PR #50528 完整报告

vllm-project/vllm

[Bugfix][Parser] Emit REASONING_END for Inkling tool calls that follow no thinking block

合并时间：2026-08-10 09:04

原文链接：http://prhub.com.cn/vllm-project/vllm/pull/50528

执行摘要

- 一句话：修复 Inkling 无思考块开头工具调用泄漏为 content
- 推荐动作：值得精读。这是一个观察“两遍解析不对称性”如何引发流式边界 bug 的典型修复案例：配置层一行事件、引擎层一次顺序重排，配合精确的 blast radius 审计与双形态门测试。重点阅读 StreamingParserEngine._on_terminal 的控制流重组，以及如何在不破坏 #51391 记账逻辑的前提下完成组合；测试设计（正反两种表形态的引擎级门测试）也值得借鉴。

功能与动机

修复 issue #50512：流式模式下，对话回合直接以工具调用开头时（判别条件为 是否在工具块前确认了推理边界 而非轮数，单轮同样泄漏），Inkling 工具块泄漏到 delta.content 且 tool_calls 为空。PR body 明确指出根因：推理 pass 以 skip_tool_parsing 运行，只有看到 REASONING_END 才交接给工具 pass，而 #49876 只让渲染可见内容的 opener（TEXT_START、TOOL_TEXT、TOOL_ERROR）确认边界，TOOL_START 被遗漏。

实现拆解

1. 配置层补齐 REASONING_END：在 vllm/parser/inkling.py 的 inkling_config() 中，将 (CONTENT, TOOL_START) 与 (MESSAGE_HEADER, TOOL_START) 两条转移的事件从 (TOOL_CALL_START,) 扩展为 (REASONING_END, TOOL_CALL_START)。理由：工具块是唯一不渲染可见内容却能直接开局的开场器，#49876 只覆盖了可见内容 opener；同时与 qwen3 / gemma4 / glm47_moe / mistral 的既有约定（CONTENT 状态下工具开场携带 REASONING_END）对齐。
2. 引擎门重排控制流：在 streaming_parser_engine.py 的 StreamingParserEngine._on_terminal 中，处于 skip_tool_parsing 且命中工具 terminal 的分支内，将 MESSAGE_HEADER 直通返回拆分为两步：先记录 leaving_message_header 并清空 _message_header_buffer，再先检查 transition.events 是否含 REASONING_END，有则返回 REASONING_END + TEXT_CHUNK 组合，否则回退到原直通行为。这样保证工具 terminal 携带的 REASONING_END 不被 header 直通吞掉；#51391 的 _in_skipped_tool_span 记账逻辑保持原样。
3. 测试配套：tests/parser/engine/test_inkling.py 新增 4 个用例——从 MESSAGE_HEADER 直达 TOOL_START（chunk_size 1 / 3 / 7 / 64 参数化）、函数名

header 场景、CONTENT 状态经文本块后的 TOOL_START、以及直接驱动 StreamingParserEngine 验证 (CONTENT, TOOL_START) 在推理 pass 中先发 REASONING_END；新增导入 EventType、StreamingParserEngine、inkling_config。tests/parser/engine/test_engine.py 新增 _message_header_config 辅助构造双形态语法（带 / 不带 REASONING_END），以及 TestSkipToolParsingFromMessageHeader 类验证“带则先发 REASONING_END、不带则保持旧直通”。

4. 验证与组合：PR 基于当前 main 并与 #51391、#49876 组合；在未应用修复的 main 上 7 個新测试失败，应用后 9 个全过；bbrowning 合并复核时跑了 4082 个 parser / reasoning / tool_parser 测试全通过，并用 live Inkling 模型确认流式无思考工具调用场景修复。测试均为纯 CPU mock，未引入新文件。

关键文件：

- vllm/parser/engine/streaming_parser_engine.py（模块 解析引擎；类别 source；类型 core-logic；符号 StreamingParserEngine._on_terminal）：共享解析引擎的核心改动位置：在 skip_tool_parsing 门内重排控制流，让携带 REASONING_END 的转移先于 MESSAGE_HEADER 直通执行，否则配置层补的事件会被直通吞掉。
- vllm/parser/inkling.py（模块 解析器；类别 source；类型 core-logic；符号 inkling_config）：配置层修复：给两个 TOOL_START 转移补上 REASONING_END，是修复的语义来源，并与其他 parser（qwen3/gemma4 等）约定对齐。
- tests/parser/engine/test_inkling.py（模块 解析器测试；类别 test；类型 test-coverage；符号 test_tool_start_from_message_header_streaming, test_function_name_header_before_tool_start_streaming, test_content_state_tool_start_streaming, test_content_tool_start_emits_reasoning_end_in_reasoning_pass）：Inkling 专属回归测试：覆盖 MESSAGE_HEADER 直达工具块、函数名 header、CONTENT 状态经文本块后的工具块，以及推理 pass 中 (CONTENT, TOOL_START) 的事件顺序。
- tests/parser/engine/test_engine.py（模块 引擎测试；类别 test；类型 test-coverage；符号 _message_header_config, TestSkipToolParsingFromMessageHeader, _skip_events, test_reasoning_end_precedes_forwarded_tool_syntax）：共享引擎门的守护测试：用合成的 MESSAGE_HEADER 语法验证“带 REASONING_END 则先发事件、不带则保持旧直通”，保护非 Inkling parser 不受影响。

关键符号：StreamingParserEngine._on_terminal, inkling_config, test_tool_start_from_message_header_streaming, test_reasoning_end_precedes_forwarded_tool_syntax

关键源码片段

vllm/parser/engine/streaming_parser_engine.py

共享解析引擎的核心改动位置：在 skip_tool_parsing 门内重排控制流，让携带 REASONING_END 的转移先于 MESSAGE_HEADER 直通执行，否则配置层补的事件会被直通吞掉。

```
def _on_terminal(self, terminal: str, value: str) -> list[SemanticEvent]:
```

```
key = (self.state, terminal)
transition = self.config.transitions.get(key)

if transition is None:
    if self._has_drops and terminal == DROP_TERMINAL:
        return []
    # 投影出的 skip 状态可能未定义 wrapper 闭合器，这里只负责退出标记。
    if self.skip_tool_parsing and terminal in self._tool_exit_terminals:
        self._in_skipped_tool_span = False
    return self._emit_for_state(value)

if self.skip_tool_parsing and terminal in self._tool_terminals:
    # Inkling 复用同一 terminal 作为工具、文本、推理的退出标记。
    # 若不在转发工具 span 内，就按普通转移处理。
    is_opener = transition.next_state in self._TOOL_STATES
    is_exit = terminal in self._tool_exit_terminals
    used_as_plain_closer = (
        is_exit and not is_opener and not self._in_skipped_tool_span
    )
    if not used_as_plain_closer:
        if is_opener:
            self._in_skipped_tool_span = True
        elif is_exit:
            self._in_skipped_tool_span = False
        leaving_message_header = self.state == ParserState.MESSAGE_HEADER
        if leaving_message_header:
            self._message_header_buffer = ""
        # 工具 terminal 若隐式结束推理，即使在 header 状态也必须上报
        # REASONING_END，否则推理 pass 永远不会把块交给工具 pass。
        if EventType.REASONING_END in transition.events:
            self.state = ParserState.CONTENT
            return [
                SemanticEvent(
                    EventType.REASONING_END, value=value, tool_index=self.tool_index
                ),
                SemanticEvent(
                    EventType.TEXT_CHUNK, value=value, tool_index=self.tool_index
                ),
            ]
        elif leaving_message_header:
            self.state = ParserState.CONTENT
            return [
                SemanticEvent(
                    EventType.TEXT_CHUNK, value=value, tool_index=self.tool_index
                )
            ]
        content_type = self.config.content_events.get(self.state)
        if content_type is not None:
            return [
```

```
                SemanticEvent(
                    content_type, value=value, tool_index=self.tool_index
                )
            ]
        return []

    if transition.skip_in_token_id_mode and self._ever_had_token_ids:
        return self._emit_for_state(value)

    return self._apply_transition(transition, value)
```

vllm/parser/inkling.py

配置层修复：给两个 TOOL_START 转移补上 REASONING_END，是修复的语义来源，并与其他 parser（qwen3/gemma4 等）约定对齐。

```
# 打开一个渲染为可见内容的块即证明推理已结束（#49876 覆盖 TEXT_START、
# TOOL_TEXT、TOOL_ERROR）；工具块不渲染可见内容，却是唯一能无前导块
# 直接开局的开场器，因此 TOOL_START 必须携带 REASONING_END。
(ParserState.CONTENT, "TOOL_START"): Transition(
    ParserState.TOOL_ARGS,
    (EventType.REASONING_END, EventType.TOOL_CALL_START),
),
# MESSAGE_HEADER 是推理 pass 下工具 terminal 经由 skip_tool_parsing
# 门转发的路径，这里同样补发 REASONING_END，与 qwen3 / gemma4 等
# 配置中 (CONTENT, TOOL_START) 的写法保持一致。
(ParserState.MESSAGE_HEADER, "TOOL_START"): Transition(
    ParserState.TOOL_ARGS,
    (EventType.REASONING_END, EventType.TOOL_CALL_START),
),
```

评论区精华

bbrowning 在首次 review 中要求 rebase 并适配 #51391，明确两点：一是必须保留 #51391 的 _in_skipped_tool_span、is_opener、is_exit、used_as_plain_closer 逻辑，不要退回旧分支；二是在 MESSAGE_HEADER 处理中先处理携带 REASONING_END 的转移，再走普通直通。其原话强调：“the essential fix here is that when we detect an actual tool call beings, we have to emit REASONING_END”。批准时 bbrowning 附上完整验证：4082 个单元测试通过，并用 live Inkling 模型复现脚本确认流式无思考 + 工具场景的最后一个失败用例被修复。mergify 曾提示合并冲突，thegoldenflow rebase 后解决。

- Rebase 与 #51391 组合要求 (design): thegoldenflow rebase 后按要求重排控制流，保留 #51391 记账逻辑，bbrowning 批准。
- 共享引擎门回归验证 (testing): 共享门改动经广泛回归测试确认无副作用。
- Merge conflict 提示 (other): thegoldenflow rebase 到 main 并更新实现与测试后解决。

风险与影响

- 风险：

1. 共享引擎控制流变更：StreamingParserEngine._on_terminal 是所有基于 engine 的 parser 的公共路径，虽然 grep 显示只有 Inkling 配置命名 MESSAGE_HEADER，但事件顺序调整理论上影响其他 parser；缓解措施是 bbrowning 跑了 4082 个相关测试（含 deepseek/gemma/glm/qwen 等）无回归。
2. 两遍解析架构不对称：Python 端依赖 skip_tool_parsing + reasoning_ended 交接，Rust 端统一单遍解析天然无此问题，长期需维护行为一致性，避免两侧漂移。
3. REASONING_END 幂等性依赖：当前 handler 是置位 self._reasoning_ended = True，重复触发无害，但若 handler 将来引入副作用（如计数、回调）需注意重复发射。
4. 测试依赖 mock tokenizer 与合成 token 流，未覆盖真实模型 tokenizer 边界与 chunk 切分不整齐的情况；live 验证虽弥补，但 CI 中未固化模型级回归测试。 - 影响：用户侧：启用 Inkling 工具调用 + 流式的 agent 框架（如 Open WebUI）不再静默丢失工具调用，tool block 不再以 raw markup 形式泄漏进 content。系统侧：仅影响 parser 事件分类，不改变模型生成、采样或精度，理论上无性能影响。团队侧：共享引擎门的行为约定更新——所有 parser 配置需保证工具开场转移携带 REASONING_END；后续新 parser 需遵循同一边界确认约定，且改动共享门时需要类似 blast radius 审计。 - 风险标记：共享引擎控制流变更，两遍解析架构不对称，依赖 mock 测试与手动 live 验证，REASONING_END 幂等性依赖

关联脉络

- PR #51391 [Bugfix][Parser] Prevent Inkling block-end leakage with tools: 前置 PR，引入 skipped-tool-span 记账逻辑（_in_skipped_tool_span、is_opener/is_exit/used_as_plain_closer），本 PR 的引擎改动嵌套在该分支内并保持其不变；测试中的精确断言依赖 #51391 消除尾部泄漏。
- PR #49876 [Bugfix][Parser] Confirm reasoning end when an Inkling content block opens: 同一边界问题的互补半：给可见内容 opener（TEXT_START/TOOL_TEXT/TOOL_ERROR）补 REASONING_END；本 PR 补齐 TOOL_START 这个唯一不渲染可见内容的工具开局器，二者合拢两遍解析的交接缺口。