

cs-console

Manual

Ephemeral, per-request interactive PTY relay with an OS password gate for System > Console and the AI helpdesk

Version 0.5.8 · October 13, 2026

Contents

1. Overview	3
2. Supported platforms	3
3. Password gate	3
4. Expect mode	3
5. Behavior and options	4
6. Use in napp-it CS	4
7. License	4

1. Overview

cs-console is the ephemeral, per-request interactive PTY relay of napp-it 4ai (client-server). It is started by `server.pl` on demand for **System > Console** and for the console mode of the AI helpdesk, opens exactly one direct connection, relays an interactive shell and then exits.

Safety by design:

- it always requires a root / Administrator parent process;
- in the interactive shell mode it additionally requires a **password confirmation through the console itself** (the password gate, chapter 3);
- the connection is restricted by IP and token and sealed with ChaCha20-Poly1305.

The design follows csweb-gui/data/howto.ai/cs-console.info. csweb-gui deploys and updates the binary per member with the menu **About > Download cs-tools**.

This is a release candidate. v0.5.5 is on GitHub with pre-built binaries for 8 targets (windows, linux, darwin, freebsd, illumos, solaris amd64; linux and darwin also arm64). The interactive console (open, OS password gate, shell) was live-verified end to end on all six amd64 platforms: Windows, Linux (Proxmox), illumos (OmniOS), macOS, FreeBSD and Solaris.

2. Supported platforms

Platform	Architecture	PTY backend	Status
Linux	amd64, arm64	creack/pty	live-verified
Windows	amd64	ConPTY	live-verified
macOS	amd64, arm64	creack/pty	live-verified
FreeBSD	amd64	creack/pty	live-verified
illumos	amd64	hand-rolled STREAMS (cgo)	live-verified
Solaris	amd64	creack/pty	live-verified

The transport is POLL (not SSE) through the Perl web server, with a dedicated rate-limit-free authentication for the `/console/*` endpoints.

3. Password gate

Before the PTY starts, the OS itself confirms the root / Administrator password: PAM on Linux, macOS, FreeBSD, illumos and Solaris, `LogonUser` on Windows. There are up to 3 attempts and a per-IP lockout of 15 s (lockout.go). The password - or any OTP that PAM asks for - never reaches the AI model, the chat transcript or the logs. The password gate relays the actual PAM/LogonUser error text (module and error names only, never secrets) instead of a bare "authentication failed".

Parallel sessions from the same frontend IP share one attempt budget: they no longer get an independent 3-attempt budget each (the concurrency fix of v0.5.7, covered by lockout_concurrency_test.go, `-race clean`).

4. Expect mode

Besides the interactive shell there are allowlisted expect-mode actions. They drive an interactive program with prepared prompts, so a secret is typed once into the program and not passed on a command line.

- `passwd_user`, `smbpasswd_user`, `ksmbd_user` - set a password of a user (root / Administrator, uid 0, is always refused).

- `zfs_create_enc_prompt` and `zfs_unlock_enc_prompt` (v0.5.8) - create or unlock a `keylocation=prompt` encrypted dataset by typing the passphrase interactively. This is needed because Solaris 11 and Windows (OpenZFS on Windows) both reject non-interactive stdin for prompt-mode ZFS key entry - an interactive prompt is the only way to drive that flow on these two platforms. The file-based create of napp-it and keysplit are unchanged; they need real key bytes on disk, which a prompt cannot provide. All four prompt strings (Solaris create/unlock, Windows create/unlock) were captured and verified live before shipping.

Robustness fixes that apply to every expect action: the ConPTY line ending on Windows (a bare `\n` never submitted a line there; now `\r\n` through a GOOS-aware `lineEnding()`), ANSI escape sequences in the middle of a prompt (`norm()` strips them before matching), and the create prompt of illumos, which differs from Windows ("Enter passphrase:" / "Re-enter passphrase:") and is now resolved separately for Solaris, Windows and the default.

5. Behavior and options

- `main.go` reads the start configuration from stdin, spawns the PTY, accepts one direct connection with IP and token authentication, relays with ChaCha20-Poly1305 and then exits.
- `NO_COLOR` is no longer forced on: the dark xterm.js theme renders ANSI colors fine. Opt back into plain text with `startConfig.no_color` or `CS_CONSOLE_FORCE_NO_COLOR=1`.
- A connection by the wrong peer is reported with an identifiable sentinel error, so a caller can translate it into a friendly message.

Source layout: `main.go` (start config, PTY spawn, relay), `pty.go` with `pty_unix.go`, `pty_windows.go` and `pty_illumos.go` (PTY backends), `session.go` and `crypto.go` (parent protocol and sealed relay), `auth.go` with `auth_unix.go` and `auth_windows.go` (password gate), `lockout.go` (per-IP brute-force lockout) and `expect.go` (allowlisted expect-mode actions).

6. Use in napp-it CS

- **System > Console** opens a console of the member in the browser. The binary is started per request and ends when the session ends.
- The **AI helpdesk** can control an interactive root console for troubleshooting and setups - only after a confirmation and a manual password entry through the console itself.
- The binary lives in `data/cs_server/tools/cs-console/<os>.<arch>/` and is distributed with **About > Download cs-tools**; it is not downloaded automatically.
- Remaining wiring still open: the web-GUI menu action for the new prompt actions and the allowlist mirroring in `server.pl` and `aihelplib`.

7. License

BSD 2-Clause - see the file LICENSE.