

System.ini 陷阱檢查工具 check_systemini

獨立 exe · 掃描 0x1A 與「值含 [] 假區段」兩大 hb_iniRead 陷阱
2026-07-23 HBSTK/build/check_systemini.py

一、訴求與現象

承接「PrnAsk 列印字型讀取懸案」:System.ini 有兩個會害 hb_iniRead 讀錯的陷阱——陷阱A 檔案混入 0x1A(Ctrl-Z/DOS EOF)截斷其後內容;陷阱B ‘PNTGO_CMD’ 等鍵的「值」含 ‘[file.prn]’/‘[printer]’, 被誤判成 ‘[區段]’ 標頭, 害排在後面的 MAIN 鍵(GDI_FONT_ADJ 等)掉進假 section 讀不到。

讀取端雖已用 IniReadSafe 免疫,但現場仍需要一支工具能快速掃各站 System.ini、指出哪一台、哪一行有問題、哪些字型鍵會被舊版讀錯,方便派工前後比對與驗證。需求是獨立 exe(免裝 Python)、雙擊即用、開頭說明清楚。

二、程式做什麼 / 選項

執行方式	行為
雙擊(不帶參數)	掃 W:\S*\System.ini(所有站台)
拖曳一個 System.ini	檢查該單一檔
帶資料夾參數	遞迴掃該資料夾下所有 System.ini(例 C:\STK)
帶單一檔參數	檢查指定檔(例 W:\S004\System.ini)

每檔報告:是否有 0x1A(第幾行)、值含 [] 的危險行、以及模擬 hb_iniRead 的解析判定哪些 MAIN 鍵會掉進假 section。結果同時存成同目錄 ‘check_systemini_report.txt’ (BIG5, 記事本可讀)。

三、踩坑與陷阱

坑	症狀	解法
主控台中文亂碼	雙擊時主控台字碼頁非 950;先試 chcp 950 / sys.stdout.reconfigure(cp950) 都無效(PyInstaller frozen stdout 編碼與主控台對不上,或設 CP 於 Python 啟動後不改既有 stdout 編碼)	★改用 ctypes WriteConsoleW 直接寫 Unicode 字元(GetStdHandle(-11)),完全繞過字碼頁→中文必對;被導向檔案時退回 cp950 bytes

坑	症狀	解法
「INIPATH=2」誤判	檢查 exe 內是否殘留診斷 log 時, 字串比對誤中符號表 CDEFINIPATH	以完整 log 字串(INIPATH1: [])比對才準, 符號表片段是假陽性
報告檔路徑	frozen 後 __file__ 指向暫存解壓目錄	_base_dir():frozen 用 sys.executable 目錄, 否則 __file__ 目錄
報告中文	主控台好了但存檔要記事本讀	報告一律 encode(big5, replace) 寫檔

四、解決方法(核心邏輯)

check_one() 讀 raw bytes, 分三段:

1. 掃 0x1A 行號(陷阱A)。
2. 掃「非註解、= 之後的值含 [」的行(陷阱B 的危險行)。
3. 模擬 hb_iniRead 的 bug:逐行 're.search(rb"\[.+\?\\]", s)' ——任何含 [...] 的非註解行都當成開一個 section, 追蹤每個 MAIN 鍵最後落在 MAIN 還是假 section, 列出「被偷走」的鍵。

主控台輸出走 cprint():

```
h = kernel32.GetStdHandle(-11)
kernel32.WriteConsoleW(h, c_wchar_p(s), len(s), byref(wrote), None)
# ■■■(■■■■)■ sys.stdout.buffer.write(s.encode("cp950", "replace"))
```

五、使用技術與部署

技術	用途
ctypes WriteConsoleW	Windows 主控台直寫 Unicode, 免字碼頁亂碼(本案關鍵)
re.search(rb"[.+]") 模擬	重現 hb_iniRead 誤判邏輯, 精準指出受害鍵
PyInstaller onefile 32-bit	Python312-32 打包獨立 exe(6.4MB), 客戶端免裝 Python
sys.frozen / sys.executable	frozen 時正確定位報告輸出目錄
uploader TOOLS 群組	exe 放 toAWI_Home\tools\ 隨 INI_Editor 派送各站 W:\TOOLS

心法:Windows

主控台要顯示中文, 別跟字碼頁(chcp/reconfigure)糾纏——直接

WriteConsoleW 寫寬字元最穩。診斷「解析陷阱」的最佳工具,就是用同一套 bug 邏輯寫個模擬器把問題檔餵進去,讓假 section 現形。