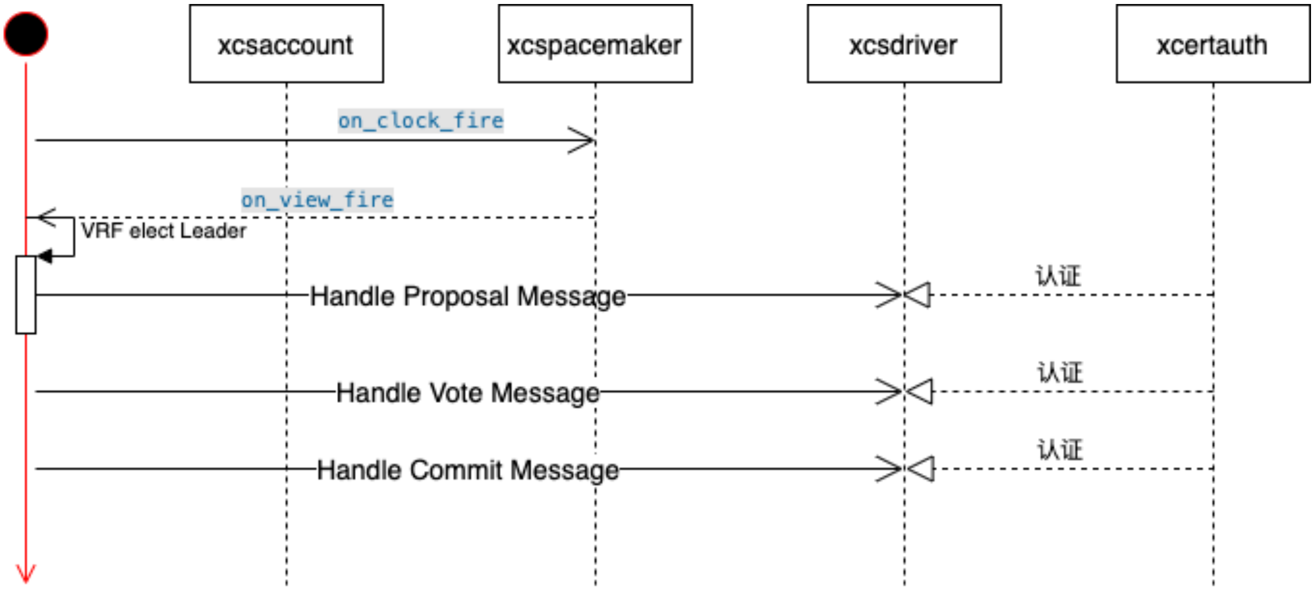
Commit Block : 连续完成了3轮BFT 的块。 这个块就是finalized的共识结果, 允许修改账号的State

Clock Block ：以固定10s为周期对时钟进行对齐的共识结果,驱动VRF 选出每一轮BFT的leader节点

BFT View ： 完成一轮BFT Round 或 持续30s 表示一个View,表示一次View Change

基本流程





代码

1. xxx/src/xtopcom/xBFT 为BFT共识engine 的代码，跨平台编译在CMakeLists.txt, XCode项目文件为xBFT-lib.xcodeproj
2. xxx/src/xtopcom/xbase/ 为基础结构&基础API定义所在
3. 测试程序 位于 xxx/src/xtopcom/xBFT/test/basic

   其中xtestnode.cpp 为测试程序的主要驱动代码所在

接口API

1. 基础对象

```
  //general consensus object
        class xcsobject_t : public base::xionode_t
        {
            virtual int     verify_proposal(base::xvblock_t *
proposal_block,base::xvqcert_t * bind_clock_cert,xcsobject_t * _from_child);
//load and execute block at sanbox

            //send clock event to child objects
            virtual bool    fire_clock(base::xvblock_t &
latest_clock_block,int32_t cur_thread_id,uint64_t timenow_ms);
            //dispatch view-change event to both upper(parent objects) and
lower layers(child objects)
            virtual bool    fire_view(const std::string & target_account,const
uint64_t new_view_id,const uint64_t global_clock,int32_t
cur_thread_id,uint64_t timenow_ms);

            //send packet from this object to parent layers
            virtual bool    send_out(const xvip2_t & from_addr,const xvip2_t &
to_addr,const base::xcspdu_t & packet,int32_t cur_thread_id,uint64_t
timenow_ms);

            //recv_in packet from this object to child layers
            virtual bool    recv_in(const xvip2_t & from_addr,const xvip2_t &
to_addr,const base::xcspdu_t & packet,int32_t cur_thread_id,uint64_t
timenow_ms);
 }
```

发送消息(pdu)到网络（单播或广播）：

```
 //send packet from this object to parent layers
            virtual bool    send_out(const xvip2_t & from_addr,const xvip2_t &
to_addr,const base::xcspdu_t & packet,int32_t cur_thread_id,uint64_t
timenow_ms);
```

从网络层收到包后的回调接口：

```
        //recv_in packet from this object to child layers
        virtual bool    recv_in(const xvip2_t & from_addr,const xvip2_t &
to_addr,const base::xcspdu_t & packet,int32_t cur_thread_id,uint64_t
timenow_ms);
```

定时时钟块事件通知入口：

```
        //send clock event to child objects
        virtual bool    fire_clock(base::xvblock_t &
latest_clock_block,int32_t cur_thread_id,uint64_t timenow_ms);
```

回调给应用层的检验Proposal入口：

```
            virtual int     verify_proposal(base::xvblock_t *
proposal_block,base::xvqcert_t * bind_clock_cert,xcsobject_t * _from_child);
//load and execute block at sanbox
```

2. 共识core对象

```
 //introduce some special events for core objects
        class xcscoreobj_t : public xcsobject_t,public base::xvaccount_t
        {
            //call from higher layer to lower layer(child)
            virtual bool  on_proposal_start(const base::xvevent_t &
event,xcsobject_t* from_parent,const int32_t cur_thread_id,const uint64_t
timenow_ms);

            //call from lower layer to higher layer(parent)
            virtual bool  on_proposal_finish(const base::xvevent_t &
event,xcsobject_t* from_child,const int32_t cur_thread_id,const uint64_t
timenow_ms);

            //note: to return false may call parent'push_event_up,or stop
further routing when return true
            virtual bool  on_consensus_commit(const base::xvevent_t &
event,xcsobject_t* from_child,const int32_t cur_thread_id,const uint64_t
timenow_ms);

            //note: to return false may call parent'push_event_up,or stop
further routing when return true
```

```
            virtual bool  on_consensus_update(const base::xvevent_t &
event,xcsobject_t* from_child,const int32_t cur_thread_id,const uint64_t
timenow_ms);


            //call from lower layer to higher layer(parent)
            virtual bool  on_replicate_finish(const base::xvevent_t &
event,xcsobject_t* from_child,const int32_t cur_thread_id,const uint64_t
timenow_ms);

            //call from lower layer to higher layer(parent)
            virtual bool  on_certificate_finish(const base::xvevent_t &
event,xcsobject_t* from_child,const int32_t cur_thread_id,const uint64_t
timenow_ms);

        public: //help function and allow called from outside

            //proposal_start_event always go down from higher layer
            bool
fire_proposal_start_event(base::xvblock_t*proposal_block);//for leader start a
proposal
            bool    fire_proposal_start_event(base::xvblock_t*
latest_commit_block,base::xvblock_t* latest_lock_block,base::xvblock_t*
latest_cert_block);//just for replica to update information
   }
```

发起start_proposal事件辅助入口：

```
            //proposal_start_event always go down from higher layer
            bool
fire_proposal_start_event(base::xvblock_t*proposal_block);//for leader start a
proposal
            bool    fire_proposal_start_event(base::xvblock_t*
latest_commit_block,base::xvblock_t* latest_lock_block,base::xvblock_t*
latest_cert_block);//just for replica to update information
```

fire_proposal_start_event 最终转化成下面的on_proposal_start 事件投递进入共识对象体系

```
            virtual bool  on_proposal_start(const base::xvevent_t &
event,xcsobject_t* from_parent,const int32_t cur_thread_id,const uint64_t
timenow_ms);
```

一个Proposal共识成功或失败的结果事件通知：

```
        //call from lower layer to higher layer(parent)
        virtual bool  on_proposal_finish(const base::xvevent_t &
event,xcsobject_t* from_child,const int32_t cur_thread_id,const uint64_t
timenow_ms);
```

一个 Block转换成Commit 状态后的结果事件(参考xcsaccount_t::on_consensus_commit的处理)

```
        //note: to return false may call parent'push_event_up,or stop further
routing when return true
    virtual bool  on_consensus_commit(const base::xvevent_t &
event,xcsobject_t* from_child,const int32_t cur_thread_id,const uint64_t
timenow_ms);
```

一个Block状态的更新通知(参考xcsaccount_t::on_consensus_update的处理)

```
            virtual bool  on_consensus_update(const base::xvevent_t &
event,xcsobject_t* from_child,const int32_t cur_thread_id,const uint64_t
timenow_ms);
```

安全规则：

xconsrules.cpp 定义了如何防止分叉，达到强一致性的规则。