DRACO7RT · FLIGHT SOFTWARE · SECURITY · 28 OCTOBER 2026

Flight Software Security

What an attacker can do to the flight software, what the vehicle already detects and refuses, how a cyber cop on board (SpaceCOP) works, and what DRACO7RT still needs

The vehicle is real cFS on RTEMS. Its lab uplink accepts any sender, so today the whole defense is on board: plausibility gates on GPS and IMU input, the Limit Checker and Stored Commands as the autonomous response, health and checksum apps, and, on the NOS3 twin, the Aerospace Corporation's SpaceCOP intrusion-detection app firing SPARTA indicators of behavior, sharing them with a sister vehicle and entering a cyber-safe mode on its own.

Slides 2 to 5: the threat, the attack surface of this vehicle, the SPARTA framework and the standards. Slides 6 to 10: the cyber cop, its 24 rules, the detect-classify-respond chain, the 14 attack scenarios and their evidence. Slides 11 to 14: what DRACO7RT defends today, authenticated commanding, software integrity, the telemetry a cyber sensor needs. Slides 15 and 16: the ordered roadmap and the rules.

THE THREAT TO FLIGHT SOFTWARE

What policy says must be protected, and what the attack frameworks say an adversary does on board

Space Policy Directive-5 (4 September 2020), section 4(b)

- Plans "to retain or recover positive control of space vehicles"
- (i) "Protection against unauthorized access to critical space vehicle functions … safeguarding command, control, and telemetry links using effective and validated authentication or encryption measures"
- (iii) "Protection against communications jamming and spoofing, such as signal strength monitoring programs, secured transmitters and receivers, authentication, or effective, validated, and tested encryption measures"
- (v) "intrusion detection methodologies for system elements such as information systems, antennas, terminals, receivers, routers"
- (vi) supply-chain risk management: trusted suppliers, counterfeit and malicious equipment

What the adversary does on board (SPARTA v4.0.1 techniques)

- **EX-0001 Replay**: "re-transmission of previously captured traffic, over RF links, crosslinks, or internal buses"; command packets and bus traffic
- **EX-0012 Modify On-Board Values**: "direct memory read/write commands, table load services, file transfers, or maintenance procedures … can be invoked to steer behavior without changing code"; 13 sub-techniques incl. memory loads, app tables, ADCS, system clock
- **EX-0013 Flooding**: valid commands, erroneous input
- **EX-0014 Spoofing**: "forges inputs that subsystems treat as trustworthy truth, time tags, sensor measurements, bus messages, or navigation signals"; **.04 PNT Spoofing**: "externally provided navigation/time that passes validity checks yet encodes a crafted trajectory or epoch"
- **EX-0008 Time Synchronized Execution**: absolute and relative time sequences (ATS and RTS abuse)
- **EX-0004 / EX-0005 / EX-0010**: boot memory, hardware commands, malicious code; **IA-0007.01** compromised on-orbit update

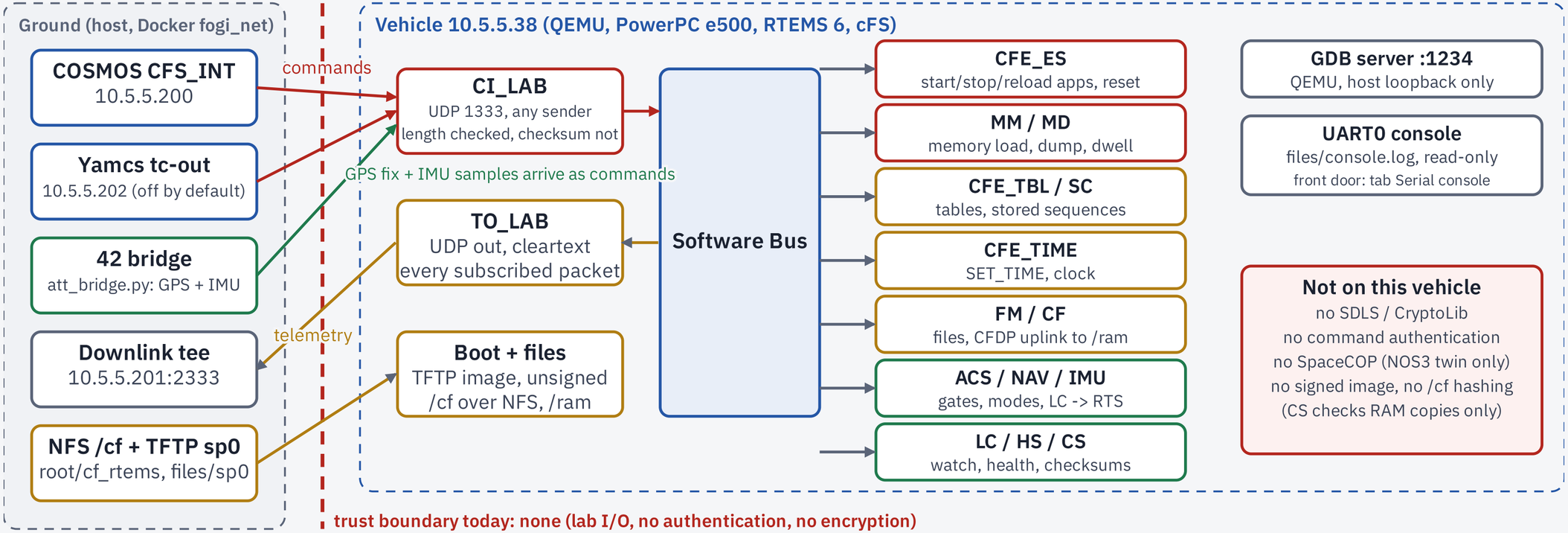
Why on-board defense at all

- Aerospace TOR-2021-01333 (April 2021): "radio frequency replay attacks are one of the largest threat vectors to the spacecraft and should always be mitigated"; "little to no security-focused capabilities for on-board monitoring, logging, and alerting"
- ESA OPS-SAT (2023): an uploaded app with a flaw reached the control layer and rotated the spacecraft
- The ground can be wrong, late or compromised; the vehicle must validate what it is told and notice what it does

Every technique on the right maps onto this vehicle: replay and flooding on the CI_LAB uplink; on-board values through MM, CFE_TBL, SC and CFE_TIME; spoofing through the GPS fix and IMU sample commands the 42 bridge uses; boot memory through the unsigned TFTP image and /cf. Slide 3 draws it.

THE ATTACK SURFACE OF THIS VEHICLE

Everything behind the uplink is reachable by whoever can send a UDP packet to port 1333



Measured 28 October 2026: CI_LAB rejects a bad length but not a bad cFE checksum (only SC checks, for stored commands). Console read-only (front door › Demo control › Serial console); GDB on host loopback. On the bus, every command is trusted.

SPARTA: THE FRAMEWORK BEHIND THE COP

Tactics and techniques for the attack, countermeasures for the design, indicators of behavior for the detector

The matrix (v4.0.1, 24 August 2026)

- Space Attack Research and Tactic Analysis, the Aerospace Corporation, approved for public release (OTR 2022-01250)
- 9 tactics: Reconnaissance 9, Resource Development 5, Initial Access 13, Execution 18, Persistence 5, Defense Evasion 12, Lateral Movement 7, Exfiltration 10, Impact 8 = **87 techniques**, 133 sub-techniques (tallied from the matrix page)
- Each technique carries a notional risk score (H | M | L), e.g. EX-0001 Replay 25 | 24 | 21
- v4.0 (DEF CON, August 2026) added ground-segment guidance and revised every countermeasure

Countermeasures (CM0001 to CM0090)

- Each: Tier I or II, **Onboard SV** or **Ground**, NIST SP 800-53 Rev 5 and ISO 27001 mappings, sample requirements
- Defense-in-depth view from TOR-2021-01333: Prevention → Ground → Comms Link → **Crypto** (authentication, encryption, bypass) → **IDS/IPS** (detection, fault-management integration, machine learning, cyber-safe mode) → **SBC/Bus/Processor** (command validation, memory protection, root of trust, bus segregation, logging, least privilege) → **S/C Software** (reproducible builds, secure coding, static and dynamic testing, code signing) → Data
- **CM0014 Secure boot** (Tier I, onboard): "a cryptographically verified chain of trust from a hardware-anchored root of trust … to the operating system or flight software image", anti-rollback, radiation-qualified

Indicators of behavior (launched April 2025)

- "IOCs are evidence of KNOWN threats … where IOBs are patterns of suspicious or malicious behavior"; about 200 IOBs in 10 categories, each with a STIX pattern and the techniques it detects; funded by DHS Science and Technology
- Categories: UACE unauthorized command execution (26), UCEB crypto bypass (11), CSNE comms and network (45), ARFS RF and authentication (12), **GNTM GNSS and time (12)**, **MIRE memory (18)**, WTRE watchdog and registers (7), SIUU software integrity and updates (26), **SMSR sensors and resources (19)**, **DISE data and storage (17)**
- GNTM-6 "Unexpected Time Delta Detected": "time should be linear with minimal drift"; GNTM-7 "Time Adjustment Commands Detected"; SMSR-3 "Unauthorized State Changes in Critical Sensors (e.g., Sun Sensor, GPS Sensor)"; DISE-1 "File or Data Integrity Check Failure"
- From the DEF CON 33 slides: "IDS's are useful for seeing if an attack is taking place, not for preventing the attack"

SPARTA cross-references the NASA best-practices guide, NIST 800-53 and D3FEND, so IOB-named detections and countermeasure-named designs land in an existing control vocabulary. The harness labels scenarios by SPARTA tactic; SpaceCOP names its rules by IOB.

THE STANDARDS THAT APPLY TO THE VEHICLE

NASA's protection standard and best-practices guide for the spacecraft, NPR 7150.2D for the software, and this project's record

NASA-STD-1006 (2019; Rev A July 2022)

- **SSPR 1**: "protect the command stack with encryption that meets or exceeds ... FIPS 140"
- **SSPR 2**: an encrypted primary command link means "any backup command link shall, at a minimum, use authentication"
- **SSPR 4**: "resilient to the complete loss of, or temporary interference with, external PNT services"
- Rationale: NASA end-of-mission experiments "found that spacecraft without encryption or authentication are particularly susceptible"

Space Security Best Practices Guide Rev B (October 2023 / January 2024): the 14 spacecraft principles, the ones that bite here

- **MI-DCO-01** Mission Cyber Actor Actions Detection: "incorporate an on-board cyber actor actions detection function in its requirements and resulting system"
- **MI-DCO-02** Fault Management: "fault management systems may be deliberately triggered in an effort to bypass the system's protective measures. For example, safehold mode operations without command-link protection"
- **MI-INTG-02** PNT Survivability: "recover from positioning, navigation, and timing jamming and spoofing attempts"
- **MI-MA-02** Cybersecurity-Safe State: "design secure vehicle fault management functions and safe mode operations"
- **MI-MALW-02** Software, PLD and Firmware Integrity: "establish and verify the integrity of it's software images"
- **MI-AUTH-02** authentication and authorization; **MI-ARCH-02** least privilege on board; **MI-SOFT-02** negative testing, abuse cases

NPR 7150.2D section 3.11 (March 2022)

- **SWE-154** identify cybersecurity risks and mitigations in flight and ground software
- **SWE-156** "software cybersecurity assessment ... including risks posed by the use of COTS, GOTS, MOTS, OSS, or reused software"
- **SWE-157** protect software with communications capabilities per NASA-STD-1006
- **SWE-159** test the mitigations ("automated security static analysis")
- **SWE-207 / SWE-185** secure coding practices, verified by static analysis
- **SWE-210** "requirements for the collection, reporting, and storage of data relating to the detection of adversarial actions": the hook an on-board IDS traces to
- Relief from 3.11 needs the SAISO or the Center CISO

DRACO7RT: Class E, not safety-critical; cppcheck on every change (SWE-185 in spirit); NAV gate = MI-INTG-02, LC → SC = MI-MA-02; SWE-156 "not assessed"; nothing traces to SWE-210, SSPR 1 or MI-MALW-02 yet. IEEE P3349: active project, no published standard.

THE CYBER COP ON BOARD

SpaceCOP: an intrusion-detection app on the cFS Software Bus, from the Aerospace Corporation

What it is

- **SpaceCOP Community Edition**, LGPL-3.0, github.com/the-aerospace-corporation/spacecop-ce; integrated here as a cFS app in the NOS3 twin (branch efsi/spacecop-integration)
- Runs like any other app: subscribes on the Software Bus, reads events and telemetry, has its own command MID 0x18FE and housekeeping
- **Monitors** (registry of callables, one *call_index* each): time controller, CPU and process monitors, file integrity (sha256), flash and memory regions, command monitors, sensor monitors, ML bridge
- **Rule table** spacecop_rule.tbl: 24 rules, each enabled flag, IOB id, mode PERIODIC or TRIGGER, period in ticks, call index, operator and tolerance; a cFE table, so it loads and validates like any other

What it emits

- **EVS 2001** event: [SPACECOP] IOB Detected: SPARTA ID=GNTM-7
- **IDS report** packet MID 0x08F9 (1,024-byte message) and HK 0x08FE: both in the COSMOS dictionary (Viewer › Mission › **SpaceCop**) and the Yamcs MDB
- **STIX 2.1 bundle** written to /cf/cti and, for HIGH severity and above, broadcast on MID 0x09FC so a sister vehicle can act on it
- **Commands**: NOOP, RESET_COUNTERS, CLEAR_CYBER_SAFE (code 2, the operator's recovery)

Why it is a cop, not a firewall

- It does not block: it **observes behaviour** (indicators of behavior, IOBs) against a SPARTA-based catalogue and **raises alarms**
- Response is delegated to the vehicle's own autonomy: an EVS event the Limit Checker and ground see, a stored sequence for cyber-safe mode
- Degrades gracefully: the 6 syscall rules and the process and memory monitors need a kernel module or /dev/mem and are inert on this host; the rest fire
- Mission-specific hooks (spacecop_mission_specific.c) are where a project adds its own telemetry rules: this project added SMSR-4 (IMU rate) and SMSR-5 (magnetometer)

Evidence of a live detection on this host: SC2's console log line 42/2/SPACECOP 2001: [SPACECOP] IOB Detected: SPARTA ID=GNTM-7 after a SET_TIME flood arrived over the crosslink (scenario S12), recorded in docs/sow/task4-multispacecraft-status.md.

THE 24 RULES, BY FAMILY

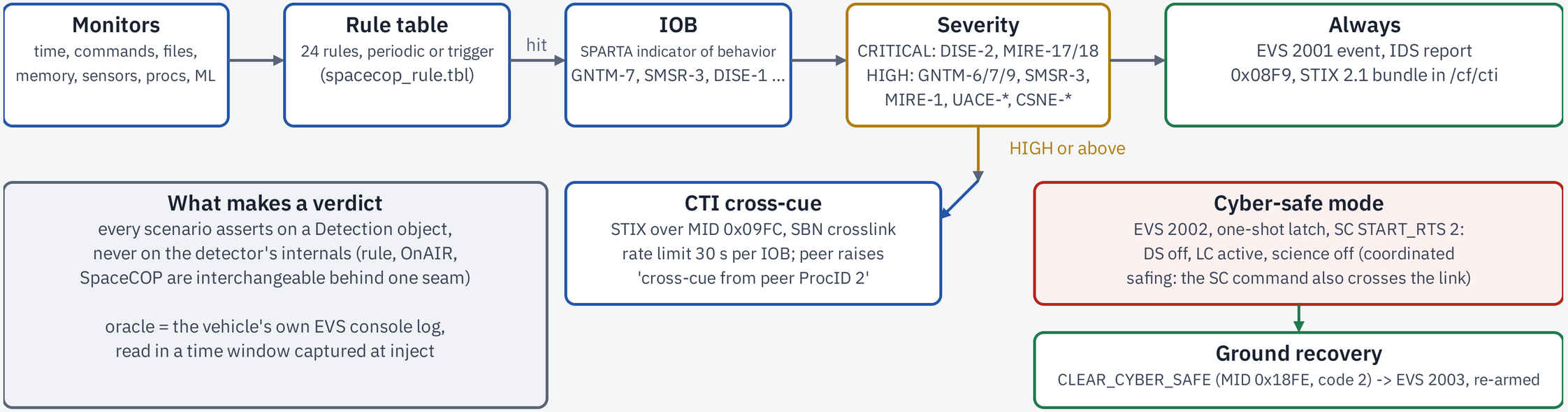
SPARTA indicators of behavior as deployed in spacecop_rule.tbl on the NOS3 twin

Family	Covers	Rules in the table	Severity	Status on this host
GNTM	Time	GNTM-6 unexpected time delta (periodic, tolerance 5 s) · GNTM-7 time-adjustment commands detected (SET_TIME flood)	HIGH	fired live: S08, S12
SMSR	Sensors and state	SMSR-2 high CPU · SMSR-3 unauthorized state change in critical sensors (app stop/start/reload) · SMSR-4 IMU rate > 5 deg/s · SMSR-5 magnetometer > 80,000 nT (both added here)	HIGH (SMSR-3)	fired live: S04, S05
DISE	Data and software integrity	DISE-1 file or data integrity check failure (sha256) · DISE-2 boot or binary integrity (deferred, D01) · DISE-4 storage exhaustion	CRITICAL (DISE-2)	fired live: DISE-1
MIRE	Memory	MIRE-1 write to payload memory · MIRE-14 abnormal memory use · MIRE-15/16 payload and boot regions · MIRE-17 flash open · MIRE-18 error detection failed in critical memory	CRITICAL (17, 18)	inert: needs /dev/mem
SIUU	Suspicious system use	SIUU-10 priority change · SIUU-11 suspicious binary · SIUU-12 kernel module load · SIUU-15 /dev/zero,/null access · SIUU-16 system commands	MEDIUM	inert: needs aerospace.ko
UACE	Unauthorized command execution	UACE-7 duplicate command packets (replay) · UACE-9 valid command flooding · UACE-16 irregular attitude-control maneuver commands · UACE-17 abnormal burn duration	HIGH (family)	S01, S02 via rule detector
CSNE	Comms and network	CSNE-17 FIFO created for exfiltration or injection · CSNE-41 high downlink latency	HIGH (family)	S09 via rule detector

Severity (spacecop_iob_severity): CRITICAL = DISE-2, MIRE-17, MIRE-18; HIGH = GNTM-6/7/9, SMSR-3, MIRE-1, every UACE and CSNE; else MEDIUM. HIGH and above gates cross-cue and cyber-safe entry. Source: ~/nos3/components/spacecop.

DETECT, CLASSIFY, RESPOND

One path from a rule hit to a safing sequence, with the operator holding the key to come back



Verified live on the twin: entry on GNTM-6/7, latch, ground clear, re-arm; idle baseline raises no cross-cue; one SET_TIME attack = one GNTM-7 and one GNTM-6 cross-cue. R12: response safe and attributable (dedicated RTS, latch, operator key).

FOURTEEN ATTACK SCENARIOS WITH EVIDENCE

cyber-scenarios/: inject one attack, observe one tap, assert on the detection, seal one JSON record

Id	Scenario	SPARTA tactic · IOB	Injection	Detection basis	Result
S01	Command injection	Execution / Initial Access · UACE	CI uplink, unauthorized valid command	counter delta without provenance	PASS
S02	Command replay	Execution (EX-0001) · UACE	resend captured frame, stale sequence	duplicate executed	PASS
S03	Malformed command	Execution · UACE	corrupt length and checksum	length rejected; checksum accepted (finding)	PASS
S04	Invalid telemetry	Defense Evasion / Impact · SMSR	rogue CAN peer, out-of-range IMU	SMSR-4	PASS
S05	Telemetry manipulation	Defense Evasion / Impact · SMSR, DISE	rogue SPI magnetometer pins the field	SMSR-5	PASS 3/3
S06	Configuration modification	Persistence / Defense Evasion · SIUU	cFS table load outside the window	TBL counters	PASS
S08	Timing manipulation	Impact (IMP-0014) · GNTM	NOS Engine clock skew	GNTM-6, GNTM-7	PASS
S09	Communications disruption	Impact / RF · CSNE	stop the radio sim	comms-loss events	PASS
S10	Combined cyber-physical	Defense Evasion + Impact · SMSR	spoof IMU, ADCS reacts vs 42 truth	SMSR + attitude divergence	PASS
S11	Autonomous response	Autonomous Response · hook	stimulus → LC actionpoint → SC RTS	RTS started	PASS
S12	Lateral movement	Lateral Movement · IMP-0014	SET_TIME flood on SC1 crosses SBN to SC2	SC2 executes; SC2 GNTM-7	PASS
S13	CTI cross-cue	Detect / Distributed	SC2 STIX over 0x09FC → SC1	cross-cue from peer ProcID 2	PASS
S14	Cyber-safe mode	Autonomous Response	HIGH IOB → safing RTS 2, latch	EVS 2002, RTS 002 Started	PASS
S07, D01-D03	Memory corruption, boot integrity, HW fault, memory hash	Persistence (PER-0001) · MIRE, DISE-2	needs MM/CS apps or the secure node		DEFERRED

Native stock-STF1 NOS3 stack, no QEMU; detector selectable (rule, onair, spacecop) behind one seam; reset soft or hard per scenario. Full suite under --reset hard: 12/12. S12, S13 need the crosslink. Injection stays outside mission FSW (SOW).

WHAT DRACO7RT DEFENDS TODAY

The vehicle already refuses implausible input and responds on its own; it does not yet know who is talking to it

Input plausibility gates (mission apps)

- **NAV** (nav_gate.c, pure): every GPS fix must pass INVALID, RADIUS, SPEED, JUMP and TIME checks; reject flags 0x01 to 0x10, FixCount, RejectCount, ValidationOn. Limits in NAV.CFG_TBL
- **IMU** (imu_check.c, pure): VALIDITY, TIME, RATE_LIMIT, ACCEL_LIMIT, STUCK, DROPOUT; the MEKF falls back to INTERNAL rate when the IMU is STALE
- **Autonomous response**: LC watchpoints → actionpoints → SC: AP0 → RTS 4 momentum dump, GPS spoof → RTS 6, IMU fault → RTS 7 (pins ACS INTERNAL); the same chain SpaceCOP uses for cyber-safe mode
- Requirements dRT-NAV-001..010, dRT-IMU-001..011; evidence in the closed runs

Health, integrity, configuration

- **HS**: application monitor table has all 32 slots at NOACT (stock), 3 restart actions max; event monitor stock
- **CS**: runs, but its app, tables and memory tables have **no enabled entries** (stock): nothing of ours is checksummed
- **Tables**: every constant is a validated table value; a load is visible in TBL housekeeping and events (configuration monitoring for free)
- **SC** validates the checksum of every stored command before execution; **CI_LAB** validates length only
- **EVS** stream to Yamcs as events (SEVERE for cFE ERROR) and to COSMOS: the detection source

Not defended

- Who sent a command: no authentication, no provenance, no authorization status
- Replay and flooding: no sequence check on the uplink
- MM load, ES stop/start, TBL load, SET_TIME: available to any sender
- Boot image and /cf: unsigned, unhashed
- Downlink: cleartext UDP
- No on-board IDS on the RTEMS vehicle (SpaceCOP is on the twin)

The gates are deliberate: after a 42 restart NAV and IMU reject on purpose (replayed time) and Demo control › 42 start resets them (recipe: notes). Never disable a gate to make a demo green: that rule is in the fsw-change skill because it was once tempting.

AUTHENTICATED COMMANDING ON THE VEHICLE

CCSDS Space Data Link Security through NASA CryptoLib: the protocol, the library, and what it would take here

The protocol

- **CCSDS 355.0-B-2** (July 2022): authentication and/or confidentiality on TC, TM, AOS and USLP transfer frames; "authentication algorithms to ensure the integrity of transmitted data and the authenticity of the data source" plus "sequence numbering to detect the unauthorized replay of previously transmitted data"
- Per Security Association: algorithm, initialization vector, **SA_sequence_number** and **SA_sequence_window**; the receiver drops frames outside the window
- **CCSDS 355.1-B-1** Extended Procedures (February 2020): key management (over-the-air rekey), SA start, stop, rekey, expire, and "Set / Read Anti-Replay Sequence Number" and its window
- ESA's SPACE-SHIELD lists the same mitigation, M2046 anti-replay, against T2019 "Telecommand a Spacecraft"

The library

- **nasa/CryptoLib** v1.5.1 (23 September 2026), NASA Open Source Agreement 1.3: "a software-only solution using ... SDLS-EP to secure communications between a spacecraft running the core Flight System (cFS) and a ground station"
- Crypto_TC_ProcessSecurity on the vehicle, Crypto_TC_ApplySecurity on the ground; TM and AOS the other direction; backends WolfSSL, LibGcrypt, JPL KMC
- NOS3 ships it as a component; it is in the twin's app list; no flight mission is named in its documentation
- Record: 20 CVEs on NVD (2025 and 2026); CVE-2025-46674: before 1.3.2 the Extended Procedures were "a Work in Progress (not intended for use during flight), potentially leading to a keystream oracle". Use a current pinned release

On DRACO7RT

1. Vehicle: CryptoLib built for powerpc-rtems6 with a backend that compiles there (LibGcrypt or WolfSSL), called by CI_LAB (or a thin TC-frame ingest in front of it) before the packet reaches the Software Bus; reject + event + counter on failure
2. Ground: Yamcs encodes TC frames (its command post-processor already adds the cFE checksum); COSMOS commands through Yamcs or a small ground-side CryptoLib service
3. Keys: generated and stored on the ground outside git; one SA for commands, one for telemetry authentication; a rekey procedure
4. Evidence: replay (S02) and injection (S01) rerun against the vehicle must now FAIL to execute and PASS to detect; the mission test still passes end to end

The twin's cryptolib app is wired for the NOS3 radio path, not for CI_LAB; here the clean seam is a TC-frame decoder between the UDP socket and CI_LAB's packet handling, which keeps CI_LAB and TO_LAB stock. This is roadmap step 5 and it is the vehicle half of the ground deck's step 7.

SOFTWARE INTEGRITY: BOOT, MEMORY, UPLOADS

What the standards require, what this vehicle does, and what the public cFS record says about the apps in this build

Required

- **SPARTA CM0014 Secure boot**: chain of trust from a hardware root of trust through each boot stage to the flight image; anti-rollback; references NIST SP 800-193
- **TOR-2021-01333**: "prevent the installation of Flight Software without verification that the component has been digitally signed using a certificate that is recognized and approved by the Program"; boot firmware "must enter a recovery routine upon failing to verify signed data"
- **NASA BPG MI-MALW-02**: verify the integrity of software images; **GR-SOFT-01**: nothing installed unless digitally signed
- **NASA BPG MI-DCO-02**: fault management must not be a bypass

This vehicle

- Boot: U-Boot fetches **files/sp0** by TFTP, unsigned; the start-up script and tables come from **/cf**, an NFS share writable from the host
- Apps statically linked into one image: the dynamic app-start path (CVE-2026-67979, arbitrary code from a shared object on storage) does not apply; **/cf being writable from the host does**
- **CS** (Checksum, stock 2.5.99): CRCs over apps, tables, cFE core, OS, EEPROM and memory regions against a baseline taken at start-up; not cryptographic; **no entries enabled here**
- **MM** (2.5.99) loads and dumps memory by command; **CFE_PSP_MemValidateRange** is in its path (mm_utils.c), the check the 2025 Aquila findings said was missing
- **CF** (3.0.99) receives uploads to /ram; **HS** (2.4.99), **DS** (2.6.99) run

The public cFS record (NVD, read 28 October 2026)

- cFS "Aquila" assessment (VisionSpace, March 2025): MM address validation and path traversal; **CVE-2025-25372** (7.5) MM crash, **CVE-2025-25373** (9.8) MM to remote code execution
- 2026, cFS 7.0.x: **CVE-2026-15352 / 18064** HS crash and processor reset on a housekeeping request (CISA ICSA-26-211-06, 30 July 2026); **CVE-2026-67970** DS path traversal; **CVE-2026-67972 / 67973** CF: attacker controls where received data is stored, DoS by replaying final PDUs; **CVE-2026-67978** SBN; **CVE-2026-67979** (9.1) ES dynamic app start; **CVE-2026-82480** SB length underflow
- 2021: no CVE; Broberg (Lockheed Martin, DEF CON 29) fuzzed CI_LAB and found crashes; CI_LAB had no authentication then and has none now
- This build: cFE v7.0.0+dev, snapshot patched by hand (Table Services ×3, CF NAK). **Each CVE above names an app in this build; whether the fix is in this snapshot is unverified and is roadmap step 8**

The honest state: on this vehicle integrity rests on the host (file permissions on root/cf_rtems and files/sp0) rather than on the vehicle. The cheapest on-board step is filling the CS tables (roadmap step 3); the real one is a signed image and a hashed /cf (step 6).

WHAT A CYBER SENSOR NEEDS TO SEE

Fourteen cyber-relevant telemetry categories from the statement of work, scored on the twin

Category	State	Where it stands
Command provenance	MISSING	origin, route, session of each command; SBN carries ProcessorId but it is not published
Command authorization status	MISSING	accepted/rejected per command with the reason; cFE accepts a bad checksum (verified)
Telemetry integrity status	PARTIAL	spoofed sensors are detected (SMSR-4/5) but no per-packet seq/checksum/staleness field
FSW execution events	DONE	CFE_EVS long events tapped; app lifecycle via SpaceCOP state-change
Process execution	PARTIAL	procmon rules registered, inert without the kernel module
Software integrity	PARTIAL	data-file sha256 works (DISE-1); binaries and boot not hashed (DISE-2 deferred)
Memory integrity	MISSING	monitors registered, inert without /dev/mem; S07 deferred
Configuration monitoring	PARTIAL	table loads seen in TBL counters; no baseline CRC or off-window classification
Hardware bus activity	PARTIAL	buses instrumented for injection; no error or unexpected-peer telemetry
Communications activity	PARTIAL	loss detected; no link statistics fed to the detector
Event logging	DONE	full EVS stream consumed; STIX written; IDS report 0x08F9
Security audit logging	MISSING	no consolidated tamper-evident trail (auth decisions + attribution + detector actions)
Time synchronization events	DONE	SET_TIME → GNTM-7 and GNTM-6, verified
Cyber health telemetry	MISSING	nothing reports that monitors are inert; degraded detection is invisible

DONE 4, PARTIAL 6, MISSING 4. The four gaps (provenance, authorization status, security audit log, cyber-health) are designed in docs/sow/task2 in three seams that never touch mission apps. DRACO7RT has the same gaps.

TWIN VERSUS VEHICLE

SpaceCOP runs on the NOS3 x86 twin; porting it to the RTEMS vehicle is a known list, not a research problem

Where things run today

- **NOS3 twin** (Linux x86, stock STF1 flight software, 11 component sims): SpaceCOP CE, the SBN crosslink, the 14 scenarios, CryptoLib in the app list
- **DRACO7RT vehicle** (QEMU PowerPC e500, RTEMS 6): the mission apps, LC/SC autonomy, CS/HS/MM/MD, CFDP; SPACECOP exists only in the COSMOS dictionary (Viewer › Mission › SpaceCop) and the Yamcs MDB (definitions audited 2026-10-22 against nos3 commit 283891d5)
- SPACECOP packets are **little-endian payload** (x86) under a big-endian CCSDS header; on the vehicle they would be big-endian, so the ground definitions change with the port

What the port takes

1. Add spacecop/fsw/cfs to cpu1_STATIC_APPLIST and the startup script (the twin needed CFE_PLATFORM_ES_MAX_APPLICATIONS 32 → 40)
2. libcrypto: SpaceCOP links OpenSSL for sha256 and STIX; RTEMS has no system libcrypto, so vendor a small sha256 or build OpenSSL for powerpc-rtems6
3. Linux-only monitors (procmon, syscall, /dev/mem, /dev/mtd) stay inert by design: the registry already tolerates absent monitors
4. File integrity over /cf (NFS) and /ram; STIX output to /ram (RTEMS NFS cannot create files on /cf; names ≤ 19 characters)
5. Cyber-safe RTS id: RTS 2 is **solar-array deploy** on DRACO7RT, so a dedicated safing sequence gets a free number (8+) and goes on the hazard list
6. Mission rules in spacecop_mission_specific.c: NAV reject flags, IMU fault flags, ACS mode changes, TBL loads

What the twin already proved

- Detection on a real cFS bus, not a simulation of one
- Two vehicles, one detection, coordinated safing
- The three cFE snapshot bugs found here (Table Services) would have hidden a table-load detection: a port is also a test of the detector

The twin and the vehicle share the cFS Software Bus, EVS and Table Services APIs, which is everything SpaceCOP touches. The difference is the operating system underneath, and that is where the monitor list shrinks.

ROADMAP FOR DRACO7RT

Cheapest and most reused first; each step is a requirement, a test and an evidence record

#	Step	What it takes	Closes
1	**Validate the command checksum and length in every app**	CFE_MSG_ValidateChecksum in the command dispatch of the mission apps (SC already does); event + counter on failure; COSMOS already computes it	the malformed-command finding (S03); a first authorization status
2	**Command provenance and authorization telemetry**	CI_LAB (or a passive cyber app) publishes per-command source address, sequence, accepted/rejected and why; one new packet in both dictionaries	SOW categories 1 and 2; attribution on the ground
3	**Turn on the Checksum app**	Fill cs_apptbl, cs_tablestbl, cs_memorytbl: the mission apps' code, the LC/SC/NAV/IMU/ACS tables; CS events become DISE-class detections	software and configuration integrity on the vehicle (no port needed)
4	**Port SpaceCOP CE to the RTEMS vehicle**	the six-item list on the previous slide; rules for NAV/IMU/ACS/TBL; a dedicated cyber-safe sequence on the hazard list; S01-S14 rerun on QEMU	an on-board IDS where the evidence is taken
5	**Authenticated commanding (SDLS)**	CryptoLib on the vehicle in front of CI_LAB; ground-side encoding in Yamcs (and COSMOS through it); anti-replay sequence; key management on the ground	SPD-5 positive control; replay and injection by design
6	**Signed image and hashed file system**	sign files/sp0, verify in U-Boot or at cFE start; hash /cf at boot and on every CFDP delivery	DISE-2, boot integrity
7	**Cyber-health telemetry and a security audit log**	SpaceCOP HK that says which monitors are live; a hash-chained log of auth decisions, commands and detector actions, forwarded to the ground SIEM (ground deck)	SOW categories 12 and 14
8	**Cybersecurity assessment under NPR 7150.2D**	the classification record marks SWE-156 'not assessed'; do it against this deck's surface and the SPARTA countermeasures; purple-team the harness against the vehicle	the assurance record

Steps 1 to 3 are ordinary flight-software changes in the loop of the Flight Software guide (edit, unit test, cppcheck, build, verify, trace). Steps 4 to 6 are projects. Nothing here changes a mission app's control logic.

RULES FOR FLIGHT SOFTWARE SECURITY ON THIS PROJECT

From the statement of work, the risk register and this project's own mistakes

Detection

- A scenario asserts on a **Detection** object, never on a detector's internals: detectors are interchangeable behind one seam
- Every detection has an **oracle** outside the detector: the vehicle's own event console in a time window captured at inject
- A pass needs positive evidence the path executed (an event line, a counter that moved), not the absence of an error
- Single-sample reads straddle housekeeping cycles (R3): poll to a stable condition over a bounded window

Response

- Autonomous response is a **dedicated, auditable sequence** with a one-shot latch and an operator recovery command (R12)
- Severity drives both the cross-cue and cyber-safe entry; one function, one floor, no per-scenario tuning
- Never disable a gate (NAV, IMU, LC, HS) to make a demo green; if a gate blocks you, say so and stop
- A new safing sequence goes on the hazard list before it is flown

Engineering

- Instrumentation and injection stay **outside mission flight software**; a cyber app is subscribe-only
- Every constant that may change is a table value, so a configuration change is visible and validated
- Controlled information (R14): the furnished SpaceCOP drop, SPARTA details under control, and any real-mission data stay out of the deliverables and the repository
- Claude Code writes under SWE-146: same tests, static analysis and human review as hand code; it never sets a requirement or signs anything

Deck sources: cyber-scenarios/README.md, docs/sow/risk-register.md, docs/sow/task2 and task4, ~/nos3/components/spacecop, cFS/apps/nav, imu, cs, hs, ci_lab, sc, and the cited public documents (slides 2, 4, 5, 12, 13). Nothing in this deck is committed yet.