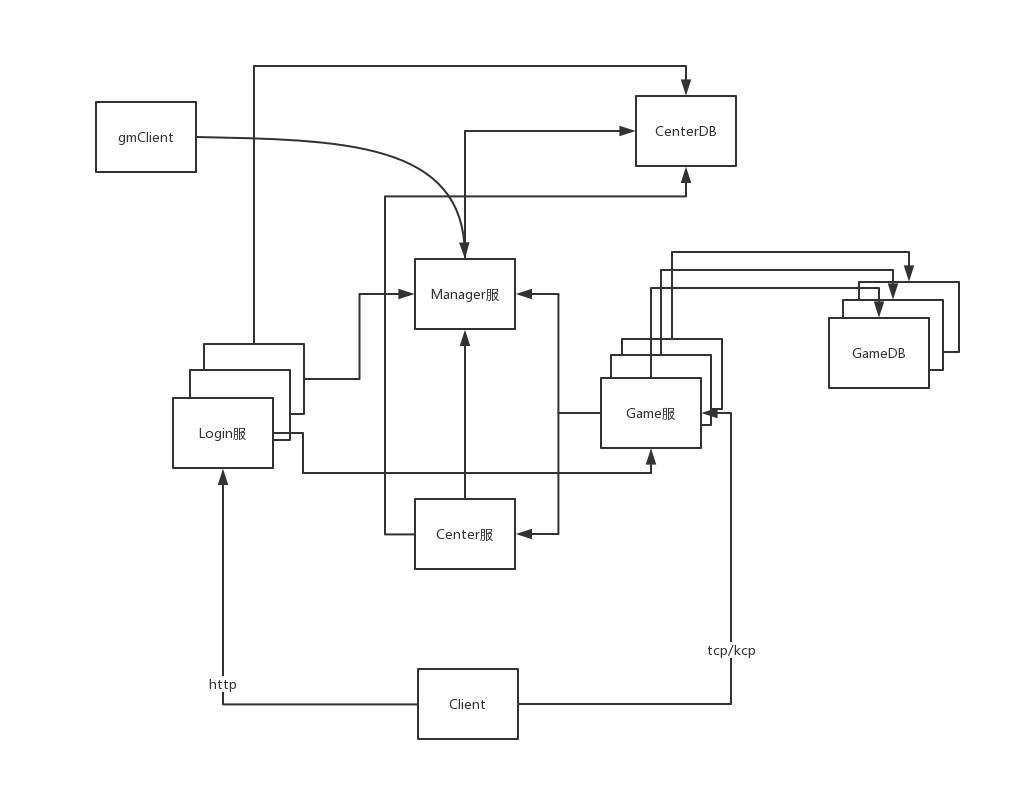
服务器架构与登陆流程

架构图:



进程种类:

Manager：管理服，唯一节点，启动服务器时，先启动该节点，作为中控的存在，也作为配置中心与服务发现管理节点,无具体业务逻辑，只有，版本更新/热更，配置更新，gm指令操作转发，热合服，等等操作，会从这里开始下达指令。center/login/game 主动连接此节点,并通过该服通知其他节点做服务发现。

Center:中心服，唯一节点，负责业务逻辑中的全局逻辑部分，如，全服唯一排行榜，全服唯一角色池，唯一匹配，全服工会列表等逻辑，（不包括登陆，登陆流程步不通过中心服）

Game:逻辑服，多节点，负责具体单服业务逻辑，无论分服模式还是大区同服模式，每个game服，对应一个数据实例（自身区服数据库），每个角色从创建时会绑定创建时的逻辑服区ID(areaID)(即使合服,areaID依然不变)，此概念可称呼为源服 (创建所在服)。每个角色的数据改变，只能由源服回写DB，其他服如需要读写其他服玩家数据，也都通过异步事务来操作。所有的Game服会连接center服，game服之间也会相互连接，连接方式是id大的主动连接id小的，如game2服会主动连接game1服。

Login:登陆服,多节点,负责客户端登陆流程，平台验证,账号创建(User),绑定游客，获取服务器列表，客户端版本号等业务。login服处理用户业务时，是按uid进行hash散列，如收到散列值与自身不相同的消息，会转发到对应的login服上进行处理。login服不连接center,每个login服都连接所有的game服,并且login服之间也会相互连接，连接方式㛑是id大的主动连接id小的。

Client:客户端

gmClient:gm平台客户端(gm用来发指令到Manager服上进行gm指令操作)

线程结构:

每个进程内的线程结构基本一致，处理具体业务的只有 主线程和池线程，其他线程都是业务无关线程。所有的线程，都是继承于java的Thread，并且线程内自行tick时间(TimeDriver)，各类时间任务回调，也均在当前线程。

下文中的n，一般指8条(或cpu核数)(无需\*2，因为线程类型多，够顶满cpu)

MainThread:主线程,1条，处理全局业务逻辑,如game服的上下线，本服排行，跨服切换等,

center服的全局业务逻辑等。

PoolThread:池线程，n条，处理主要业务逻辑(可分片的，吃性能的)，如game服的每个Player的业务逻辑与场景战斗逻辑。

IOThread:读写线程，n条,处理网络收发包，序列化/反序列化等。

DBThread:数据库操作线程,n条，处理常规 增删改查操作（可Batch的）。

DBWriteThread:数据库同步线程，1条，每隔一定时间(默认30s)，同步当前内存中数据到数据库中，会将所有项，对应的进行Batch，然后以事务方式提交，如失败，此次操作回滚。