

PR #51446 完整报告

vllm-project/vllm

Preserve revision pins in secondary artifact loaders

合并时间: 2026-08-11 12:33

原文链接: <http://prhub.com.cn/vllm-project/vllm/pull/51446>

执行摘要

- 一句话: 修复次要加载器丢失 revision 钉住的安全问题
- 推荐动作: 值得精读。这是一次针对安全公告的跨模块修复, 虽然每个点的改动都很机械 (多传一个参数), 但整体呈现了系统化的审计修复模式: 枚举所有 "次级读取入口"、统一透传 revision/code_revision、在外部下载边界用记录桩做 RED-to-GREEN 验证、并主动删除低价值的模型专属测试以控制维护成本。重点关注三处设计:
ModelArchConfigConvertorBase 构造器签名变更对子类的影响、
_PROCESSOR_CHAT_TEMPLATES 缓存键与 revision 的关系、以及 tokenizer_revision 与 revision 的语义区分。

功能与动机

PR body 明确指出: "An operator can use --revision and --code-revision to select reviewed model artifacts. Several secondary loaders honored the primary model selection but fetched related configuration, processor, quantization, or export files from the repository's changing default branch." 具体后果是 "A pinned deployment could behave differently from the commit the operator reviewed: quantized layer selection, exported tensorizer contents, audio preprocessing, chat templates, and architecture settings could come from another revision." 例如 DeepSeek 的架构修复在未携带 revision 的情况下重新打开 config.json, 可能用 main 分支的数据覆盖钉住的值。该问题由 Patch the Planet (Trail of Bits + OpenAI 合作) 发现并通过 GPT-5.5-Cyber 辅助定位, 对应 GHSA-3hxx-64c2-p36x; PR 同时声明修复并不声称可变仓库或 --trust-remote-code 本身安全。

实现拆解

变更入口是 ModelConfig 的两个方法与模型加载链路, 修订以参数透传的方式贯穿所有远端读取调用点, 共 6 个站点 + 测试配套:

1. 架构配置转换器携带 revision (vllm/transformers_utils/model_arch_config_convertor.py 与 vllm/config/model.py): ModelArchConfigConvertorBase.__init__ 以及 MimoV2ModelArchConfigConvertor、MimoV2MTPModelArchConfigConvertor 的构造器新增 revision: str | None = None 参数; ModelConfig.get_model_arch_config 用 getattr(self, "revision", None) 传入。_get_qk_rope_head_dim 修复 transformers v5.4+ attribute_map bug 时把 self.revision 传给 get_hf_file_to_dict("config.json",

model_path, ...), 确保恢复 qk_rope_head_dim 读取的是被钉住的提交而非默认分支。构造器签名变更是 data-contract 级改动, 所有子类与调用方需同步。

2. processor 与 chat-template 助手 (vllm/renderers/hf.py) :
_PROCESSOR_CHAT_TEMPLATES 缓存键从 (name_or_path, trust_remote_code) 扩展为 (name_or_path, revision, code_revision, trust_remote_code);
_try_get_processor_chat_template 与 resolve_chat_template 把 model_config.revision / code_revision 透传给 cached_get_processor, 避免处理器与模板文件来自未钉住的提交。
3. FunAudioChat/Tarsier2 处理器 (vllm/model_executor/models/funaudiochat.py) :
feature_extractor 改用 WhisperFeatureExtractor.from_pretrained(..., revision=self.ctx.model_config.revision); speech_tokenizer 改用 TokenizersBackend.from_pretrained(..., revision=self.ctx.model_config.tokenizer_revision), 钉住音频预处理相关文件。
4. generation-config 回退 (vllm/config/model.py 与 vllm/transformers_utils/config.py) :
try_get_generation_config 新增 code_revision 参数,
ModelConfig.try_get_generation_config 在 GenerationConfig.from_pretrained 失败回退到 get_config 时保留并传递 code_revision, 避免回退路径丢失代码钉住。
5. AutoGPTQ/AutoAWQ 量化元数据 (vllm/config/vllm.py) : 在共享调用点 VllmConfig._get_quantization_config 中把 model_config.revision 传给 quant_config.maybe_update_config(...); PR body 说明 AutoGPTQ 与 AutoAWQ 的补丁字节级一致 (PTP-VLLM-143/144), 故合并为一个共享实现提交。
6. Tensorizer 导出 (vllm/model_executor/model_loader/tensorizer.py) :
serialize_extra_artifacts 新增 revision 参数并用于 hf_api().snapshot_download(..., revision=revision); serialize_vllm_model 从 model_config.revision 传入。
7. 测试配套: 新增 tests/test_tensorizer_revision_pin.py、tests/test_quantization_revision_pin.py; 在 tests/config/test_model_arch_config.py 增加 test_qk_rope_head_dim_recovery_uses_model_revision, 在 tests/transformers_utils/test_config.py 增加 test_model_config_generation_fallback_fowards_code_revision; 并按评审删除 tests/model_executor/test_funaudiochat_revision_pins.py。所有聚焦测试在外部下载边界使用记录桩 (fake_open_stream、Mock 的 hf_api/snapshot_download), 无需网络即可断言 revision 被正确转发。

关键文件:

- vllm/transformers_utils/model_arch_config_convertor.py (模块 架构转换; 类别 source; 类型 data-contract; 符号 ModelArchConfigConvertorBase.init, ModelArchConfigConvertorBase._get_qk_rope_head_dim, MimoV2ModelArchConfigConvertor.init, MimoV2MTPModelArchConfigConvertor.init) :
安全修复的主站点之一: ModelArchConfigConvertorBase 构造器新增 revision 参数, _get_qk_rope_head_dim 在恢复被 transformers 损坏的 MLA 字段时从钉住的提交读取原始 config.json; 构造器签名变更影响全部子类。
- vllm/config/model.py (模块 模型配置; 类别 source; 类型 core-logic; 符号 ModelConfig.get_model_arch_config, ModelConfig.try_get_generation_config) : 把 revision 传入架构转换器构造器, 并在 generation-config 回退路径上保留 code_revision

；是两处修复的中枢调用点。

- vllm/renderers/hf.py (模块 渲染器; 类别 source; 类型 core-logic; 符号 `_try_get_processor_chat_template`, `resolve_chat_template`) : `processor_chat_template` 缓存键与调用同时纳入 `revision/code_revision`, 避免缓存命中错误修订的模板。
- vllm/model_executor/model_loader/tensorizer.py (模块 模型加载; 类别 source; 类型 core-logic; 符号 `serialize_extra_artifacts`, `serialize_vllm_model`) : `Tensorizer` 导出辅助文件时 `hf_api().snapshot_download` 补上 `revision`, 避免导出内容来自默认分支。
- vllm/model_executor/models/funaudiochat.py (模块 多模态; 类别 source; 类型 core-logic; 符号 `FunAudioChatProcessingInfo.feature_extractor`, `FunAudioChatProcessingInfo.speech_tokenizer`) : `FunAudioChat/Tarsier2` 的 `feature_extractor` 与 `speech_tokenizer` 分别钉住 `revision` 与 `tokenizer_revision`, 防止音频预处理文件来自其他提交。
- vllm/config/vllm.py (模块 全局配置; 类别 source; 类型 core-logic; 符号 `VllmConfig._get_quantization_config`) : `AutoGPTQ` 与 `AutoAWQ` 元数据加载的共享调用点 `VllmConfig._get_quantization_config` 补上模型 `revision`。
- vllm/transformers_utils/config.py (模块 配置工具; 类别 source; 类型 data-contract; 符号 `try_get_generation_config`) : `try_get_generation_config` 新增 `code_revision` 参数并透传给 `get_config`, 支撑 `generation-config` 回退路径的代码钉住。
- tests/test_tensorizer_revision_pin.py (模块 模型加载; 类别 test; 类型 test-coverage; 符号 `test_serialize_vllm_model_forwards_model_revision`, `fake_open_stream`) : 新增的聚焦测试: 用 `fake_open_stream` 与 `Mock hf_api` 在下载边界记录 `snapshot_download` 调用, 验证 `Tensorizer` 导出时 `revision` 被透传。
- tests/test_quantization_revision_pin.py (模块 量化; 类别 test; 类型 test-coverage; 符号 `test_quantization_config_forwards_model_revision`) : 新增聚焦测试: 验证 `AutoGPTQ` 量化配置加载时 `maybe_update_config` 收到 `model_config.revision`。
- tests/config/test_model_arch_config.py (模块 架构转换; 类别 test; 类型 test-coverage; 符号 `test_qk_rope_head_dim_recovery_uses_model_revision`) : 新增测试验证 `qk_rope_head_dim` 恢复时 `get_hf_file_to_dict` 收到 `pinned-revision`, 覆盖架构修复站点。
- tests/transformers_utils/test_config.py (模块 配置工具; 类别 test; 类型 test-coverage; 符号 `test_model_config_generation_fallback_forwards_code_revision`) : 新增测试验证 `generation-config` 回退到 `get_config` 时同时保留 `revision` 与 `code_revision`。

关键符号: `ModelArchConfigConvertorBase.init`,
`ModelArchConfigConvertorBase._get_qk_rope_head_dim`,
`ModelConfig.get_model_arch_config`, `ModelConfig.try_get_generation_config`,
`_try_get_processor_chat_template`, `resolve_chat_template`, `serialize_extra_artifacts`,
`serialize_vllm_model`, `FunAudioChatProcessingInfo.feature_extractor`,
`FunAudioChatProcessingInfo.speech_tokenizer`, `VllmConfig._get_quantization_config`,
`try_get_generation_config`

关键源码片段

vllm/transformers_utils/model_arch_config_convertor.py

安全修复的主站点之一：ModelArchConfigConvertorBase 构造器新增 revision 参数，_get_qk_rope_head_dim 在恢复被 transformers 损坏的 MLA 字段时从钉住的提交读取原始 config.json；构造器签名变更影响全部子类。

架构配置转换器：把 HF PretrainedConfig 转换为 vLLM 的 ModelArchitectureConfig。

安全修复（GHSA-3hxx-64c2-p36x）：转换器现在持有部署时选定的 revision，

当它需要从原始 config.json 恢复被 transformers 损坏的字段时，

会从被审计的提交读取，而不是从仓库的 main 分支读取。

class ModelArchConfigConvertorBase:

```
    def __init__(
        self,
        hf_config: PretrainedConfig,
        hf_text_config: PretrainedConfig,
        revision: str | None = None, # 新增参数：模型 revision 钉住
    ):
```

```
        self.hf_config = hf_config
```

```
        self.hf_text_config = hf_text_config
```

```
        self.revision = revision # 供 _get_qk_rope_head_dim 等恢复逻辑使用
```

```
def _get_qk_rope_head_dim(self) -> int:
```

```
    """获取 qk_rope_head_dim，修复 transformers v5.4+ attribute_map bug."""
```

```
    cfg = self.hf_text_config
```

```
    qk_rope_head_dim = getattr(cfg, "qk_rope_head_dim", 0)
```

```
    qk_nope_head_dim = getattr(cfg, "qk_nope_head_dim", 0)
```

```
    # 合法 MLA 配置中两个字段不应相等，相等即说明被 attribute_map 损坏
```

```
    if qk_rope_head_dim == 0 or qk_rope_head_dim != qk_nope_head_dim:
```

```
        return qk_rope_head_dim # 未损坏，直接返回
```

```
    # 从原始 config.json 找回正确值
```

```
    from vllm.transformers_utils.repo_utils import get_hf_file_to_dict
```

```
    model_path = self.hf_config.name_or_path
```

```
    if not model_path:
```

```
        return qk_rope_head_dim
```

```
    # 修复点：把 self.revision 透传给 get_hf_file_to_dict，
```

```
    # 否则这里会从默认分支读 config.json，覆盖掉钉住的架构配置
```

```
    raw = get_hf_file_to_dict("config.json", model_path, self.revision)
```

```
    if raw and "qk_rope_head_dim" in raw:
```

```
        correct = raw["qk_rope_head_dim"]
```

```
        if correct != qk_rope_head_dim:
```

```
            logger.info(
```

```
                "Fixing qk_rope_head_dim: %d -> %d "
```

```
                "(transformers v5.4+ attribute_map bug)",
```

```
                qk_rope_head_dim,
```

```
                correct,
```

```
            )
```

```

        # 就地修补配置，保证下游模型层拿到一致的值
        cfg.qk_rope_head_dim = correct
        return correct
    return qk_rope_head_dim

```

vllm/model_executor/model_loader/tensorizer.py

Tensorizer 导出辅助文件时 hf_api().snapshot_download 补上 revision，避免导出内容来自默认分支。

```

def serialize_extra_artifacts(
    tensorizer_args: TensorizerArgs,
    served_model_name: str | list[str] | None,
    revision: str | None = None, # 新增：与模型权重一致的钉住 revision
) -> None:
    """把仓库中的辅助文件一并写入 tensorizer 输出目录。"""
    if not isinstance(served_model_name, str):
        raise ValueError(
            f"served_model_name must be a str for serialize_extra_artifacts, "
            f"not {type(served_model_name)}."
        )

    with tempfile.TemporaryDirectory() as tmpdir:
        hf_api().snapshot_download(
            served_model_name,
            revision=revision, # 修复点：此前未传 revision，会从默认分支抓取
            local_dir=tmpdir,
            ignore_patterns=[
                "*.pt",
                "*.safetensors",
                "*.bin",
                "*.cache",
                "*.gitattributes",
                "*.md",
            ],
        )
        for artifact in os.scandir(tmpdir):
            if not artifact.is_file():
                continue
            with (
                open(artifact.path, "rb") as f,
                open_stream(
                    f"{tensorizer_args.tensorizer_dir}/{artifact.name}",
                    mode="wb+",
                    **tensorizer_args.stream_kwargs,
                ) as stream,
            ):
                logger.info("Writing artifact %s", artifact.name)
                stream.write(f.read())

```

tests/test_tensorizer_revision_pin.py

新增的聚焦测试：用 fake_open_stream 与 Mock hf_api 在下载边界记录 snapshot_download 调用，验证 Tensorizer 导出时 revision 被透传。

聚焦测试：验证 Tensorizer 导出时把钉住的 revision 透传给 snapshot_download。

技巧：在外部下载边界用记录桩（fake_open_stream + Mock 的 hf_api），

不依赖网络即可断言参数转发，这正是 PR body 所说的 recording stubs。

```
def test_serialize_vllm_model_forwards_model_revision(
    monkeypatch: pytest.MonkeyPatch,
):
    @contextlib.contextmanager
    def fake_open_stream(*args, **kwargs):
        yield io.BytesIO()

    snapshot_download = Mock()
    monkeypatch.setattr(tensorizer, "open_stream", fake_open_stream)
    monkeypatch.setattr(tensorizer, "TensorSerializer", Mock())
    monkeypatch.setattr(
        tensorizer,
        "hf_api",
        lambda: SimpleNamespace(snapshot_download=snapshot_download),
    )

    model = torch.nn.Module()
    tensorizer_config = SimpleNamespace(
        _construct_tensorizer_args=lambda: SimpleNamespace(
            tensorizer_uri="/tmp/model.tensors",
            tensorizer_dir="/tmp/out",
            stream_kwargs={},
        ),
        encryption_keyfile=None,
        _is_sharded=False,
        serialization_kwargs=None,
    )
    model_config = SimpleNamespace(
        served_model_name="repo/model",
        revision="pinned-revision",
    )

    tensorizer.serialize_vllm_model(model, tensorizer_config, model_config)

    # 断言下载辅助文件时使用了钉住的 revision
    assert snapshot_download.call_args.args == ("repo/model",)
    assert snapshot_download.call_args.kwargs["revision"] == "pinned-revision"
```

评论区精华

评审主要由维护者 DarkLight1337 完成。整体 review 反馈为 "The approach looks good but please remove low-value tests", 随后对新增的模型专属测试给出内联评论 "Let's remove the model-specific test"。作者在提交 38cae11571 中删除 tests/model_executor/test_funaudiochat_revision_pins.py 并说明 "the generic revision-pin tests cover the shared loader behavior", 最终 DarkLight1337 APPROVED ("Thanks")。另外 claude[bot] 提示 fork PR 自动 review 被禁用, 改由维护者人工评审。

- 删除低价值测试: FunAudioChat 模型专属测试 (testing): 作者在提交 38cae11571 中删除该测试, 理由是通用的 revision-pin 测试 (处理器 / 聊天模板与 tensorizer 等边界桩测试) 已覆盖共享 loader 行为。
- Fork PR 自动 review 被禁用 (other): 改由维护者 DarkLight1337 人工评审, 最终 APPROVED ("Thanks")。

风险与影响

- 风险:
 - 兼容性风险 (低): 所有改动本质是 "多传一个参数", revision/code_revision 为 None 时行为与修复前完全一致; 唯一结构性变更是 ModelArchConfigConvertorBase 构造器签名, 第三方自定义 convertor 子类若未同步签名可能报错。
 - tokenizer_revision 语义差异: vllm/model_executor/models/funaudiochat.py 的 speech_tokenizer 使用 model_config.tokenizer_revision 而 feature_extractor 使用 model_config.revision, 若 --tokenizer-revision 未显式设置且 ModelConfig 未回落到 revision, 两者可能来自不同提交。
 - 缓存键结构变更: vllm/renderers/hf.py 的 _PROCESSOR_CHAT_TEMPLATES 是模块级全局缓存, 键从二元组变四元组, 运行期无迁移问题; 但该全局缓存本身仍无并发保护 (非本 PR 引入)。
 - 测试覆盖局限: 新增测试在下载边界用 mock 记录桩断言参数转发, 不覆盖真实远端仓库行为; PR body 以 RED-to-GREEN 证据表补充说明每个站点的修复前后对照。
 - 安全边界: 修复只保证钉住时从钉住的提交取文件, 不承诺可变仓库或 --trust-remote-code 场景的安全性。
 - 影响: 对用户: 使用 --revision/--code-revision 的部署将获得与审计提交一致的量化层选择、tensorizer 辅助内容、音频预处理、聊天模板与架构配置, 消除 "审查内容与实际运行内容不一致" 的供应链类风险, 这是本次安全公告的核心诉求。对系统: 涉及模型加载主链路 (配置转换 -> tensorizer -> 量化 -> processor/renderer), 但每个改动点局部且默认值 None 保证未钉住时零行为变化。对团队: 作为 Patch the Planet 审计协作的落地补丁, 为后续同类审计发现提供了可复用的模式——把 revision 钉住透传到所有远端文件读取入口, 并在外部边界打桩做红绿对照测试。
 - 风险标记: 安全修复 (GHSA-3hxx-64c2-p36x), 构造器签名跨类层级变更, tokenizer_revision 与 revision 语义差异, 测试依赖 mock 桩无真实网络覆盖

关联脉络

- PR #49519 [Bugfix][Model Loader] Defer post-load attention weight processing: 同属模型加载生命周期与量化 / 后处理链路的修复: 一个修正后处理执行时机, 一个修正后处理读取的文件来源, 二者共同收敛 " 模型加载辅助步骤与主模型不一致 " 的问题。
- PR #51727 [Bugfix] Fix DeepSeek V4/3.2 tokenizer vocab size overcount crashing guided decoding: 同属 DeepSeek 系列模型配置正确性修复: 本 PR 明确以 DeepSeek 架构修复 (config.json 重读) 为例说明 revision 旁路危害, 两者都属于 " 模型配置与实际加载内容不一致 " 类问题的收敛。