>BJ Bruce Jackson
CONSULTING

The 12 Things Your Agentic Config Needs Before It Touches IaC

If you can't tick every box, the agent is going to break something a code reviewer can't catch.

FREE CHECKLIST — brucejacksonconsulting.com

> *Source: 3.5 months running a single AI assistant config across shell, Python/Rust, Rust, and Terraform/Ansible — and the IaC gate categories that caught silent failures the standard linter missed. Sanitized; no host, role, or credential specifics quoted.*

- [ ] 01 **Plan-before-apply is enforced, not requested.**
The agent proposes a `terraform plan` and waits for human approval before `terraform apply`. A hook in the permission layer, not a line in the system prompt. Prompts get ignored under context pressure; permissions don't.

- [ ] 02 **`prevent_destroy` lifecycle blocks on every stateful resource.**
State storage, managed databases, DNS zones, secret stores. The lifecycle block is the only thing that fails the plan instead of silently scheduling a destroy.

- [ ] 03 **Secrets live in a vault; the agent has no read path to them.**
The agent reads the variable name; the runtime reads the value. If the agent can read the secret, your incident-response surface just expanded to include every session it ran in.

- [ ] 04 **Variable substitution is verified by the linter, not assumed correct.**
The highest-impact silent failure: a role that hard-coded the wrong account name in tasks meant for different users. Base linter: clean. Second-pass CoP rule: caught. Invisible to syntax-only review.

- [ ] 05 **Idempotency is asserted, not declared.**
Every task touching a system command needs `changed_when` written explicitly. "It probably won't change anything on re-run" is the most expensive sentence in infrastructure work.

- [ ] 06 **Undefined-variable fallbacks fail closed.**
A package-source key URL referencing an undefined variable renders to empty string in default Jinja and silently falls back to a deprecated key. Set the template engine to error on undefined; pin every variable in `defaults/` or `vars/`.

- [ ] 07 **Variable prefixes match the role that owns them.**
`run_updates` is a bug waiting to happen — two roles defining the same unprefixed name silently overwrite each other depending on inclusion order. The agent won't enforce this; the linter rule must.

- [ ] 08 **File-mode defaults are validated against the runtime, not the developer.**
Mode `0701` looks safer than `0755`. It also blocks the containerized service that owns it from writing — silent failure the deploy reports as success.

- [ ] 09 **Template format-sensitivity is respected.**
Some parsers (DNS zone files, certain JSON-driven daemons) reject valid templates if a managed-by-Ansible comment header lands in the wrong position. Carve out format-sensitive templates explicitly; the agent will helpfully add the header everywhere.

- [ ] 10 **Exit codes propagate through every wrapper script.**
One un-propagated exit code in a `Makefile` wrapper makes the entire CI gate fraudulent: green when it should be red. Audit every wrapper that calls a lint or safety tool.

- [ ] 11 **Hooks fail closed, not open.**
A hook that exits 0 on internal error is worse than no hook — it costs you the assumption the gate ran. Every hook: `set -e` for fail-fast, explicit exit codes, no `|| true` on the critical path.

- [ ] 12 **The agent proposes; it does not apply.**
This rule subsumes the other eleven. Enforcement belongs at the credential boundary. If the agent has apply credentials, no upstream gate matters.

This is the floor, not the ceiling — the gates without which an AI agent touching IaC is a production incident on a delay. It does not cover cost controls, code-repo review patterns, multi-language toolchain enforcement, or the cross-tool portability problem.

Those are in the long-form teardown: *3.5 Months Training One AI Assistant Across Shell, Python/Rust, Rust, and Terraform/Ansible* — at **brucejacksonconsulting.com/teardown**