

SAPIENT C-UAS Implementation Guide

BSI Flex 335 v2.0 and STANREC-aligned Counter-UAS integration.

NCIA C-UAS Team

Guide version 2026.0.1
BSI Flex 335 v2.0:2024-03

Operational baseline
STANREC 4869 Edition A Version 1.0

Index

- | Purpose and scope
- | Design Principles
- | Conformance baseline
- | Field Presence and Data quality
- | Operational roles
- | Message lifecycle
- | Registration and status
- | Detection and track metadata
- | Tasking, alerts, and errors
- | Hierarchy, units, and checklists



Purpose and scope

| The Implementation guide narrows optional SAPIENT behaviour, so (independent) C-UAS nodes can interoperate predictably.

What it covers

Framing, registration, status, detections, tasking, alerts, errors, hierarchy, C-UAS metadata, and mission configuration.

What stays outside

Raw media transport, national rules of engagement, effector safety authorization, and HMI design except where they affect message content.

| Joint effort between industry, NATO and nations.

Design principles

- | Use SAPIENT at the information level (detections/tasking, not raw sensor data)
- | Edge node autonomous enough to execute a task locally after receiving it
- | Registration is the capability contract. Only transmit what is advertised
- | Fusion node hierarchy is level-agnostic. Child fusion nodes forward aggregated information as it was an edge node
- | Use SAPIENT default fields first. Only when needed use CUAS extension fields.

Conformance baseline

Area	Profile rule
Protocol files	Generate code from the official BSI Flex 335 v2.0 .proto files.
Transport and encoding	Support TCP/IP, Google Protocol Buffers, and UTF-8 strings.
Framing	Prefix every serialized SapienMessage with a 4-byte little-endian length.
Time and identifiers	Use UTC mission NTP, UUID v4 node_id values, and ULIDs for report/task/object/alert/region identifiers.
ICD version	Populate Registration.ICD_version with the agreed BSI Flex and STANREC baseline. (‘BSI Flex 335 v2.0 / NATO STANREC 4869 Ed A V1.0’.)

Field Presence and Data quality

Case	Receiver behavior	Response
Malformed frame or undecodable protobuf	Discard the message. Do not forward or partially process.	Send Error if the source can be identified; otherwise log locally.
Missing mandatory field	Discard the message.	Send Error describing the missing field.
Mandatory enum set to UNSPECIFIED/0	Treat as not set and discard the message.	Send Error.
Invalid optional field	Ignore only that field when the rest of the message is valid and safe to use.	Send Error with warning severity in the text; continue processing valid content.
Operational rejection of a valid registration, task, or alert	Do not use Error.	Use RegistrationAck, TaskAck, or AlertAck with rejection reason.
Unknown string value where registration defined an allowed set	Treat as invalid field. If the field is mandatory, discard message; if discretionary, ignore field.	Send Error describing expected values.

Operational roles

| Operational roles are represented through capability, not vendor identity.

Role	SAPIENT representation	Primary responsibility
Sensor edge node	RADAR, CAMERA, PASSIVE_RF, ACOUSTIC, LIDAR, or OTHER	Register capability, report status, send detections, and accept supported sensing tasks.
Effector edge node	JAMMER, CYBER, KINETIC, LDEW, RFDEW, or OTHER	Register effect modes, report arming/activity status, and accept only authorized tasking.
Platform node	MOBILE_NODE or POINTABLE_NODE with dependent_nodes	Expose movement or pointing capability and status for dependent nodes.
Dual-role edge node	NodeDefinition includes all applicable sensor and effector types.	Separate mode definitions shall make clear which modes sense, effect, or do both.
Child fusion node Level 1 C2	FUSION_NODE	Aggregate lower-level capability upward and manage lower-level nodes downward.
Top-level fusion node Level 2 or 3 C2	FUSION_NODE Server role in the hierarchy	Accept registrations, task nodes, fuse detections, and manage lifecycle and errors.
Message handling application Middleware	Does not register. Is transparent (proxy).	Forward transparently, maintain connectivity/status, optionally validate and persist messages

Message lifecycle

- | An edge node cannot send operational content until its registration has been accepted.
- | A fusion node treats unregistered operational messages as protocol faults.

Edge node startup

Open TCP connection to the assigned fusion node or message handler.

Send exactly one Registration before operational messages.

Wait for RegistrationAck.acceptance=true.

Send initial StatusReport after successful registration.

Fusion node startup

Maintain registration state for each node_id.

Discard non-registration messages from unregistered nodes.

Accept registration only when mandatory fields, version, definitions, and config_data satisfy the profile.

Reject valid but unsupported registrations with RegistrationAck, not Error.

Message lifecycle

I Normal operations

Message	Start condition	Transmission rule	Receiver action
StatusReport	After RegistrationAck and initial status.	Periodic at registered interval; additionally on change for on_change fields.	Update node health, active mode, task, location, coverage, and performance state.
DetectionReport	After RegistrationAck and when detection/tracking is active.	On new detection and on meaningful update; C-UAS profile caps updates for the same object at 10 Hz unless locally tightened.	Validate against registered detection definition, fuse/store, and forward upward when applicable.
Task	Fusion node has an operational reason and node registered support.	Single task message requiring TaskAck. Use destination_id.	Accept, reject, complete, or fail through TaskAck messages.
Alert	Node or fusion node detects a non-detection event requiring notification or permission.	Send when active; response required unless AlertType=INFORMATION and local policy says no ack.	Acknowledge through AlertAck or return Error if malformed.
Error	Invalid message received.	Send immediately where source is known.	Sender shall log, correct, and avoid retrying the same invalid content.

Message lifecycle

I Shutdown and reconnect

Condition	Required behavior
Planned node shutdown	Send <code>StatusReport.system=GOODBYE</code> , then close the connection after the message is flushed.
Connection lost	Both sides monitor connectivity. A disconnected node shall attempt reconnect every 10 seconds.
Reconnect within 2 minutes	Registration resend is not required, but the node shall send a fresh <code>StatusReport</code> immediately.
Reconnect after more than 2 minutes	Node shall repeat full Registration before any other operational message.
Missed heartbeat	If the fusion node receives no status within three registered status intervals, it shall close the socket, mark the node stale, and wait for reconnection.

Registration and status

- | Registration is the capability contract; nodes should fill it to the maximum truthful extent.

Fill to the max extend

The node definition, modes, commands, detection definitions, status definitions, and dependent nodes bound every later message.

Aggregate capability upward

A child fusion node registers combined lower-level capability rather than leaking every internal implementation detail to a parent.

Registration and status

Registration element	C-UAS profile guidance
node_definition	Include every relevant role. A child fusion node uses FUSION_NODE; if it is physically on a mobile or pointable platform, include that platform type too.
name	Name should contain Unit/Company, followed by a descriptive name of the capability. For example: 'CompanyA_RF_Sensor'
short_name	Short name should be an abbreviation of the Unit/Company name, followed by an abbreviated capability name. If the Unit/Company has multiple identical capabilities, it should be followed by a number. For example: 'CA_RF_01'
capabilities	Use category/type/value/units to declare operationally useful details such as sensor band, effector frequency range, coverage sector, maximum transmit power, supported media, or safety interlocks.
status_definition	Declare status_interval between 1 and 60 seconds. In normal cases, implementation should use 5 seconds for mobile/effecting nodes and 10 seconds for fixed sensors unless bandwidth requires otherwise.
mode_definition	Declare exactly one DEFAULT mode. Mode names are case-insensitive operationally but shall be sent exactly as registered.
detection_definition	Declare location type, class taxonomy, behaviour taxonomy, velocity type, and additional track/object fields actually sent by the node.
task	Declare concurrent_tasks honestly. If concurrent_tasks=1, reject a second incompatible task rather than silently replacing it unless the command definition says it overrides.
dependent_nodes	Use for platform nodes and child fusion nodes that aggregate or control lower nodes. The parent uses this to reason about task side effects.
config_data	Populate manufacturer, model, serial/product identifier, hardware_version where physical hardware exists, and software_version for all software components.

Registration and status

- | Status keeps operators and fusion logic aligned with live capability.
- | All nodes send regular StatusReport messages after registration, with active modes, location, coverage, and field-of-view changes represented explicitly.

Status expectations

Use the registered mode name; comma-separate multiple concurrent modes.

Set info=NEW for material, field-of-view or coverage changes.

If no new info, set info=UNCHANGED

Provide node_location initially and whenever it changes.

Make effector activity visible through StatusReport.mode.

Effector extension

Use StatusType=OTHER when active mode does not fully describe effector state.

Represent object_id and track_id references in status_value.

Include releasable authority state where it matters.

Do not rely only on task acknowledgements for live effector state.

Registration and status

- | C-UAS Effector status extension
- | When active mode does not fully describe the state or when related to a specific object/track.

Key	Value rule	Example
mode_parameters	Brief registered parameter string or safe operational summary.	mode_parameters:sector-3
object_id	Repeat the key for each affected SAPIENT object_id.	object_id:01J...
track_id	Repeat the key for each local or STANREC track identifier.	track_id:TN-0042
authority	Optional local authorization state where releasable.	authority:operator-approved

Detection and track metadata

| DetectionReport carries both single detections and track updates.

Field	Profile rule
report_id	ULID unique to the report; timestamp component changes on each report.
object_id	ULID persistent across a tracklet, track, or track with re-identification.
location_oneof	Provide Location or RangeBearing; child fusion nodes convert RangeBearing before upward reporting where required.
detection_confidence	Use a 0.0 to 1.0 probability.
classification / subclassification	Use namespace:value strings; default namespace is sapient_core. If default is used, namespace prefix may be omitted.
traceability	Use associated_detection and derived_detection to preserve fused-source relationships.

A sensor that cannot maintain identity across detections generates a new object_id for each report.

Detection and track metadata

| TrackObjectInfo extensions carry C-UAS identity and origin metadata.

Extension	Allowed values or unit	Use
Identification	Pending, Unknown, Assumed Friendly, Friendly, Suspect, Hostile	Operational identity aligned with local STANREC practice.
Exercise Indicator	True or False	Marks exercise scenario tracks.
Simulated Indicator	True or False	Marks simulated or injected tracks.
Originator position	Latitude, longitude, altitude	Required when forwarding RangeBearing-originated reports upward.
Originating Node ID	UUID v4	Preserves original edge node identity through child fusion.
Local Track ID	Local string	Optional bridge to non-SAPIENT track numbers; not a replacement for object_id.

The extension only carries strings, so numbers and Booleans are transmitted as strings

Tasking

- | Tasking uses acknowledgements for decisions and Error for protocol faults.
- | A valid task must match a capability advertised in registration.
Rejection, completion, and failure state travel through TaskAck.

Task command	When to use	Acknowledgement expectation
REQUEST; command_parameter: Registration	Fusion node needs current capability contract.	Node sends TaskAck ACCEPTED, then sends Registration.
REQUEST; command_parameter: Status	Fusion node needs full current state.	Node sends TaskAck ACCEPTED, then sends StatusReport with all supported current fields.
REQUEST; command_parameter: Arm	Prepare an effector for later start.	Effector sends TaskAck and reports armed mode/status. Requires local authority.
MODE_CHANGE	Change to a registered operating mode.	Reject if mode not registered or unsafe; status shall show new mode after change.
LOOK_AT	Cross-cue a steerable sensor or pointable platform.	TaskAck ACCEPTED/REJECTED; detection report if object detected; COMPLETED without detection means nothing found.
FOLLOW	Follow the object ID as defined in the task	TaskAck ACCEPTED/REJECTED; detection report if object detected;
MOVE_TO/PATROL/FOLLOW	Control a mobile or pointable platform.	TaskAck ACCEPTED/REJECTED and subsequent status updates for Transiting, Patrolling, Following, Stuck, or Drifting states.
DETECTION_TRESHOLD/ DETECTION_REPORT_RATE/ CLASSIFICATION_TRESHOLD	Set the requested threshold or rate to the specified value.	TaskAck ACCEPTED/REJECTED

Tasking

I Taskings are controlled.

Task Control	When to use	Acknowledgement expectation
START	Start or schedule a task, resume paused task, or start command execution.	TaskAck ACCEPTED or REJECTED promptly; COMPLETED later if execution has a finite completion point.
STOP	Stop the task, remove its definition, and revert to previous/default task.	TaskAck ACCEPTED if the stop was applied; COMPLETED may follow when physically complete.
PAUSE	Suspend task, go back to default or previous, but keep definition available for restart.	TaskAck ACCEPTED if paused; status shall show not detecting or platform state where relevant.

Alerts and errors

- | Alerts notify operators; Errors diagnose invalid messages.
- | Separates operational events from protocol repair so senders know whether to acknowledge, retry, or correct content.

Alert behavior

Use Alert for non-detection events, warnings, and permission requests.

Do not use Alert as a substitute for DetectionReport.

Acknowledge alert types unless INFORMATION or local policy suppresses acknowledgement.

Rejected permission alerts should include a retry instruction only when retry is allowed.

Error behavior

Send Error for dropped invalid messages when the source is known.

Use clear patterns for missing fields, invalid values, unsupported versions, framing defects, or unregistered messages.

After Error, the sender corrects the fault rather than blindly retransmitting.

Hierarchy, units, and checklists

| Keep parent C2 focused on deliverable capability.

Topic	Profile rule
Bottom-up registration	Preferred sequence is edge nodes to local fusion, then child fusion to parent.
Aggregate capability	A child fusion node registers only capability it can deliver through current lower-level nodes or its own processing.
Coverage changes	Lower-node loss updates StatusReport.coverage and status within one status interval.
Platform dependencies	Mobile and pointable platform nodes declare dependent_nodes so task side effects can be evaluated.
Task decomposition	A child fusion node accepts parent tasks only when it can autonomously decompose and supervise them.

RangeBearing-originated reports forwarded upward include originator location metadata in TrackObjectInfo extensions.

Hierarchy, units, and checklists

Data	Unit/resolution profile
Timestamp	UTC google.protobuf.Timestamp. Source content time, not transmission time.
Latitude/longitude	Decimal degrees for LAT_LNG_DEG_M. Recommended precision 6 decimal places or better when available.
Altitude	Meters above declared datum. Recommended precision 0.1 m where available.
Range	Meters. Detection RangeBearing allows 0.01 m resolution; cone/range region values may use finer precision where supported.
Angles	Degrees unless registration declares radians. Recommended 4 decimal places for point measurements and 5 for cone extents where supported.
Velocity	ENU east/north/up rates in m/s for this profile. KPH may be accepted only when advertised.
Frequency	Use Hz-derived SI units in registration. If MHz is used for Signal fields by legacy convention, declare MHz explicitly in detection definition.
Probability/confidence/ranking	Float 0.0 to 1.0.
Battery level	Integer percentage 0 to 100.
Status interval and task durations	Use Duration with explicit TimeUnits; seconds preferred.

Hierarchy, units, and checklists

Pre-Integration

Generate code from official .proto files.

Implement 4-byte little-endian framing.

Use UUID v4 node_id and ULID report/task/object/alert identifiers.

Populate registration with config_data, intervals, modes, task definitions, detection definitions, and metadata fields.

Test invalid-message handling, GOODBYE status, and reconnection behavior.

Operator acceptance

Confirm NTP synchronization for all nodes.

Verify every node registers before operational data.

Show node health, active mode, coverage, field of view, and stale-node state.

Exercise task acceptance, rejection, completion, and failure.

Implementation guide takeaway

- | Turn a flexible standard into an operational contract.
- | First step, use next exercises to further improve.

Contact us

E-mail C-UAS_Exercises@nr.ncia.nato.int

Phone