

錯誤回報 FTP 上傳加速

連線重用 + 掃描解耦 + ERR_SCAN_SEC 可調 專案:StkAgent 日期:2026-07-21
編譯:hbm2 stk_agent.hbp -comp=bcc(BUILD OK)

一、訴求與現象

- 客戶回報「錯誤訊息上傳 FTP 太慢」。錯誤畫面(DefError / 手動 Alt+C)會把 trigger + txt + bmp 寫到 error_pending 目錄,再由 StkAgent 掃描並上傳到 FTP(/H/stk-error/{KMSN}/{站})供監控端接收。實測從錯誤發生到上傳完成耗時很久。

二、踩坑與陷阱

- 兩個獨立根因疊加:(1) 偵測延遲——ProcessErrorPending 原本綁在 600 秒的網路維護區塊,錯誤最久要等 10 分鐘才開始上傳;(2) 上傳延遲——FtpUploadFile 每個檔案都重開一次完整連線(connect / login / PASV / STOR / QUIT),一筆錯誤含 3-4 個檔就要 3-4 次握手加登入。
- shared\ftp_helper.prg 原檔是 LF 行尾(非 CRLF),新增函式段誤用 CRLF → 因原檔無 CRLF,全檔統一 replace 為 LF;而 stk_agent.prg 反而是 CRLF。
- 10 秒太頻繁的顧慮:網路站(W:\)每 10 秒一次 SMB 目錄查詢會讓連線一直保持活躍,抵觸原作者對 Win10 第二條 SMB 連線與 20 連線上限的設計(600 秒維護 + idle 斷線)。
- INI_EDITOR 同步鐵律:新增 ini key 必須同步 ini_editor.py 的 INI_META、ini_template.json(實際 4 份,且位於 stk_agent.ini 群組而非頂層)、以及 stk_agent.ini 範本 2 份。

三、解決方法

- 偵測解耦:抽出獨立 nLastErrScan,每 s_nErrScanSec 秒掃一次 error_pending(掃描僅一次 Directory() 呼叫,無 trigger 檔時零成本),與 600 秒網路維護完全分離。
- 連線重用:新增 FtpUploadFiles(aLocalFiles),一次登入、一次 CWD,控制連線重用逐檔只做 FtpStorOne(PASV + STOR),最後才 QUIT;ProcessErrorPending 與 ERROR_REPORT 兩處都改用。

- 間隔可調:掃描間隔預設 30 秒(錯誤仍在數十秒內上傳,遠優於原 10 分鐘),並新增 ERR_SCAN_SEC ini key(<=0 保底 30),各站可自行調整免重編;同步更新 INI_EDITOR(ini_editor.py + 4 份 JSON + 2 份 stk_agent.ini 範本)。

四、使用技術與用法

- Harbour hb_socket* 手寫 FTP:PASV 被動模式,資料連線每檔獨立(FTP 協定要求),但控制連線(已登入)可重用是本次加速關鍵。
- 節流模式:主迴圈每輪(relay recv timeout 2 秒)呼叫,內部以 $\text{Abs}(\text{Seconds}() - \text{nLast}) > \text{間隔}$ 判斷是否執行;Abs 防止跨午夜 Seconds() 歸零造成 delta 變負而永不觸發。
- ini 可調:stk_agent.ini 加 ERR_SCAN_SEC=30;agent LoadAgentConfig 讀 hb_DirBase()stk_agent.ini,解析器會 skip [section] 純比對 key 名,故 section 只是外觀分組。

修改檔案清單

檔案	修改內容
SOCKET\shared\ftp_helper.prg	新增 FtpUploadFiles()/FtpStorOne() 控制連線重用批次上傳
SOCKET\StkAgent\source\stk_agent.prg	error_pending 掃描解耦每 s_nErrScanSec 秒(預設30);新增 ERR_SCAN_SEC ini(<=0保底30);兩處改用 FtpUploadFiles
HBSTK\build\ini_editor.py	INI_META [FTP] 群組新增 ERR_SCAN_SEC(預設30)
ini_template.json × 4	各檔 stk_agent.ini 群組新增 ERR_SCAN_SEC 區塊(default 30)
stk_agent.ini 範本 × 2	[FTP] 段新增 ERR_SCAN_SEC=30