PR #33172 完整报告

sgl-project/sglang

runtime_context: per-role namespace enforcement behind SGLANG_ROLE_NAMESPACES

合并时间：2026-08-01 23:59

原文链接：http://prhub.com.cn/sgl-project/sglang/pull/33172

执行摘要

- 一句话：配置读取按角色强约束，三模式审计与 fail-closed 落地
- 推荐动作：值得精读。这是 config-namespace 重构系列的关键拼图，展示了三个高价值设计决策：①用 " 单条可剪枝比较 + 外联检查 " 在默认路径上兼容 dynamo fullgraph 捕获；②record/enforce 两阶段 rollout——先审计后约束，且审计数据即时持久化以对抗信号杀进程；③fail-closed 的防御纵深（publish 时校验 + 读取时再防御）。建议结合 #33011 / #33012 / #33013 及 base PR #33171 一起阅读，理解完整演进脉络。

功能与动机

PR body 明确说明：publish(role=...) 自命名空间拆分以来只记录角色而未实施约束，本 PR 接线该角色被保留所对应的强制逻辑，"a config read outside a process type's declared surface fails loud instead of silently coupling process types"——即让跨进程类型的配置读取从静默耦合变为失败关闭，为后续按角色收窄命名空间投影铺路。

实现拆解

实现按 5 步拆解：

1. 环境开关注册与导入期校验：在 python/sglang/srt/environ.py 的 Envs 类新增 SGLANG_ROLE_NAMESPACES（默认 off）与 SGLANG_ROLE_NAMESPACES_OUT 两个 EnvStr 字段，遵守项目 " 新 SGLANG_* 开关必须注册到 Envs 并经由 envs.X.get() 读取 " 的约定。runtime_context.py 在模块导入时通过 _role_ns_mode_from_env() 一次性解析并调用 _validated_role_ns_mode() 校验（strip + lower，非法值直接抛 ValueError，空字符串同样失败而非回退到 off），得到模块级常量 _ROLE_NS_MODE，保证 config_bag 里的模式门在 dynamo 下可被常量折叠。
2. 读取路径接线：RuntimeContext.config_bag() 在返回 bag 前追加 if _ROLE_NS_MODE != "off" 检查并调 _check_role_namespace(name)；检查逻辑刻意外联成独立方法，使 off 模式下保持单条死分支可剪枝判断（config_bag 会运行在编译后的模型前向中）。_check_role_namespace 按模式分流：record 模式在非编译态下调 _record_namespace_read(role, name) 收集审计；enforce 模式查 ROLE_NAMESPACE_SETS，角色缺失或命名空间越界即抛带修复指引的 ValueError。
3. 角色命名空间表：新增 ROLE_NAMESPACE_SETS 声明各 publish 角色可读的顶级命名空间集合。scheduler / launcher / test 为 None（全树）；dp_controller 依据 record 模式

冒烟审计（普通 + DP-attention）收窄为 frozenset({"exec"})，与模块静态读取集一致（弹性 EP 门）；tokenizer（当前零 bag 读取，按设计读 self.server_args）与 encoder / expert_backup / weight_cache_daemon（部署形态未审计）保守保留 None。parallel 经 get_parallel() 独立服务且各进程均合法读取拓扑配置，不纳入本表。

4. record 审计管线：_record_namespace_read 以 (role, name) 集合去重；_append_role_ns_out 对每对新 pair 立即追加写 SGLANG_ROLE_NAMESPACES_OUT（worker 常被信号终止、跳过 atexit，因此不能依赖退出时统一落盘），写失败打印 stderr 警告且条目保留在内存集合中；_ensure_record_dump_registered 在 publish 时即注册 atexit handler _dump_recorded_namespace_reads，按角色汇总并在 stderr 输出，且始终包含进程自身 publish 角色（零读取角色输出空行）；publish 时额外写 {role} - 标记行，使被信号杀死的零读取 worker 也可被识别。
5. enforce fail-closed 与测试配套：publish 时若角色不在 ROLE_NAMESPACE_SETS 中立即抛错（防止拼错角色静默继承全树），读取路径再做防御。测试在 test/registered/unit/test_runtime_context_config_bags.py 新增 TestRoleNamespaceEnforcement，覆盖三模式行为、表外角色拒绝、模式值校验、record 注册时机，以及两个 torch.compile(fullgraph=True, backend="eager") 探针，分别钉死 off 模式可剪枝与 record 模式在追踪下无副作用；另以 enforce 模式跑 DP-attention 冒烟验证零违规启动与服务。

关键文件：

- python/sglang/srt/runtime_context.py（模块 运行时上下文；类别 source；类型 dependency-wiring；符号 _check_role_namespace, _validated_role_ns_mode, _role_ns_mode_from_env, _is_compiling）：核心实现文件：在 config_bag 读取路径接入模式门与 _check_role_namespace，新增 ROLE_NAMESPACE_SETS 角色表、record 审计管线与 enforce fail-closed 逻辑，并更新 publish 语义。
- test/registered/unit/test_runtime_context_config_bags.py（模块 配置测试；类别 test；类型 test-coverage；符号 TestRoleNamespaceEnforcement, test_off_mode_ignores_declared_sets, test_enforce_blocks_reads_outside_the_declared_set, test_enforce_full_tree_role_and_roleless_install_are_unrestricted）：新增 TestRoleNamespaceEnforcement 共 9 个用例，覆盖三模式行为、表外角色拒绝、模式值校验、record 注册时机，以及两个 torch.compile(fullgraph=True) 探针分别钉死 off 可剪枝与 record 无副作用。
- python/sglang/srt/environ.py（模块 环境变量；类别 source；类型 core-logic；符号 SGLANG_ROLE_NAMESPACES, SGLANG_ROLE_NAMESPACES_OUT）：按项目约定将两个新 SGLANG_* 开关注册到 Envs 类：SGLANG_ROLE_NAMESPACES（默认 off）与 SGLANG_ROLE_NAMESPACES_OUT，使开关可被发现、可 override，并支持导入期冻结供 dynamo 剪枝。

关键符号：config_bag, _check_role_namespace, _validated_role_ns_mode, _role_ns_mode_from_env, _is_compiling, _ensure_record_dump_registered, _append_role_ns_out, _record_namespace_read, _dump_recorded_namespace_reads, publish

关键源码片段

python/sglang/srt/runtime_context.py

核心实现文件：在 config_bag 读取路径接入模式门与 _check_role_namespace，新增 ROLE_NAMESPACE_SETS 角色表、record 审计管线与 enforce fail-closed 逻辑，并更新 publish 语义。

```
# runtime_context.py —— 按角色命名空间强制检查的核心片段

def config_bag(self, name: str) -> _ConfigBag:
    # 配置命名空间统一读取入口；模式门刻意保持为单条比较，
    # 因为 config_bag 会运行在编译后的模型前向里，off 模式下
    # 这条分支必须能被 dynamo 死分支剪枝（由 fullgraph 测试钉死）
    bags = self._config_bags
    if not bags or name not in bags:
        raise ValueError(f"config namespace {name!r} not published")
    if _ROLE_NS_MODE != "off":
        self._check_role_namespace(name)
    return bags[name]

def _check_role_namespace(self, name: str) -> None:
    # 检查逻辑外联成独立方法，让 off 模式的门保持单条可剪枝判断
    role = self._publish_role
    if _ROLE_NS_MODE == "record":
        # 记录有 Python 副作用（set 变更、文件 I/O、atexit 注册），
        # 在 dynamo 追踪下必须剪掉，否则 fullgraph 捕获不合法；
        # 代价是编译区域内的读取不会被审计（已在表注释中声明）
        if not _is_compiling():
            _record_namespace_read(role, name)
    elif _ROLE_NS_MODE == "enforce" and role is not None:
        # fail closed：表外角色在 publish 时已被拒绝，这里再做防御
        if role not in ROLE_NAMESPACE_SETS:
            raise ValueError(
                f"publish role {role!r} has no ROLE_NAMESPACE_SETS entry; "
                "declare its namespace set (None for the full tree)."
            )
        allowed = ROLE_NAMESPACE_SETS[role]
        if allowed is not None and name not in allowed:
            # 错误信息带修复指引：要么扩展表，要么把读取挪到别的进程
            raise ValueError(
                f"config namespace {name!r} is outside the declared set "
                f"for publish role {role!r} ({sorted(allowed)})."
            )

# record 审计管线 —— 面向信号杀进程场景的即时持久化设计

def _append_role_ns_out(role: str | None, name: str) -> None:
    # 立即追加写：worker 常被信号终止而跳过 atexit，审计必须幸存
    out = envs.SGLANG_ROLE_NAMESPACES_OUT.get()
```

```
    if not out:
        return
    try:
        with open(out, "a") as f:
            f.write(f"{role} {name}\n")
    except OSError as e:
        # 条目仍留在内存 set 中，退出摘要依然覆盖它；
        # 写失败必须可见，否则会 seed 出过窄的 enforcement 集
        print(
            f"[role-namespaces] pid={os.getpid()} failed to append "
            f"({role}, {name}) to {out!r}: {e}",
            file=sys.stderr,
            flush=True,
        )

def _record_namespace_read(role: str | None, name: str) -> None:
    if (role, name) in _RECORDED_NS_READS:
        return
    _RECORDED_NS_READS.add((role, name))
    _append_role_ns_out(role, name)
    _ensure_record_dump_registered()

def _dump_recorded_namespace_reads() -> None:
    # 每个进程 dump 一次；进程自身 publish 角色总是包含在内，
    # 零读取角色输出空行而不是与 " 从未录制 " 混淆
    by_role: dict = {}
    own_role = _CONTEXT._publish_role
    if own_role is not None:
        by_role.setdefault(own_role, set())
    for role, name in _RECORDED_NS_READS:
        if name == "-": # publish 时写的标记行，不是命名空间读取
            by_role.setdefault(role, set())
            continue
        by_role.setdefault(role, set()).add(name)
    for role in sorted(by_role, key=str):
        print(
            f"[role-namespaces] pid={os.getpid()} role={role} "
            f"read={','.join(sorted(by_role[role]))}",
            file=sys.stderr,
            flush=True,
        )
```

test/registered/unit/test_runtime_context_config_bags.py

新增 TestRoleNamespaceEnforcement 共 9 个用例，覆盖三模式行为、表外角色拒绝、模式值校验、record 注册时机，以及两个 torch.compile(fullgraph=True) 探针分别钉死 off 可剪枝与 record 无副作用。

```
# test_runtime_context_config_bags.py —— enforce 与 record 模式的关键测试
```

```
class TestRoleNamespaceEnforcement(CustomTestCase):
    """SGLANG_ROLE_NAMESPACES: off（默认）自由；record 收集逐角色读取审计；
    enforce 对角色声明集合之外的 bag 读取失败关闭。"""

    def setUp(self):
        rc.reset_context()

    def tearDown(self):
        rc.reset_context()

    def test_enforce_blocks_reads_outside_the_declared_set(self):
        # 用 mock 切换模式与表，避免真实环境变量影响测试隔离
        self._publish("test")
        with mock.patch.object(rc, "_ROLE_NS_MODE", "enforce"), mock.patch.dict(
            rc.ROLE_NAMESPACE_SETS, {"test": frozenset({"serving", "schedule"})}
        ):
            rc.get_serving()
            rc.get_schedule()
            with self.assertRaisesRegex(ValueError, "outside the declared set"):
                rc.get_exec()

    def test_off_mode_bag_read_traces_under_torch_compile(self):
        # config_bag 运行在编译后的模型前向里；off 模式下模式门必须
        # 保持死分支可剪枝，否则 fullgraph 捕获会失败
        import torch

        self._publish("test")

        @torch.compile(fullgraph=True, backend="eager", dynamic=False)
        def probe(x):
            if rc.get_schedule().max_running_requests is None:
                return x + 1
            return x * 2

        self.assertEqual(probe(torch.zeros(())).item(), 1.0)

    def test_record_mode_bag_read_traces_under_torch_compile(self):
        # record 模式有 set 变更 / 文件 I/O 副作用，_is_compiling() 探针
        # 必须把它们从追踪中剪掉，保持 fullgraph 捕获合法
        import torch

        self._publish("test")
        with mock.patch.object(rc, "_ROLE_NS_MODE", "record"), mock.patch.object(
            rc, "_RECORDED_NS_READS", set()
        ):

            @torch.compile(fullgraph=True, backend="eager", dynamic=False)
            def probe(x):
                if rc.get_schedule().max_running_requests is None:
```

```
            return x + 1
        return x * 2

    self.assertEqual(probe(torch.zeros(())).item(), 1.0)
```

评论区精华

review 由 Codex 自动审查与作者 ch-wan 逐条回应构成，共 25 条评论，核心交锋如下：

1. record 模式与 torch.compile 冲突（Codex P2）：记录逻辑含 set 变更、文件 I/O、atexit 注册等 Python 副作用，与 Dynamo full-graph 捕获不兼容。作者用 torch.compiler.is_compiling() 惰性探针在追踪下剪枝录制，并补 fullgraph 回归测试。
2. 编译区域内读取不可记录（Codex P2 后续）：is_compiling() 的剪枝导致编译前向内访问的命名空间永不入审计，可能 seed 过窄的 enforcement 集。作者承认这是 "in-band recording" 的固有限制——"side-effect-free tracing and observing reads are mutually exclusive"，改为在模式表注释中显式声明并要求审计时关闭编译，而非静默。
3. 表外角色静默继承全树（Codex P2）：dict.get() 返回 None 与显式声明全树无法区分，拼错角色会静默放行。作者在 publish 时拒绝表外角色并在读取时再防御，补测试。
4. 模式值校验（Codex P2，两轮）：先是 Enforce/ 尾随空格未规范化，后是 SGLANG_ROLE_NAMESPACES= 空字符串被 or "off" fallback 静默吞掉。两轮均修复：strip + lower + 空串也抛错，"absence still defaults to off"。
5. 零读取角色的审计可见性（Codex P2，两轮）：先补 publish 时注册 atexit，保证零读取角色输出空行；再补 signal 杀进程场景——record 模式 publish 即写 {role} - 标记行到 OUT 文件。
6. OUT 写失败静默（Codex P2 + 作者 suggestion 双线程）：失败后 pair 已入内存 set 不再重试。作者改为写失败打印含路径与 errno 的 stderr 警告，并明确接受 " 先入 set 再写 " 的权衡——否则会连 atexit 摘要一起丢失。
7. 项目约定：作者自提两条 suggestion——新 env 开关须注册到 Envs 类经 envs.X.get() 读取（已落实），以及 tokenizer 表注释夸大存在 get_disagg() 调用（已软化为 "may need"）。

- record 模式与 torch.compile fullgraph 捕获冲突 (correctness): 作者用 torch.compiler.is_compiling() 惰性探针在追踪下剪枝录制，并补 fullgraph 回归测试；后一轮 Codex 指出编译区域内读取将不被审计，作者承认是固有限制并文档化。
- enforce 模式下表外角色静默继承全树 (correctness): publish 时拒绝表外角色（抛 ValueError），读取路径再防御；补 test_enforce_rejects_roles_missing_from_the_table。
- 模式环境变量值校验（含空字符串 fallback） (correctness): _validated_role_ns_mode 做 strip + lower，非 off/record/enforce 一律抛错；移除空串 fallback，缺省由 EnvStr 默认值 off 提供。
- 零读取角色的审计可见性（atexit 注册时机与信号杀进程） (correctness): publish 时即注册 atexit 且在 dump 中恒含自身角色；再补 publish 写 {role} - 标记行到 OUT 文件，信号杀死的零读取 worker 也可区分。
- OUT 文件写入失败静默丢弃审计 (correctness): 失败时打印带路径与错误信息的 stderr 警告；作者明确接受 " 先入 set 再写 " 的权衡——否则 atexit 摘要也会丢失。

- 新 SGLANG_* 开关须注册到 Envs 类 (design): 注册到 Envs（SGLANG_ROLE_NAMESPACES = EnvStr("off")、SGLANG_ROLE_NAMESPACES_OUT = EnvStr(None)），模式仍导入期冻结、仍用 _validated_role_ns_mode 校验而非 EnvField 的 warn-and-default。
- tokenizer 表注释夸大存在 get_disagg 调用 (documentation): 软化为 may need 措辞并说明基于零读取审计做限制是猜测，tokenizer: None 的保守选择不变。

风险与影响

- 风险：
  1. record 审计盲区（正确性）：_is_compiling() 剪枝使 torch.compile 编译区域内的 bag 读取不被记录，若直接基于此类审计收窄 enforce 集合，可能将合法读取拒之门外。缓解：模式表注释已显式声明 "audits must run with compilation disabled"，且 enforce 默认关闭。
  2. OUT 写入失败无重试（可观测性）：_append_role_ns_out 失败仅打 stderr 警告，pair 已入内存 set 不再重试；若随后进程被信号杀死，该 pair 丢失。这是作者明确接受的设计权衡。
  3. enforce 误配即启动失败（运维）：enforce 模式下拼错模式值、角色缺失、任意越界命名空间读取都会在 publish 或首次读取时抛错，进程无法启动——这是 fail-closed 的有意代价，切换前必须用 record 模式完成全量审计。
  4. dp_controller 已前置收窄：dp_controller 被限定只读 exec 命名空间，若未来 DP 控制器（如 dataparallel_controller 的弹性 EP 路径）需要读取其他命名空间，会直接抛错，需要同步扩展 ROLE_NAMESPACE_SETS。
  5. 性能：off 模式仅多一条可剪枝比较；record 模式去重后每新 pair 一次文件追加；enforce 模式每次 config_bag 一次集合查找，均不在前向热路径内，风险可忽略。 - 影响：影响范围中等偏小：默认 off 模式零行为变化，对现有部署无感；enforce/record 均为 opt-in 环境变量，主要影响配置开发者与运维。对系统而言，config_bag 读取路径新增一道模式门，为后续按角色收窄命名空间投影、消除跨进程类型配置耦合奠定机制基础；对团队而言，确立了两条长期约定——新 SGLANG_* 开关须注册 Envs 类、fail-closed 路径须在 publish 时校验而非首次读取时暴露；dp_controller 是本 PR 唯一被实际约束的角色，已通过冒烟验证。 - 风险标记：核心配置读取路径变更，record 审计对编译区域盲区，OUT 写失败无重试，enforce 误配即启动失败，dp_controller 已前置收窄

关联脉络

- PR #33012 runtime_context: record the publishing process role: 直接前置：让 publish(role=...) 记录进程角色并迁移全进程入口到角色化 publish，本 PR 的 enforce 正是基于该角色信息接线。
- PR #33013 config: read resolved config via namespace accessors: 同系列：将 160 个文件 628 处配置读取迁移到命名空间 bag 访问器，本 PR 的 config_bag 强制检查正是作用于这一新读取路径。

- PR #33011 config: preserve resolved config across nested publishes + mutation ratchets: 同系列配置生命周期护栏：配置快照恢复与双基线 ratchet 与本 PR 的 fail-closed 强制共同构成 config-namespace 迁移的安全网。