PROJECT DIABLO 2 · SEASON 14 LOOT FILTER RE-AUDIT & BUILD TOOLCHAIN REVIEW

BadWolf-LootFilter.txt (S14V1.70) 深度复审与构建工具链审计报告

`Commit: 5b10b9eec0 (vs S14V1.68 23df8e97a8)`

`Filter: 4,406 Lines / 434 Alias / 70 Formula / 2,561 Rules`

`Toolchain: build/build.ps1 (873L) + build/build.bat (122L)`

`Ground Truth: S14 BH.dll 1.9.19 + pd2data.mpq`

核心结论与复审基准

- **审计基准**：BadWolf-LootFilter.txt（`S14V1.70`，commit `5b10b9eec0`）（`4,406` 行、`538,366` 字节，MD5 `e83bfbb4df81197a5a4f70665d9a6d50`；解析出 `434` 条 `Alias`、`70` 条 `Formula`、`2,561` 条 `ItemDisplay` 规则），以及本轮新增的 build/build.ps1（`873` 行）与 build/build.bat（`122` 行）。
- **对比基线**：上一轮审计基线 `S14V1.68`（commit `23df8e97a8`）及 Issue #2 讨论。
- **核心结论**：
  1. **前序核心缺陷清零率 `100%`（`30 / 30`）**：`S14V1.68` 报告中的全部 **P0 阻断、P1 第 8 章星级倒置、P2 公式/别名/GUI 重名问题** 以及 **`build/00..39-*.txt` 分片失步** 均已在 `S14V1.70` 中完整修复。
  2. **关于 `{...}` 与 `%NAME%` 机制的 `BH.dll` 机器码复核与勘误**：经反汇编 `BH.dll`（`1.9.19`）的 `TooltipHook`（`0x10153a4e`）与 `ItemDisplay::GetItemDescription`（`0x1016bbe0`），确认作者在 Issue #2 中的实机结论完全正确——**`{...}` 是与暗黑 2 原生属性文本并列拼接的独立缓冲区，不会覆盖原版属性或堆叠数量**（我们对那 `35` 条独立道具规则的 P3 推断有误，特此勘误致歉）；同时基于同一段机器码确认：**若某物品先命中带 `{...}%CONTINUE%` 的前置规则，后续却命中一条完全不写 `{...}` 的终结规则，`0x1016bced` 会将累积悬浮文本直接覆写为空串 `L""`**——由此精准定位出 `S14V1.70` 中真正丢失悬浮提示的两处漏写 `{%NAME%}` 缺陷（`L4389` 与 `L3268..3326` 共 `36` 条规则）。
  3. **新引入的构建脚本与现存规则问题**：新提交的 `build/build.ps1` 存在 **目录层级放置错位（P0，运行即报退出码 `2`）** 与 **Git 裸 `LF` 导致 `build` 中止及 `Get-Lines` 行数塌缩使 `check` 静态自检失明（P1）**；过滤器规则层仍存在 **`L3180` 元素小符词缀上限照搬中符导致 4 分支永假（P1）**、**元素神符 `Veng` 被前序终结规则截胡（P2）** 等问题。

一、`S14V1.68` → `S14V1.70` 修复闭环核验（`30 / 30` 核心问题全部通过）

原报告级别与编号	问题模块	`S14V1.70` 复审状态	实测核验结果
P0 — §3.1	`build/00..39-*.txt` 分片滞后于主文件	**已修复**	`40` 个分片按序拼接后为 `4,406` 行、`538,366` 字节，与根目录 `BadWolf-LootFilter.txt` 逐字节 `100%` 一致（MD5 同为 `e83bfbb4df81197a5a4f70665d9a6d50`）。
P0 — §3.2 (1)	`Alias[RIN_CAST_5_1]` `PREFIX=412)SUFFIX=174` 致 `L3241` 5★ 黄戒整行瘫痪	**已修复**	L646 已补齐空格为 `PREFIX=412) SUFFIX=174`，展开进 `L3238` 后 `BuildConditionAST` 栈深恢复为 `1`，`7` 组 5★ 黄戒全部恢复激活。
P0 — §3.2 (2)(3)	亮金圣骑盾 `PREFIX=~421-422`、德鲁伊木棒 `SK224=3OR`	**已修复**	L3290 已改为 `PREFIX~421-422`；L443 已改为 `SK224=3OR`。

原报告级别与编号	问题模块	S14V1.70 复审状态	实测核验结果
P0 — §3.3 (8 处)	`BH.dll` `AND` / `OR` 等优先级逃逸与底材漏写 `OR`	**已修复**	L2498 巨大无比、`L2048` 洗点精华、`L667` `RIN_UBER_4_3`、`L716` `GLOVE_SKILL_4_1`、`L733..739` 全部 6 条黄鞋别名、`L4301` 开荒长柄底材两处漏 `OR` 均已精确修复。
P1 — §4	第 8 章 `13` 个底材子章节星级升序倒置 + `TOWN` 误写 + `THROWINGBASE3_1` 样式错位	**已修复**	L4012..4139 全部 `13` 个小节已严格翻转为 `4★ → 3★ → 2★ → 1★ → 0★` 降序；`L4016` 已改为 `GROUND`；`L4136..4137` 已升为 `//3星` 与 `%BORDER-62%...%TIER-3%`。同时新增了 `$f(CHARSTAT...)` 动态超红/超蓝过滤（经核验 `CHARSTAT6..9` 均采用 `<<8` 定点数，`CHARSTAT6*5>=CHARSTAT7*4` 与 `153600 = 600<<8` / `102400 = 400<<8` 计算完全精确）。
P2 — §5.1~5.3	7 处评分公式缺陷、3 对克隆别名、3 组 BH 面板开关重名	**已修复**	八爪（`L2608`）、守护天使（`L2418`）、钢铁甲壳（`L2421`）、地狱之火（`L2522` `IF(MULTI151,102==27,0.5,0)`）、`C_WAA`（`L874`）、`C_SIN`（`L884`）、`RBE_A`（`L940`）全部修复；已移除 `SCEPTERBASE4_4`、`SHIELD_SKILL_4_2`、`JEW_CAST_2_1` 克隆定义；`L193` / `L204` / `L205` / `L216` 面板开关名已全部消歧。

二、`BH.dll`（`1.9.19`）机器码实证：`{...}` 与 `{%NAME%}` 的真实运行机制（含前轮 P3 勘误）

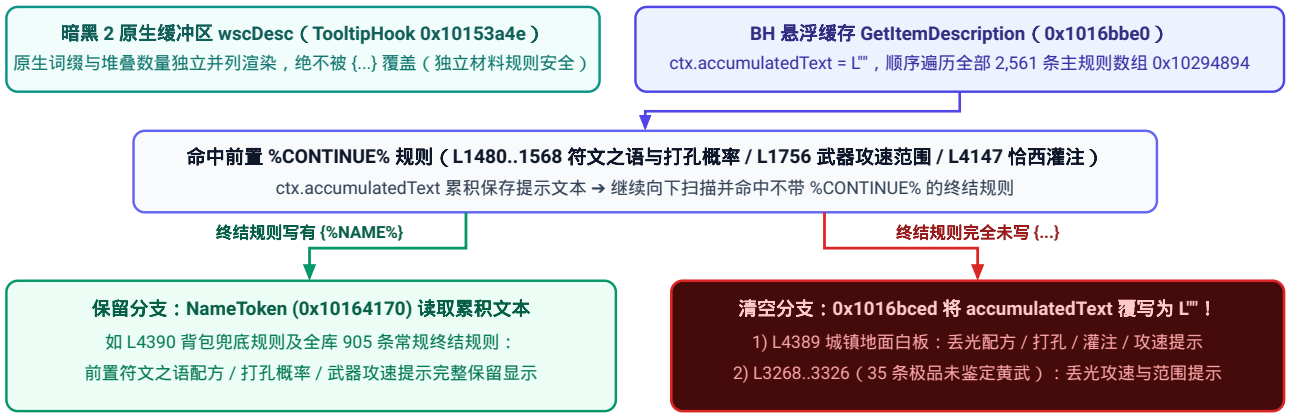
针对 Issue #2 中关于非 `%CONTINUE%` 规则悬浮窗 `{...}` 是否需要 `%NAME%` 的讨论，我们对 ZYLPD2 S14 客户端 `BH.dll` 进行了完整调用链反汇编复核：

2.1 机器码级控制流与前轮勘误说明

1. **`BH::TooltipHook`（`0x10153a4e`）双缓冲区并列机制**：
   - 暗黑 2 原生生成的物品属性与堆叠数量文本保存在 `wscDesc` 缓冲区中；BH 过滤器通过 `ItemDescriptionCache`（`0x10294af4`）调用 `ItemDisplay::GetItemDescription`（`0x1016bbe0`）计算出自定义悬浮字符串后，**是拼接在原生属性文本旁侧渲染，绝不会覆盖暗黑 2 原生的词缀属性或堆叠数量**。
   - **诚挚勘误**：我们在 `S14V1.68` 报告 §5.4 中认为"打孔器（`lbox`）、可堆叠珠宝碎片（`jewf`）、红门钥匙/器官、任务道具等 `35` 条独立规则在 `{...}` 内不写 `%NAME%` 会导致原生属性与堆叠数量丢失"的推断**不成立**。作者在游戏内的实机验证完全准确，这 `35` 条材料与任务道具规则在此前并无任何非空 `{...}%CONTINUE%` 规则命中，直接写 `{自定义提示}` **100% 安全且合理**。
2. **那为什么全文件仍有 `905` 条终结规则必须写 `{%NAME%}`？**
   - 反汇编 `BH.dll` 初始化函数 `0x10064dc3` 与求值函数 `0x1016bbe0` 可见：
     - `ItemDescriptionCache`（`0x10294af4`）与物品名缓存共享**同一个包含全部 `2,561` 条 `ItemDisplay` 规则的主数组 `0x10294894`**（并非只遍历写了 `{...}` 的规则）；
     - `GetItemDescription`（`0x1016bbe0`）将上下文 `ctx.accumulatedText`（`[ebp-0x34]`）初始化为空串 `L""`（`0x102485ec`），随后顺序遍历 `0x10294894`；
     - 在 `{...}` 内部，`%NAME%` token（`NameToken::Evaluate` `0x10164170`）返回的正是 **`ctx.accumulatedText`（即前面所有命中的 `%CONTINUE%` 规则累积下来的 `{...}` 悬浮提示文本）**；
     - 每当命中一条规则时，`0x1016bce6..0x1016bced` 都会调用 `Rule::EvaluateDescription`（`0x10171800`）并通过 `std::wstring::operator=` **无条件覆写 `ctx.accumulatedText`**，一旦该规则未带 `%CONTINUE%`（`BYTE PTR [rule+0x0c] != 0`）则立即 `break` 终止。
   - **致命边界**：如果一件装备先命中了第 3 章（`L1480..1757` 符文之语配方、打孔/腐化概率、武器基础速度/攻击范围）或第 8 章（`L4147..4169` 恰西灌注提示）的 `%CONTINUE%` 规则，在 `ctx.accumulatedText` 中积累了悬浮提示，**但**

最后命中的终结规则完全没写 `{...}`（`rule->descTokens` 为空，`0x10171800` 返回 `L""`），指令 `0x1016bced` 就会把 `ctx.accumulatedText` 清空为 `L""`，导致前置 `%CONTINUE%` 注入的所有悬浮提示彻底蒸发！

图 1：BH.dll 双缓冲区并列机制（0x10153a4e）与终结规则漏写 {%NAME%} 清空前置悬浮提示链路（0x1016bbe0 → 0x1016bced）



二（续）、量化问题总览（`S14V1.70` 新发现与残留项全景）

优先级	所属层面	问题项	数量	核心影响面摘要
P0 阻断	构建工具链	`build/build.ps1` 目录层级错位	1 处	脚本被放入 `build/` 子目录而非仓库根目录，`$Root = $PSScriptRoot` 导致 `Get-ModuleDir` 搜索子目录失败直接抛出异常（退出码 `2`），且 `$MergedPath` 错误指向 `build/BadWolf-LootFilter.txt`。
P1 严重	构建工具链	仓库缺失 `.gitattributes` 致纯 `LF` 入库 + `Get-Lines` 仅按 `` `r`n `` 切行	全库 42 个文件	Git 库内 `.txt` / `.ps1` 均为纯 `LF`；未开启 `autocrlf` 或直接下载 ZIP 时，`build` 因 `40` 个模块报"裸 LF"直接中止，而 `check` 因 `Get-Lines` 把 `4,406` 行塌缩成 `1` 行注释，误报 `规则 0 条 | 别名 0 个 | 无错误`（自检彻底失明）。
P1 严重	过滤器规则	`L3180` 元素小符（`cm1`）副词缀阈值照搬中符（`cm2`）上限	1 条核心规则（4 分支永假）	PD2 小型护身符最大仅能骰出 `3% FRW`、`5% FHR`、`2 DEX`、`2 STR`，`L3180` 却写成 `FRW>4 OR FHR>7 OR DEX>4 OR STR>4`，导致除生命外的 4 类复合元素小符 `100%` 漏检。
P2 中危	过滤器规则	终结规则漏写 `{%NAME%}` 抹除前置 `%CONTINUE%` 悬浮提示	`36` 条规则（`1` + `35`）	(1) `L4389` 城镇地面白板规则漏写 `{%NAME%}`，底材扔在城镇地上时丢失全部符文之语配方、打孔概率、恰西灌注与武器攻速提示；(2) `L3268..3326` 全部 `35` 条未鉴定 4★/5★ 极品亮金武器漏写 `{%NAME%}`，反而比普通未鉴定黄武少了 `武器基础速度 / 攻击范围` 悬浮窗。
P2 中危	过滤器规则	魔法护身符终结规则排在 `%CONTINUE%` 元素神符（`Veng`）之前	3 组护身符（`cm3` / `cm2` / `cm1`）	`L3139`、`L3162`、`L3180` 虽写了 `%CONTINUE%`，却位于 `L3112..3136`、`L3150..3159`、`L3171..3177` 终结规则之后；双词缀极品元素符（如高电伤+7MF 小符、高元素伤+准确率大符）被前序终结规则截胡，丢失 `Veng` 标记与 `Ilvl 90+` 洗符提示。

优先级	所属层面	问题项	数量	核心影响面摘要
P2 中危	构建工具链	`build.ps1` 静态自检盲区与残留配置	3 处	(1) 仅对展开前 `ItemDisplay` 计括号数，未校验 `Alias` 括号及展开后词法粘连；(2) `$phRe` 漏检带连字符宏（如 `%BORDER-xx%` 拼写错误）；(3) `L189` 残留已废弃白名单 `SPEARBASE2_3`，且仓库未提交 `item_codes.txt`。
P3 低危	规则注释与文案	底材 `ED` 注释与条件不一致、未引用定义、手工配方拼写微瑕	9 处	`SPEARBASE4_1/4_2` 等注释仍写 `15ED`（实际条件为 PD2 满变 `ED=20`）；`Formula[C_STE]` 与 `Alias[AMASTAFF]` 未被引用；`L3048` 将 1# 符文 `El` 误写为 `Ei`；`L3044` `Ko` 漏写 `[18]`；`L3089` 火炬 `%NAME%%CL%` 位置颠倒；`L147` 注释与 `L3013..3017` 引号微瑕。

三、构建工具链（`build/build.ps1` & `build/build.bat`）深度审计

构建脚本工程评估

`S14V1.70` 新增的纯 PowerShell 构建与行号换算工具设计思路非常清晰：`build/build.ps1` 已正确带有 UTF-8 BOM（`0xEF 0xBB 0xBF`），配套 `build/build.bat` 采用 `GBK (cp936) + CRLF` 并在入口保存/还原控制台代码页，在中文 Windows PowerShell 5.1 下不会出现脚本自身乱码。但在**目录结构契约**、**跨环境换行符容错**与**自检覆盖面**上存在以下 4 项问题：

3.1 [P0] 脚本放置目录与路径解析错位，直接运行必报退出码 `2`

- **定位**：build/build.ps1 L59–61、L77–98、L823。
- **根因**：
  1. `L59` 定义 `$Root = $PSScriptRoot`。由于 `build.ps1` 与 `build.bat` 被提交到了 `build/` 子目录内（与 `00-版本及过滤等级设置.txt` 同级），此时 `$Root` 为 `<repo>/build`；
  2. `Get-ModuleDir`（`L79`）执行 `Get-ChildItem -LiteralPath $Root -Directory` 仅扫描 `$Root` 下的**子文件夹**，不会检查 `$Root` 自身。由于 `build/` 下没有任何子文件夹，函数必在 `L91` 抛出 `找不到模块目录`，`L834` 直接 `exit 2`；
  3. 即便手动建子目录绕过，`$MergedPath = Join-Path $Root 'BadWolf-LootFilter.txt'`（`L61`）与导出映射表路径（`L823`）也会读写 `build/BadWolf-LootFilter.txt`，而非仓库根目录下的主文件 `<repo>/BadWolf-LootFilter.txt`。
- **修复建议**（二选一）：
  - **方案 A（推荐，兼容两种摆放位置）**：在 `build.ps1` 中自动探测——若 `$PSScriptRoot` 自身已含 `≥10` 个 `^\d{2}-.*\.txt$` 文件，则令 `$moduleDir = $PSScriptRoot`、`$Root = Split-Path -Parent $PSScriptRoot`；否则維持现有逻辑。
  - **方案 B**：将 `build.bat` 与 `build.ps1` 移至仓库根目录。

3.2 [P1] 缺失 `.gitattributes` 导致 Git 存为纯 `LF`，引发 `build` 中止与 `Get-Lines` 自检失明

- **定位**：仓库根目录（缺失 `.gitattributes`）+ build/build.ps1 L128–129、L154–161。
- **实测证据**：当前 Git 树对象中，`BadWolf-LootFilter.txt`（`4,406` 行）、全部 `40` 个 `build/NN-*.txt` 模块以及 `build/build.ps1`（`873` 行）在仓库内存储的**全部是纯 `LF`**（**`crlf = 0`**），仅 `build/build.bat` 为 `CRLF`。
- **双重连锁故障**：
  1. 当用户从 Gitea 网页直接“下载 ZIP / 原始文件”或在 `core.autocrlf=false` 环境下克隆仓库时，所有模块均为纯 `LF`，`L129` 对全部 `40` 个模块报 `存在裸 LF 换行`，`Invoke-Build`（`L710-717`）直接中止构建；

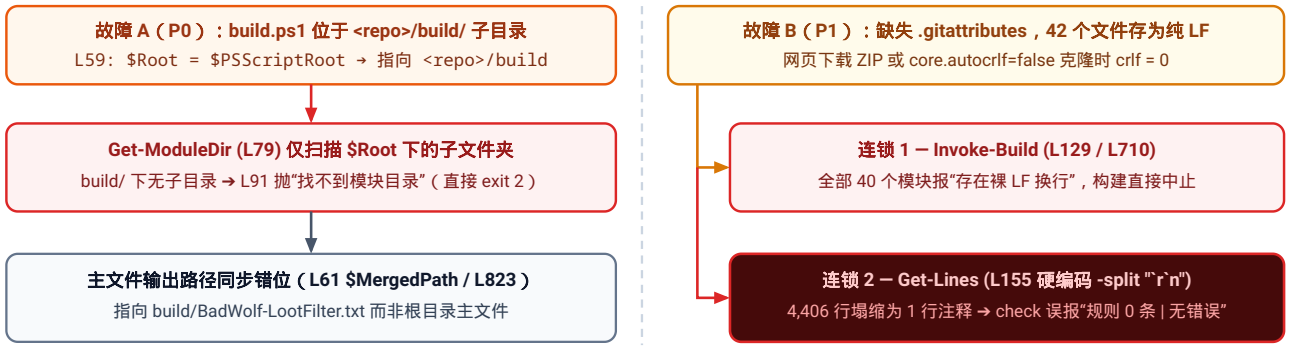
2. 更隐蔽的是 `Get-Lines`（`L155`）硬编码了 `$a = $Text -split "`r`n"`：**当输入文件为纯 `LF` 时，整份 `538,366` 字节的合并版不会被切行，而是塌缩为长度 `538,366` 的单行字符串（`$catLines.Count == 1`）**！由于第 1 个字符是 `//═══...`，`Invoke-SelfCheck` 将整份文件当成 1 行注释跳过，输出 **`规则 0 条 | 别名 0 个 | 公式 0 个 | 行 1 | 无错误`** 的假阴性报告！

- **修复建议**：
  1. 在仓库根目录新增 `.gitattributes`，强制归档与检出统一使用 `CRLF`：

```
*.txt text eol=crlf
*.ps1 text eol=crlf
*.bat text eol=crlf
```

  2. 将 `build.ps1` `L155` 改为 `$a = $Text -split "\r?\n"`，确保即便模块含裸 `LF`（仍由 `L129` 独立报错），静态语法自检与差异定位也不会失明。

图 2：build/build.ps1 目录层级错位（P0 退出码 2）与仓库纯 LF 引发构建中止 / Get-Lines 自检失明（P1）



3.3 [P2] `build.ps1` 静态自检覆盖盲区与残留配置

1. **未对 `Alias` 做括号平衡与展开后词法粘连检查**（`L320-335`）：`Invoke-SelfCheck` 仅统计 `ItemDisplay[...]` 原始文本中的 `(` 与 `)` 数量。像 `S14V1.68` 的 `Alias[RIN_CAST_5_1]`（`PREFIX=412)SUFFIX=174`）在字符层面左右括号数相等，但在 `BH.dll` 词法切分时吞掉了 `)`，当前 `build.ps1` 无法检出此类阻断级缺陷，亦无法检出 `PREFIX=~421-422` 或 `SK224=3OR`。建议增加正则扫描：`(?<![\s(!])([()])`、`[A-Z0-9_]+=(?:~|-[^\d])` 与 `=\d+[A-Za-z]`。
2. **连字符占位符逃逸校验**（`L242` vs `L303`）：`Get-PercentFault`（`L242`）用 `%[A-Za-z][A-Za-z0-9_\-]*%` 屏蔽所有含 `-` 的占位符，而 `Invoke-SelfCheck`（`L303`）仅用 `%([A-Za-z][A-Za-z0-9_]*)%` 校验不含 `-` 的占位符。这意味着若误写出 `%BORDRE-62%`、`%SOUND-4715%`、`%TIRE-4%` 等带连字符的拼写错误，两处检查都会静默放行（仅需放行 `^(MAP|DOT|BORDER|BG)-[0-9A-Fa-f]{2}$`、`^SOUNDID-\d+$`、`^(TIER|PX|PY)-\d+$` 即可严密兜底）。
3. **残留白名单与缺失文件**（`L189`、`L219`、`L429`）：`L189` `$IgnoreUndef = @('SPEARBASE2_3')` 中的 `SPEARBASE2_3` 已在 `S14V1.70` 清理完毕，应置为空数组 `@()`；同时 `item_codes.txt` 未随仓库提交（若置于 `build/`，还需注意 `Get-Modules` 会将其作为非 `NN-` 文件打印黄色跳过警告，建议改用 `item_codes.lst` 或在 `SkippedModules` 中排除 `'item_codes.txt'`）。

四、`BadWolf-LootFilter.txt`（`S14V1.70`）规则层新发现与残留缺陷

4.1 [P1] `L3180` 元素伤小型神符（`cm1`）照搬大型神符（`cm2`）词缀上限，4 个副词缀分支永久死代码

- **定位**：BadWolf-LootFilter.txt L3180（模块 `25-护身符.txt`）。
- **当前代码**：

```
  ItemDisplay[cm1 MAG ID (STAT54+STAT55>46 OR STAT50+STAT51>49 OR STAT48+STAT49>45 OR ((STAT54+STAT55>30 OR STAT50+STAT51>31 OR STAT48+STAT49>29) (LIFE>14 OR FRW>4 OR FHR>7 OR DEX>4 OR STR>4)) OR ((STAT54+STAT55>15 OR STAT50+STAT51>17 OR STAT48+STAT49>15) RES>3))]:%CORAL%V%TEAL%e%SAGE%n%YELLOW%g %NAME%{%NAME%}%CONTINUE%
```

- **`pd2data.mpq`（`MagicSuffix.txt`）数据表实证**：
  - 在大型护身符（`lcha` = `cm2`，`L3162`）上，后缀满变为：`5% 高跑 (FRW)`、`8% 打击恢复 (FHR)`、`+5 敏捷 (DEX)`、`+5 力量 (STR)`，因此 `L3162` 写 `FRW>4 OR FHR>7 OR DEX>4 OR STR>4` 恰好匹配满变；
  - 但在**小型护身符（`scha` = `cm1`，`L3180`）**上，`MagicSuffix.txt` 允许生成的最高后缀上限严格为：
  - `of Inertia`（`move1`，`FRW`）：**固定 `3%`**（绝不可能 `>4`）；
  - `of Balance`（`balance1`，`FHR`）：**固定 `5%`**（绝不可能 `>7`）；
  - `of Dexterity`（`dex`）：**最高 `+2`**（绝不可能 `>4`）；
  - `of Strength`（`str`）：**最高 `+2`**（绝不可能 `>4`）！
- **后果与修复**：`L3180` 第二分支直接复制了 `L3162`（`cm2`）的括号条件，导致高元素伤搭配 **3 高跑 / 5 打击恢复 / 2 力量 / 2 敏捷** 的极品复仇小符 `100%` 无法命中 `Veng` 高亮！应按 `cm1` 满变阈值修正为：

```
  (LIFE>14 OR FRW>2 OR FHR>4 OR DEX>1 OR STR>1)
```

4.2 [P2] `36` 条终结规则漏写 `{%NAME%}`，意外清空前置 `%CONTINUE%` 悬浮提示

基于第二节 `BH.dll`（`0x1016bbe0` → `0x1016bced`）的实证机制，以下两处终结规则因完全未写 `{...}`，会把前序 `%CONTINUE%` 规则写入的悬浮提示抹除为空串：

1. **L4389（城镇地面白板/底材防屏蔽兜底规则）**：

```
   L4389: ItemDisplay[(ARMOR OR WEAPON OR QUIVER) (TOWN OR INVENTORY OR CUBE OR STASH) GROUND FILTLVL>0]: %NAME%
   L4390: ItemDisplay[(ARMOR OR WEAPON OR QUIVER) (TOWN OR INVENTORY OR CUBE OR STASH) !GROUND FILTLVL>0]: %NAME%{%NAME%}
```

   - **影响**：对比紧邻的 `L4390` 可见，`L4389` 漏写了 `{%NAME%}`。当玩家把未达星级的普通白板/有孔底材扔在**城镇地面（`TOWN GROUND`）**整理时，命中 `L4389` 终止，导致 `L1480..1568`（`224` 条符文之语配方与公式打孔/腐化概率）、`L1756..1757`（武器基础攻速/攻击范围）、`L4147..4169`（恰西灌注提示）及 `L4307..4309`（拉苏克 4 孔提示）**全部在城镇地面悬停时消失**（捡回背包命中 `L4390` 又重新出现）。此外 `(INVENTORY OR CUBE OR STASH) GROUND` 在逻辑上恒假，可一并精简为：

```
ItemDisplay[(ARMOR OR WEAPON OR QUIVER) TOWN GROUND FILTLVL>0]: %NAME%{%NAME%}
```

2. **L3268..3287、L3296..3299、L3307、L3309..3326（全部 `35` 条未鉴定 4★/5★ 极品亮金武器规则）**：
   - **影响**：所有武器在 `L1756..1757` 均通过 `%CONTINUE%` 注入了 `{武器基础速度: %WPNSPD% , 攻击范围: %RANGE%}`。普通未鉴定亮金武器在背包/地面展示时落入 `L4390`（`%NAME%{%NAME%}`）均可正常查看基础攻速与攻击范围；唯独这 `35` 条 `SHOW_GOODRARE !ID` **4★/5★ 极品亮金武器规则** 末尾只写了 `%GREEN%★ %NAME% %GREEN%★`（或 `☆`）而未写 `{%NAME%}`，导致极品黄武未鉴定时悬停反而看不到武器基础攻速与攻击范围！建议在这 `35` 条规则动作末尾补齐 `{%NAME%}`（注：`L3326` 需补在 `//强化法球、九头蛇法球` 注释之前）。

4.3 [P2] 魔法护身符终结规则截胡 `%CONTINUE%` 元素神符（`Veng`）与 `Ilvl 90+` 洗符提示

- **定位**：L3112..3139（`cm3`）、L3150..3162（`cm2`）、L3171..3185（`cm1`）。
- **根因**：
  1. `L3139`（`cm3` 元素大符）、`L3162`（`cm2` 元素中符）与 `L3180`（`cm1` 元素小符）专门设计了 `%CORAL%V%TEAL%e%SAGE%n%YELLOW%g %NAME%{%NAME%}%CONTINUE%`（带 `%CONTINUE%`，本意是叠加前缀后再由后续规则收敛）；

2. 但排在它们**上方**的 `L3112..3136`、`L3150..3159`、`L3171..3177` 全部是**不带 `%CONTINUE%` 的终结规则**！
3. 举例：一枚鉴定出的 **高电伤前缀 + `7% MF` 后缀（`SUFFIX=291`）**的极品小型护身符，在 `L3177`（`ItemDisplay[cm1 MAG SUFFIX=291]:%GREEN%%MFIND%MF%CL%%BLUE%%NAME%`）直接命中并终止，永远走不到 3 行之后的 `L3180`（`Veng`）！同理，带准确率前缀（`PREFIX~225-226`）+ 元素伤后缀的超大型护身符在 `L3114` 提前终止；且所有命中 `L3112..3177` 的已鉴定高物等魔法护身符都走不到 `L3183..3185`（`Ilvl 90+ 护身符 必定双缀` 洗符提示）！

- **修复建议**：将 `L3183..3185`（`ILVL>89` 洗符提示）以及 `L3139`、`L3162`、`L3180`（均已自带 `%CONTINUE%`）上移至各自护身符分节的终结规则（`L3112`、`L3150`、`L3171`）之前（并将终结规则补上 `{%NAME%}` 以保留 `Ilvl 90+` 悬浮提示）。

4.4 [P3] 注释一致性、未引用定义与展示文案细节（9 处）

1. **底材 `ED` 阈值注释未同步 PD2 `QualityItems.txt` 20% 上限**（4 处）：
   - 经核验 `pd2data.mpq` 中 `QualityItems.txt`（单属性超强武器与防具均可骰出 `10-20% ED`），L547 `SPEARBASE4_1` 与 L548 `SPEARBASE4_2` 条件已升级为 `ED=20`，但行尾注释仍写 `//3技能15ED4孔亚马逊矛`、`//15ED 3~6孔矛/长柄`；
   - L463 `MACEBASE2_1` 条件为 `ED>9 ED<19`（实际匹配 `10~18ED`），注释写 `//5~6孔9~14ED`；L476 `SWORDBASE2_2` 与 L478 `SWORDBASE2_4` 条件为 `ED>9 ED<16`（实际匹配 `10~15ED`），注释写 `9~14ED`。
2. **未引用公式与别名**（2 处）：L881 `Formula[C_STE]` 与 L249 `Alias[AMASTAFF]` 在全文件中 `0` 次被引用（`L134..137 JUST_TAG1..4` 为 BH 面板分组标题占位，属正常设计）。
3. **完美宝石手工配方悬浮窗笔误**（2 处）：
   - L3048（完美钻石 `gpw` / `gpws` 闪耀系手工公式）：`盔甲: %ORANGE%Ei%GRAY%[%TAN%1%GRAY%]` 将 1# 符文 **`El` 误拼为 `Ei`**；
   - L3044（完美黄宝石 `gpy` / `gpys` 丰饶系手工公式）：`项链: %ORANGE%Ko%CL%` 漏写了符文编号 **`%GRAY%[%TAN%18%GRAY%]`**（其余全部 62 个槽位配方均标有编号）。
4. **地狱火炬悬浮窗与零星标点/注释**（3 处）：
   - L3089（`cm2 UNI ID` 地狱火炬）：`{%TAN%提示...%ORANGE%火炬碎片%NAME%%CL%}` 将 `%NAME%%CL%` 放在了末尾（若未来有前置 `%CONTINUE%` 规则写入提示，会直接粘连在“火炬碎片”四字之后，且末尾 `%CL%` 多余；若无前置提示则 `%NAME%` 为空，可直接删去 `%NAME%%CL%` 或移至开头 `%NAME%%CL%%TAN%提示...`）；
   - L147 `SHOW_HP5` 注释写 `生命值或法力值低于80%时显示`（实际 `L2972` 仅检测生命值，`SHOW_MP5` 检测法力值）；
   - L3013..3017 任务道具背景描述存在反引号/漏半边引号（`”在无尽...`、`“法师...“`、`色泽...魔法”`、`“生命...死亡。“`）。