WHITE PAPER:
Wire-Level Validation of Storage Security

Delivering Auditor-Ready Proof for Storage Security Compliance

Summary

Secure data removal and encryption-at-rest are critical for protecting sensitive information and meeting compliance mandates. Features like SANITIZE and Self-Encrypting Drives (SED) based on TCG standards ensure that data cannot be recovered once a device leaves active service. These capabilities are essential for preventing data breaches, safeguarding intellectual property, and satisfying regulatory requirements such as NIST SP 800-88 and FIPS 140-3.

While these features are powerful, proving they work as intended is equally important. Traditional validation methods (host utilities, firmware logs, side-channel benches) confirm only intent—not the actual protocol exchanges on the wire. Wire-level captures and targeted fault injection provide auditor-ready evidence that the commanded method (e.g., Overwrite, Block Erase, Crypto Erase) and control bits (IMMED, AUSE) executed correctly, including error paths and retries.

This white paper first explains the technologies (SAS/SSP transport, SCSI SANITIZE, TCG SED flows), then identifies validation challenges, and finally shows how Teledyne LeCroy's analyzer + exerciser + jammer approach delivers repeatable, auditor-ready proof aligned with NIST SP 800-88 media sanitization guidance, TCG Storage specifications, and CMVP/FIPS evaluation practices.

Technology Primer

1.1 SAS/SSP Transport in Brief

Serial Attached SCSI (SAS) transports SCSI Information Units over SSP frames: COMMAND, XFER_RDY, DATA, and RESPONSE govern ordering, retries, and timing end-to-end. An analyzer observing these IUs is the authoritative record of what actually occurred on the link.

1.2 SCSI SANITIZE Command (opcode 0x48)

SANITIZE is defined in SCSI Block Commands with method/service-action semantics and flags that alter execution and reporting. Common methods include Overwrite, Block Erase, Crypto Erase, and Exit Failure Mode; flags such as IMMED and AUSE control immediate return and allow unrestricted sanitize. Host tools map user options to CDB fields—but only a protocol analyzer proves the CDB bytes carried on the wire and how the target reported progress/completion.

1.3 TCG SED Flows over SAS/SATA

Self-Encrypting Drive (SED) operations—ownership, authentication, ranges/bands, lock/unlock, revert—are carried via SECURITY PROTOCOL IN/OUT exchanges. Wire-level decoding validates payload fields, sequence integrity, and device responses beyond what SDK logs alone can show.

Why it Matters: Compliance and Risk Context

Auditor expectations: Certifiers (e.g., CMVP/FIPS) rely on functional evidence and documentation. Timestamped, decoded on wire traces increase confidence and reduce review cycles by proving exact command bytes, method/flag state, error handling, and completion status.

NIST SP 800 88: Requires rigorous media sanitization consistent with organization policy and device capabilities. Wire level traces show precisely which method the drive executed and how it reported status.

TCG Enterprise/Opal: Defines SED behaviors—including authentication, band configuration, and revert—that must be verifiably correct on the link.

Evidence package: Captures, decodes, and configuration logs demonstrate controls actually exercised on the device—not just intended by host software—supporting CMVP/Common Criteria engagements.

The Real-World Validation Challenges

Host utilities like sg_sanitize and vendor SDKs report what they asked the device to do; they cannot prove CDB layout, IU ordering, transport-layer retries, or CRC anomalies. Without on-wire visibility, sense transitions and failure handling (e.g., CHECK CONDITION for unsupported methods) are easy to misinterpret, delaying debug and audits. Some devices don't fully implement opcode discovery; capturing the actual SANITIZE frames is often the most reliable way to confirm capability and behavior.

Solution Overview - Analyzer + Exerciser + Jammer

Objective: Deliver auditor-ready proof of SANITIZE and SED behaviors while enabling negative/pathological testing to validate robustness and compliance.

Architecture:

• Protocol Analyzer captures SAS/SSP frames and decodes SANITIZE, SECURITY PROTOCOL IN/OUT, sense data, and timing.
• Exerciser issues repeatable SANITIZE and TCG sequences, including workload/background traffic to emulate realistic conditions.
• Jammer injects controlled faults (e.g., bit flips, CRC errors, field changes) to validate device error handling and standards-conformant responses.

Automation: Tie analyzer bookmarks, exerciser scripts, and jammer configurations into a single repeatable playbook for regression and certification evidence packages.

Repeatable Test Workflows

SANITIZE Positive Paths:

- Crypto Erase with IMMED=1. Capture COMMAND IU (CDB[0]=0x48), progress polling, and RESPONSE IUs; verify completion and sense transitions.
- Overwrite (count = 1) with IMMED=1. Confirm method execution, device reporting, and completion status on the link.

SANITIZE Negative/Pathological:

- Flip IMMED to 0 in-flight to validate blocking behavior and matching host/device responses.
- Request unsupported method (e.g., Block Erase on HDD) and expect CHECK CONDITION with appropriate sense (e.g., ILLEGAL REQUEST).
- Inject CRC error on COMMAND IU and confirm transport retries or failure path and error logging.

TCG Enterprise/Opal Flows:

- Ownership & Authentication: Use SECURITY PROTOCOL IN/OUT to establish credentials; capture payload fields and sequence for audit.
- Range Lock/Unlock & Revert: Exercise band configuration and transitions; verify protocol replies and sense codes.
- Fault Injection: Malform payloads (e.g., length/nonce) to validate device error responses (e.g., AUTHORITY_ LOCKED_OUT or ILLEGAL REQUEST).

Auditor-Ready Deliverables

- **Trace Package:** Timestamped SAS captures showing COMMAND / XFER_RDY / DATA / RESPONSE information units with annotated CDB fields (e.g., CDB[0]=0x48, method selection, IMMED/AUSE flags), retries, sense data, and completion status.
- **Decoded Evidence Views:** Protocol-decoded trace views suitable for audit review, including SANITIZE and SECURITY PROTOCOL IN/OUT exchanges, command sequencing, and sense transitions. Trace data may be exported (e.g., CSV/Excel) for external reporting, documentation, or customer-specific audit packages.
- **Configuration Log:** Exerciser and jammer configuration details, including command parameters, fault injection settings, background traffic conditions, and expected vs. observed outcomes.
- **Standards Mapping:** Cross-reference of observed on-wire behavior to relevant specifications and guidance, such as NIST SP 800-88 sanitization methods, applicable SPC/SBC SANITIZE sections, and TCG Enterprise/Opal command flows. Mapping focuses on substantiating claims with captured protocol evidence rather than inferred host-level intent.
- **Repeatable Test Workflows:** Defined, repeatable test sequences that can be re-executed across firmware revisions, platforms, or devices to demonstrate consistent behavior. These workflows may include both positive and negative test conditions and can be extended to support customer-, audit-, or compliance-driven validation needs.

Business Impact

Wire-level evidence helps mitigate audit disputes and late-stage rework by demonstrating actual on-wire execution rather than inferred intent. Defensible protocol captures can reduce back-and-forth with certification labs during CMVP or Common Criteria evaluations. Reproducible negative and fault-injection tests shorten debug cycles for firmware and driver interoperability issues and strengthen confidence in storage-security implementations.

Practical Notes and Known Variations

- **Opcode discovery caveat:** REPORT SUPPORTED OPERATION CODES may be incomplete; capture real SANITIZE exchanges for certainty.
- **Host-tool convenience vs. wire truth:** Utilities (e.g., sg_sanitize) map UI options to CDB fields; only analyzers confirm bytes, IUs, and transport details.
- **Media sensitivity:** SSDs often support Block Erase/Crypto; HDDs typically support Overwrite—use jammer to force mismatches and validate error handling.

Conclusion

Security features like SANITIZE and SED/TCG are only as strong as their on-wire execution. Teledyne LeCroy's analyzers, exercisers, and jammers provide the missing proof layer—turning compliance assertions into auditor-ready evidence. As storage vendors and hyperscalers scale secure workflows, wire-level validation becomes standard operating procedure.

Appendix A - SANITIZE CDB (Quick Field Map)

Byte 0:
0x48 — SANITIZE opcode.
Byte 1:
Bits 7–6: SANITIZE flags
AUSE (bit 7): Allow Unrestricted Sanitize Exit
IMMED (bit 6): Return status immediately; sanitize runs in background
Bits 5–0: SANITIZE Method
0x00 — Overwrite
0x01 — Block Erase
0x02 — Crypto Erase
0x1F — Exit Failure Mode
Bytes 2–31:
Additional method specific parameters (overwrite count, pattern length, pattern data, etc.).

References

1. UNH-IOL: Serial Attached SCSI (SAS) Clause 9 — SSP frame types and transport behavior. https://www.iol.unh.edu/sites/default/files/knowledgebase/sas/SAS_Clause_9.ppt
2. Seagate: Serial Attached SCSI (SAS) Interface Manual, Publication 100293071 Rev. C. https://www.seagate.com/staticfiles/support/disc/manuals/Interface%20manuals/100293071c.pdf
3. sg3_utils: sg_sanitize man page — SANITIZE command semantics and options. https://man.archlinux.org/man/sg_sanitize.8.en
4. Oracle Solaris: sg_sanitize man page. https://docs.oracle.com/cd/E88353_01/html/E72487/sg-sanitize-8.html
5. Seagate TCGstorageAPI — TCG Enterprise/Opal operations over SAS/SATA. https://github.com/Seagate/TCGstorageAPI
6. TCG Storage: Enterprise SSC Specification (v1.01 r1.00). https://trustedcomputinggroup.org/wp-content/uploads/TCG_Storage-SSC_Enterprise-v1.01_r1.00.pdf
7. NIST CMVP: Program overview and validated modules database. https://csrc.nist.gov/Projects/Cryptographic-Module-Validation-Program/Validated-Modules
8. SBC-3: SCSI Block Commands — SANITIZE command definition. https://www.t10.org/ftp/t10/document.05/05-344r0.pdf
9. FIPS 140-3: Security Requirements for Cryptographic Modules. https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.140-3.pdf
10. NIST CMVP: Implementation Guidance for FIPS 140-3. https://csrc.nist.gov/CSRC/media/Projects/cryptographic-module-validation-program/documents/fips%20140-3/FIPS%20140-3%20IG.pdf