

PR #39001 完整报告

sgl-project/sglang

[Router] Fleet-wide sampling contract 2/3: enforce and inject per request

合并时间: 2026-09-16 15:29

原文链接: <http://prhub.com.cn/sgl-project/sglang/pull/39001>

[Router] Fleet-wide sampling contract 2/3: 把采样契约落到每个请求

执行摘要

本 PR 是 fleet 级采样契约三步走的第二步，在 `experimental/sgl-router` 的 chat 入口把 `--override-sampling-params` 从「只有配置面」变成「真正生效」：请求省略受管采样字段时注入运维声明的值（显式 `null` 视同省略），`--sampling-param-conflict reject` 下客户端值与契约不一致时在准入前直接 400，`allow` 下原样转发。为支撑这一点，请求体探测层被改造成在反序列化期就把受管键定形为三态 `ProbedValue`，并新增了 `sampling_contract_violation` 错误码与按参数打标的拒绝计数器。flag 未设置（默认）时请求路径行为不变，风险集中在 `reject` 灰度的 fail-closed 取舍，以及 review 中发现并已修复的一次契约绕过。

功能与动机

PR body 把整个改造拆成三步，并明确 step 1 (#39000, 已合并) 只加了类型、解析器和两个 flag: "this is the PR that makes the flag do something"。动机可归为三层：

1. 防止引擎默认值漂移：只要运维声明了契约，请求省略字段时必须注入配置值，"so the engine's defaults cannot drift from what the operator declared"。
2. 拒绝发生在准入前：`reject` 模式下客户端显式发送不同值时直接 400，"costing no queue slot and no engine round-trip"；`allow` 模式则一个字都不改写客户端值。
3. 灰度必须可观测：把 `reject` 滚到整个 fleet 会把原本正常服务的流量变成 400，"how much and on which parameter has to be answerable without reading logs"，因此需要独立错误码与按参数打标的计数器。

此外 PR 澄清了与引擎自身 `--preferred-sampling-params` 的分工：即便引擎侧修好，也只覆盖单引擎的注入半边；「准入前拒绝、band、跨 fleet 统一契约」这半边只能留在路由器。

实现拆解

1. 探测结构扩容 + 手写反序列化 (`experimental/sgl-router/src/server/routes/chat.rs`)：
`RequestProbe` 从 `#[derive(Debug, Deserialize)]` 改为 `#[derive(Debug, Default)]` 加手写 `Deserialize` (`ProbeVisitor::visit_map`)，新增 `sampling: [ProbedValue; SamplingField::ALL.len()]`，按 `SamplingField::index` 索引，治理新参数无需改结构；同时删掉原先独立的 `HashMap<String, IgnoredAny>` 形状锚定 pass，因为 `deserialize_map` 本身就拒绝 `null / [] / "hi"`，多 MiB 的 body 不再被走两遍。新增 `ProbeKey` 用借用的

&str 匹配出判别式，未识别键不产生 String。原因：这个探测在每个请求上都跑，无论是否配置了契约。

2. 值的三态定形：ProbedValue = Absent / Number(f64) / Unusable，全部在反序列化期解析完成，不保留任何客户端大小的数据（visit_seq / visit_map 用 IgnoredAny 排空后返回 Unusable）。visit_str 走新增的 parse_as_engine_number；visit_bool 把 true / false 读成 1.0 / 0.0；visit_unit 把 null 归为 Absent。解析限长 MAX_SAMPLING_NUMERIC_LEN = 64，去下划线写入固定栈缓冲，客户端可控长串不进入分配路径。
3. 逐参数判定与注入：apply_sampling_overrides 遍历所有受管参数——Absent 注入配置值（两种冲突模式都注入）；Number 与配置不一致时 reject 400、allow 放行；Unusable 在 reject 下被拒。即使多个参数同时违规，每个参数都会被判定并计数，客户端只被告知一个。注入集合由 chat_completions 穿进既有的 build_outgoing_body 解析路径，绕开该解析的零拷贝注入留给 stack 3。重复键语义刻意区分：采样键 last-wins（对齐引擎的 json.loads），RoutingKey（stream / model / max_tokens / max_completion_tokens）重复即 400。
4. 错误面与指标：experimental/sgl-router/src/server/error.rs 新增 ApiError::SamplingContract { param, detail }，映射为 400 + x-router-error-code: sampling_contract_violation，与畸形 JSON、缺 model 的 bad_request 区分；experimental/sgl-router/src/server/metrics.rs 新增 record_sampling_contract_rejection 与 sgl_router_sampling_contract_rejections_total {param} 渲染，param 来自 &'static str 固定枚举，标签基数有界。
5. 容错、测试与配套修复：探测这 8 个键让 serde_json 开始转换其数字字面量（此前 IgnoredAny 只扫描），{"temperature": 1e400} 会让整包解析失败并报出与请求体不符的「body must be a JSON object」，因此 parse_probe 退回不转换的 probe_without_sampling_values。测试上，chat.rs 内 21 个单元测试覆盖拒绝契约、band 准入、allow 既不掩盖也不拒绝、null 视为省略、双向重复键、超大非数字值排空不物化、探测值归一化为数字；numeric_strings_are_read_the_way_the_engine_reads_the_m 的 63 个输入期望值由 pydantic 2.13.5 实跑生成；error.rs / metrics.rs 各补一个错误码与计数器测试，cargo test 793 passed。配套上，tests/proxy/pd_protocol_binding.rs 的 config() 补 sampling_overrides: Default::default()，修掉 #39000 与 #39006 的语义冲突造成的 main 编译红灯。

experimental/sgl-router/src/server/metrics.rs

新增 sampling_contract_rejections_total 字段、record_sampling_contract_rejection 记录函数与 Prometheus 渲染段，使 reject 的灰度具备「挡了多少、挡在哪个参数」的可观测性；并带一个按参数打标的单元测试。

experimental/sgl-router/src/server/metrics.rs：注册表字段、记录函数与渲染段，构成完整的计数器闭环。

```
// MetricsRegistry 新增字段：`param` 是固定枚举的 wire name，标签集合因此有界。  
sampling_contract_rejections_total: Mutex<HashMap<&'static str, Arc<AtomicU64>>>>，
```

```
impl MetricsRegistry {
```

```

/// 递增 `sgl_router_sampling_contract_rejections_total{param}`。
///
/// 仅在 fleet 级采样契约于 `--sampling-param-conflict reject` 下拒绝请求时记录。
/// 这是该 flag 的灰度指标：它回答「契约挡掉了多少客户端流量、挡在哪个参数上」——
/// 这个问题否则无法回答，因为拒绝和别的 400 在客户端看来一模一样。
pub fn record_sampling_contract_rejection(&self, param: &'static str) {
    let mut guard = self.sampling_contract_rejections_total.lock();
    let counter = guard
        .entry(param)
        .or_insert_with(|| Arc::new(AtomicU64::new(0)))
        .clone();
    drop(guard);
    counter.fetch_add(1, Ordering::Relaxed);
}

// render_with_workers 中的渲染段：真正发生过拒绝之前不输出任何 series。
out.push_str(
    "# HELP sgl_router_sampling_contract_rejections_total Requests refused by the fleet-wide
    sampling contract (--override-sampling-params under --sampling-param-conflict reject), by
    parameter.\n",
);
out.push_str("# TYPE sgl_router_sampling_contract_rejections_total counter\n");
let guard = self.sampling_contract_rejections_total.lock();
let mut entries: Vec<(&str, u64)> = guard
    .iter()
    .map(|(k, v)| (*k, v.load(Ordering::Relaxed)))
    .collect();
entries.sort_by_key(|entry| entry.0);
for (param, value) in entries {
    out.push_str(&format!(
        "sgl_router_sampling_contract_rejections_total{{param=\"{}\"}} {}{}\n",
        escape_label(param),
        value,
    ));
}
drop(guard);

```

experimental/sgl-router/src/server/error.rs

新增 `ApiError::SamplingContract` 变体并映射到独立的 400 + `sampling_contract_violation` 错误码，使契约拒绝可与畸形 JSON、缺 `model` 的 `bad_request` 区分并独立告警；同时补上区分性测试。

`experimental/sgl-router/src/server/error.rs`：新的错误变体、状态码映射，以及保证「可独立告警」的测试。

```

/// 被 fleet 级采样契约拒绝的请求（`--override-sampling-params` 配合
/// `--sampling-param-conflict reject`）。
///

```

```

/// 有意与 [`Self::BadRequest`] 分开：把契约滚到整个 fleet 上会把原本正常服务的
/// 流量变成 400，而运维的第一个问题就是「多少、挡在哪个参数」。折进 `bad_request`
/// 就答不出来——那个 code 已经被畸形 JSON 和缺 `model` 字段占用了。
/// `param` 是 [`crate::config::SamplingField::wire_name`] 的 `&'static str`，
/// 因此还能安全地当有界的指标标签用。
#[error("{param} violates this deployment's sampling contract: {detail}")]
SamplingContract { param: &'static str, detail: String },

impl ApiError {
    fn status_and_code(&self) -> (StatusCode, &'static str) {
        match self {
            ApiError::BadRequest(_) => (StatusCode::BAD_REQUEST, "bad_request"),
            // 独立 code：客户端可从 `x-router-error-code` 头读到它，
            // 不与 bad_request 混在一起。
            ApiError::SamplingContract { .. } => {
                (StatusCode::BAD_REQUEST, "sampling_contract_violation")
            }
            ApiError::ModelNotFound(_) => (StatusCode::NOT_FOUND, "model_not_found"),
            // ... 其余分支保持不变
        }
    }
}

// 测试：拒绝必须与普通 bad_request 可区分，且参数与实际值要回传客户端。
#[test]
fn sampling_contract_has_a_distinct_code_from_other_bad_requests() {
    let err = ApiError::SamplingContract {
        param: "temperature",
        detail: "got 0.5, expected 1 (or omit the field)".into(),
    };
    let (status, code_header, env) = parse_envelope(err.into_response());
    assert_eq!(status, StatusCode::BAD_REQUEST);
    assert_eq!(code_header.as_deref(), Some("sampling_contract_violation"));
    assert_eq!(env.error.code, "sampling_contract_violation");
    assert_ne!(env.error.code, "bad_request");
    assert!(
        env.error.message.contains("temperature") && env.error.message.contains("0.5"),
        "got: {}",
        env.error.message
    );
}

```

评论区精华

sherlockwu: "The **reject** contract can be bypassed because the router and engine interpret some values differently." 他给出对照表: "**temperature**": **false** 在路由器侧是 **Unusable** (不检查也不注入)，引擎用 0.0; "**temperature**": "0.5_0" 同理，引擎用 0.5。Proposed fix: add some normalization: convert booleans to 0/1, and support Pydantic's numeric-string syntax, including underscores.

Kangyan-Zhou: "Good catch — confirmed both, and the fix is pushed." 他用 pydantic 2.13.5 对 sglang 自己的字段声明 (`temperature: Optional[float] = None`, 无 validator、无 strict) 实跑 33 个输入, 7 例分歧, 恰好都在 sherlockwu 指出的两类; 随后反驳提案的核心: "Implementing support Pydantic's numeric-string syntax, including underscores with Python's own numeric-literal rule ... gets 5 of 18 cases wrong. Pydantic does not use Python's rule." 因此修复重心从「把规则补全」改为「拒绝读不出来的残差」 (fail-closed), `parse_as_engine_number` 只负责「让引擎接受的输入里有多少被精确回答」。

评审结论: sherlockwu 给出 APPROVED (无附加正文)。整个讨论收敛为一条明确判据——契约强度不能建立在对第三方未文档化强转表的保真度上。

风险与影响

风险	具体位置	影响
reject 误拒 (fail-closed)	<code>apply_sampling_overrides + ProbedValue::Unusable</code>	引擎接受但探测读不出的值会被 400; pydantic 无版本下限, fleet 内行为可能不一致, 需靠新计数器灰度观察
注入改写客户端语义	两种冲突模式下的注入分支	设置 flag 后, 省略字段的请求采样行为被静默改为运维声明值; <code>allow</code> 的「契约强度」实际来自注入而非拒绝
解析路径回归面	<code>parse_probe / probe_without_sampling_values</code>	已出现 <code>1e400</code> 破坏整包解析并被误报成 body 形状错误; 新增受管参数时需重复验证探测与 <code>IgnoredAny</code> 扫描的差异
重复键行为不对称	<code>RoutingKey</code> vs <code>SamplingField</code>	采样键 last-wins、路由键 400, 需在发布说明中明确; 是否有客户端依赖旧行为材料中未验证
缺少代理层 e2e	仅 <code>chat.rs / error.rs / metrics.rs</code> 单元测试	400 信封、error code header、计数器只在单元层断言, 跨进程转发路径留待 stack 3
跨 PR 语义冲突	<code>tests/proxy/pd_protocol_binding.rs</code> 的穷尽 <code>ModelConfig</code> 字面量	<code>#39000 + #39006</code> 令 main 编译红灯, 同类冲突在配置面快速演进期仍会复发

影响面: 用户侧默认零变化 (flag 关闭), 开启后可通过错误码与消息自助排查; 系统侧 chat 入口的两次 `deserialize` pass 合并为一次, 未配置契约时零分配, 拒绝发生在准入前; 运维侧首次获得「挡了多少、挡在哪个参数」的量化视图。

关联脉络

- #39000 (stack 1, 已合并) : 提供 SamplingOverrides / SamplingField / ConflictPolicy 与两个 flag, 本 PR 才让它们生效; 探测数组长度、wire name 与错误码参数类型都从那里继承。
- #39002 (stack 3) : 按 PR body, 它负责「绕开 build_outgoing_body 解析的零拷贝注入、e2e proxy 测试与 README」, 本 PR 明确把这两项缺口留给它。
- #39004 / #39006 一线: 本 PR 第 4 个 commit 修的
tests/proxy/pd_protocol_binding.rs 出自按 worker 解析 h2c/HTTP1 协议的这条工作线, 两个 PR 只在文本上不冲突、语义上冲突——这也是本 PR 顺带解除 main 红灯的原因。
- 趋势判断: router 正在从「转发器」演化为「fleet 策略执行点」——采样契约、HiCache 分层路由、wire protocol 解析都在同一入口收敛; 本 PR 贡献的「前置探测 + 显式 fail-closed + 按参数计量」模式, 大概率会被后续的边缘策略 (如缓存准入、band 治理) 复用。