Bytom协议扩展性设计

比原链
朗豫

区块链扩展性

全网协议可升级

故障可收敛处理

可持续发展

维持区块链账本记录可靠

区块链协议升级

- 节点之间协调：如何使全节点都能适应最新规则
- 协议设计安全：新协议是否引入了漏洞，是否改变了一些预先限制
- 协议实现安全：新协议是否实现可靠，在程序中存在BUG
- 协议特性安全：新的特征是否如描述中的安全， 是否考虑了系统环境条件

协议升级措施

Soft Fork

核心是“Compatible”

是前向兼容的模型，概括特征：

- 以前不能用的现在还不能用
- 以前能用的现在可能不能用
- 非优雅升级
- 代价较小，升级持续时间长

Hard Fork

核心是“Revolution”

是完全兼容的模型，概括特征：

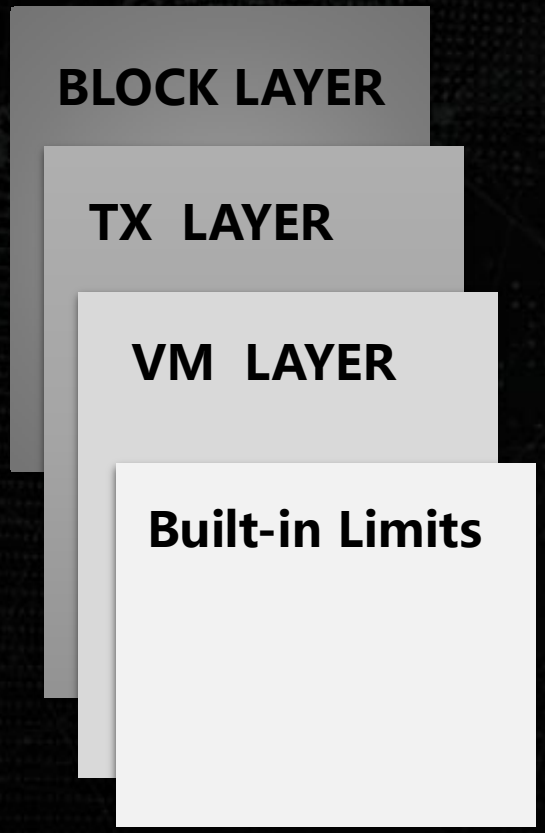
- 以前不能用的现在能用
- 优雅升级
- 代价巨大，升级快速

BIP99

详细说明各种分叉的特点和要求

HARD FORK的影响

Upgrade Affect



交易相关的软件可能可用， Signing或者Validation

几乎没有任何软件兼容

根据实际情况而定， Opcode和VM验证方式改变

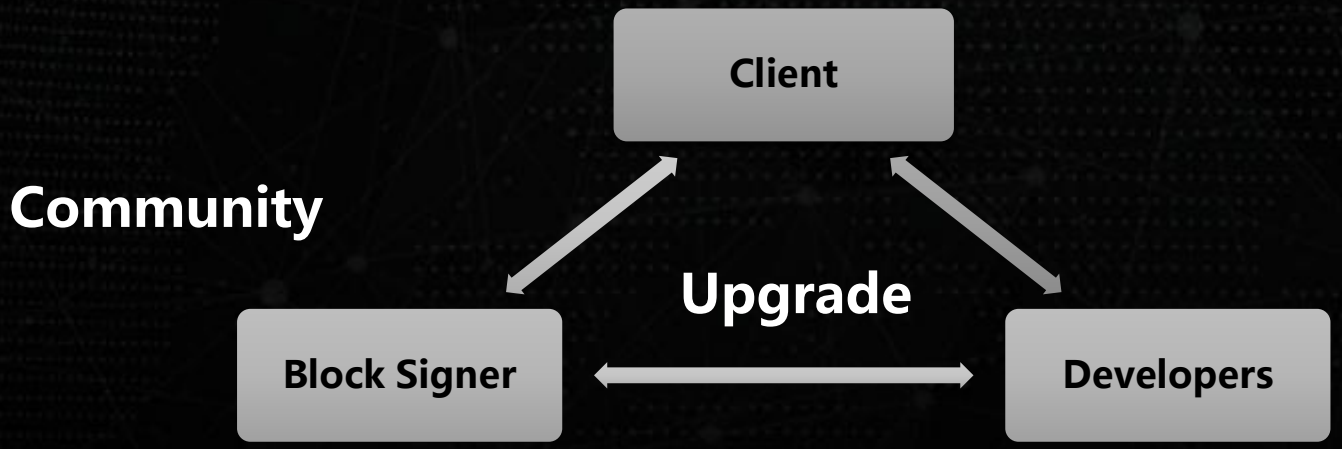
少量检验合法性的软件兼容

HARD FORK的不确定性

- 理想的HF：在通知所有节点升级程序， 共识维护者完全知晓，且绝对算力支持
- 不理想的HF：并非大部分节点做好准备，系统停止运转，共识维护者产生分歧，算力分为两派
- 失败的HF： 非计划的BUG， 发生非意料的双花和分叉，极难的监控， 导致资金受损

安全的SOFT FORK

- 扩展性升级：增加新的规则和特征，一些以前保留的数据位被使用
- 约束性升级：现存的规则被部分废除，过滤“无效”交易，常用于性能优化和功能淘汰



BYTOM升级实现

“Tick-Tock”循环模式

Alert 紧急消息

系统发生影响范围大的BUG，需要通过全网广播通知时， 发布相应Msg信息

Alert在Bitcoin 0.14中被设计为Deprecate特征， 但始终未被移除，考虑了弹性(Resilient)

BYTOM升级实现

Bytom从以下数据结构中设计升级策略:

1. Block
2. Transaction
3. Asset
4. ControlProgram

BLOCK_VERSION	协议版本
BLOCK_COMMITMENT	协议相关结构
[illegible]	[illegible]
TX_VERSION	交易版本
TX_COMMITMENT	交易组成数据
TX_WITNESS	交易辅助验证

ASSET_VERSION	资产版本
ASSET_COMMITMENT	“存,取”辅助数据
ASSET_WITNESS	额外辅助验证
VM_VERSION	虚拟机版本
NOP* INSTRUCTIONS	保留指令

UPGRADE EXAMPLE

场景: Bytom系统中主要使用SHA3算法来进行做核心数据的消息摘要， 由于漏洞出现SHA3 要进行升级成 “SHA4”

BLOCK HEADER	BLOCK HEADER	TRANSACTION
BLOCK_COMMITMENT SHA3(MERKLE_HASH) SHA3(PERV_HASH)	BLOCK_WITNESS SHA3_BITS SHA3_NOUNCE	TX_COMMITMENT SHA3(ASSET_ID) SHA3(VM_PROGRAM)
SHA4(MERKEL_HASH) SHA4(PERV_HASH)	SHA4_BITS SHA4_NOUNCE	SHA4(ASSET_RELATED)

UPGRADE EXAMPLE

THEN :

设定升级最后高度期限, 通过Alert信号通知客户端升级

BLOCK HEADER

INCR_BLOCK_VERSION

TX HEADER

INCR_TX_VERSION

到某个高度时升级成功， 未更新的客户端无法使用， 协议切换成SHA4支持模式

ONE MORE THING...

- 所有的改变都需要的算力和社区的共识: 通过算力投票是唯一解决方法， 相比于传统比特币的coinbase Signal 方法， Bytom设计了MintProgram的合约，在进行复杂条件的决策环境下，更加简单

- Bytom 近况: 11月底release Testnet Beta版， 代号SPARK（“晓”），欢迎大家体验

https://github.com/Bytom/bytom **所有代码已经开源**


THANKS