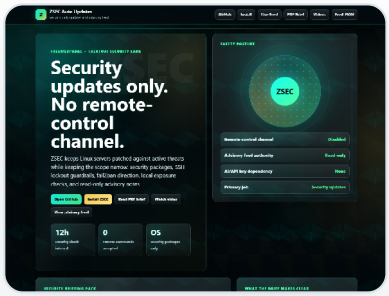
FOR SERVER OWNERS, HOSTING TEAMS, REVIEWERS, AND TECHNICAL STAKEHOLDERS

Security updates without a remote command channel.

ZSEC is a public security-only Linux update helper and advisory-feed workflow. The value is the narrow scope: apply OS security packages, review advisory notes, keep SSH/fail2ban/backups visible, and avoid remote-control ambiguity.

- **Security packages only.**
- **Read-only public advisory feed.**
- **No AI runtime, no provider API key, and no remote command channel.**



THE POINT

Narrow scope is the trust signal

Small servers fail because routine work gets missed.

A VPS can become risky through stale packages, unclear ownership, weak SSH habits, poor backup discipline, or one-off manual fixes that nobody records. ZSEC is framed as a routine, not a replacement for security operations.

- **Use a regular security-check rhythm, currently presented publicly as a 12-hour check interval.**
- **Keep access paths recoverable before hardening-sensitive changes.**
- **Treat status and advisory notes as owner-review material.**

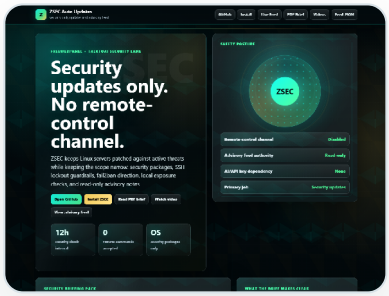
THE POINT

Routine beats panic patching

The safest updater says what it does not do.

ZSEC does not replace monitoring, backups, firewall policy, SSH key hygiene, incident response, or human review. The public feed is data and attribution, not authority to execute remote instructions.

- **No remote commands accepted.**
- **No arbitrary feature upgrades.**
- **No public secrets, private logs, customer data, or browser-supplied upstream authority.**

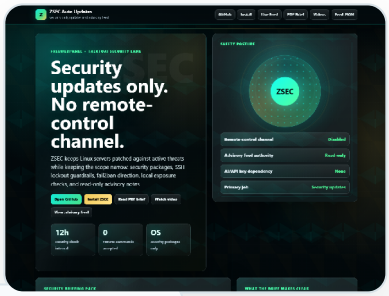


THE POINT

No remote control

The install and status path is intentionally simple.

The live ZSEC page gives a one-command installer and a local status command. That is enough for a public review path while keeping private configuration and server-specific evidence off the public site.



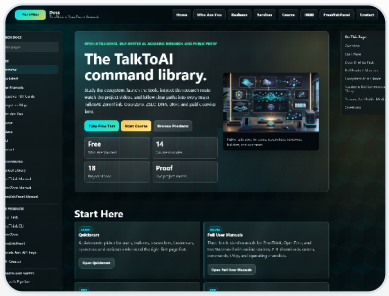
- **Install: curl -fsSL https://raw.githubusercontent.com/ResearchForumOnline/ZSEC/main/install.sh | sudo bash**
- **Check: sudo zsec status**
- **Read source before running on sensitive machines and keep rollback access available.**

Install, check, review

ZSEC uses normal Linux package managers.

The public explanation is deliberately plain: check local OS security state, read advisory signals, back up sensitive access material, apply security updates through apt or dnf, then report local status.

- **Check local package state and public advisory signals.**
- **Back up SSH config, authorized keys, and server snapshots before hardening-sensitive work.**
- **Patch operating-system security packages, not broad feature upgrades without review.**



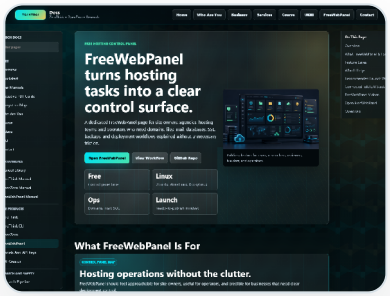
THE POINT

apt/dnf, local report

Access safety must be visible before hardening.

The live page calls out SSH lockout guardrails and fail2ban direction because server owners can break access while trying to improve security. ZSEC messaging should keep that risk front and center.

- **Confirm at least one working admin login route before SSH/firewall changes.**
- **Keep authorized keys and config recoverable.**
- **Use fail2ban-style thinking as part of an owner-reviewed server hygiene routine.**



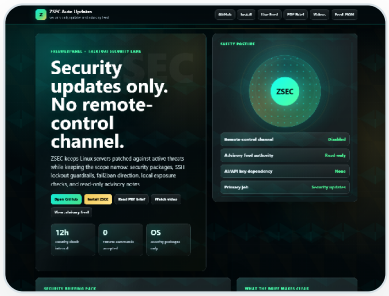
THE POINT

Do not lock yourself out

The feed is a signal layer, not a control plane.

The live ZSEC page describes normalized CISA KEV and security-news signals for local review notes. ZSEC clients can consume the data, but the feed does not tell machines to run arbitrary commands.

- **Read-only JSON/feed behavior.**
- **Attribution and public source signals stay inspectable.**
- **Local machines decide through local update tooling and owner policy.**



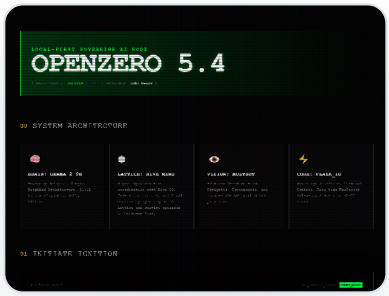
THE POINT

Data, not authority

ZSEC protects the hosting and AI infrastructure story.

OpenZero nodes, FreeWebPanel servers, WordPress deployments, public docs, and ZeroThink-related services all benefit when the security story is explicit and bounded.

- **FreeWebPanel can install or sit beside ZSEC, but ZSEC does not depend on panel code.**
- **OpenZero operators should read ZSEC and security docs before exposing any endpoint publicly.**
- **Business reviewers get a clear answer to what is automated and what remains human-controlled.**



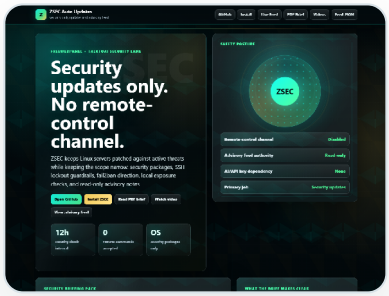
THE POINT

Hosting + AI needs server hygiene

Before installing, review the operational basics.

The most useful ZSEC PDF page is a checklist: backups, SSH access, scope, source, feed boundary, OS family, package manager, status output, and incident-response expectations.

- **Backups: snapshot and recoverable SSH material.**
- **Scope: security packages and advisory notes only.**
- **Review: GitHub source, feed JSON, status output, and what changed after patching.**



THE POINT

Review before automation

Inspect the live page, source, feed, and videos.

ZSEC is strongest when readers can see exactly what is public: the install route, GitHub source, advisory feed, docs module, and videos. That keeps the promise reviewable.

- **Open talktoai.org/zsec for public positioning.**
- **Open the ResearchForumOnline/ZSEC GitHub repository for source review.**
- **Use the full video for explanation and the short for quick security awareness.**

https://talktoai.org/

https://talktoai.org/course/

https://talktoai.org/zsec/

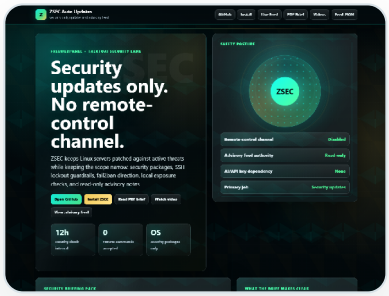
https://docs.talktoai.org/

https://docs.talktoai.org/freewebpanel/

https://docs.talktoai.org/manuals/freewebpanel-user-manual/

https://openzero.talktoai.org/

https://zerothink.talktoai.org/faq



THE POINT

Source and feed are inspectable