Skyfort AI — Service Catalog & Operations

Monitoring · Vulnerabilities · AI Remediation · Baseline Hardening · Telegram Bot · Lite & SIEM Portals

Version 1.11 (EN) · 2026-10-28 · portal.skyfortcyber.com · @Skyfortreport_bot · Provider commercial/operations document

1. Overview

Skyfort AI is Skyfort Cyber's security operations (SecOps) platform: a lightweight agent installed on client machines (Linux and Windows), a gateway with AI-assisted analysis on the provider's infrastructure, a **Telegram bot** for day-to-day operations, a **web portal** for executive visibility, and a complementary **Wazuh sensor** for detection and compliance assessment (SCA).

Components

- **Skyfort Agent** — collects metrics, inventory, CIS and vulnerabilities; fetches remediation commands every 2 min.
- **Gateway (AI)** — multi-tenant API, command queue, vulnerability fusion, alerts.
- **Telegram Bot** — reports and 1-click remediation from your phone.
- **Portal v3** — client dashboard (SOC, SCA, NCs, patches).
- **Wazuh** — sensor: ~1,600 SCA non-conformities mapped to the Skyfort catalog.

Offered services (summary)

- 24×7 monitoring with daily report and proactive alerts
- Multi-source vulnerability management (Skyfort + CIS + Wazuh)
- **AI remediation** — from detection to verified fix in ≤ 2 min, with human approval
- Baseline hardening Top 15 Emergencial / Armor / Shield / Fortress with rollback
- Fleet batch operations (servers × workstations)
- Executive portal and compliance inventory

2. Plans & portal by plan

Ladder structure, following the US MSP/MSSP market pattern: **per-endpoint/month pricing**, increasing value per tier and the **full SIEM anchored on the top plan** — the most expensive asset is the upgrade motive, not the entry pattern. Monitoring, bot and AI-assisted remediation exist on every plan.

Feature	Essentials	Professional	Enterprise (SIEM)
Client portal	**Lite** — posture score, machines, pending updates	**Lite** — + vulnerability inventory	**SIEM v3** — SOC, threat detection, FIM, events, maturity score
Detection	Essential Skyfort catalog (EOL, SMBv1, updates, baseline drift)	Full Skyfort catalog (proprietary framework)	+ CIS Benchmarks + Wazuh sensor (~1,600 NCs) merged
Remediation	Top 15 Emergencial + Armor baselines (apply + re-apply on drift)	**Full playbook catalog** (safe + approval, 1-click AI)	+ fixes mapped to CIS/Wazuh NCs and server × workstation batch
Baseline	Top 15 Emergencial + Armor	+ Shield	+ Fortress
Reports	Daily (Telegram)	+ weekly executive summary	+ monthly executive / compliance

> **Autonomous add-on**: the AI automatically fixes *safe*-risk items (no individual confirmation); *approval* items keep human approval — a configurable *human-in-the-loop* per client policy.

Competitive edge: AI remediation (closed-loop)

Available from **Professional** up (Skyfort catalog); on Enterprise the AI also crosses CIS/Wazuh NCs to suggest fixes.

1. Detect	2. Suggest & fix	3. Guarantee
Agent + Wazuh + CIS feed the gateway AI — a multi-source, deduplicated, severity-ranked view.	AI suggests the exact playbook; 1-tap approval on Telegram; execution in ≤ 2 min with automatic post-fix validation.	Audited result (validate), 90-minute impact window and rollback by client decision.

Sales message (US market): "*AI-powered remediation — from alert to verified fix in under 2 minutes, with human-in-the-loop approval and one-click rollback.*"

3. Telegram Bot — @Skyfortreport_bot

The client's operations channel. The chat is **bound per tenant**: unknown chats are blocked and logged. Binding: send any command to the bot and report the `chat_id` to the provider.

Commands

Command	What it does
`/start` · `/help`	Welcome, tenant name, plan and command list.
`/status`	Report of the tenant's machines: active alerts, RAM and per-disk usage, with alert details.
`/vulns`	Vulnerability inventory summary (Skyfort + CIS + Wazuh merge, per plan), by severity.
`/vuln <host>`	Machine top-10 with severity, CIS control and → **suggested fix** — when a playbook exists, suggests `/remediate <id> <host>`.
`/playbooks`	Available remediation catalog (with each item's risk level).
`/remediate <id> <host> [confirm]`	Queues a playbook for execution on the machine. **safe**: runs immediately. **approval**: requires `confirm` as the 3rd argument. Result in ≤ 2 min with automatic post-execution validation. **Revert:** `/remediate <id> <host> rollback confirm` undoes the fix (state-changing playbooks carry their own reversal, with a pre-change backup). Post-execution confirmation arrives as "✔ NC fixed" or "↩ Reversal applied", with numbers.

Proactive communications

- **Daily report** — 8:00 AM, per-tenant summary with each machine's state.
- ✔ **NC fixed** — automatic confirmation after successful remediation (with the validate result).
- ⚠ **Baseline impact** — after applying Top 15 Emergencial / Armor / Shield / Fortress, for 90 min any service that stops triggers a single alert: "*NOTHING was reverted — the decision is yours — if you identify impact run* `/remediate SKYFORT_ROLLBACK <host> confirm`".
- **Isolation** — only the chat bound to the tenant receives data; requests from other chats are refused.

4. Remediation catalog (playbooks)

Playbooks are auditable procedures executed by the agent on the machine, with a **risk level** that defines gating, **automatic validation** after execution and **idempotency** (safe re-execution). Risks:

`safe` runs directly from the bot `approval` requires client confirmation `manual` not script-remediable (provider follow-up)

> **Executable rollback on every playbook** — procedures that change state carry their own reversal (pre-change backup + restore steps): `/remediate <id> <host> rollback confirm`. Exceptions (no state change): channel tests (`*_AGENT_PING`) and `WIN_OS_EOL` (manual).

ID	Action	Risk	Target OS
`AUDITD_SERVICE`	Install and enable auditd (audit trail)	safe	Linux
`PASS_MAX_DAYS`	Password expiration ≤ 365 days	safe	Linux
`DEFAULT_UMASK`	umask 027 in login.defs	safe	Linux
`SSH_MAXAUTHTRIES`	SSH MaxAuthTries ≤ 4	safe	Linux
`SSH_PASSWORD_AUTH`	Disable SSH password authentication (key only)	approval	Linux
`SSH_PERMIT_ROOT_LOGIN`	Prohibit direct root login via SSH	approval	Linux
`FIREWALL_INACTIVE`	Enable ufw guaranteeing the SSH rule first	approval	Linux
`PENDING_UPDATES`	Apply pending updates (maintenance window)	approval	Both
`WIN_WU_DISABLED`	Re-enable the Windows Update service	safe	Windows
`WIN_SMB1`	Disable SMBv1	approval	Windows
`WIN_OS_EOL`	Unsupported OS — requires migration/upgrade	manual	Windows
`LINUX_AGENT_PING` / `WIN_AGENT_PING`	End-to-end test of the command channel	safe	Linux / Windows
`SKYFORT_SHIELD`	Level-1 baseline, product **Top 15 Emergencial** (on Linux: **Armor**) — Windows: INF policy per OS version · **Linux: per-distro CIS package, item by item (Ubuntu 22.04/24.04 and Debian 13 — 67/78/84 controls)**	approval	Windows + Linux
`SKYFORT_ARMOR`	Level-2 baseline, product **Armor** (cumulative)	approval	Windows
`SKYFORT_FORTRESS`	Level-3 baseline, product **Shield** (cumulative, maximum)	approval	Windows
`SKYFORT_PADRAO`	**Fortress** — official CIS baseline (Windows 11, 407 items · Windows Server 2016/2019/2022/2025, 320/341/336/357 items) plus HardenedPaths rules	approval	Windows
`SKYFORT_ROLLBACK`	Rollback: restores the state from before the last baseline (Windows: policy + registry; Linux: files, from backup)	approval	Windows + Linux

5. Hardening baseline — Top 15 Emergencial · Armor · Shield · Fortress

Hardening packages applied through the same bot/portal queue: **Windows** via local policy (secedit/INF) and **Linux** via a per-distro CIS package, applied **item by item** by the executor — with prior backup, version stamp, **continuous drift monitoring** and rollback under client decision. Tiers are **cumulative** (Armor includes Top 15 Emergencial; Shield includes the previous ones); Fortress is the official CIS baseline — **Windows 11** (407 controls) and **Windows Server 2016/2019/2022/2025** (320/341/336/357 controls). On **Linux**, Armor has a per-distro package (the entry tier on Linux): **Ubuntu 22.04** (67 controls), **Ubuntu 24.04** (78) and **Debian 13** (84).

OS version	Top 15 Emergencial (N1)	Armor (N2)	Shield (N3)	Fortress (CIS)
Windows Server 2008 R2	✓	—	—	—
Windows Server 2012 / 2012 R2	✓	✓	—	—
Windows Server 2016 / 2019 / 2022 / 2025	✓	✓	✓	✓
Windows 11	✓	✓	✓	✓
Windows 10	package not published in this framework version			
Ubuntu 22.04 LTS (Linux)	**✓ Armor (67 CIS controls, item by item)**	—	—	—
Ubuntu 24.04 LTS (Linux)	**✓ Armor (78 CIS controls, item by item)**	—	—	—
Debian 13 (Linux)	**✓ Armor (84 CIS controls, item by item)**	—	—	—
Other Linux (Ubuntu 26.04, Debian 12, RHEL)	per-distro package in upcoming phases — the executor warns when a distro has no package yet			

How it works

- **Backup before everything** — **Windows**: policy export (`pre-secedit.inf`) + snapshot of the target registry; rotation of the last 5. **Linux**: faithful copy of every file about to be touched (`pre-baseline/<tier>`, with directory mode/owner registered).
- **Stamp** — **Windows**: `PSS\Baseline` registry value. **Linux**: `baseline-stamp.json` with tier, version and distro.
- **Item-by-item verification** — Windows: ≈2,787 items across manifests (registry checked in HKLM + policy via export, before × after), 3,126 including the per-distro Linux and macOS manifests. Linux: per-distro manifest with per-item check and fix (Armor: Ubuntu 22.04 67 · Ubuntu 24.04 78 · Debian 13 84); reports fixes/drift with the same numbers on the portal and Telegram.
- **Drift monitor** — every collection checks against the manifest; drift becomes finding `SKYFORT_BASELINE` and fixes return as `corrected` once normalized (Windows and Linux, same format).
- **Rollback** — `/remediate SKYFORT_ROLLBACK <host> confirm` restores the state from before the last application (Windows: policy + registry; Linux: files, tolerant per file) and removes the stamp. **Never automatic**: the decision is always the client's.
- **Per-machine exceptions** — hosts that route traffic (hypervisor, Docker, Tailscale) skip the network-forwarding/kernel-module items: the exception is per machine, the package is the same for any client.
- **Domain (DC)** — GPO may overwrite on refresh; document exceptions.
- **Per-OS documentation** — `portal.skyfortcyber.com/skyfort-framework/docs/` (Top 15 Emergencial / Armor / Shield / Fortress × each version).

> **Batch application (provider operation)**: the internal `skyfort-baseline` tool applies the baseline across the entire fleet separating **servers × workstations**, with per-OS package validation and duplicate-execution blocking.

6. Client portal — portal.skyfortcyber.com

Two portals depending on the plan: **Lite** (Essentials/Professional) and **SIEM v3** (Enterprise).

Downloads & installation stay INSIDE the tenant (v1.6, 28/09): the client portal's **Downloads** menu opens *that* tenant's install page — both agents (Skyfort AI + Wazuh sensor) with **the tenant's own credentials embedded** in the commands, the **associated agents live**, and the framework documentation. No global entry in the navigation; every machine installed there is born bound to the right tenant.

SIEM Portal (v3) — Enterprise

- **Executive Summary** and **Security Posture** — executive view.
- **Security Maturity Score** — maturity evolution.
- **SOC** — Threat Detection Rules, Event Activity, FIM.
- **Vulnerability Management** — NCs overview and detailed (Wazuh + Skyfort).
- **SCA** — per-machine configuration assessment.
- **Patch Management** — pending updates per host.
- **Infrastructure Visibility** — Monitored Hosts and inventory.
- **Risk Prioritization (KEV + EPSS)** — queue ranked by real-world exploitation (CISA KEV + EPSS), with drill-down: every number opens the full list and every item opens its detail (affected machines, remediation, NVD).
- **Reports + executive PDF** — 6 downloadable reports (CSV) and a printable Full Security Report stamped with the client's environment name.
- **Multi-tenant access** — each client signs in with their environment key and sees only their own machines (end-to-end validated isolation).
- **Remediation console (closed-loop)** — dedicated results page: compliance by policy (exact numbers per benchmark), bulk apply by OS with checkboxes, per-machine detail with evidence, visible queue with Telegram confirmation; fix buttons on CVEs plus the remediation/baseline catalog (Top 15 Emergencial / Armor / Shield / Fortress with rollback), executed within 2 minutes by the agent.

Lite Portal — Essentials / Professional

- Posture score and simple fleet view
- Top non-conformities and pending updates
- **Web fixes** — apply and revert baseline per machine/batch, item-by-item audit per machine and per package, with numbers on the portal and bot
- Direct access to the bot and downloads

Download & integration areas

- `/downloads/` — agent installers (Windows/Linux) and framework.
- `/skyfort-framework/docs/` — baseline package documentation.
- `/skyfort-ai/` — gateway API (per-tenant authentication).
- Portal v1/v2 — legacy, under maintenance (Intune install and legacy reports).

Corporate integration

- Windows agent deployment via **Intune** (per-tenant management script, pre-injected key).
- CIS export/import for compliance pipelines (Qualys/Intune) on demand.

One key per client environment: each environment (tenant) receives its own `agent_api_key` — it identifies where each machine reports and what the portal and bot display. Each environment has isolated visibility (dashboard, remediations and inventory show only that environment's machines); a machine installed with another environment's key lands in the wrong environment and is invisible to the right one. The key is delivered by the provider (Telegram) and the single-use install link (24 h) embeds the environment's key — the client copies, pastes and the machine is born in the right environment. Rotation: a new key in the configuration invalidates the old one immediately.

7. Gateway API (integrations)

Public base: `https://portal.skyfortcyber.com/skyfort-ai/` · Authentication: `Authorization: Bearer <tenant agent_api_key>` · Keys per tenant, never in URL/argv.

Endpoint	Function
`POST /api/v1/report`	Receives the agent report (metrics, CIS, findings, baseline).
`GET /api/v1/status`	Fleet: tenants, machines and last-report age.
`GET /api/v1/vulns`	Merged vulnerability inventory (Skyfort + CIS + Wazuh).
`GET /api/v1/commands?host=`	Host's pending command queue (consumed by the executor).
`POST /api/v1/command`	Queues a playbook (`host` , `playbook` , `confirm` ; `action: "rollback"` to revert) — risk, plan and host revalidated server-side.
`POST /api/v1/command/result`	Execution result (output + validate).

8. Onboarding — from purchase to value, by plan

Standard journey after signing: provisioning, agents, sensor and hardening in dated phases. First results (visible machines + daily report) arrive within 48 h on all plans. By Telegram the client receives the access key, the portal login and their **own tenant Downloads & installation page** (agents + documentation).

Phase	What happens	Plans
0 · Contracting Day 0	Tenant provisioned on the gateway (plan + key), Telegram chat bound to the bot, portal access created. Kickoff: machine inventory (server × workstation), maintenance windows and contacts.	All plans
1 · Skyfort Agent Day 1-2	**Windows**: installer with per-tenant pre-injected key (pilot machine → fleet via Intune) · **Linux**: SSH installer (key via stdin). Verification: all machines reporting on the bot (`/status`). Assisted install: **single-use link (24 h)** that installs agent and sensor with the environment's key embedded — no typing by the client.	All plans
2 · Wazuh sensor Day 2-3	Per-tenant agent group on the Wazuh Manager, Wazuh agent deploy (MSI/package) with enrollment, active SCA — non-conformities start merging into the inventory (bot and portal).	Enterprise
3 · Hardening Week 1-2	Laddered baseline: Top 15 Emergencial → Armor → Shield → Fortress (Enterprise includes server × workstation batch). 90-min impact window monitored, rollback ready.	All (ladder per plan)
4 · AI routine Ongoing	Daily report 8:00 AM, AI remediation in daily ops (`/vuln` → `/remediate`), weekly executive summary (Professional) and monthly + maturity review (Enterprise).	All (cadence per plan)

Onboarding model by plan

Step	Essentials	Professional	Enterprise (SIEM)
Provisioning + bot + portal	Day 0	Day 0	Day 0
Skyfort Agent	Self-serve (installer/Intune)	Assisted (Intune/fleet)	Full fleet white-glove
Wazuh (SCA/SOC)	—	—	SCA + FIM + threat detection tuned
Initial baseline	Top 15 Emergencial + Armor	+ Shield	+ Fortress and server × WKS batch
Reports from day one	Daily	+ Weekly	+ Monthly / compliance
Time to operational	24 h	48 h	~1 week

> **Client prerequisites**: machine list with function, administrative access (Admin/Intune and SSH), maintenance window for hardening and a technical contact to bind the bot.

9. Recommended operational flow

#	Step	Where / how
1	Install agent	Per-OS installer at `/downloads/skyfort-agent/` (or Intune for the Windows fleet).
2	Bind Telegram chat	First command to the bot → provider copies the `chat_id` and binds it to the tenant.
3	Daily routine	8:00 AM report on Telegram + check on Portal v3.
4	Inventory	`/vulns` (summary) and `/vuln <host>` (top-10 with suggested fix).
5	Remediate	`/remediate <id> <host> confirm` → result and validation in ≤ 2 min.
6	Hardening	Baseline Top 15 Emergencial → Armor → Shield → Fortress per machine (or server × workstation batch, provider-operated).
7	Impact?	Alert within 90 min; if the client identifies impact: `/remediate SKYFORT_ROLLBACK <host> confirm`.

10. Security & guarantees

- **Credentials** — one `agent_api_key` per environment (tenant): the key identifies the client's environment everywhere (agent→gateway, portal, bot); delivered by the provider (Telegram), never in argv, URL or chat; rotation kills the old one instantly (401); isolation validated (invalid key → 401, wrong tenant → 403); the single-use install link embeds the environment's key with no typing.
- **Server-side approval** — approval/manual playbooks are revalidated server-side; approval without confirmation is refused; **manual never executes** via script.
- **Automatic validation** — every playbook carries post-execution verification (the result shows an explicit *validate*).
- **Reversibility** — baseline only applies with a valid prior backup (Windows: policy+registry; Linux: files, per item); state-changing playbooks carry their own executable rollback; rollback is always the client's decision.
- **Management preserved** — playbooks never lock management access or take down services: firewall rules guarantee the ports before applying.
- **Audit** — every execution is logged (queue, output, validate) and the daily report consolidates the state.