# Software Test Plan (STP)

**Project:** System Performance Tracker (Linux/Unix) — Test Plan v1.0

**Version:** 1.0  
**Authors:** Karan M, K Adithya Shetty, Pavan Kumar G, Purandar  
**Date:** 08-09-2025  
**Status:** Sample / Draft

## 1. Introduction

Purpose: This document defines the test plan for System Performance Tracker v1.0. It outlines objectives, scope, strategy, resources, schedule, and responsibilities for testing a desktop/terminal-based application designed to monitor and analyze Linux/Unix system performance in real-time.

Scope: Testing covers all core monitoring features including CPU usage tracking (per-core), memory utilization, disk I/O statistics, network monitoring, process management, authentication/privilege validation, alerting mechanisms, and data export (CSV/JSON). Excluded: kernel internals, third-party hardware firmware, and distributed network infrastructure beyond the local host.

References:

* System Performance Tracker SRS v1.0
* Design Specifications v1.0
* Linux Performance Monitoring Tools & Best Practices (top, vmstat, iostat, mpstat, pidstat, /proc)
* Security Best Practices for Unix-like systems

Definitions: SPT (System Performance Tracker), CPU (Central Processing Unit), HDD (Hard Disk Drive), PID (Process Identifier), PPID (Parent Process Identifier), TTY (Teletypewriter), CLI (Command Line Interface), TUI (Terminal User Interface), NFR (Non-Functional Requirement).

## 2. Test Items

* Authentication and privilege validation module
* CPU monitoring module (per-core usage tracking)
* Memory monitoring module (RAM, swap, cache utilization)
* Disk monitoring module (usage and I/O statistics)
* Network monitoring module (bandwidth, packets)
* Process management module (listing, sorting, termination)
* Alerts and threshold management module
* Configuration management module
* Data export module (CSV/JSON)
* User interface module (interactive TUI and CLI modes)
* Logging & audit module (structured logs, rotation, retention)
* Security enforcement module (RBAC, sanitization)

## 3. Features to be Tested

Features mapped to SRS requirement IDs:

**Authentication & Security:**

* SPT-F-001: Validate admin users via sudo
* SPT-F-002: Restrict access to privileged metrics
* SPT-F-003: Deny insufficient privileges with error message
* SPT-F-004: Log privileged access attempts
* SPT-SR-001: Role-based access control for process termination for privileged users
* SPT-SR-002: Sanitize configuration input to prevent buffer overflow or injection

attacks

* SPT-SR-003: Append-only audit logs
* SPT-SR-004: Secure default file permissions
* SPT-SR-005: Protect sensitive environment variables

**Core Monitoring:**

* SPT-F-005: Display real-time CPU usage per core
* SPT-F-006: Display memory utilization (used, free, cached, swap)
* SPT-F-007: Display disk usage and I/O statistics
* SPT-F-008: Display network utilization (bandwidth, packets)

**Process Management:**

* SPT-F-009: List running processes with PID, CPU%, Memory%
* SPT-F-010: Allow sorting of processes by various criteria

**Alerts & Configuration:**

* SPT-F-011: Trigger CPU threshold (85%) warnings
* SPT-F-012: Trigger memory threshold warnings
* SPT-F-013: Configure custom thresholds via config file

**Usability & Export:**

* SPT-F-014: Provide interactive and non-interactive modes
* SPT-F-015: Export monitoring data in CSV/JSON format

**Non-Functional Requirements:**

* SPT-NF-001: Metrics update latency ≤1 second
* SPT-NF-002: System recovery within 5 seconds after transient faults
* SPT-NF-003: Privileged metrics require administrative permission
* SPT-NF-004: Structured logging with rotation
* SPT-NF-005: UI accessibility and scaling support

## 4. Features Not to be Tested

* OS kernel internals and vendor kernel module source code
* Third-party hardware vendor firmware
* External network infrastructure beyond the local host (routers, switches)
* Hardware failures like catastrophic disk failure or power loss (only simulated behavior)

## 5. Test Approach / Strategy

**Testing Levels:**

* Unit tests (individual module testing)
* Integration tests (module interactions)
* System tests (end-to-end functionality)
* User acceptance tests (UAT)

**Testing Types:**

* Functional testing (all core features)
* Security testing (privilege validation, input sanitization)
* Performance testing (response time, resource overhead)
* Usability testing (UI clarity, accessibility)
* Compatibility testing (selected Linux distros / kernel versions)
* Stress testing (high system load scenarios)
* Regression testing

**Entry Criteria:**

* Stable build delivered with all modules integrated
* Test environment(s) configured (listed in Section 6)
* Test data and user accounts prepared
* Testing tools installed and configured

**Exit Criteria:**

* >=90% of planned test cases executed
* 0 critical defects open
* All acceptance criteria satisfied
* Performance benchmarks met
* Security requirements validated

## 5.1 Security Validation

* Validate privilege escalation mechanisms (sudo/admin rights)
* Verify role-based access controls for sensitive operations
* Test input sanitization for configuration files
* Validate audit logging for privileged actions
* Test protection of sensitive process information
* Verify secure file permissions for configuration files
* Penetration testing for privilege bypass attempts

## 6. Test Environment

**Hardware:**

* Linux machines with varying specifications
* Multi-core processors for CPU monitoring tests
* Systems with different memory configurations (4GB, 8GB, 16GB+)
* Various disk types (HDD, SSD, NVMe)
* Network-connected systems for network monitoring tests

**Software:**

* Ubuntu 22.04 LTS (kernel 5.15)
* System Performance Tracker application v1.0
* Administrative and standard user accounts

**Tools:**

* Automated testing framework for CLI applications
* Process monitoring tools (Task Manager, Resource Monitor)
* Network simulation tools
* Log analysis tools
* Performance benchmarking utilities
* Security testing tools

**Test Data:**

* Configuration files with various threshold settings
* Sample processes for testing process management
* Network traffic generators for monitoring tests
* System load generators for stress testing

## 7. Test Schedule

Milestones:  
- Test case design completion: 29-Sep-2025

- Test environment setup: 7-Oct-2025

- Unit testing: 14-Oct-2025 to 21-Oct-2025

- Integration testing: 21-Oct-2025 to 28-Oct-2025

- System testing: 5-Nov-2025 to 10-Nov-2025

- Security testing: 10-Nov-2025 to 15-Nov-2025

- Performance testing: 15-Nov-2025 to 20-Nov-2025

- UAT: 20-Nov-2025 to 22-Nov-2025

- Final validation and sign-off: 22-Nov-2025

## 8. Test Deliverables

* Test Plan (this document)
* Test Cases (manual and automated)
* Test Scripts and automation frameworks
* Test Data sets and configuration files
* Test Execution Reports
* Defect Reports and tracking logs
* Performance Test Results
* Security Assessment Report
* Test Summary Report
* Requirements Traceability Matrix (RTM)

## 9. Roles and Responsibilities

| Role | Name | Responsibility |
| --- | --- | --- |
| QA Lead | karan | Prepare plan, coordinate execution |
| Test Engineer | adithya | Design & execute test cases, log defects |
| Developer | pavan | Support defect fixes and triage |
| Product Owner | purandar | Approve test results, sign-off readiness |

## 10. Risks and Mitigation

| **Risk** | **Mitigation** |
| --- | --- |
| Inconsistent behavior across distros / kernel versions | Early compatibility matrix testing; run automated tests in containers for each target distro |
| Performance testing affected by noisy host | Use dedicated test machines or isolated VMs; baseline measurements first |
| Security testing requires elevated privileges | Use isolated test lab & coordinated access; sandboxed tests |
| Network tests affected by external conditions | Use simulated network environments (tc/netem) and local loopback tests |
| Long retention scenarios (90 days) not testable in real time | Use time-shift simulation or retention configuration tests to validate rotation logic |

## 11. Assumptions & Dependencies

* systems with sufficient privileges for testing will be available
* Test systems will have stable hardware configurations during testing
* Administrative access will be provided for privilege testing
* Network connectivity will be available for network monitoring tests
* Development team will provide timely fixes for critical defects

## 12. Suspension & Resumption Criteria

**Suspend testing if:**

* Test environment unavailable for >6 hours
* Build is too unstable (blocks >40% of test cases)
* Critical security vulnerability discovered
* Major architectural changes required

**Resume testing if:**

* Blocking defects are resolved and verified
* Test environment is restored and validated
* Updated build passes smoke tests
* Security issues are addressed

## 13. Test Case Management & Traceability

Requirements Traceability Matrix (RTM) ensures mapping of SRS requirements to test cases.

**Examples:**

* SPT-F-001 (Admin validation) → TC-Auth-01, TC-Auth-02, TC-Auth-03
* SPT-F-005 (CPU monitoring) → TC-Mon-01, TC-Mon-02, TC-Mon-03
* SPT-F-009 (Process listing) → TC-Proc-01, TC-Proc-02
* SPT-NF-001 (Performance) → TC-Perf-01, TC-Perf-02
* SPT-SR-001 (Role-based access) → TC-Sec-01, TC-Sec-02

## 14. Test Metrics & Reporting

**Metrics Collected:**

* Test case execution rate (% executed vs planned)
* Pass/fail rates by module and priority
* Defect density by module
* Defect resolution time and aging
* Requirements coverage percentage
* Performance benchmark results
* Security test results

**Reports:**

* Daily test execution status
* Weekly progress reports
* Performance benchmark reports
* Security assessment results
* Final Test Summary Report with recommendations

## 15. Approvals

| Role | Name | Signature / Date |
| --- | --- | --- |
| QA Lead |  |  |
| Dev Lead |  |  |
| Product Owner |  |  |