

PR #54935 完整报告

vllm-project/vllm

[Security] Cap GLMGA video sampling to prevent request-driven resource exhaustion

合并时间: 2026-09-04 20:02

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/54935>

执行摘要

- 一句话: 为 GLMGA 视频后端添加采样上限, 防止请求驱动的资源耗尽。
- 推荐动作: 此 PR 值得精读。它展示了如何为用户可控参数添加多层防御以防止资源耗尽攻击, 模式清晰。但 CodeRabbit 关于中间列表分配的评论指出了一个尚未解决的性能 / 稳定性问题, 值得关注。建议在后续 PR 中重构采样逻辑, 直接生成有界索引。

功能与动机

修复一个安全漏洞 (GHSA-58v5-2m8f-94pr)。根据 PR body 描述, `GLMGAVideoBackend.compute_frames_index_to_sample` 计算 `extract_t = min(int(duration * target_fps), max_frames)`, 其中 `target_fps` 和 `max_frames` 均可通过请求级 `media_io_kwargs` 控制。当 `total_frames_num < extract_t` 时, 列表推导式 `range(extract_t)` 会创建攻击者控制的、远超实际帧数的中间列表分配, 造成拒绝服务风险。

实现拆解

1. 源码变更 (`vllm/multimodal/video.py`): 在 `GLMGAVideoBackend` 类中新增两个类级别常量: `_MAX_FRAMES = 640` 和 `_MAX_FPS = 30`。修改 `compute_frames_index_to_sample` 方法: 将 `target_fps` 钳制到 `min(target_fps, cls._MAX_FPS)`; 将 `max_frames` 钳制到 `min(kwargs.get("max_frames", cls._MAX_FRAMES), cls._MAX_FRAMES)`; 在计算 `extract_t` 时, 增加对 `total_frames_num` 的钳制 `min(extract_t, max_frames, total_frames_num)`。
2. 测试配套 (`tests/multimodal/test_video.py`): 新增测试类 `TestGLMGASamplingCaps`, 包含四个测试用例: `test_extreme_values_bounded_by_total_frames` (验证极端参数下索引数不超过实际帧数)、`test_class_cap_overrides_kwargs_max_frames` (验证类上限覆盖请求参数)、`test_class_cap_overrides_target_fps` (验证类上限覆盖请求 fps)、`test_normal_operation_unchanged` (验证正常功能不受影响)。

关键文件:

- `vllm/multimodal/video.py` (模块 多模态视频; 类别 source; 类型 core-logic; 符号 `GLMGAVideoBackend`, `_MAX_FRAMES`, `_MAX_FPS`, `compute_frames_index_to_sample`): 核心安全修复文件, 为 `GLMGAVideoBackend` 添加了输入钳制常量和逻辑, 防止资源耗尽。

- tests/multimodal/test_video.py (模块 多模态测试; 类别 test; 类型 test-coverage; 符号 TestGLMGASamplingCaps, _source, test_extreme_values_bounded_by_total_frames, test_class_cap_overrides_kwargs_max_frames) : 新增回归测试类, 专门验证安全修复的有效性和正常功能的兼容性。

关键符号: GLMGAVideoBackend.compute_frames_index_to_sample

关键源码片段

vllm/multimodal/video.py

核心安全修复文件, 为 GLMGAVideoBackend 添加了输入钳制常量和逻辑, 防止资源耗尽。

```
# vllm/multimodal/video.py 中 GLMGAVideoBackend 类的关键变更
@VIDEO_LOADER_REGISTRY.register(
    "glmga",
    video_processor="GlmgaVideoProcessor",
)
class GLMGAVideoBackend(VideoBackend):
    # 新增类级别安全常量, 防止请求参数导致资源耗尽
    _MAX_FRAMES: ClassVar[int] = 640
    _MAX_FPS: ClassVar[int] = 30

    @classmethod
    def compute_frames_index_to_sample(
        cls,
        source: VideoSourceMetadata,
        target: VideoTargetMetadata,
        **kwargs,
    ) -> list[int]:
        total_frames_num = source.total_frames_num
        duration = source.duration
        original_fps = source.original_fps
        # 将请求的 fps 钳制到类级别上限, 防止创建过大中间列表
        target_fps = min(target.fps, cls._MAX_FPS)
        max_frame_idx = source.total_frames_num - 1
        # 将请求的 max_frames 钳制到类级别上限, 双重保险
        max_frames = min(kwargs.get("max_frames", cls._MAX_FRAMES), cls._MAX_FRAMES)

        duration = duration or round(max_frame_idx / original_fps) + 1

        extract_t = int(duration * target_fps)
        # 关键安全修复: 将 extract_t 同时钳制到 max_frames 和视频实际帧数
        # 这确保后续生成的索引列表不会超过视频实际内容, 即使前两个钳制被绕过
        extract_t = min(extract_t, max_frames, total_frames_num)
        # ... 后续采样逻辑不变 ...
```

tests/multimodal/test_video.py

新增回归测试类, 专门验证安全修复的有效性和正常功能的兼容性。

tests/multimodal/test_video.py 中新增的测试类

class TestGLMGASamplingCaps:

"""Regression tests for GHSA-58v5-2m8f-94pr: request-controlled fps and max_frames must not create intermediate allocations larger than the actual frame count."""

@staticmethod

def _source(

total_frames: int, fps: float = 30.0, duration: float = 0

) -> VideoSourceMetadata:

if duration == 0 and fps > 0 and total_frames > 1:

duration = round((total_frames - 1) / fps) + 1

return VideoSourceMetadata(total_frames, fps, duration)

def test_extreme_values_bounded_by_total_frames(self):

核心安全测试: 2 帧视频, 请求极大 fps 和 max_frames, 确保索引数不超过 2

source = self._source(total_frames=2, fps=2.0, duration=1.0)

target = VideoTargetMetadata(num_frames=-1, fps=500_000, max_duration=-1)

indices = GLMGAVideoBackend.compute_frames_index_to_sample(

source,

target,

max_frames=500_000,

)

assert len(indices) <= 2

def test_class_cap_overrides_kwargs_max_frames(self):

验证 _MAX_FRAMES 类常量覆盖请求参数

source = self._source(total_frames=10_000, fps=30.0)

target = VideoTargetMetadata(num_frames=-1, fps=30, max_duration=-1)

indices = GLMGAVideoBackend.compute_frames_index_to_sample(

source,

target,

max_frames=100_000,

)

assert len(indices) <= GLMGAVideoBackend._MAX_FRAMES

def test_class_cap_overrides_target_fps(self):

验证 _MAX_FPS 类常量覆盖请求 fps

source = self._source(total_frames=10_000, fps=30.0)

target = VideoTargetMetadata(num_frames=-1, fps=500_000, max_duration=-1)

indices = GLMGAVideoBackend.compute_frames_index_to_sample(

source,

target,

)

assert len(indices) <= GLMGAVideoBackend._MAX_FRAMES

def test_normal_operation_unchanged(self):

验证正常视频处理功能不受安全限制影响

source = self._source(total_frames=1000, fps=30.0, duration=33.0)

```
target = VideoTargetMetadata(num_frames=-1, fps=2, max_duration=-1)
indices = GLMGAVideoBackend.compute_frames_index_to_sample(
    source,
    target,
)
assert 0 < len(indices) <= GLMGAVideoBackend._MAX_FRAMES
assert all(0 <= idx < 1000 for idx in indices)
```

评论区精华

CodeRabbit 在 review 中提出了两个重要意见：

1) 在 `vllm/multimodal/video.py` 第 819 行指出，虽然 `extract_t` 被钳制，但循环仍会生成大量中间索引列表（在最终重采样前），建议重构采样逻辑以直接生成有界索引列表。2) 在 `tests/multimodal/test_video.py` 第 1517 行指出，当前 `test_class_cap_overrides_target_fps` 测试无法区分 fps 上限和 `_MAX_FRAMES` 限制，建议使用更精确的测试用例。这两个建议均未被采纳，但 PR 作者 Isotr0py 在审批时表示“LGTM，可能需要在后续 PR 中重构以识别哪些 kwargs 可被运行时覆盖”，暗示了这些技术债务。

- 测试未能有效区分 fps 上限和帧数上限 (testing): 未采纳，测试保持不变。PR 作者在审批时暗示可能在后续 PR 中重构相关 kwargs 处理逻辑。
- 中间索引列表可能仍然过大 (performance): 未采纳。这属于未解决的技术债务，可能在后续 PR 中优化。

风险与影响

- 风险：
 1. 功能回归风险：虽然新增了正常操作测试，但对现有使用 GLMGAVideoBackend 的用户或模型，如果其依赖超过 640 帧或 30 fps 的采样，行为将被强制限制，可能影响视频处理质量。
 2. 中间分配风险：如 CodeRabbit 指出，当 `total_frames_num` 远大于 `extract_t` 时，现有循环仍可能生成大量中间索引（尽管最终会缩减），但此次修复未解决此点。
 3. 兼容性风险：此修复为安全加固，属于必要变更，但可能影响一些边缘用例。
- 影响：
 1. 用户影响：所有使用 GLM-GA 视频处理的用户将受到采样上限约束，确保请求安全。
 2. 系统影响：增强了 vllm 多模态处理层的资源安全性，防止通过恶意视频请求导致服务拒绝。
 3. 团队影响：为处理类似资源控制问题提供了模式参考（类级别常量 + 多层钳制），但 CodeRabbit 的反馈指出了进一步优化的方向。 - 风险标记：安全加固可能影响边缘用例，中间列表分配风险未完全解决

关联脉络

- 暂无明显关联 PR