

Relay 連線取代 Race 修復(agent 同時死、逐台重啟才復活)

專案:SOCKET/Relay 日期:2026-06-14 檔案:relay_server.py

一、訴求與現象

客戶的 WebClient 與本機測試 Commander/WebScreen 連線異常:Commander 顯示 agent 全部失聯、WebScreen

連不上。關鍵特徵:(1)客戶機與測試機「同時」死亡;(2)重啟 RELAY_SERVER 無效,反而是逐台「移除+重啟 AGENT」才一台一台慢慢恢復;(3)agent log 是「connected OK 但 90

秒靜默→重連」鬼打牆。此問題長期反覆發生、修過很多次。

二、診斷過程(被使用者的反證逐步收斂)

假設	被推翻的依據
relay 半死殭屍	使用者實測:重啟 relay 無效、重啟 agent 才有效
各機 I: 網路磁碟卡住	客戶+測試機「同時」死,兩台 I: 不可能同時壞→觸發點是共用的東西(relay)
relay 持鎖 blocking send 凍結全域	「逐台各自重啟才復活、一個個慢慢好」推翻全域凍結(否則修一隻全復活)→ 問題是 per-agent
★ 連線取代 race(正解)	唯一能同時解釋:同時死(共同觸發)+per-agent(各自卡)+重啟 relay 無效+重啟 agent 有效+反覆復發

三、根因(connection-replacement race)

agent 連線 client_key = KMSN:STATION。relay 收到同 key 新連線會「Replaced old connection」關掉舊 socket。而 remove_client(client_key) 在連線結束的 finally 裡無條件 del clients[key], 不檢查要刪的 socket 是不是自己這條。

Race 步驟: agent 舊連線 C1 半開→agent 90s 判死、關 C1、重連 C2; relay 的 C2 執行緒先註冊 clients[K]=C2(並關掉 C1); relay 的 C1 舊執行緒這時才發現 C1 被關→進 finally→remove_client(K)→但

clients[K] 現在是 C2→把剛連好的 C2 關掉並刪除; agent 的 C2 又死→再重連→重複。每隻 agent 各自鬼打牆, 只有乾淨重啟 process(舊執行緒排乾)才打破。大規模同時重連(relay 重啟/網路抖動)時所有 agent 一起中→逐台重啟才一個個復活。

四、解決方法

讓 remove_client 只刪「還是自己這條 socket」的連線, 被較新連線取代過就不要碰:

	作法
remove_client	加 my_sock 參數;只在 clients[key]['sock'] is my_sock(或未傳)時才 close+del
4 個呼叫端	finally / cleanup_thread(stale) / forward_to_agents(failed) / forward_to_commanders(failed) 都把自己的 socket 傳進去
驗證	py_compile 通過;PyInstaller 重 build relay_server.exe;判準=relay log「Registered 後不再緊接著 Removed disconnected」

五、使用技術與用法

技術	用法
身分比對而非按 key	用 sock 物件身分(is)判斷, 避免「key 相同但已是新連線」被誤刪—分散式連線管理的通則
鎖內收集、鎖外 I/O	forward_to_commanders 本來就是對的範本(鎖內收集 socket、鎖外 send);forward_to_agents 仍鎖內 send 為後續可再優化
診斷心法	信使用者的實證反證(同時死/逐台復活)逐一淘汰理論, 比硬套單一假設可靠
部署	build 出 relay_server.exe→上傳 SERVER→乾淨重啟(kill 全部→copy 到 A/B→start_relay.bat)

結論:長年反覆的「agent 莫名失效」根因是 remove_client 的連線取代 race;以 socket 身分比對根治, 不再「全部同時死、逐台重啟才復活」。