多阶段漏洞链路重建 - 5 个整体错位样本修复

概述

本 PR 针对 GitHub Issue #7【2026 犀牛鸟】重建整体错位和多阶段利用样本的调用链，对 5 个存在"整体错位"或"多阶段利用"的 VulnGym 样本进行调用链重建。这些样本的共同特征是：节点能找到代码，但链路语义错位——要么入口/sink 定位在无关位置，要么单一入口点难以表达完整的多阶段利用过程。

修复目标：

- **entry-00185**：n8n ReadWriteFile 节点写入 .git 目录导致 RCE（整体错位）
- **entry-00197**：openclaw sandbox TOCTOU race condition（多阶段利用）
- **entry-00290**：openclaw 悬空符号链接沙箱逃逸（多阶段利用 + 定位争议）
- **entry-00320**：langflow 文件上传路径穿越（整体错位）
- **entry-00391**：fastmcp OpenAPIProvider SSRF + 路径穿越（整体错位）

每个 entry 的 entry_point / critical_operation / trace 均经 **GitHub raw 远程字节级匹配** + **本地 SCHEMA 校验** 双重复核（33 个 {file,line,code} 节点全部精确匹配），并补全 desc 字段。

交付物

核心交付物（Issue #7 要求）

文件	说明
`data/entries.fixed.jsonl`	修复后的 JSONL 文件，包含 5 个修正后的 entry（共 5 条）
`rebuild_report.md`	逐条说明修复前问题、修复后链路、关键判断依据
`rebuild_fix_diff.csv`	字段级修改记录（15 条，5 entry × 3 fields）
`manual_review.csv`	人工复核清单（本轮无未决项，空表头）

补充材料

文件	说明
`scripts/verify_entries.py`	本地 SCHEMA 校验脚本（474 项检查）
`scripts/verify_github_raw.py`	远程 GitHub raw 源码字节级匹配脚本（33 节点）

链路重建策略

本 PR 采用 **双重验证** + **客观化表述** 的策略：

1. **双重验证机制**
   - 本地 SCHEMA 校验：474 项检查（字段完整性、禁止字段、格式验证、节点结构、链路连续性等）

- 远程 GitHub raw 匹配：从 GitHub raw API 拉取对应 commit 的源码，进行字节级精确匹配

2. **多阶段表达**
   - 对多阶段样本，在 `desc` 和报告中明确区分前置阶段、主触发阶段和最终危险操作
   - entry-00197（TOCTOU）：显式表达 check(L104) → open(L47) 两阶段
   - entry-00290（CHECK→ESCAPE）：区分 dangling symlink fail-open 根因和最终 writeFile sink
3. **客观化表述**
   - entry-00290 使用"critical 定位争议"而非"critical 定位错误"
   - 说明两种理解均合理（检查点 vs sink），解释选择理由
   - 删除主观推断性描述（如"不含任何针对 .git 的安全过滤"）
4. **无关节点清理**
   - 删除或替换原链路中的无关日志、静态声明、返回值、闭合括号等节点
   - 确保每个 trace 节点都在真实数据流上

样本处理摘要

entry_id	project	entry_point	critical_operation	trace 节点	处理重点
entry-00185	n8n	`write.operation.ts:70`	`file-system-helper-functions.ts:128-131`	5	从静态 append 默认值改为用户可控 fileName，并定位到最终写盘 sink
entry-00197	openclaw	`server.ts:36`	`server.ts:57`	6	显式表达 check(L104)/use(L47) 两阶段 TOCTOU，删除 open 后无关节点
entry-00290	openclaw	`apply-patch.ts:94`	`apply-patch.ts:152`	5	区分 dangling symlink fail-open 根因和最终 writeFile sink，注明 pre-fix 状态
entry-00320	langflow	`files.py:162`	`local.py:120`	6	将入口上移到上传文件名捕获处，删除死分支，按调用序排列 trace
entry-00391	fastmcp	`director.py:23-28`	`director.py:216`	6	替换 logger 噪声节点，定位 path 参数拼接和 urljoin 归一化

合计：33 个独立节点全部精确匹配

验证结果

本地 SCHEMA 校验

```
# 运行本地验证脚本
E:\project_store\2_day\tencent\.venv\Scripts\python.exe
E:\project_store\2_day\tencent\submission\验证脚本\verify_entries.py
```

验证结果：

```
============================================================
汇总报告
============================================================

总检查项： 474
通过： 474
失败： 0
警告： 0

各 Entry 结果：
  entry-00185: PASS (pass=91, fail=0, warn=0)
  entry-00197: PASS (pass=98, fail=0, warn=0)
  entry-00290: PASS (pass=91, fail=0, warn=0)
  entry-00320: PASS (pass=97, fail=0, warn=0)
  entry-00391: PASS (pass=97, fail=0, warn=0)

============================================================
最终判定： ALL PASS - 所有 entry 符合 GitHub Issue #7 验收标准
```

```
总检查项：474
通过：474
失败：0
警告：0
<img width="511" height="473" alt="image" src="https://github.com/user-attachments/assets/583e0456-7728-419e-af0b-56f56efb4fbd" />

最终判定：ALL PASS - 所有 entry 符合 GitHub Issue #7 验收标准
```

检查类别分布：

- SCHEMA 字段完整性：70 项
- SCHEMA 禁止字段：60 项
- SCHEMA 格式验证：30 项
- SCHEMA 节点结构：150 项
- Issue#7 源码匹配：37 项
- Issue#7 链路完整性：25 项
- Issue#7 多阶段表达：10 项
- Issue#7 无关节点清理：5 项
- Issue#7 链路连续性：15 项
- entry_point==trace[0] && critical_operation==trace[-1]：10 项
- 其他检查：62 项

远程 GitHub raw 源码匹配

```
# 运行远程验证脚本
E:\project_store\2_day\tencent\.venv\Scripts\python.exe
E:\project_store\2_day\tencent\submission\验证脚本\verify_github_raw.py
```

验证结果：

```
33 个 {file, line, code} 节点全部精确匹配
entry-00185: PASS (5 节点)
entry-00197: PASS (6 节点)
entry-00290: PASS (5 节点)
entry-00320: PASS (6 节点)
entry-00391: PASS (6 节点)

最终判定：ALL PASS - 所有节点在 GitHub raw 上精确匹配
```

6 条验收标准达标情况

验收标准	要求	结果
1	源码匹配 + desc 准确	✅ 37 节点全部逐字匹配
2	链路完整性	✅ 所有链路完整闭环
3	多阶段利用表达	✅ entry-197(TOCTOU) + entry-290(CHECK→ESCAPE)
4	无关节点清理	✅ 已删除所有噪声节点
5	人工复核标记	✅ 所有 verify=0，manual_review.csv 为空
6	SCHEMA 约束	✅ 10 条 Invariants 全部满足
