FALYS

ENTERPRISE FILE INTEGRITY MONITORING (FIM) SYSTEM

User Guide

Version 2.2

Prepared By

Sheeza Alam Khan

Maimoona Shah Khan

Real-Time Filesystem Anomaly Detection • Enterprise-Grade Threat Response

1. Introduction

1.1 About FALYS

FALYS (File Integrity Monitoring System) is a real-time File Integrity Monitoring (FIM) platform. It continuously watches designated directories on Windows and Linux endpoints, detects create/modify/delete/move activity, attributes each change to the responsible OS account wherever possible, scores every event through a multi-signal detection engine, and alerts the right people the moment something looks wrong.

1.2 Key Features

- Lightweight Windows and Linux agents, deployable as single compiled binaries — no Python required on the endpoint.
- Five-stage detection engine: context collection, rule-based scoring, behavioral anomaly detection, event correlation, and confidence-based classification.
- Per-agent alert routing — each deployed agent sends alerts to its own assigned focal-person email.
- Real-time dashboard with live event streaming (Server-Sent Events), historical search/export, and a correlated-incident feed.
- OS-level user attribution via Linux audits and Windows Security event logs, clearly labeled Verified / Unverified.
- LAN auto-discovery so agents keep finding the server even after its IP changes, with a replay-resistant, time-bucketed discovery protocol (see Chapter 9).
- HTTPS by default, with a self-signed certificate the server generates and manages automatically, and certificate pinning on every agent (see Chapter 9).
- Per-agent API keys, individually revocable, so one compromised endpoint doesn't compromise the whole fleet (see Chapter 9).
- Admin-editable detection rules and engine tuning — no code changes needed to adjust sensitivity.

1.3 Target Users

IT Administrators, Security Operations (SOC) Analysts, enterprise compliance teams, and technical evaluators who need continuous, explainable visibility into unauthorized or unexpected file-system activity.

2. System Architecture

2.1 Components

Component	Responsibility
Agent (Windows / Linux)	Watches the configured folder tree, computes file hashes and permissions, resolves OS-level user attribution, queues events offline, and ships them to the server over HTTPS.
Flask Server	Ingests events over HTTPS (HTTP fallback if TLS is explicitly disabled) with per-agent API-key authentication, runs the detection engine, stores events, serves the dashboard, and dispatches email alerts.
Detection Engine	Five in-process stages that turn a raw event into a classification, confidence score, and explainable reasons.
Dashboard (Web Console)	Session-authenticated UI for live monitoring, history, agent management, detection review, and configuration.
Discovery Service	UDP broadcast protocol that lets agents relocate the server automatically after a DHCP IP change, protected by a rotating time-bucketed proof (see Chapter 9).
Email Service	Sends alert emails via Gmail SMTP (STARTTLS on port 587), with per-agent routing and burst throttling.

2.2 Data Flow

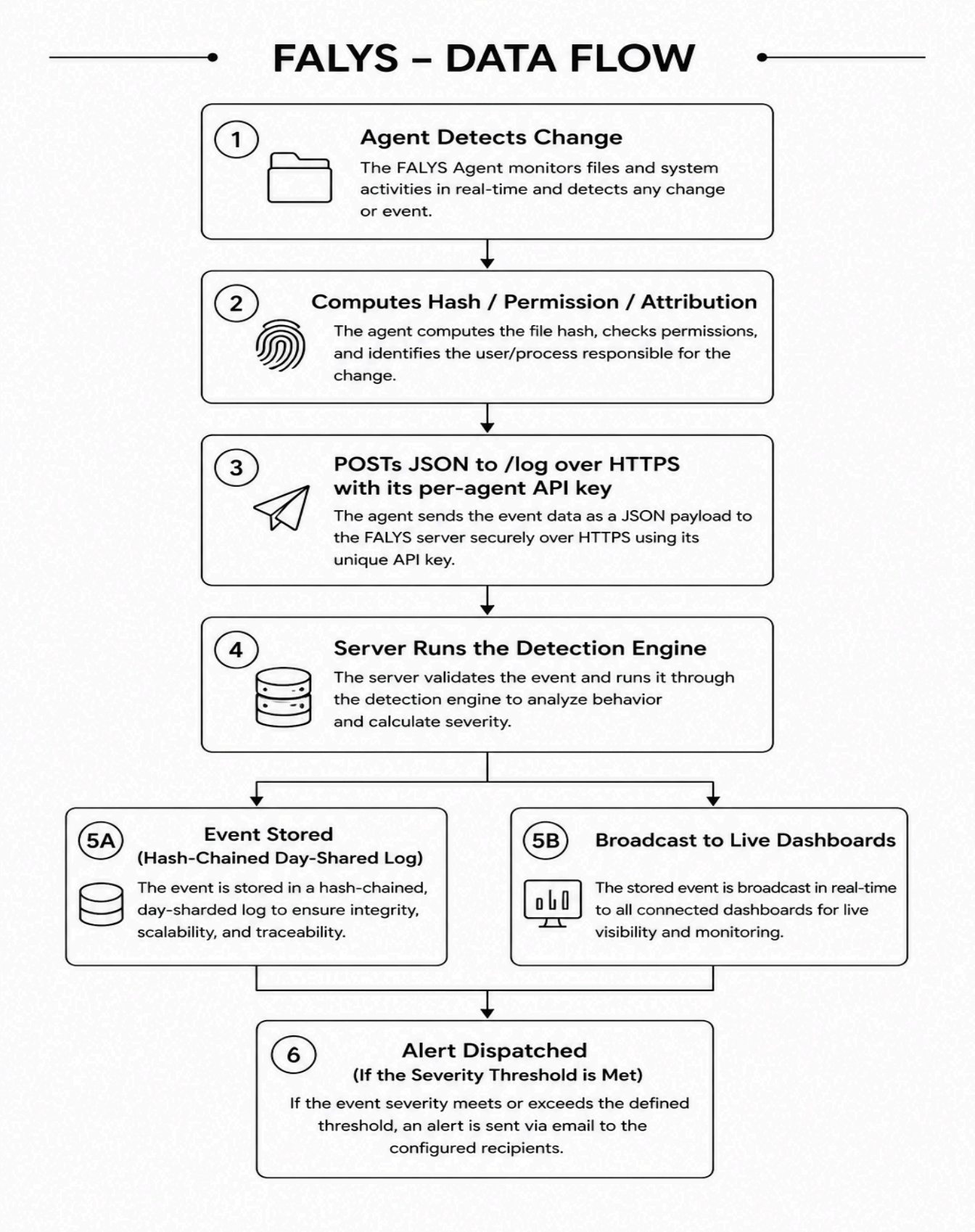
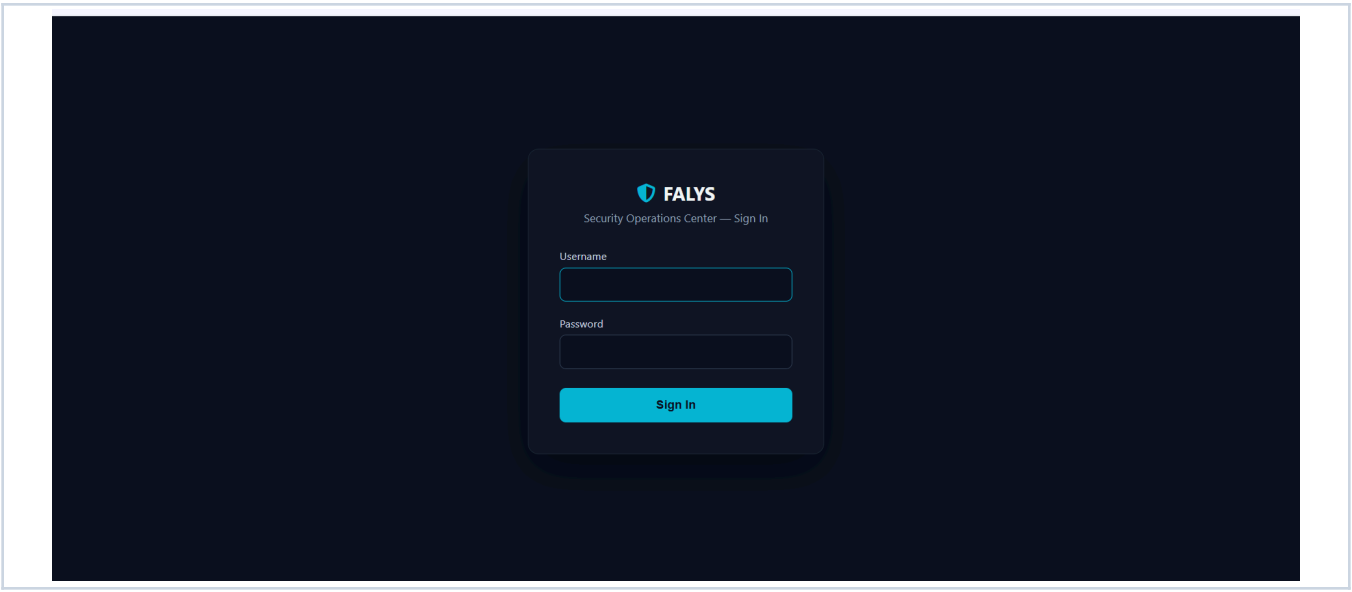


Figure 2.1 — The FALYS Security Operations Center home page, showing protected-endpoint count, total events logged, threats detected, and alerts sent at a glance.

3. Installation & Deployment

3.1 Server Installation

- For a production deployment, run the pre-built server binary directly — no Python installation required on the server machine: FalysServer.exe on Windows, or the falys-server binary on Linux.
- Double-click FalysServer.exe (Windows) or run ./falys-server (Linux) — it starts on port 8088 (HTTPS by default) and immediately begins broadcasting its LAN presence. For development or contributing to the source, python server.py works identically, provided the dependencies in server/requirements.txt (Flask, waitress, cheroot, cryptography, requests) are installed first.
- On the first run, any secret you haven't set yourself (admin password, agent API key, discovery token) is generated automatically and printed once to the console, then saved to logs/generated_secrets.json for reuse on restart.
- Sign in at the login page using the printed administrator credentials, or the username/password you set via FALYS_ADMIN_USERNAME / FALYS_ADMIN_PASSWORD.



3.1.1 Running the Server as a Windows Service (Optional)

For deployments that need the server to survive a logoff or restart automatically on boot — the same expectation as Wazuh Manager or Splunk Enterprise — FalysServerService.exe registers FALYS as a real Windows Service, as an addition to (not a replacement for) the plain double-click FalysServer.exe.

FalysServerService.exe install then FalysServerService.exe start registers and starts the service; stop and remove reverse it. Double-clicking the exe once, with no arguments, does the same install-and-start automatically after one UAC elevation prompt. Once installed, it appears in services.msc as “FALYS SOC Server” with Startup type Automatic, and is managed the same way as any other Windows service (sc query FalysServer confirms its state).

3.1.2 Linux Server Portability

The falys-server binary is built with PyInstaller against a specific glibc version and will run on that version or newer, but will fail with a GLIBC_2.XX not found error on an older distribution than the one it was built on. Build it on the oldest Linux distribution version you need to support — the same portability rule that already applies to the falys-agent binary (see Chapter 10, Troubleshooting).

3.1.3 Preparing a Clean Install for Distribution

When copying a built server (FalysServer.exe, FalysServerService.exe, or falys-server) to a new machine, copy only the binary and the build_output folder. Do not copy an existing logs folder along with it — logs holds that specific instance’s generated admin password, discovery token, TLS private key, issued agent API keys, and event history. A fresh install should generate all of these itself on first run; copying someone else’s logs folder means the new install starts already carrying another deployment’s live secrets and data.

3.2 Deploying an Agent

From the Agents page, an administrator selects a platform (Windows or Linux), enters the folder to watch and a focal-person alert email, and clicks Generate & Download. FALYS ships back a ready-to-run package: a single compiled binary (FalysAgent.exe or falys-agent), a pre-filled config.json, and the server's pinned TLS certificate (cert.pem) — no

Python or manual setup required on the endpoint. Each generated package now also receives its own unique, individually revocable API key (see Chapter 9, Section 9.3).

3.2.1 Windows Agent Deployment

Double-click FalysAgent.exe. It self-installs as a Windows service, applies the audit policy needed for user attribution, and starts immediately.

3.2.2 Linux Agent Deployment

chmod +x falys-agent && sudo ./falys-agent. It installs itself to /opt/falys-agent, configures auditd, and registers a systemd service.

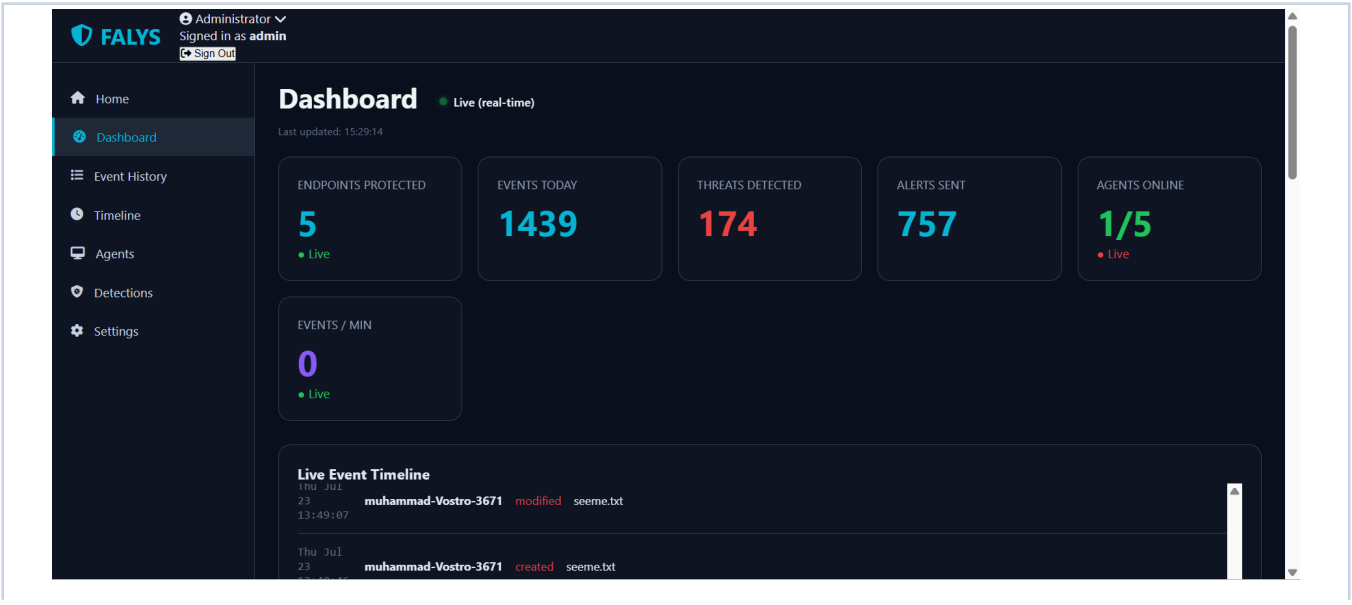
3.2.3 OS-Level User Attribution Setup

Optional — for verified per-user attribution, run setup_audit_windows.ps1 (Windows) or setup_audit_linux.sh (Linux) once on each endpoint.

4. Dashboard Guide

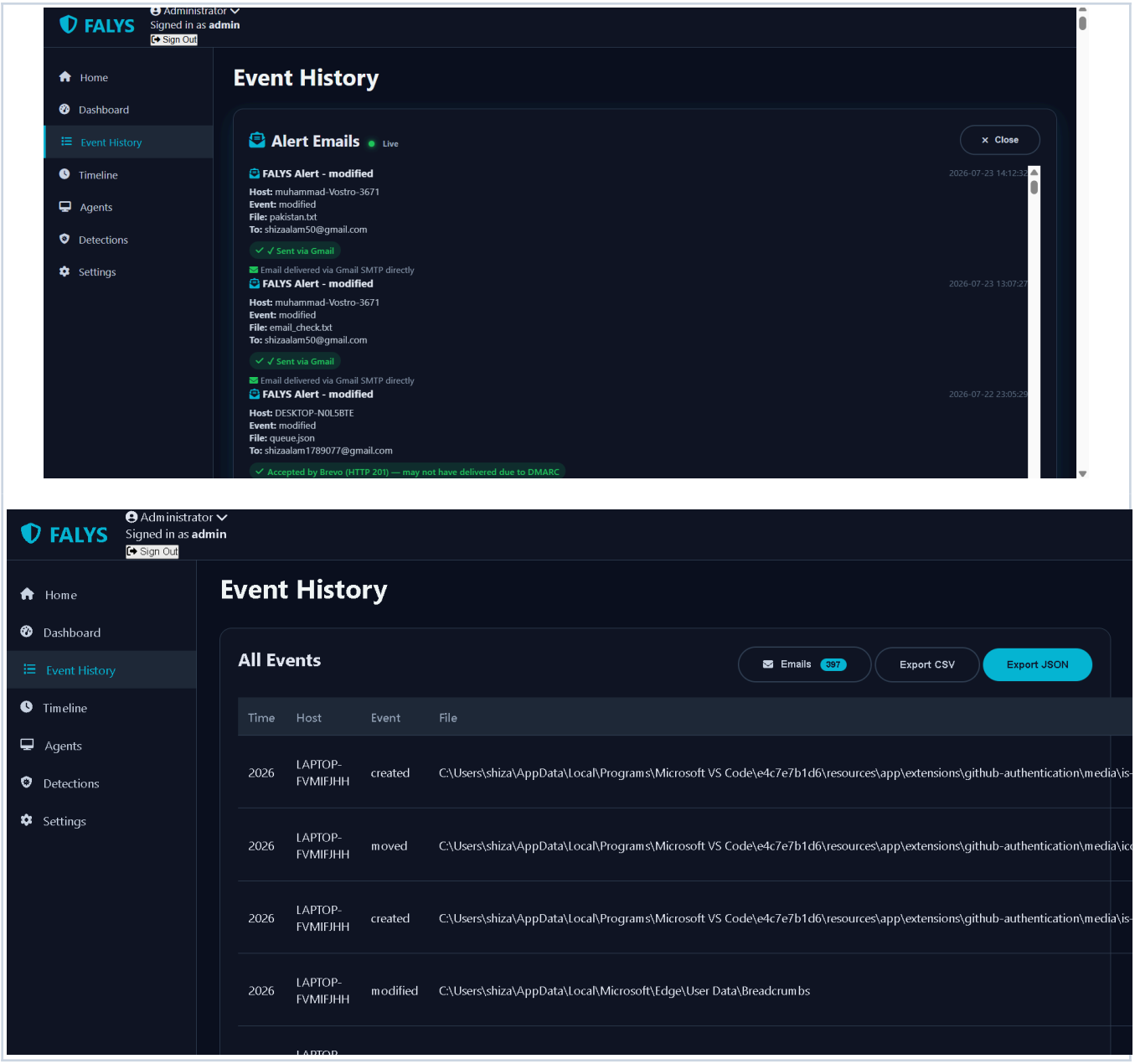
4.1 Live Dashboard

Shows real-time KPIs (events today, threats detected, alerts sent, agents online, events/minute), a live event timeline pushed over Server-Sent Events, and 24-hour trend, distribution, and severity charts.



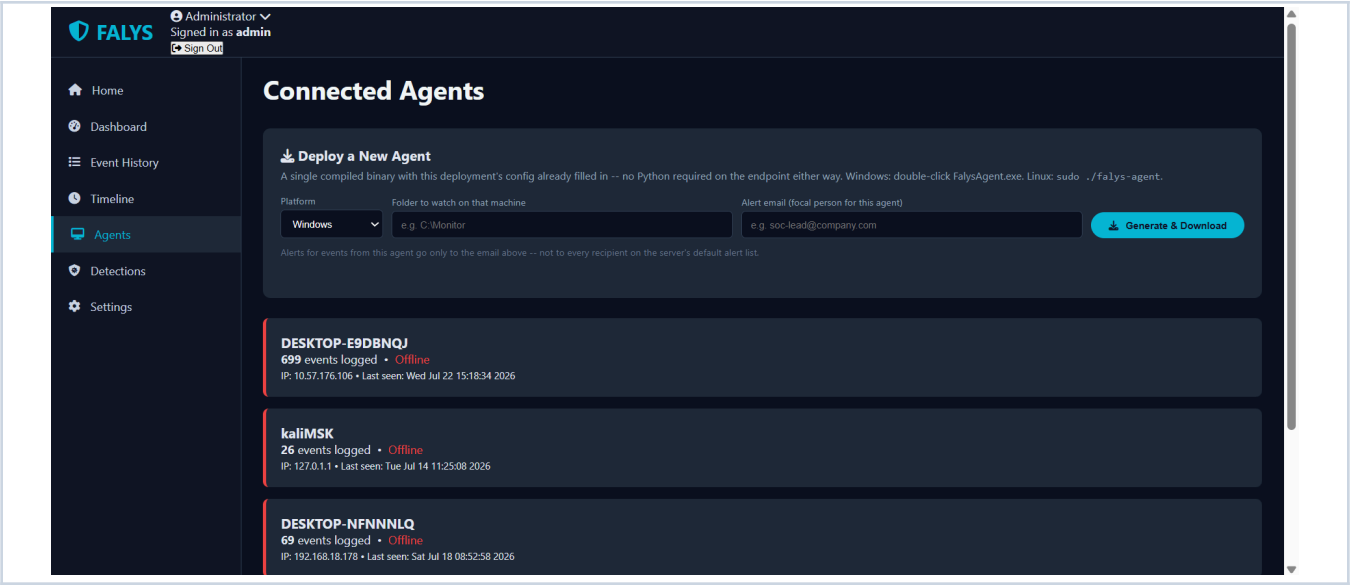
4.2 Event History

A searchable, exportable (CSV/JSON) log of every event ever received, including host, event type, file path, and the attributed user with a clear Verified / Unverified badge. An expandable panel shows the full email-alert history per event, including Gmail delivery status.



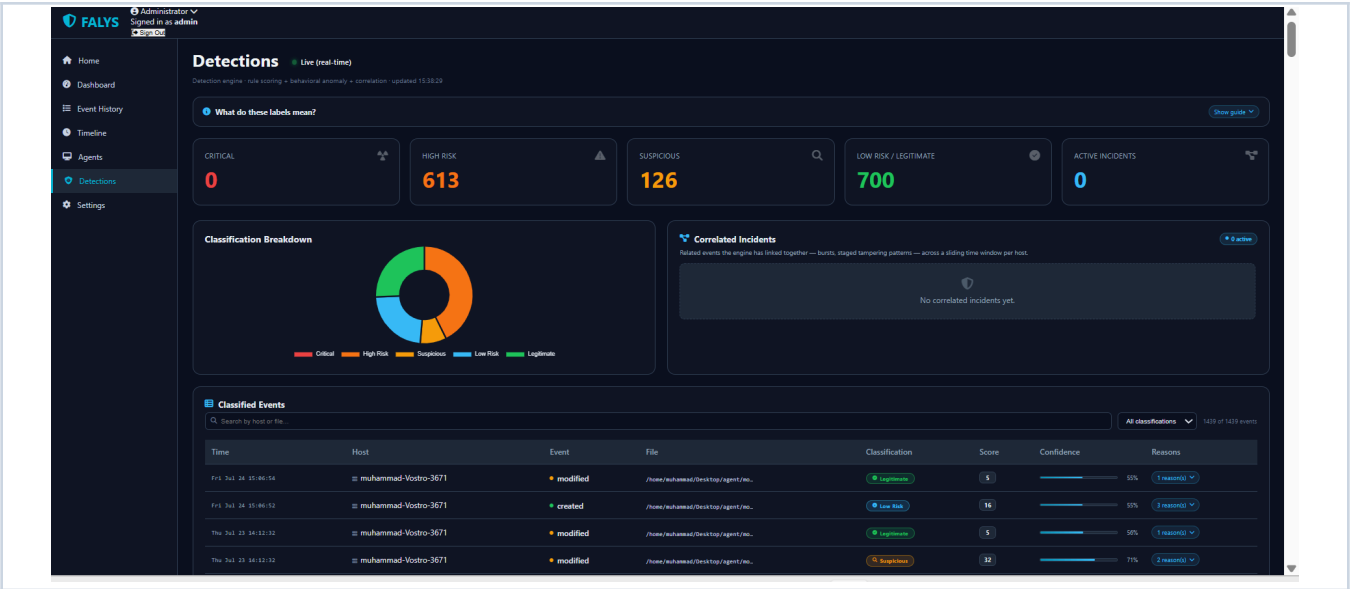
4.3 Connected Agents

Lists every agent that has ever reported in, with live Online/Offline status (configurable offline threshold), event counts, and last-seen IP. This is also where new agent deployment packages are generated, each now with its own individually revocable API key.



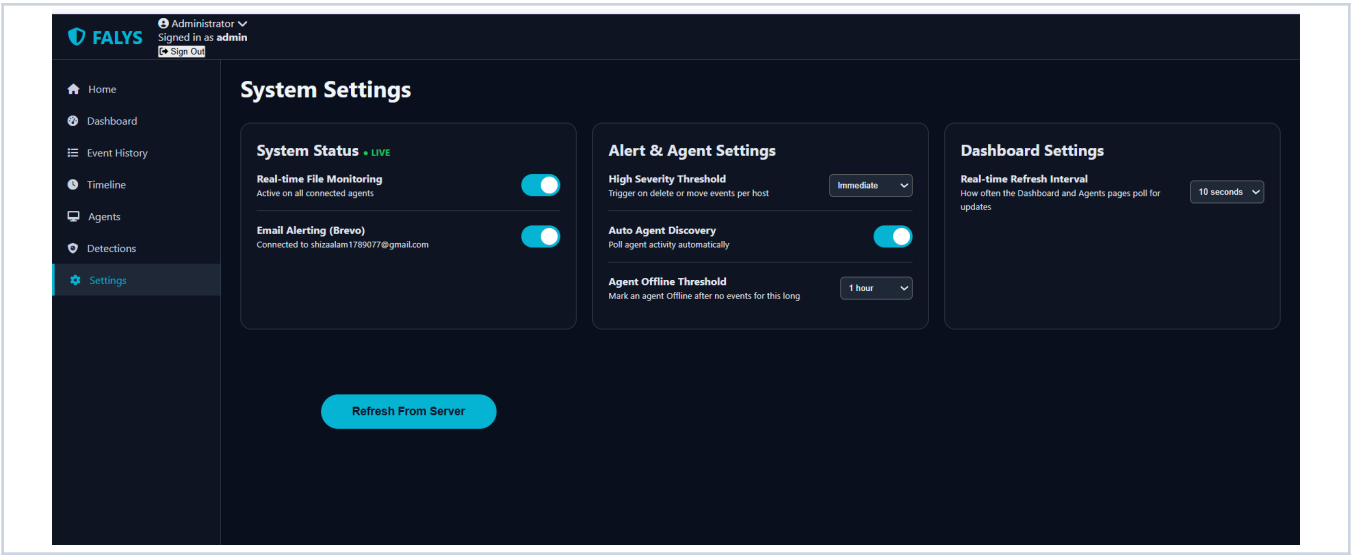
4.4 Detections

The detection engine's output: KPI cards by classification tier, a classification-breakdown chart, the correlated-incidents feed, and a filterable table of every classified event with its score, confidence, and expandable reasons.



4.5 Settings

Administrator controls for real-time monitoring, email alerting, high-severity alert threshold, auto agent discovery, agent-offline threshold, and dashboard refresh interval.



5. Monitoring & the Detection Engine

5.1 What Gets Monitored

Every Created, Modified, Deleted, and Moved event is captured. For each one, the agent records a SHA-256 hash, the file's permission string, and — where OS auditing is configured — the real account responsible for the change (not just the account the agent runs as).

5.2 The Five-Stage Detection Pipeline

Stage	Module	Purpose
1	Context Collection	Gathers situational facts: is this host/user new, is it business hours, is the host a tagged critical asset.
2	Rule-Based Scoring	Deterministic, admin-editable rules: event-type baseline, sensitive-path matches, permission changes, burst frequency.
3	Behavioral Anomaly Detection	Learns each host's own statistical baseline (event rate, active hours) and flags deviations from it.
4	Event Correlation	Groups related events into incidents: bursts, staged tampering patterns, and cross-host activity by one user.
5	Confidence-Based Classification	Combines all three scores into one 0–100 risk score, a five-tier classification, and a confidence value.

5.3 Risk Classification Tiers

Tier	Score	Meaning
Legitimate	0–9	Normal, everyday activity. No action needed.
Low Risk	10–29	A little unusual, but minor. Worth a glance.
Suspicious	30–54	Uncommon enough that a person should take a look.

Tier	Score	Meaning
High Risk	55–79	Strong signs something is wrong — investigate soon.
Critical	80–100	Very likely a real security problem — act now.

NOTE: Confidence is not severity. Confidence measures how much history backs a score, separate from how dangerous it looks. A Critical result at low confidence usually means the host is new and FALYS doesn't have much baseline for it yet.

6. Email Alerts

6.1 Per-Agent Alert Routing

Every deployed agent carries its own alert_email, set once at package-generation time. Alerts for events from that agent go only to that address — never to every recipient on a shared list.

6.2 Gmail SMTP Delivery

Emails are sent directly through Gmail SMTP (smtp.gmail.com, port 587, STARTTLS) using an app password, which avoids DMARC rejection issues seen with third-party relay providers when sending as a Gmail address. No Gmail credential ships with FALYS — FALYS_GMAIL_SENDER and FALYS_GMAIL_APP_PASSWORD must be set via the environment before alerting is active; until then, the server logs alerts as skipped rather than failing silently.

6.3 Alert Throttling

To prevent alert storms, only the first event to a given recipient in a 30-second window triggers an email; the rest are logged as throttled and remain visible in Event History.

7. Configuration Reference

7.1 General Settings

Setting	Description
Real-Time File Monitoring	Master gate — when off, no incoming events can trigger alerts.
Email Alerting	Enables/disables outgoing alert emails globally.
High Severity Threshold	How many high-severity events per host before an alert fires: Immediate, After 1, or After 3.
Auto Agent Discovery	Whether the Agents page polls for live status automatically.
Agent Offline Threshold	How long without an event before an agent is marked Offline.
Dashboard Refresh Interval	How often dashboard/agent pages poll for updates.

7.2 Risk Rules (admin-editable, no code changes)

- Default severity by event type (created / modified / deleted / moved).
- Sensitive-path glob patterns (e.g. *.pem, *shadow, *wp-config.php) with an assigned severity.
- Permission-change severity and the burst frequency rule (events per host per time window).
- Critical-hosts list for tagging high-value assets.

7.3 Engine Configuration

Admin-editable weights and thresholds for how the five detection stages combine into a final score — adjustable without redeploying agents.

8. REST API Reference

Endpoint	Method	Purpose
/log	POST	Agent event ingestion. Requires X-API-Key header (per-agent key; the legacy shared key is no longer accepted).
/health	GET	Server identity/discovery proof — used by agents to verify the server.
/logs	GET	Recent stored events (session-authenticated).
/api/stream	GET	Server-Sent Events stream for live dashboard updates.
/api/agents	GET	Aggregated per-host status (Online/Offline, event counts).
/api/stats	GET	Dashboard KPI and chart data.
/api/incidents	GET	Recent correlated incidents from the correlation engine.
/api/risk-rules	GET / POST	Read or patch the rule-scoring configuration.
/api/engine-config	GET / POST	Read or patch detection-engine weights and thresholds.
/api/settings	GET / POST	Read or patch dashboard/alerting settings.
/api/generate-agent/windows-exe	POST	Builds and downloads a Windows agent deployment package with its own per-agent API key.
/api/generate-agent/linux-bin	POST	Builds and downloads a Linux agent deployment package with its own per-agent API key.
/api/agent-keys	GET	Lists issued per-agent API keys (label, created, last-used, revoked) — never the key itself.
/api/agent-keys/<key_id>/revoke	POST	Revokes one agent's key without affecting any other agent.
/api/email-logs	GET	Full email-alert delivery history.

All /api/* routes and the dashboard pages require an authenticated session. /log authenticates separately via API key, since agents cannot perform an interactive login.

9. Troubleshooting

Problem	Likely Cause	Solution
Agent shows Offline	No event received within the offline threshold	Confirm the endpoint is running and can reach the server URL/port
Email alerts not received	No alert_email on the agent, throttled, or Gmail credentials not configured	Redeploy the agent with a valid focal-person email; confirm FALYS_GMAIL_SENDER/FALYS_GMAIL_APP_PASSWORD are set; check Event History's email panel
User shows Unverified	OS-level audit trail not configured on that endpoint	Run setup_audit_windows.ps1 or setup_audit_linux.sh on that machine
Agent can't find server	Server IP changed and discovery hasn't completed yet	Confirm the discovery UDP port is open on your firewall (see your administrator for the exact port); events queue locally until the server is found
Browser shows a certificate warning	Expected — the certificate is self-signed for a LAN-only deployment	Proceed past the warning, or install a certificate from your organization's internal CA instead
Linux agent or server binary won’t start	Older glibc than the build machine	Rebuild falys-agent or falys-server on the oldest supported distro version
Agent gets HTTP 401 from server	Agent was deployed before per-agent API keys existed; the legacy shared key it was using has since been removed entirely	Redeploy this agent from the dashboard’s Agents page to receive a fresh per-agent key
Server ignores an .env file placed next to it	A frozen/compiled server binary resolves its own real folder correctly, but always double-check the .env file sits directly next to the .exe / binary, not in a subfolder	Confirm the .env file's location, or set the same values as real environment variables instead (setx on Windows) — this always works regardless of where .env is placed

10. FAQ & Glossary

10.1 Frequently Asked Questions

Q: Does FALYS need Python installed on endpoints? A: No, deployed agents are compiled, standalone binaries.

Q: What happens if the network drops? A: Events are queued locally (queue.json) and flushed automatically once connectivity returns.

Q: Can each site have its own alert recipient? A: Yes, via the per-agent alert_email set at deployment time.

Q: How is 'confidence' different from the risk score? A: The score says how risky an event looks; confidence says how much history/agreement backs that call.

Q: Can detection sensitivity be tuned without redeploying agents? A: Yes, all rule and engine weights are admin-editable server-side.

Q: Is the connection between agents and the server encrypted? A: Yes, by default. Agents connect over HTTPS and pin the server's specific certificate rather than trusting any certificate a Certificate Authority has signed.

Q: What happens if an endpoint is compromised? A: Revoke that specific agent's API key from the dashboard — no other agent is affected.

Appendix A — FALYS Project Reference

A.1 Developed By

Sheeza Alam Khan and Maimoona Shah Khan

10.2 Glossary

Term	Definition
Attribution	Resolving which OS account actually performed a file change, via auditd or the Windows Security log.
Correlation Window	The sliding time window in which related events are grouped into one incident.
Incident	A correlated group of related events (e.g. a burst, or a staged tampering pattern).
Focal Person	The specific recipient assigned to receive alerts for one particular agent.
Baseline Maturity	How much historical data backs a host's behavioral anomaly score.
Certificate Pinning	Trusting one specific, known certificate directly instead of validating against a public Certificate Authority chain.
Replay Attack	Re-sending a previously captured, legitimate message to trick a system into acting on it again.

HOW FALYS WORKS

01 **Agent Detects Change**
The FALYS agent monitors files and folders in real-time and detects any change or suspicious activity.

02 **Computes Hash / Permission / Attribution**
It computes the file hash, checks permissions, and captures attribution details of the change.

03 **POSTs JSON to /log over HTTPS with its per-agent API key**
The agent securely sends the event data to the server using HTTPS and its unique per-agent API key.

04 **Server Runs the Detection Engine**
The server analyzes the event using intelligent rules and behavior analysis to determine its severity.

05 **Event Stored (Hash-Chained Day-Shared Log) and Broadcast to Live Dashboards**
The event is securely stored in a hash-chained, day-sharded log and instantly broadcast to all connected dashboards.

06 **Alert Dispatched if the Severity Threshold is Met**
If the event meets or exceeds the configured severity threshold, an email alert is sent to the configured recipients.

Always Watching. Always Protecting.
FALYS ensures the integrity of your critical files and gives you the visibility and control you need to stay secure—always.

EVENT LOG & INTEGRITY

Hash-Chained Logs
Each event is cryptographically linked to the previous one, ensuring tamper-proof integrity.

Day-Shared Storage
Logs are stored in daily shards for better performance, scalability, and easy management.

Real-Time Dashboard
All events are displayed in real-time on a unified dashboard with advanced filtering and search.

SECURITY & TRUST

Per-Agent API Key
Each agent has a unique API key for secure identification and data submission.

HTTPS Encryption
All data is transmitted over HTTPS to ensure end-to-end encryption.

Tamper-Proof Logging
Hash-chained logs ensure that any manipulation is immediately detectable.

USE CASES

Ransomware Detection
Detect unauthorized file modifications and deletions.

Insider Threat Monitoring
Identify suspicious activities by internal users.

Compliance & Audit
Maintain file integrity and meet security compliance requirements.

Critical Data Protection
Protect sensitive files and directories from unauthorized changes.

FEATURES AT A GLANCE

Real-Time Monitoring	Monitor file system events in real-time including create, modify, delete, move, and permission changes.
Threat Detection	Detect suspicious and malicious activities using smart rule engine and behavior analysis.
Email Alerts	Instant email notifications for critical events and policy violations.
Cloud & On-Premise	Deploy on-premise or use our secure cloud for centralized management.
Centralized Dashboard	Unified dashboard for logs, alerts, reports, and device management.
Easy Deployment	Lightweight agent with simple setup for any environment.
Enterprise Ready	Scalable, secure and reliable solution for organizations of all sizes.

BUILT FOR ENTERPRISES. DESIGNED FOR SECURITY.

FALYS is a File Integrity Monitoring (FIM) solution that helps organizations detect, respond, and recover from threats—before damage occurs.

