openGauss 社区需求check-in评审报告

评审纪要

时间：

特性名称：

特性责任人：

评审人：

设计checklist

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
GUC参数	功能	新增GUC参数适配项	①guc.cpp里需要新增guc参数的相应代码，包括参数的校验/赋值等功能函数； ②如果要加入postgres.conf，需要修改postgresql_single.conf.sample 路径：src/common/backend/utils/misc/ 一般以注释方式加入，如果非常确认使用某个默认值，也可以以非注释方式加入 ③如果要允许gs_guc工具设置，请修改cluster_guc.conf 路径：src/bin/gs_guc/cluster_guc.conf	不涉及	完成
升级	兼容性	升级脚本兼容性	对于新增的升级脚本，需要检查是否兼容A、B、C、PG这几类数据库（默认兼容O*）。	不涉及	完成
升级	兼容性	版本号变更	src/common/backend/utils/init/globals.cpp ：GRAND_VERSION_NUM	不涉及	完成
升级	兼容性	前向兼容性	1）涉及系统表修改的特性，必须提供系统表升级脚本和回滚脚本，修改版本号文件 2）对于涉及修改持久化数据（如日志）的特性，必须考虑新老版本共存时的兼容性场景，必	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			要情况下，需要增加版本号以进行识别 3）对于涉及修改执行态数据格式（如syscache结构）的特性，必须考虑新老版本共存时的兼容性场景，必要情况下，需要增加版本号以进行识别		
升级	兼容性	验证升级正常	涉及到升级时需要验证如下场景： 1. 集群环境安装：该步骤会验证集群环境安装是否正常 2. 集群环境升级：从openGauss老的版本（1.1.0、2.0.0、3.0.0版本）升级到最新版本是否正常；回滚是否正常；再升级和升级提交是否正常	不涉及	完成
升级	兼容性	前向兼容性	1）涉及系统表修改的特性，必须提供系统表升级脚本和回滚脚本，修改版本号文件 2）对于涉及修改持久化数据（如xlog）的特性，必须考虑新老版本共存时的兼容性场景，必要情况下，需要增加版本号以进行识别 3）对于涉及修改执行态数据格式（如syscache结构）的特性，必须考虑新老版本共存时的兼容性场景，必要情况下，需要增加版本号以进行识别	不涉及	完成
升级	功能	升级工具功能或相关功能变动	1）涉及升级工具gs_upgradectl的修改，需要验证升级到最新版本的就地和灰度升级、回滚、再升级和升级提交是否正常 2）涉及集群管理工具(例如om或cm)的公共函数修改，需要验证升级到最新版本的就地和灰度升级、回滚、再升级和升级提交是否正常动	不涉及	完成
内存	内存	新创建共享内存	注意不要在main.cpp中创建（ProcessMemoryContext处），需要在postmaster.cpp中InitializeWorkloadManager()后进行创建，否则，共享内存不受资源管理的管控。	不涉及	完成
内存	内存	内存大容量边界设计	对于内存使用要考虑大容量边界的设计： 1.缓存大量数据场景。	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			2.长事务/循环中分配大量小块内存需要显式调用pfree_ext释放。		
内存	内存	析构函数	GAUSSDB在线程异常退出时，会释放线程上变量的内存并递归调用线程上所有类的析构函数，为避免double free的情况，需通过判断CurrentResourceOwner是否为空，如果为空，则表示线程退出中，此时仅需要将变量置为NULL即可	不涉及	完成
公共数据结构	功能	pg_try	PG_TRY使用注意事项 ① PG_CATCH中不能使用return，否则会出现随机core； ② PG_CATCH中一定要记得清理eredata（FlushErrorState();），否则会出现暴栈；	不涉及	完成
公共数据结构	功能	hashtable	hashtable使用注意事项 ①在使用hash表时，entry必须封装成一个结构体，第一个成员必须是hash key 否则会造成查找失败！！ ②封装的结构体必须只有两个成员：key+value，即value需要封装起来 否则hash表在search的时候会core！！ ③hash表在创建的时候，entry 的size一定要正确 否则在enter的时候会造成写越界，不会报错但是其它变量内存会被踩，不易发现！！ ④在遍历hash表的循环过程中，使用hash_seq_init初始化游标时重点审视是否存在重复问题，是否符合预期。	不涉及	完成
公共数据结构	功能	禁止使用commandTag 判断Query的类型	需要对SELECT\INSERT\UPDATE\DELETE\MERGE INTO等query类型进行识别判断时，应该使用CmdType进行判断或新增标识。 禁止使用commandTag作为判断标识，因为commandTag在带returning*语句执行中变化。	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
系统函数	兼容性	系统表升级是否正确	通过对比initdb和升级后的db中pg_proc每一列的值，避免出现升级和非升级不一致的情况，尤其是strict、voliatle属性，并且升级场景中函数注释也需要添加。	不涉及	完成
系统函数	兼容性	新增数据类型适配FENCED函数	新增数据类型如果是FENCED函数使用，需要修改UDFArgumentHandler函数，把数据类型添加到对应收发分支。	不涉及	完成
系统函数	功能	NULL值处理	对于非strict函数允许接收NULL入参，在函数逻辑中应对NULL值进行细致的处理	不涉及	完成
前向兼容性	兼容性	SQL语法兼容性	禁止对已经支持的语法进行更改，包括禁用、语法变更。包括： 1. 对gram.y已经支持的语法进行删除或修改。 2. 对pl_gram.y已经支持的语法进行删除或修改。	不涉及	完成
前向兼容性	兼容性	FUNCTION兼容性	禁止对函数的对外接口更改（如函数名、入参个数、参数类型）	不涉及	完成
前向兼容性	兼容性	Node相关数据结构删减	禁止对Node相关数据结构（如Query、RangeTblEntry）数据成员删减	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
兼容性	兼容性	兼容性设计原则	目前支持兼容性PG,A,B,C;对于每一个兼容点要求： 1、如果该兼容点为非冲突兼容点，应该是各个兼容模式下均支持，不需要开关控制 2、如果该兼容性点为冲突兼容点，应该由兼容开关控制。	不涉及	完成
查询解析	兼容性	添加关键字场景	为了尽可能的防止移进规约冲突，最大化的用关键字作为对象名，关键字分有4类：非保留关键字、列名关键字、类型函数关键字、保留关键字。他们的约束逐渐加强，比如保留关键字不可以用做表名。添加关键字需要考虑如下： 1. 添加新的关键字最好先作为非保留关键字添加 2. 如果要加其他关键字类型那么需要考虑是否有前向兼容问题。比如排查客户场景是否有用该关键字作为对象名。 3. 关键字还需要加入kwlist.h文件中。	不涉及	完成
查询解析	兼容性	DML语句targetlist补全场景	如果新加DML类型语句（如INSERT/UPDATE/DELETE/MERGEINTO），需要考虑用户输入列可能没有完全覆盖目标表列的场景，比如t1表有两列(a int, b int)但是用户只插入一列insert into t1(a) values(1)，这时targetlist会有缺失列的情况，需要对缺失的列补全默认值。	不涉及	完成
查询解析	兼容性	Gram.y中处理语法冲突	1、合理使用a_expr/b_expr/c_expr使用场景 为了尽可能防止冲突，划分了a_expr/b_expr/c_expr，比如BETWEEN a_expr AND b_expr中 AND属于a_expr范畴，b_expr中不含AND。否则就无法区分BETWEEN之后是Bool判断还是范围。c_expr是a_expr和b_expr的交集。所以 尽量用a_expr这样支持的范围最广。如果用b_expr/c_expr 那需要考虑潜在的用户问题场景（参考BETWEEN...AND...例子）并在手册中写清楚约束。	不涉及	完成
查询	功能	Var获取	var的获取除了使用pull_var_clause()，函数无法获取sublink子查询（Query）中的var，需要：	不涉	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
重写			1.检查投影中sublink中的子查询（Query） 2.检查Where/On/using中的sublink中的子查询（Query） 如果没有获取到sublink中子查询（Query）的var，将会导致对这些var缺少对应的处理，例如：levelsup, varno等，造成后续处理的错误。	及	
查询重写	功能	var levelsup场景	考虑子查询中包含var的varlevelsup > 0的场景，需要特殊处理，否则var找错层次可能导致找不到 1.Var的levelsup在子查询提升时是否已经根据新的层次做了调整 2.Var的varno在子查询提升之后是否根据新的位置做了调整	不涉及	完成
查询重写	功能	join alias var场景	考虑查询中包含显式join的场景（即语句中有 t1 join t2 on 的形式），此时会生成Join relation，包含joinaliasvar，此时需要考虑在targetlist和qual中引用基表的var和join表的joinaliasvar 1.在preprocess_expression函数之前，join alias var需要注意处理 2.在preprocess_expression函数之后，join alias var会被基表的列取代，需要注意处理 3.没有基表的compute Var需要注意处理	不涉及	完成
查询重写	功能	等价转换关于NULL和空集的设计	设计新的查询重写规则，要考虑NULL值和空集合的正确性，否则会引起结果错误。 1.在设计查询重写时避免只考虑Inner Join的情况，Left/Semi/Anti/Full Join由于会引入NULL值，需要注意查询重写过程中对NULL值的处理。 2.查询重写必须增加和NULL值/空集相关的测试用例	不涉及	完成
查询	功能	Query所有成员变量测试	考虑Query中所有成员变量是否为空的场景，并设计用例进行测试	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
重写					
路径生成	功能	等价类的使用	1、在单机环境中等价类用于判断两列是否等价 2.和数据类型相关的函数、操作符修改时，需要考虑对等价类的影响，修改相关重分布函数 3.使用等价类时请确认使用的是最终等价类（等价类使用指针）	不涉及	完成
路径生成	功能	dummy rel	路径生成过程中可能会生成dummy rel，行数、path等都是空的，可以通过is_dummy_rel()函数判断，避免访问空指针等问题	不涉及	完成
路径生成	功能	append rel	考虑append rel的路径，一般是hdfs内表的扫描场景，或者多分支targetlist数据类型一致的union all场景 1.append rel作为子节点通常是参照一个模板拷贝出来的，因此需要注意调整它的Var的信息	不涉及	完成
路径生成	功能	数据稳定性	1.limit场景：路径生成时需要考虑个DN数据不稳定的场景，比如replicate表limit场景，每个DN在limit之后数据可能不同，如果limit之后再join可能造成数据增多的情况发生。其他还包括排序算法不稳定，造成非排序列在各DN顺序不一致引发的问题。如上场景需要考虑在单DN执行，从而获得一致的结果。	不涉及	完成
计划生成	功能	append剪枝计划	在setop场景中要进行DN裁剪要考虑各分支计划的执行节点的设置，必须一致。由于执行层每个线程对应的producer必须一致，所以要求执行setop算子的DN必须与每个分支stream算子consumer的DN一致才行	不涉及	完成
计划生成	功能	出现重复列的场景	包含group by列的场景，考虑targetlist中包含多个重复列（其中一个为group by列），及group by列中引用重复列的边角场景，可能在聚集处理时由于删除掉相同列出现找不到的问题，例如：select a,a,a from t group by 1,1,1; 中group	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			by 列是相同的一列。 1. 需要构造相应的SQL语句包含group by重复列的用例		
计划生成	功能	数据类型处理	考虑数据类型处理时要考虑到Relabeltype, FuncCall场景，要防止计算过程中对表达式处理时在targetlist上增加基表var导致上层找不到 1. 增加新的Var时不能直接增加基表的Var，需要参考计划的各层投影的类型来判断是否需要增加的Var是对中间层投影的引用	不涉及	完成
计划生成	功能	在路径上/计划上人为添加算子	一般不建议在路径/计划上强制添加算子，比如强制添加Broadcast算子，而是让优化器自动生成。如果必须添加算子，那么需要注意执行器是否支持这些算子组合。 1. 不要添加两个连续的stream算子，比如Gather&&BroadCast算子。 2. 确认执行器支持这种算子组合。	不涉及	完成
计划生成	功能	与其它典型场景交互	考虑与AP函数，count(distinct)，windowagg等典型计划生成场景的交互	不涉及	完成
计划生成	DFX	DN裁剪问题	如果DN裁剪发生在DN 上时（如 with recursive），需要考虑 1. 同一线程需要push down execnodes 2. explain DFX 如何正确清楚的显示。 否则出现hang时用explain 定位不了。	不涉及	完成
计划生成	功能	subplan dop 问题	目前subplan不支持并行，smp修改涉及subplan、initplan时增加相应场景测试。如果有特殊的dop场景，需要进行适配。需要： 1. 单独考虑并测试subplan initplan场景	不涉及	完成
行数估算	性能	考虑NULL值	常用的统计信息DISTINCT值、MCV等都不包含NULL值，因此在实际计算时需要先对表的行数乘以(1-nullfrac)才是非NULL值行数，null值不可以和任何值（包括null值）join	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
行数估算	性能	考虑多列相关性	当存在两个以上对同一个表的过滤、连接条件时，考虑是否需要使用多列统计信息，或者在一些不方便使用多列统计信息的场景（例如计算hashbucket时）使用补偿手段（例如0.75折减或者取最小值等）	不涉及	完成
代价模型	性能	算子代价模型修改	算子代价模型修改要慎重，需要确保如下几点做到位： 1. 有明确的问题场景。 2. 根因比较明确。 3. 修改之后需要全量验证测试看护场景，如TPCC、TPCH、TPCDS等，以防性能看护场景劣化。	不涉及	完成
代价模型	性能	新增算子代价模型	新增算子代价模型需要考虑如下几点： 1. 适配SMP自适应、内存自适应。 2. 不同dop下折减系数。	不涉及	完成
ANALYZE	功能	涉及场景	更改Analyze的核心流程时，需要对下列场景进行测试： 1. 百分比统计信息、非百分比统计信息 2.行存、列存、hdfs内表、hdfs外表(hdfs内表分为主表、delta表)、obs外表。 3.Hash表、复制表	不涉及	完成
ANALYZE	功能	delta表的表名在每个DN不同	不能假设HDFS表中delta表的表名在各个DN相同，如果做此假设，可能会导致某些DN找不到相应的delta表。	不涉及	完成
ANALYZE	功能	等概率采样	采样算法需要保证每行元组均已等概论被选中。 考虑采样算法在下列场景的适用性： 多次执行批量插入、批量删除。这可能导致表中有大量连续空页面、大量连续的满页面。	不涉及	完成
其他	功能	优化器中全局变量使用	尽量不要用全局变量，如果实在要用全局变量，那么需要考虑若干问题： 1. subquery_planner中的全局变量，每个子查询都会递归调用，这样就需要全局变量适配子查询	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			2.异常退出之后全局变量内存释放和置空，否则会在下次用时造成非法内存访问。 3.PBE场景和一般的exec_simple_query流程不同，需要单独做测试。		
其他	功能	PBE执行方式	考虑PBE执行方式并进行测试，包括通过JDBC/ODBC接口，以及在存储过程中使用等方式	不涉及	完成
其他	功能	新增已有的数据结构成员适配序列化与反序列化	注意适配：src/common/backend/nodes/copyfuncs.cpp src/common/backend/nodes/equalfuncs.cpp src/common/backend/nodes/outfuncs.cpp src/common/backend/nodes/readfuncs.cpp 其中，outfuncs.cpp需要使用版本号workingVersionNum判断，readfuncs.cpp使用宏IF_EXIST判断。	不涉及	完成
执行器	性能	if判断加速	尽量不在高频函数上增加if判断，如果一定要增加，则必须用likely()/unlikely()做branch miss优化。	不涉及	完成
执行器	性能	耗时处信号响应	在很耗时的逻辑处，在不特别影响性能的情况下要响应cancle信号	涉及	完成
执行器	性能	模板	对于耗时（执行中反复碰到的逻辑），要利用C++模板特性，讲在运行时不变的量抽取到模板函数的参数列表里，加速查询	不涉及	完成
执行器	性能	内存分配加速	如果有很多小块内存分配，可以分配一个大块，然后算法逻辑里进行切分，避免反复调用内存分配接口	涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
执行器	性能	初始化函数	在算子的初始化函数里不要进行大段时间消耗的操作，如果有移到运行时去做	不涉及	完成
执行器	兼容性	explain performance的适配	新算子一定要记得适配explain performance，并且因为performance多种格式展示，需要逐个适配	不涉及	完成
执行器	兼容性	COST设计	新算子需要适配优化的COST设计，根据当前算子的算法适配其COST估算	不涉及	完成
执行器	兼容性	行存和向量化	新算子也需要适配两个不同的执行框架，行存和向量化	不涉及	完成
执行器	兼容性	active SQL适配	对于activesql中视图的监控项目（如下盘大小等）需在设计时考虑适配	不涉及	完成
执行器	兼容性	架构适配	算子需要考虑SMP 1 如果涉及数据扫描，要进行数据切分处理 2 如果不涉及数据扫描，考虑算法过程中涉及不涉及输入输出流的切分	不涉及	完成
执行器	兼容性	codegen适配	向量化hashjoin/agg/cstorescan 如果核心数据结构和算法改变需要进行codegen适配	不涉及	完成
执行器	兼容性	内存估算	需要考虑内存估算，适配内存自适应	不涉及	完成
执行	内存	内存不足场景	算子关键算法处理，需要考虑分配大块内存但是无法分配的场景。	不涉	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
器				及	
执行器	内存	TOP memorycontext分配	执行内存不允许在TOP MEMORY CONTEXT上分配	涉及	完成
执行器	内存	跨线程分配	原则上不允许跨线程的内存传递，如果一定要传递请在share memory context里面分配	涉及	完成
执行器	内存	内存生命周期管理	内存分成长寿命和短寿命两种不同的生命周期 短寿命的一定记得及时reset context，否则会造成内存堆积	不涉及	完成
执行器	内存	初始化函数	在初始化函数里面不要进行大块内存分配，如果有移到运行时去做	不涉及	完成
执行器	内存		不允许用原生的malloc/free进行内存分配、释放	涉及	完成
执行器	DFX	第三方组件	使用第三方组件进行初始化或者处理的时候，需要用PG_TRY/PG_CATCH进行包裹，以免第三方组件throw exception没有捕获导致系统异常	涉及	完成
执行器	并发控制	临界区访问	多线程对于临界区访问，统一都要加锁，不能依赖volatile语义，改成无锁算法需要进行评审	涉及	完成
公共	功能	GUC参数开关和默认值	新增guc参数，如果是功能开关需要调查清楚用户需求，是否所有用户都需要，是否要默认打开。 新增guc参数默认值要充分分析场景，设置合理。	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
公共	功能	验证升级正常	升级涉及到内核模块、临接模块如ReDis、OM的行为变更，需要验证CM机制是否受影响	不涉及	完成
公共	安全	命令行执行	使用system函数，popen函数，注意命令输入来源，避免产生外部注入风险。	不涉及	完成
公共	安全	随机数	使用随机数生成函数，如果是用于加密等安全场景，请不要使用伪随机数函数rand()和random()。	不涉及	完成
公共	日志	日志规范性	关键流程、关键状态数据要有日志，日志简洁规范，便于分析定位，避免过多过杂，循环流程中避免不要打印日志。	涉及	完成
公共	日志	日志规范性	在非预期行为发生的路径是否增加了ERROR及以上级别的日志打印	涉及	完成
公共	日志	日志规范性	新增日志中是否打印了用户密码等隐私信息	不涉及	完成
集群管理	性能	避免功能在主备实例上重复执行	CM Server有主备，例行功能不要在CMserver主备上重复执行，只需在主上执行即可。 例如磁盘检测功能，不需要在CM Server主备上重复执行，影响性能。	不涉及	完成
集群管理	性能	避免功能在主备实例上重复执行	修改流程、调整实例的状态检测时间、心跳超时时间、仲裁延时时间，要参照RTO公式文档， 充分考虑对角色仲裁、故障恢复性能的影响。	不涉及	完成
集群	兼容性	与老版本兼容可升级	新增功能特性是否与老版本兼容，pg_class等系统表的修改要经过社区sig组评审，做好版本升级等代码适配修改。	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
管理					
集群管理	内存	结构体设计要合理	新增结构体设计要合理，结构体中的变量尽量精简，避免包罗万象，实际使用时只用其中一部分。	不涉及	完成
集群管理	功能	消息通信	cm_ctl, cm_agent与cm_server之间消息通信，要考虑建立连接超时，消息发送失败重发，重要消息发送要应答确认等增强可重入机制。	不涉及	完成
集群管理	功能	对周边模块影响和依赖	新增功能，分析对周边模块的影响性，拉通周边模块一起设计，解决依赖项，避免遗漏。	不涉及	完成
集群管理	功能	新增或变更命令行	新增命令行，要有超时退出机制，避免发生故障时，命令hang住，不退出，不清理执行结果。cm_ctl 和 cm_server前后端超时时间要一致。	不涉及	完成
集群管理	功能	新增或变更命令行	要考虑不同命令行之间是否有冲突，设计上需要建立互斥机制，避免同一时间执行不同的命令相互冲突。例如cm_ctl build 的同时，不能执行cm_ctl switchover.	不涉及	完成
集群管理	功能	端口号、资源分配	在进程中，尽量避免fork子进程，子进程会继承父进程的端口等资源， 避免父进程退出，子进程没有退出，占用资源不释放的问题。	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
集群管理	DFX	异常场景	要考虑无ETCD，CM Server主备之间网络断开场景，数据未同步对功能的影响。	不涉及	完成
集群管理	DFX	异常场景	充分考虑各类异常场景处理，检查每一种故障场景，故障处理无遗漏。	不涉及	完成
集群管理	DFX	关键信息可查询	关键状态信息，配置文件，环境变量等提供诊断命令、日志打印等查询手段，方便快速分析定位。	不涉及	完成
集群管理	DFX	故障告警和告警恢复	实例故障，设备故障等情况，及时上报告警，引导用户及时接入排除问题。	不涉及	完成
集群管理	安全	端口防攻击	监听端口接收消息要考虑防攻击，消息鉴权，消息长度检测防止读越界，设置IP白名单等防护措施。	不涉及	完成
通信模块	性能	新增需求是否涉及消息转发	代理转发过程如果涉及锁，系统调用等操作，需要评估代理转发过程是否会成为性能瓶颈（需要考虑不同的并发场景和包的大小）	不涉及	完成
通信模块	性能	性能数据是否可以衡量	对于时延类问题的分析是否有打点数据的统计，对于可能耗时较长的逻辑是否增加到explain performance报告中	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
通信模块	前向兼容	端口配置变更是否前向兼容	修改涉及增减端口的问题时需要注意升级适配，包括：（a). 集群环境安装：该步骤会验证集群环境安装是否正常 (b). 集群环境升级：从老的版本升级到最新版本是否正常（包括小版本升级和大版本升级）	不涉及	完成
通信模块	前向兼容	端口配置变更是否前向兼容	是否修改系统表	不涉及	完成
通信模块	内存	内存是否受管控	通讯库内部的内存使用是否使用了内存上下文的统一管理机制	不涉及	完成
通信模块	内存	数据缓存buffer是否可以动态调整	对于长时间空闲的数据缓存buffer，是否可以动态释放并在需要的时候再次申请	不涉及	完成
通信模块	内存	内存计数是否准确	通讯库内部内存申请和释放过程中的内存记录是否准确	不涉及	完成
通信模块	并发	慢启动	对于瞬间的高并发作业的场景，是否保证慢启动，避免瞬间压力过大导致通讯库内存爆涨，丢包率上升等问题	不涉及	完成
通信模块	并发	拥塞控制	高并发场景是否针对全部链接有拥塞控制算法，避免数据大量缓存在通讯库内部中	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
通信模块	并发	锁的并发管理	锁的使用顺序应按照锁的粒度单调下降，最小范围的锁下不应有任何等锁的逻辑	不涉及	完成
通信模块	并发		锁的使用是否输出对应加锁放锁的流程图，确保多线程之间不会互锁	不涉及	完成
通信模块	DFX	通讯信息是否可追溯	对于每一个连接的收发包总数量，收发字节大小，创建时间，对应的线程等信息是否有视图等工具可以追溯	不涉及	完成
通信模块	DFX	单元测试和系统测试是否完善	新增的需求是否给定了对应的UT和系统测试用例设计	不涉及	完成
通信模块	功能	链接复用	对于复用的链接，新的链接的数据缓存Buffer是否保证被清空，避免使用残留数据	不涉及	完成
通信模块	功能	连接建立&销毁	建立的socket在异常场景下是否被正常关闭，不会造成fd泄露	不涉及	完成
通信模块	功能	连接建立&销毁	是否由同一个线程去关闭连接，不会存在double close造成的fd错关问题	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
通信模块	功能	连接建立&销毁	建立和使用的逻辑是否有时间窗口，避免使用老的连接发送数据（如主备切换场景下，消息发给了备机）	不涉及	完成
通信模块	功能	连接建立&销毁	保存连接的数据结构在建立连接之前是否初始化（注意：0为有效fd，因此在palloc0后需要手动将socket初始化为-1），连接关闭后socket是否有复位	不涉及	完成
通信模块	功能	连接池，内存池的使用	在获取和释放的时候都有准确的修改池内可用数据结构的个数	不涉及	完成
通信模块	功能	连接池，内存池的使用	在使用时判断数组下标是否超出最大Size，避免踩踏内存	不涉及	完成
通信模块	功能	新增对外接口	接口中是否有加锁的逻辑，如果有是否屏蔽了信号，并且在接口返回前检查了屏蔽信号期间未处理的信号	不涉及	完成
通信模块	安全	新增协议	新增网络协议时，是否考虑了消息的完整性校验，是否符合安全可信原则	不涉及	完成
通信模块	安全	新增的通讯链路	对于新增的网络连接，对集群外部的连接，是否对收发数据的接口增加了相关的安全算法的校验； 对于内部的连接，是否包括kbr鉴权过程	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
公共子系统	内存管理	接口函数与外部声明函数是否一致；	新增内存上下文或者其他内存管理接口函数需要对外暴露的接口是一致的，用户便于理解；	不涉及	完成
公共子系统	内存管理	新增内存上下文算法用途和使用方式是否有详细说明；	新增内存上下文对外暴露接口，需要在接口声明部分给出详细说明，例如palloc的结果需要检验等。	不涉及	完成
公共子系统	内存管理	使用步骤是否简洁，是否提供全流程管理（如分配、释放、检查等）	在实现内存上下文算法时，必须将函数集合里面的所有接口定义进行实现	不涉及	完成
公共子系统	内存管理	内存上下文的初始化内存大小是否符合要求	高效内存使用，防止浪费，各模块负责	涉及	完成
公共子	内存	新增内存上下文初始化是否	所有内存上下文的使用需要在初始化后进行，否则不受内存上限管控	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
系统	管理	在逻辑内存管理初始化之后进行			
公共子系统	内存管理	内存上下文生命周期是否满足预期	每个内存上下文在声明时需要给出生命周期，例如是线程级别、session级别、事务级别、算子级别等	不涉及	完成
公共子系统	内存管理	共享内存上下文上使用内存是否过大、是否存在上限管控	共享内存上下文不能分配过大，如超过2G，否则应考虑设计是否合理；对于一些内存上下文用于保留信息，可以通过上限管控防止内存使用超过设计阈值	不涉及	完成
公共子系统	内存管理	对于超1G内存使用，是否将内存实时记录到内存管理器，供查询	便于查找是否存在超大内存使用，若只通过内存上下文很难定位	不涉及	完成
公共子	内存管理	非数据库中内存上下文中内存使用，	该内存不受内存管理上限控制，在内存泄露场景下，需要给出定位手段	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
系统		是否提供接口供查询内存使用			
公共子系统	内存管理	局部变量内存释放，非程序结尾位置，释放进行置空；全局内存变量释放，无论什么位置均需要置空	防止内存使用出现use-after-free模式，内存越界	涉及	完成
公共子系统	内存管理	对于使用数组、下标访问内存操作场景，是否提前判断下标符合内存空间要求	防止出现heap overflow场景	涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
公共子系统	内存管理	内存分配报错日志中，是否需要新增信息，便于分析报错原因	内存报错场景下，快速定位内存报错原因	涉及	完成
资源管理	兼容性	只能新增系统表参数，不可以删除已有的参数	前向兼容	不涉及	完成
资源管理	功能	新增系统表参数的默认初始化数值要合理，不影响已有的功能	前向兼容	不涉及	完成
资源管理	兼容性	修改资源池系统表参数后，对升级自验	是否适配就地、灰度升级的脚本	不涉及	完成
资源管理	功能	对外暴露的参数，在易用性	提供好理解的参数列表，对外提供一致性的能力	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
		上满足客户要求			
资源管理	功能	机器重启后，在低内核版本（suse11、redhat6）上，控制组会消失，需要重新加载，需要提供机制保证重启机器后，加载控制组必须有效。	未合理加载控制组，将影响集群启动	不涉及	完成
资源管理	兼容性	新增的控制组件是否已在控制组内包含，受统一的管理	例如cm/gaussdb均由gs_cgroup控制组来管控，对外提供统一的接口	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
资源管理	功能	控制组的默认资源配比是否合适	数据库与非数据库、查询作业与常驻线程间资源配比要满足大部分场景要求	不涉及	完成
资源管理	兼容性	控制组配置文件中，若新增数据结构或成员，需要适配升级脚本	gs_cgroup中upgrade参数来适配改动，否则影响升级、扩容和节点替换后，控制组成员信息不一致	不涉及	完成
资源管理	内存	控制组配置文件中，字符串成员，是否通过8字节地址对齐	通过映射文件访问结构体成员，地址必须8字节对齐，否则可能出现SIGBUS的core	不涉及	完成
资源管理	性能	资源池内只需要对复杂语句进行资源管控，对简单语句不需要资源管控，	简单语句对外查询视图无法获取内存使用信息	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
		否则引起性能下降			
资源管理	功能	CPU资源统计需要是间隔方式统计，而非定点查询统计	降低统计时引起的误差，对于时间间隔（一个时间周期内）进行统计，可以有效降低偏差率	不涉及	完成
资源管理	功能	CPU、CPUSET和CPUACCT控制组层次结构成员必须一致	对于相同控制组，进行cpu资源分析时，需要考虑各控制组的依赖，若不一致，则分析结果有误	不涉及	完成
资源管理	功能	明确查询语句是否受IO资源管控，查询视图要统一	对外暴露一致的视图信息，查询该语句是否受IO资源管控	不涉及	完成
资源管理	功能	IO资源统计是否根据磁盘类型或使用场景做区分	对于虚拟盘和物理盘，IO资源管控检查是不一致的，需要针对不同场景给出不同IO资源统计	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
资源管理	兼容性	新增IO资源管控类型要保持对外接口一致	在管控、统计等维度，IO资源管控提供统一的信息视图	不涉及	完成
资源管理	DFX	对于统计存储空间，是否提供校准机制	统计类型的实现会出现误差，通过校准有效避免统计误差过大情况	不涉及	完成
资源管理	功能	超过控制阈值后，是否对终止查询语句提供一致性的触发机制	获取查询语句信息列表，对所有要取消的语句发送cancel信号	不涉及	完成
资源管理	DFX	新增并发排队机制是否有后台校准能力	在排队不准确时释放可以后台校准	不涉及	完成
资源管理	DFX	新增并发排队机制是否可以线程内部自愈	出现异常分支时，可以在入口和出口地方，进行线程级别的自愈处理	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
资源管理	功能	并发排队中简单和负载语句判断条件是否合理	对外体现合理的判断条件：例如内存使用、CPU和IO评估等	不涉及	完成
资源管理	功能	特殊排队类型是否在查询视图中有标注	如事务块、存储过程等特殊语句，在查询视图中需要标注	不涉及	完成
资源管理	功能	是否存在视图中状态显示与查询语句信息不符	状态更新需要和查询语句执行流一致	不涉及	完成
资源管理	DFX	异常场景下的负载管理是否满足预期，例如内存泄露场景下，内存占用过高，如何进行负载调度	特殊情况下，仍提高负载调度机制，通过排队机制，尽可能避免频繁内存报错	不涉及	完成
资源	性能	记录语句是否超过	提高定时清理和复用机制，避免空间过载	不涉	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
管理		可用空间大小		及	
资源管理	功能	实时查询视图和历史查询视图信息是否记录所有查询语句	记录语句是通过resource track设置的标记完成，若存在语句未记录情况，则说明有些语句没有走判断流程	不涉及	完成
资源管理	性能	内部临时表定时清理	历史信息存到临时表情况，需要定时清理，防止空间膨胀	不涉及	完成
资源管理	功能	用户层级管理和资源池匹配	用户和资源池关系是多对1，用户层级需要和资源池层级匹配	不涉及	完成
资源管理	功能	查询规则使用通过控制组关联使用	新增查询规则，通过控制组来指定和使用	不涉及	完成
资源管理	功能	管理员用户不受资源池管控	管理员用户因权限问题，不受资源管控，故只有普通用户执行作业才使用给定资源	不涉及	完成
资源	功能	逻辑集群下用户、逻辑集	用户关联逻辑集群、资源池关联逻辑集群均满足下，用户和资源池管理，三者条件一致	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
管理		群、资源池关系必须完全一致			
资源管理	功能	逻辑集群foreign资源池针对非本逻辑集群用户使用	只有外部用户使用的资源才进foreign资源池，其他均使用本地资源	不涉及	完成
GUC参数	兼容性	前向兼容性	1）新增GUC参数，需要考虑必要性，必须经过社区sig组评审，对于需要在安装时根据环境取值的，应知会升级、安装等OM操作进行相应适配 2）修改GUC参数默认值必须慎重，应尽可能避免，必须经过sig组评审，且需要知会升级、安装等OM操作进行相应适配 3）修改GUC参数取值范围、取值单位、生效行为的，应尽可能避免，必须经过社区sig组评审，必须出具升级兼容性测试报告，且需要知会升级、安装等OM操作进行相应适配 4）原则上不允许删除GUC参数（可以废弃不用）	不涉及	完成
异常捕获	功能	打断计数恢复	1）在PG_TRY之前必须记录当前的打断计数，尤其是在PG_TRY前后有HOLD_INTERRUPT和RESUME_INTERRUPT的场景 2）在异常捕获过程中，会把打断计数清零，因此，在PG_CATCH中，需要恢复打断计数	不涉及	完成
异常捕获	DFX	资源释放	如果PG_CATCH中不再向上层RE TRHOW异常，那么一定要确认所有在PG_TRY中可能申请的资源在PG_CATCH中得到合理释放，包括内存、锁、表引用计数、BUFFER引用计数、syscache引用计数	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
信号响应和处理	并发	中断响应	对于执行时间较长的函数，比如文件读写操作，需要定期响应信号CHECK_FOR_INTERRUPT，防止HANG了以后查询取消不了	涉及	完成
信号响应和处理	并发	失效消息	打开表和CHECK_FOR_INTERRUPT等操作会响应失效消息，可能会修改和释放正在使用的smgrRelation。因此，如果在smgropen之后，在使用smgrRelation之间，如果可能会响应失效消息，必须再次调用smgropen	不涉及	完成
行存	功能	GETSTRUCT(TUP)正确性	GETSTRUCT(TUP) 使用仅适用于系统表，并且仅适用于靠前的、连续的定长字段（与C结构体可保持一致）	不涉及	完成
行存	性能	行存表属性列排布	创建的行存表将定长类型的列尽可能连续的、放置在靠前位置，变长类型的列靠后放置，以提高性能	不涉及	完成
行存	性能	syscache使用	涉及扫描系统表的操作，尽量使用索引列和扫描键匹配的syscache，以加速扫描。	不涉及	完成
行存	功能	行存表属性列标号	用户列AttrNumber是从1开始计数的，访问数组TupleDesc.attrs[]时要减1。系统列（隐藏列）AttrNumber是负数。0用来表示无效的列。	不涉及	完成
行存	功能	引用计数	对引用计数增减要成对匹配调用。例如RelationIncrementReferenceCount()要与	不涉	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			RelationDecrementReferenceCount调用次数要匹配	及	
行存	功能	relation打开和关闭	relation_open要有对应的relation_close	不涉及	完成
行存	功能	cachelist	SearchSysCache*与ReleaseSysCache的调用要成对匹配. SearchSysCacheList*与ReleaseSysCacheList的调用要成对匹配 . SearchSysCacheCopy*返回一个元组的副本，需调用heap_freetuple进行释放.	不涉及	完成
行存	功能	Xlog	涉及新增日志模式的，需要check主备行为是否一致	不涉及	完成
行存	功能	ResourceOwnerCreate	调用ResourceOwnerCreate函数时，parent参数传入为NULL要谨慎, 可能造成资源泄露	不涉及	完成
锁	并发	SpinLock	1.使用SpinLock的时候持锁时间要尽量短（因为spinlock机制是拿不到继续spin，持锁时间过长可能导致CPU资源的空转浪费）。务必注意持锁和解锁成对出现(在哪个函数里面拿锁就在哪个函数里面放锁)。 2.SpinLock一定记得持锁和解锁成对出现，SpinLock不记录任何持锁线程的信息，即线程本身不知道自己是否已经持有锁，别的线程也不知道谁持有锁。所以不具备检测死锁的功能。为了避免造成死锁的情况，务必注意拿锁和放锁成对出现。	不涉及	完成
锁	并发	LWLock	LWLock维持线程等待队列，持锁线程本身知道自己持有锁，但等待线程不知道谁持有锁。所以不具备死锁检测功能。虽然在事务结束时会统一释放，但是我们在使用时依旧要遵守在不需要锁保护的时候及时放锁的原则。性能上可以考虑是否有更好无锁算法或者是降低冲突的算法。如果一定	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			要持锁操作，尽量使持锁时间较短，将不必要放在锁保护中执行的代码移到外面。		
锁	并发	Regular Lock	1.常规锁维护等待线程队列和持锁线程队列，在单一节点上具备死锁检测功能。但是编码使用常规锁的过程中不要过分依赖事务结束时锁的统一释放，应该在不再需要锁保护的时刻及时放锁。 2.常规锁获取方式中包括等待和不等待两种模式。等待表示如果拿不到锁，进入睡眠，等待唤醒继续尝试拿锁或等锁超时；不等待表示拿不到锁立即返回失败。使用时需要考虑特性适合哪种模式。 3.如果通过searchSysCache得到的tuple信息在真正使用前有一定的空窗期，需要对系统表拿适当的锁，防止出现因有线程并发修改系统表内容导致的不可预知的错误。（当前代码中可能还存在一些这样的情况存在隐患）	不涉及	完成
事务回滚	DFX	错误清理	新增错误清理的逻辑不要放到aborttransaction函数中添加。禁止在回滚阶段进行复杂操作，禁止申请内存，禁止申请锁资源。	不涉及	完成
中断信号屏蔽	并发	事务commit/prepare	事务prepare,commit阶段中有很多地方是hold interrupt，目的是实现原子的操作，不希望被任何信息打断/出错。在hold interrupt中加入等待，网络等操作应仔细斟酌。	不涉及	完成
HA设计	DFX	原则	1.假定失效的设计：假定任何环节都可能出问题，然后倒推依次设计，确保应用可以连续的工作。 2.多可用区设计：提供多个可用区，某个可用区故障之后可以快速切到另一个可用区。 3.自动扩展设计：增删节点（备机）自动化，支持在线不影响业务，可以动态调整高可用能力。 4.自我修复设	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			计：故障后自动检测自动修复，人工不参与，尽量保证对各种异常场景处理逻辑的覆盖。 5.松耦合设计：耦合度越小，扩展性越好，容错能力越大，依赖的组件越多，问题场景越复杂越难以处理。		
HA用例	功能	ha check	1.HA相关设计实现需要严格系统的测试。 2.需要保证最简单的功能验证，包括搭建主备环境，确认主备可以正常同步，以及跑ha check用例，确认所有用例可以跑过。 3.需要完成各种故障场景的验证，设计详细的故障库，确保覆盖各个场景。 4.需要完成压力场景下的性能验证。	不涉及	完成
HA功能	功能	日志同步	1.日志是按照顺序发送的，通过日志号LSN来保证，控制文件中的日志号需要重点维护，同时，控制文件中的checkpoint点（决定恢复的初始位置）和minrecovery点（决定对外提供服务的位置）等同样需要重点维护，错误修改可能会导致数据丢失等异常情况。 2.主备日志的同步位置通过流复制槽来维护，决定了主备同步的位置以及日志回收的位置，错误更新可能会导致主备同步错乱或者日志回收异常。另外，流复制槽对于增量build的执行至关重要，相关的修改需要考虑对其可能造成的影响。 3.必须有视图查看主备同步的进度，进一步增强主备同步异常的提示，提高DFX能力，尽量避免人工从日志中筛选可用的定位信息。 4.必须有工具校验日志的一致性，增强信息显示，方便问题定位。	不涉及	完成
HA功能	功能	数据重建	1.数据重建需要校验主备机的信息，比对主备机的文件，找到合适的位置，拷贝修改的文件和日志。需要考虑该过程中每个可能的故障点，设计实现中需要确保在任意点故障后重建过程可重入。 2.哪些目录是要删除的，哪些目录不需要删除，而是需要从对端拷贝。一般状态文件需要保留，数据文件需要拷贝，当新增特殊类型文件时，需要考虑build时是否需要从对端拷贝。 2.增量build依赖流复制槽，对流复制槽的修改需要重点关注可能对增量build的影响，更新过早	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			或者不及时都可能导致增量build寻找日志分叉点不准确，增量build无法正常运行。 3.重建的时长与拷贝的数据量大小有关，在大量小文件的场景下，耗时会比较长，避免引入大量无效的问题，相关的修改需要考虑效率的优化。 4.dn build dn最为常见。刚初始化的主备实例要进行全量build，主机故障备机升主导致日志分叉要进行增量build。build太频繁会严重影响性能，相关的修改需要尽可能减少build次数，导致频繁build的修改是无法接受的。 5.增强重建过程进度显示等一些辅助信息，在关键位置打印日志，提高DFX能力。		
HA功能	功能	主备倒换	1.主要包括switchover和failover。switchover先要备机通知主机降备，然后降备主机通知备机升主，过程中主备需要进行交互，逻辑比较复杂，相关的修改不能打破整体的逻辑顺序，避免出现异常情况，比如双主。另外，switchover中降备的过程需要完成线程退出、内存清理等大量工作，如果线程退出不掉或者内存释放不掉，会造成降备以及switchover失败，相关的修改需要避免产生类似的问题。Failover直接备机，不存在与其他备机同步的逻辑，但是需要通过一定的手段保证日志的完整性和一致性（当前是根据LSN大小，需要进一步优化）。	不涉及	完成
HA性能	性能	性能设计	1.HA性能涉及日志复制、主备倒换、数据重建等多个方面，需要统筹考虑，使用多种并行方式（比如pipeline、batch、multi threads等）提高HA性能，同时要考虑优化HA整体架构，使用业界领先的解决方案大幅改进HA性能。 2.一主多备，同步备越多，主机事务提交时需要等待备机日志落盘的代价越大，同步性能会有一定的影响。 3.数据重建，拷贝数据量大小相关，特别是大量小文件的场景，应避免引入无用的文件或者大量生成很多文件（引入crc校验出现过该问题）。频繁数据重建，性能影响很大，应尽可能避	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			免，同时，应对build过程本身性能进一步优化。 4.redo比较耗时，特别是日志量比较大的情况下，会导致启动或者故障切换的速度比较慢，增加新类型的日志或者增加原有日志写的频率都会加剧这个问题，相关的修改需要评估对redo速度的影响。 5.通过HA check的运行时间可以大概估计是否相关的修改对HA同步性能造成了影响，确定会对HA性能造成影响的修改需要进行专门的性能测试发现引入的性能瓶颈点。		
HA与其他模块的交互	DFX	其他关联模块包括集群管理、通信、元信息等	1.集群管理负责实例的状态检测和自动恢复，一种常见的场景是发现主机故障、仲裁升主备机、下发升主命令，还有一种是下发build命令，进行备机重建。集群管理相关的修改，如监听超时时间、仲裁选主机制等的修改，会影响整个HA的预期行为，需要充分验证在不同故障场景下HA行为是否符合预期。 2.通信负责主备日志和数据的传输以及主备之间的心跳检测，同时，还包括CMA和DN之间等模块间的通信，稳定的通信对于相关模块的正常运行至关重要，通信机制或者参数（超时）的变更会影响HA相关模块的预期行为，需要充分验证在不同网络压力场景下HA行为是否符合预期。 3.一主多备修改pgxc_node存储格式以及与pooler、stream模块的修改适配导致了较多的问题，由于本身相关模块复杂比较高，相关的修改需要给予充分的测试时间。	不涉及	完成
HA部署	DFX	推荐形态和原则	1.HA的部署方案需要保证故障后有冗余节点替换，同时还需要保证切换的速度足够快，即同时满足RPO和RTO的要求。如果副本越多，可用性和可靠性越好，但是资源利用率越低。因此，需要根据实际情况选择最优方案，既要部署一定数量的冗余，又要通过支持只读业务等方式提高整体资源利用率。 2.不同机房部署不同数量的主机，既可以做到负载均衡，又可以在机房故障时快速切换业务到另一机房。 3.主备机交叉部署，均衡负载，同时，避免单点故障时，所有主	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
			机都要切换。 4.ETCD不与DN部署在一起，避免DN压力过大，造成ETCD不可用。 5.其他组件也应该在条件允许的情况下分开部署。		
fastcheck用例	功能	fastcheck	1.可以添加自测用例的，提交PR必须带有自测用例，并确保测试用例、用例结果的正确性。2.在src/test/regress下新增的用例集，需要添加到parallel_schedule0A/0B/0C的其中一个，并且在没有互相影响的情况下，添加到已有的测试组中（一个测试组可允许5~10个用例并行执行），提升fastcheck效率。非必要不新增测试组。	不涉及	完成
编译方式	功能	支持CMAKE编译	新增源代码文件场景下，除了支持基本的make外，还需要适配支持cmake编译。	不涉及	完成
打包适配	功能	注意企业版和轻量版均需支持	新增插件场景下，需要在build/script下的打包列表中添加插件相关文件，包括插件so、sql脚本和control文件等。除特殊情况外，轻量版也应当支持，对应脚本是aarch64_lite_list和x86_64_lite_list。	不涉及	完成
安全	访问通道控制	是否新增侦听端口	新增侦听端口需刷新通信矩阵	不涉及	完成
安全	访问通道	是否新增进程或组件间通信	新增进程或组件间通信刷新通信矩阵	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
	控制				
安全	访问通道控制	是否新增认证方式	新增认证方式需刷新通信矩阵及产品文档	不涉及	完成
安全	权限控制	是否涉及创建文件或目录	创建文件或目录须显式指定文件或目录的访问权限	不涉及	完成
安全	未公开接口	是否新增GUC参数	新增GUC参数需刷新产品文档	不涉及	完成
安全	未公开接口	是否新增或修改函数、视图、系统表	1.新增或修改函数、视图、系统表需刷新产品文档，考虑权限控制 2.新增函数、系统表的oid避免选择[7000,8000]区间内的。	不涉及	完成
安全	未公开接口	是否新增SQL语法	新增SQL语法需刷新产品文档，支持记录审计日志	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
安全	未公开接口	是否新增内部工具	新增内部工具需刷新产品文档	不涉及	完成
安全	未公开接口	是否存在口令硬编码	禁止口令硬编码	不涉及	完成
安全	未公开接口	脚本中是否存在注释代码	Shell/Python等解释性语言禁止注释代码，注释代码需要删除	不涉及	完成
安全	未公开接口	是否存在隐藏命令、参数、端口等接入方式	对于现网维护期间不会使用的命令/参数、端口等接入方式（包括但不限于产品的生产、调测、维护用途），必须删除（如通过编译宏）	不涉及	完成
安全	未公开接口	禁止在产品对外部用户发布的软件（包含软件包/补丁包）中提	禁止在产品对外部用户发布的软件（包含软件包/补丁包）中提供可修改任意用户口令、具有“口令破解能力”（指口令暴力破解、利用系统/算法漏洞恶意破解口令）、对包含敏感数据的文件（如包含密钥的配置文件、数据库）进行解密的功能或工具。	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
		供破解类、网络嗅探类工具。			
安全	未公开接口	禁止在产品对外部用户发布的软件（包含软件包/补丁包）中提供破解类、网络嗅探类工具。	禁止在系统中保留第三方的网络嗅探工具tcpdump、gdb、strace、readelf网络、进程调试工具，cpp、gcc、dexdump、mirror、JDK开发/编译工具和仅在调测阶段使用的自研调试工具/脚本（例如：仅在调试阶段使用的加解密脚本、调测功能、可以提权的命令），由于业务需要必须保留的，需要进行严格的访问控制。同时在资料中说明保留的原因、使用的场景、风险。	不涉及	完成
安全	敏感数据保护	认证凭据不允许明文存储在系统中，应该加密保护。	认证凭据（如口令/私钥等）不允许明文存储在系统中，应该加密保护。	不涉及	完成
安全	敏感数据保护	用于敏感数据传输加密的密钥，不能硬编码在代码中。	用于敏感数据传输加密的密钥，不能硬编码在代码中，系统应提供密钥管理（含密钥修改）或密钥协商更新的机制。	不涉及	完成

模块	类别	检查项	检查项详细说明	是否涉及	是否完成检查
安全	敏感数据保护	密码算法中使用到的随机数必须是密码学意义上的安全随机数。	密码算法中使用到的随机数必须是密码学意义上的安全随机数。	不涉及	完成
安全	敏感数据保护	禁止在系统中存储的日志、调试信息、错误提示中明文打印认证凭据	禁止在系统中存储的日志、调试信息、错误提示中明文打印认证凭据（口令/私钥/预共享密钥）。	不涉及	完成

特性LLT测试方案及验证结论

基本测试

基本功能测试

回归测试验证

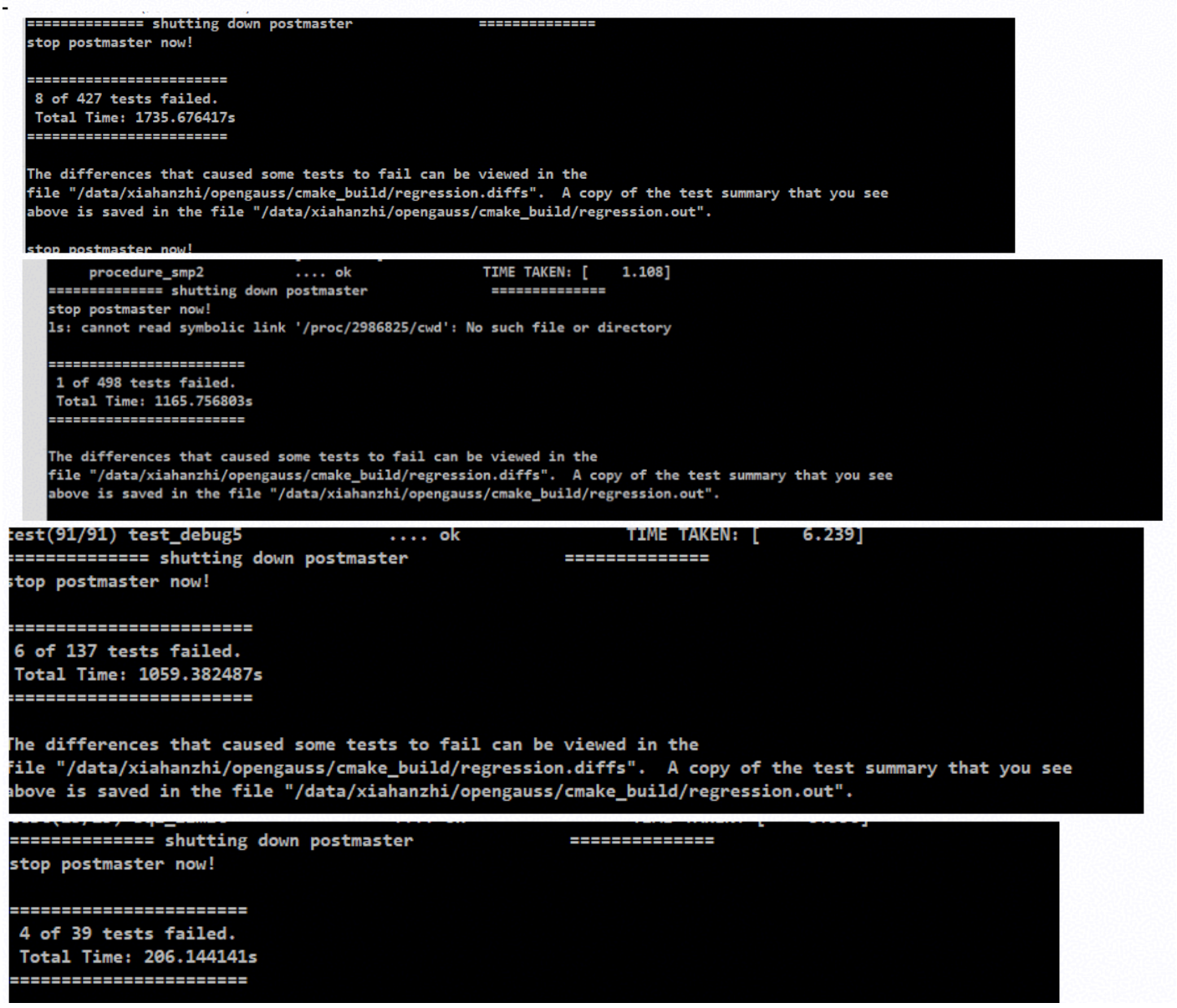
测试集	结论
fastcheck	通过
memcheck	通过

测试集	结论
hacheck	通过
subscription/check	通过

Fastcheck

参考社区博客获取fastcheck/memcheck/hacheck/subcheck的执行方法：

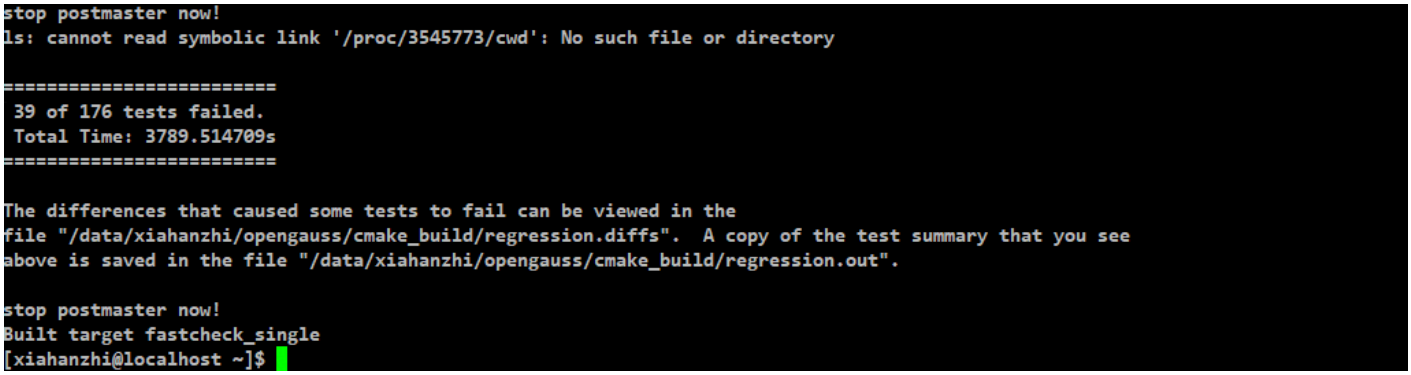
https://opengauss.org/zh/blogs/xiteming/HowtorunFastcheck.html



Memcheck

```
3356824==HINT: if you don't care about these errors you may set ASAN_OPTIONS=detect_odr_violation=0
MMARY: AddressSanitizer: odr-violation: global 'XLogSegmentSize' at /data/xiahanzhi/opengauss/openGauss-server/src/common/port/fio_dss.cpp:81:8
============================================================
3356824==ERROR: AddressSanitizer: odr-violation (0xaaaae4ba9b80):
[1] size=1 'g_enable_dss' /data/xiahanzhi/opengauss/openGauss-server/src/common/port/fio_dss.cpp:78:6
[2] size=1 'g_enable_dss' /data/xiahanzhi/opengauss/openGauss-server/src/common/interfaces/libpq/fio_dss.cpp:78:6
ese globals were registered at these points:
[1]:
  #0 0xaaaae3d75558 in __asan_register_globals ../../../../libsanitizer/asan/asan_globals.cpp:341
  #1 0xaaaae3ef7b1c in _sub_I_00099_1 (/data/xiahanzhi/opengauss/openGauss-server/src/test/regress/tmp_check/install/data/xiahanzhi/opengauss/install/bin/gsql+0x217b1c)
  #2 0xffff896fb1d0 in __libc_start_main (/usr/lib64/libc.so.6+0x2b1d0)
  #3 0xaaaae3d645ac in _start (/data/xiahanzhi/opengauss/openGauss-server/src/test/regress/tmp_check/install/data/xiahanzhi/opengauss/install/bin/gsql+0x845ac)

[2]:
  #0 0xaaaae3d75558 in __asan_register_globals ../../../../libsanitizer/asan/asan_globals.cpp:341
  #1 0xffff89fc834c in _sub_I_00099_1 (/data/xiahanzhi/opengauss/openGauss-server/src/test/regress/tmp_check/install//data/xiahanzhi/opengauss/install/lib/libpq_ce.so.5.5+0xd8
  #2 0xffff8ad0ab68  (/lib/ld-linux-aarch64.so.1+0x3b68)
  #3 0xffff8ad0ac50  (/lib/ld-linux-aarch64.so.1+0x3c50)
  #4 0xffff8ad20774  (/lib/ld-linux-aarch64.so.1+0x19774)

3356824==HINT: if you don't care about these errors you may set ASAN_OPTIONS=detect_odr_violation=0
MMARY: AddressSanitizer: odr-violation: global 'g_enable_dss' at /data/xiahanzhi/opengauss/openGauss-server/src/common/port/fio_dss.cpp:78:6
============================================================
3356824==ERROR: AddressSanitizer: odr-violation (0xaaaae4ba9a00):
[1] size=344 'g_dss_device_op' /data/xiahanzhi/opengauss/openGauss-server/src/common/port/fio_dss.cpp:77:17
[2] size=344 'g_dss_device_op' /data/xiahanzhi/opengauss/openGauss-server/src/common/interfaces/libpq/fio_dss.cpp:77:17
ese globals were registered at these points:
[1]:
  #0 0xaaaae3d75558 in __asan_register_globals ../../../../libsanitizer/asan/asan_globals.cpp:341
  #1 0xaaaae3ef7b1c in _sub_I_00099_1 (/data/xiahanzhi/opengauss/openGauss-server/src/test/regress/tmp_check/install/data/xiahanzhi/opengauss/install/bin/gsql+0x217b1c)
  #2 0xffff896fb1d0 in __libc_start_main (/usr/lib64/libc.so.6+0x2b1d0)
  #3 0xaaaae3d645ac in _start (/data/xiahanzhi/opengauss/openGauss-server/src/test/regress/tmp_check/install/data/xiahanzhi/opengauss/install/bin/gsql+0x845ac)

[2]:
  #0 0xaaaae3d75558 in __asan_register_globals ../../../../libsanitizer/asan/asan_globals.cpp:341
  #1 0xffff89fc834c in _sub_I_00099_1 (/data/xiahanzhi/opengauss/openGauss-server/src/test/regress/tmp_check/install//data/xiahanzhi/opengauss/install/lib/libpq_ce.so.5.5+0xd8
  #2 0xffff8ad0ab68  (/lib/ld-linux-aarch64.so.1+0x3b68)
  #3 0xffff8ad0ac50  (/lib/ld-linux-aarch64.so.1+0x3c50)
  #4 0xffff8ad20774  (/lib/ld-linux-aarch64.so.1+0x19774)

3356824==HINT: if you don't care about these errors you may set ASAN_OPTIONS=detect_odr_violation=0
MMARY: AddressSanitizer: odr-violation: global 'g_dss_device_op' at /data/xiahanzhi/opengauss/openGauss-server/src/common/port/fio_dss.cpp:77:17
test(208/213) pgstat_opt                .... ok                TIME TAKEN: [   23.682]
============== shutting down postmaster               ==============
top postmaster now!

=========================
 10 of 427 tests failed.
 Total Time: 5154.668688s
=========================

The differences that caused some tests to fail can be viewed in the
file "/data/xiahanzhi/opengauss/cmake_build/regression.diffs".  A copy of the test summary that you see
above is saved in the file "/data/xiahanzhi/opengauss/cmake_build/regression.out".

top postmaster now!
Built target fastcheck_single
[xiahanzhi@localhost ~]$
[xiahanzhi@localhost ~]$
```

```
ls: cannot read symbolic link '/proc/3366889/cwd': No such file or directory

=========================
 5 of 498 tests failed.
 Total Time: 4718.756156s
=========================

The differences that caused some tests to fail can be viewed in the
file "/data/xiahanzhi/opengauss/cmake_build/regression.diffs".  A copy of the test summary that you see
above is saved in the file "/data/xiahanzhi/opengauss/cmake_build/regression.out".

stop postmaster now!
Built target fastcheck_single
```



非本次修改导致,无core

hacheck

不涉及

subscription/check

不涉及

升级验证

验证集	结论
openGauss LTS版本升级到最新master	不涉及
社区最新master回滚到旧版本	不涉及
openGauss LTS版本再升级到最新master	不涉及
openGauss LTS版本升级提交到最新master	不涉及

openGauss LTS版本升级到最新master

openGauss最新master回滚到旧版本

openGauss LTS版本再升级到最新master

openGauss LTS版本升级提交到最新master

代码检视结论

编码规范检查

编程规范

请对照社区安全编码规范进行排查：

https://gitee.com/opengauss/security/blob/master/guide/SecureCoding.md

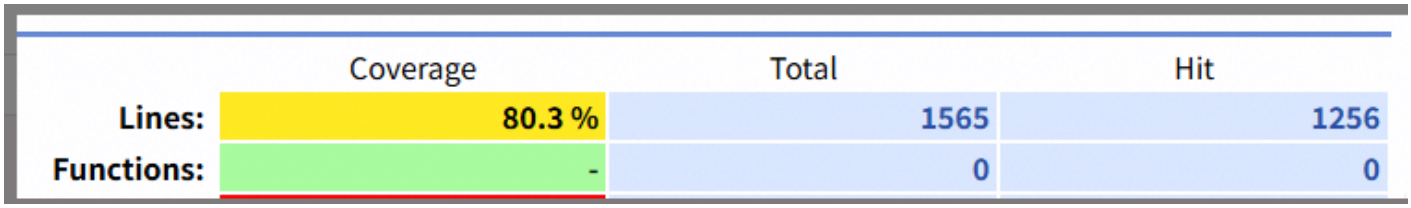
https://gitee.com/opengauss/security/blob/master/guide/SecureCompile(C&C++).md

内存使用排查

相关内存均在指定上下文内申请和释放
排查结果正确

覆盖率

增量代码测试用例覆盖率80%+



	Coverage	Total	Hit
Lines:	**80.3 %**	**1565**	**1256**
Functions:	**-**	**0**	**0**

参考社区博客获取增量代码覆盖率的执行方法：

https://opengauss.org/zh/blogs/totaj/%E5%A6%82%E4%BD%95%E8%B7%91%E5%A2%9E%E9%87%8F%E4%BB%A3%E7%A0%81%E8%A6%86%E7%9B%96%E7%8E%87.html

鲲鹏平台乱序排查

编号	排查依据	排查结果
1	**是否为多线程并发场景?**	是
2	**是否涉及线程间共享数据?**	是
3	**是否未对共享数据加锁保护?**	已加锁
4	线程间共享数据是否存在无锁同步模式?	无
5	是否 **未** 在正确的位置插入合适的内存屏障	是

代码检视意见

提出人	意见	答复详情

遗留问题

测试建议