

Correct in Isolation, Broken in Translation: Exploiting Cross-Schema Vulnerabilities in LTE and 5G Base Stations

Qiqing Huang, Xingyu Wang, Xiaofeng Chen, and Hongxin Hu
University at Buffalo
{*qiqinghu, xwang282, xchen326, hongxinh*}*@buffalo.edu*

***Abstract*—Protocol translators bridge two specifications by decoding from one schema and re-encoding into another. Cellular base stations are one such instance: an LTE eNB or 5G gNB receives uplink messages in the RRC schema and re-encodes selected fields into S1AP or NGAP for the core network. Because each schema is maintained by a different 3GPP working group on independent release cycles, no specification enforces consistency across them; each is correct in isolation, but their boundary inside the base station is unprotected. This boundary, which we call the translation surface, is hard to discover: it is invisible to single-schema parser fuzzing and compliance testing, it lives inside a single binary without language-level type enforcement across schemas, and reachability depends on per-implementation handling of cross-schema values. CELLBREAKER addresses these challenges by combining static cross-schema divergence analysis, which identifies candidate divergent value pairs across protocol boundaries, with dynamic boundary-targeted testing on running base-station implementations. Across four protocol boundaries (LTE RRC-S1AP and 5G NR RRC-NGAP, NGAP-F1AP, NGAP-E1AP), CELLBREAKER finds divergent pairs at every boundary and uncovers three distinct boundary failures in open-source and commercial stacks: a 6-byte pre-authentication message crashes an open-source 5G gNB; an LTE eNB silently changes the meaning of a reserved RRC establishment cause when re-encoding it into a legitimate S1AP cause code on the wire, an existence proof rather than an end-to-end exploit; and a 10-byte OTA frame causes a 30-second cell-wide outage in a commercial Baicells/Open5GS small-cell deployment.**

1. Introduction

A single six-byte uplink RRC payload, sent after random access from a device with no SIM card, no credentials, and no established connection, can crash a widely used open-source 5G base station (OpenAirInterface [1]) and drop every user attached to that cell. The payload is not malformed in the usual sense: it passes the CRC, is decoded successfully by the generated RRC parser, and reaches ordinary base-station control logic. The crash occurs later, when the base station tries to translate a value accepted on the radio side into a different protocol format the core network expects. Because a base station is shared infrastructure, this translation failure is not confined to one endpoint; it can take the serving cell offline.

This paper studies the attack surface exposed by these translation failures. A cellular base station is not merely a parser for one protocol. It decodes UE-facing Radio Resource Control (RRC) messages, stores selected fields in implementation data structures, and later re-encodes some of them into core-facing protocols: S1AP in LTE and NGAP in 5G. Disaggregated 5G deployments add further mediation points: F1AP between the Distributed Unit and Centralized Unit, and E1AP between the CU control-plane and user-plane. The subtlety is that these protocols are not different encodings of one shared type system. Each has its own ASN.1 schema, maintained by a separate 3GPP working group and evolving on an independent release cadence. The generated encoder and decoder for each schema may be correct in isolation, while the C/C++ glue code connecting two schemas is not. A value may be legal in the source schema but unrepresentable, reserved, or semantically different in the destination schema. Parsing each schema is largely a solved engineering problem; preserving correctness across independently maintained schemas is not.

Existing cellular testing techniques are not designed to search this space systematically. Prior work has found parser bugs [2], [3], downlink baseband exploits [4], [5], [6], specification-compliance violations [7], [8], and physical-layer uplink attacks [9]. These approaches are effective within a protocol context: they exercise decoders, state machines, conformance rules, or radio procedures. They do not systematically ask whether a schema-valid value remains representable, semantically equivalent, or properly contained after base-station mediation into another protocol context—so failures that manifest only at re-encoding, forwarding, or cross-context mediation boundaries remain outside their primary search space.

The missing piece is that a single base station contains multiple mediation boundaries—some between independently maintained ASN.1 schemas, others between forwarding domains meant to remain opaque—which we call the **translation surface**. At this surface, typed fields require *value-space preservation*: a value accepted by the source schema should remain representable and semantically equivalent in the destination schema. Opaque inter-layer containers, such as NAS PDUs, require *containment isolation*: the base station should forward the payload without interpretation

while preserving its association with the correct UE and core-facing context. This is the cellular instance of a broader interface-mismatch pattern previously surfaced in differential TLS parsing [10] and HTTP request smuggling [11]: two components can be correct in isolation yet fail at their shared boundary.

In this paper, we present CELLBREAKER, a boundary-aware framework for discovering translation-surface vulnerabilities. CELLBREAKER combines cross-schema divergence analysis, opaque-container boundary analysis, analyst-assisted reachability tracing, and boundary-targeted dynamic testing. Rather than fuzzing one parser in isolation, CELLBREAKER constructs inputs whose danger appears only after the base station accepts and mediates them into another protocol context.

Across every examined protocol boundary—LTE RRC↔S1AP; 5G NR RRC↔NGAP; NGAP↔F1AP; and NGAP↔E1AP—CELLBREAKER successfully identified divergent value pairs. Its systematic scan identifies 87 paired enumeration types, of which 36 (41%) exhibit value-space divergences; 30 are prioritized for testing, seven are exercised, and Table 9 classifies all 30 by evidence level, from structural divergence and implementation correctness loss to demonstrated boundary failures. These divergences lead to three qualitatively different outcomes across open-source and commercial stacks: a pre-authentication 5G NR gNB crash in OAI that reproduces deterministically across two software versions released 14 months apart; an LTE wire-level silent-corruption existence proof in srsRAN 4G [12], where a reserved RRC value is invisibly translated into a legitimate S1AP cause code; and a 30-second cell-wide denial of service on a commercial Baicells Nova 430H / Open5GS deployment, where the eNB's spec-compliant transparent NAS forwarding exposes the core to an unauthenticated out-of-context payload (crashing the Open5GS MME) and amplifies the resulting SCTP teardown via a fixed 30-second reconnection timer (§5.3). We treat the LTE silent-corruption case as a wire-level existence proof rather than an end-to-end exploit; the corrupted value reaches the core on the wire, but neither tested core acts on it, so no downstream security effect is demonstrated. Our decoder-only baseline does not reach the mapping and re-encoding stages where these boundary failures arise (Appendix E).

The key contributions of this paper are as follows:

1) **Cellular translation-surface systematization.** We identify and systematize the translation surface inside LTE and 5G base stations: the mediation boundary where independently maintained ASN.1 schemas and opaque forwarding domains meet (§3). We formalize two safety properties at this boundary: value-space preservation for mapped fields and containment isolation for opaque payloads. We focus on horizontal (cross-working-group) divergences and discuss vertical (cross-release) divergences as an extension (Appendix D). This framing unifies value-space and containment failures that otherwise appear as unrelated implementation bugs.
2) **Boundary-aware analysis and testing.** We introduce CELLBREAKER, which combines cross-schema divergence analysis, opaque-container boundary analysis, reachability tracing, and boundary-targeted dynamic testing on running base stations (§4).
3) **Cross-generation findings.** We report three boundary failures across LTE and 5G NR: a pre-authentication OAI gNB crash, wire-level silent semantic corruption in srsRAN 4G, and a commercial CBRS denial of service (§5). The first and third directly affect availability, while the srsRAN finding demonstrates semantic-integrity loss across the RRC→S1AP boundary. The same translation surface can produce crashes, silent lossiness, or cell-wide disruption depending on mediation and downstream handling (§6).

We release CELLBREAKER's divergence analyzer, safe static reproducers, and non-commercial proof-of-concept test inputs; commercial OTA reproduction artifacts are not released.[1]

2. Background

2.1. Control-Plane Mediation in LTE and 5G

Radio Resource Control (RRC) is the UE-facing control-plane protocol, running between UE and gNB in 5G NR (TS 38.331 [13]) and between UE and eNB in LTE (TS 36.331 [14]). Uplink RRC traffic reaches the base station on UL-CCCH for early establishment messages such as `RRCSetupRequest`, and on UL-DCCH for dedicated signaling after connection establishment.

After decoding RRC, the base station forwards selected control-plane information toward the core. In LTE, the eNB uses S1AP (TS 36.413 [15]) toward the MME; in 5G the gNB uses NGAP (TS 38.413 [16], [17]) toward the AMF. Thus even a monolithic base station sits between a UE-facing RRC context and a core-facing application-protocol context.

Modern 5G deployments add internal control-plane boundaries. TS 38.401 [18] allows the gNB to split into a Distributed Unit (**gNB-DU**) and a Centralized Unit (**gNB-CU**); the CU may further split into Control-Plane (**gNB-CU-CP**) and User-Plane (**gNB-CU-UP**) components. F1AP connects the DU and CU-CP (TS 38.473 [19]), while E1AP connects the CU-CP and CU-UP (TS 38.463 [20]). Figure 1 shows the resulting mediation points.

2.2. ASN.1 Schemas and Generated Codecs

RRC and the application protocols above are specified in ASN.1 [21] and encoded with Packed Encoding Rules [22], unaligned for RRC and aligned for the application protocols. ASN.1 defines enumerations, integer ranges, size constraints, `OPTIONAL` fields, and extension markers (`...`); compilers such as `asn1c` [23] generate encoders and decoders that enforce one schema at a time.

The relevant schemas are defined in separate 3GPP specifications: NR RRC in TS 38.331, LTE RRC in TS 36.331,

1. https://doi.org/10.5281/zenodo.22814506

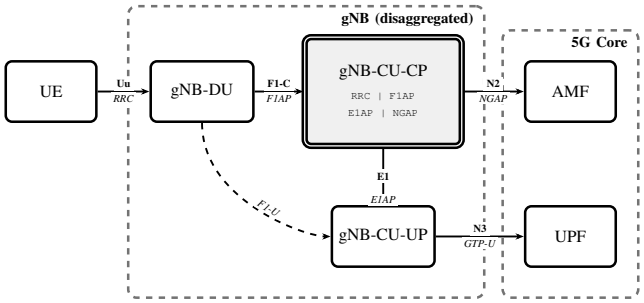


Figure 1. The 5G NR gNB after the CU/DU/CU-UP split (TS 38.401). RRC, F1AP, E1AP, and NGAP meet inside or around the gNB-CU-CP. The LTE eNB is the simpler non-disaggregated counterpart: F1 and E1 collapse into implementation calls, leaving RRC, S1AP, and NAS forwarding as the main control-plane paths.

NGAP in TS 38.413, S1AP in TS 36.413, F1AP in TS 38.473, and E1AP in TS 38.463. They are owned by separate 3GPP working groups—RAN WG2 owns RRC, RAN WG3 owns NGAP/S1AP/F1AP/E1AP, and CT WG1 owns NAS—and evolve on independent release cadences. The standards define each protocol, but no single normative artifact guarantees that every implementation-level field mapping across protocols is mutually representable or semantically equivalent.

2.3. Field Mapping Across Schemas

Generated codecs validate a message against the schema being decoded or encoded, but base-station implementations must connect those generated types: a typical path decodes an RRC message into generated structures, copies selected fields into C/C++ state, and later constructs an S1AP, NGAP, F1AP, or E1AP message.

Here protocol-local validity and cross-protocol compatibility can diverge: a source schema may define a value as ordinary, reserved, or spare, while the destination schema omits it, places it at a different numeric index, or assigns that index a different symbolic meaning. Because mediation code often uses plain integers, C enums, or implementation-specific sentinels between generated decoder and encoder calls, a value accepted by the source parser is not automatically safe to encode at the destination.

2.4. Opaque NAS Forwarding

Some information crossing the base station is not translated field by field; the Non-Access Stratum (NAS) layer runs end-to-end between the UE and the core network: EMM/ESM in LTE per TS 24.301 [24] and 5GMM/5GSM in 5G per TS 24.501 [25]. Each NAS message begins with a one-byte Protocol Discriminator (PD) that selects its sub-protocol; some PD values denote legacy circuit-switched procedures (TS 24.007 [26]) and are not valid EPS/5GS NAS discriminators. NAS messages are carried through the base station inside RRC on the radio side and inside S1AP or NGAP on the core-facing side.

The base station is not the semantic endpoint for NAS: TS 36.413 §8.6.2.3 requires the NAS-PDU IE to be "transferred without interpretation in the eNB", and TS 38.413 §8.6.1.2 imposes the analogous requirement on the NG-RAN node, making NAS an opaque payload from the base station's perspective. Unlike mapped fields, opaque NAS forwarding creates an implicit trust-bridging and amplification hazard: strict adherence to the "without interpretation" mandate turns a spec-compliant base station into an unthrottled conduit that projects unauthenticated radio bytes directly onto the trusted Core interface, while multiplexing all UEs over a single SCTP association allows a single-UE core fault to take down the entire cell.

3. The Translation Surface

Building on the protocol boundaries laid out in §2.1–§2.4, we define the **translation surface**: the base-station mediation boundary where information accepted under one protocol context is carried into another. A parser boundary asks whether an input is valid under one schema. A translation boundary asks whether, after a value or payload has passed local validity checks, it remains representable, semantically equivalent, or safely contained when the base station maps, re-encodes, or forwards it in the next protocol context.

3.1. Motivating Example

Consider the `establishmentCause` field in every `RRCSetupRequest`. This field tells the base station why the UE is requesting access. RRC (TS 38.331 v17.3.0 [13]) defines it as a fixed 16-value enumeration with six `spare` slots and no extension marker. NGAP (TS 38.413 v15.8.0 [16], the version used by OpenAirInterface) defines the corresponding field as an extensible enumeration with only 11 defined values: 0–9 plus `notAvailable` as the lone extension. Values 11–15 (`spare5` through `spare1`) are therefore valid in RRC but have no representation in NGAP.

OpenAirInterface illustrates why this is a translation-surface failure rather than an RRC parsing failure. When a UE sends `establishmentCause = 12` (`spare4`), OAI first accepts the message in the generated RRC decoder and stores the decoded value as a C integer in UE state. The failure appears only later, when that stored value is carried to the RRC-to-NGAP mediation boundary, and OAI attempts to construct the NGAP message. On OAI 2026.w16, the mediation guard checks whether the stored cause is smaller than `NGAP_RRC_CAUSE_LAST`. Since `NGAP_RRC_CAUSE_LAST` is 12, the input value 12 fails the check ($12 < 12$ is false), and the `AssertFatal` guard aborts the gNB. Value 11 follows a different path: it passes the guard but later fails at the NGAP APER encoder because the destination schema still cannot encode it. Both paths share the same root cause: the RRC value space exceeds the NGAP value space.

The same bug class also survives local symptom patches. In OAI 2025.w05, the relevant C enum was non-contiguous,

Prior testing scope	What it exercises	Trigger condition
Parser fuzzing	Decoder on one schema	Malformed input under one schema
Logic / handler	Single-protocol decision	Reachable state in one protocol
Spec compliance	Conformance to one spec	Behavior diverges from that spec
Memory safety	Buffer / lifetime bugs	Corruption inside one component
State machine	Transition table of one protocol	Out-of-order or replayed message
Translation surface	**Cross-schema mapping or opaque forwarding**	**Schema-valid value or opaque payload becomes unsafe after translation or forwarding**

TABLE 1: Testing-scope distinction: where prior cellular work focuses vs. the translation surface trigger condition. A boundary failure may manifest through any of the rows above; the distinction is whether the trigger condition spans two protocol contexts.

and the sentinel allowed some unmapped values to pass the guard, where they failed downstream at NGAP encoding. In OAI 2026.w16, the C enum was made contiguous, and the visible failure moved to the guard above, but the NGAP ASN.1 schema still did not acquire representations for RRC spare values 11–15. The patch moved the crash site; it did not remove the schema-pair mismatch.

3.2. Definition

A **translation surface vulnerability** exists at a base-station protocol boundary when an input that passes layer-local validity checks later causes a crash, semantic corruption, or loss of containment after the base station translates or forwards it across the boundary. The defining feature is that the trigger condition spans two protocol contexts: a single-schema parser sees a valid input, but the value or payload becomes unsafe only after the base station mediates it into another schema or forwarding domain.

Two safety properties follow, one for each kind of thing a boundary carries. *Value-space preservation* applies to a mapped field: every value the source schema can encode must have a representation in the destination that preserves its meaning. *Containment isolation* applies to an opaque payload: a container the base station forwards without interpreting must not carry a fault beyond the UE that sent it. The first is a property of the schema pair and the converter between them; the second is a property of the fault domain the boundary maintains when it forwards onto shared transport. A boundary can satisfy either and violate the other, and the taxonomy of §3.5 follows that split, with Type-I and Type-II as value-space failures and Type-III as a containment failure.

Table 1 contrasts this boundary-centric trigger condition with prior testing scopes confined to one protocol context. The comparison is intra-domain and does not claim disjointness from the broader interface-mismatch pattern discussed in §8.

3.3. Protocol Boundaries in the gNB

The architecture in Figure 1 induces the concrete translation surfaces studied in this paper. For typed-field mediation, we examine boundary pairs where values accepted under one schema are mapped or re-encoded into another: RRC↔NGAP for UE-originated information forwarded to the core, F1AP↔NGAP for causes raised on the DU–CU interface and propagated toward the core, E1AP↔NGAP for CU-UP/CU-CP causes mediated toward the core, and RRC↔S1AP in the LTE eNB. Endpoints operate within one schema, while the base station bridges two. The NAS-over-S1AP/NGAP forwarding path adds the opaque-container boundary analyzed for containment failures.

3.4. Why the Translation Surface Exists

The translation surface exists because protocol-local correctness does not imply cross-protocol compatibility. For value-space failures such as RRC→NGAP, F1AP→NGAP, and RRC→S1AP, the divergence is visible in published specifications: source and destination schemas are defined independently, on separate release cycles, with no complete normative mapping table. Individual crashes or corruptions are implementation-mediated, but the compatibility gap each implementation must bridge exists upstream of any single codebase.

Three properties explain why the surface persists. First, incompatible schema designs allow one schema to define values the other cannot represent or assigns a different meaning. The RRC/NGAP `establishmentCause` pair illustrates this directly: RRC admits spare values that NGAP cannot encode. Second, there is no cross-schema type safety. Generated ASN.1 code enforces one schema at a time; once decoded values enter C/C++ mediation code, they are commonly represented as plain integers, C enums, or passed through `static_cast` between unrelated enum types, then guarded by sentinels, range checks, or switch statements. The compiler cannot warn that a value decoded from one schema is not encodable or semantically equivalent in another. Third, spot patches move the bug, not the class.

For opaque forwarding, the transparency mandate gives the RAN edge no role in validating NAS, directly exposing core-network parsers to pre-authentication radio inputs. When the core fails to isolate the fault, the base station's shared S1AP/NGAP association and reconnection backoff timers amplify a single-packet core crash into a prolonged cell-wide outage.

3.5. Taxonomy of Translation Surface Attacks

The definition above yields two mechanism classes and three concrete failure types. This paper focuses on horizontal translation failures: values or containers crossing schemas or forwarding domains owned by different working groups in the same deployment. A second, vertical dimension can arise when the same schema evolves across releases; we defer that extension to Appendix D.

The first mechanism class is the value-space class.

- **Type-I: Value-space crash.** Schema A defines enumeration values absent from Schema B at the same boundary; the unrepresentable value triggers an encoder assertion or abort. Example: §5.1.
- **Type-II: Silent semantic corruption.** Both schemas define a value at the same index but with different



Figure 2. Threat model. A radio-proximate attacker UE (SDR, no SIM, no credentials) sends a single pre-authentication uplink message on Uu that the base station decodes and forwards to the core over NG/S1.

semantics; the base station silently forwards the misinterpreted value without error. Example: §5.2.

The second mechanism class is the containment class. An opaque inter-layer payload is forwarded as the specification requires, but the forwarding path fails to preserve isolation at the correct UE or transport context.

- **Type-III: Transparent forwarding without containment.** A base station transparently forwards an unauthenticated opaque payload onto a shared core-network association as mandated by 3GPP. This exposes core parsers to out-of-context radio inputs without RAN-edge filtering, and amplifies any resulting core-side crash into a cell-wide outage via shared transport teardown and RAN reconnection delays. Example: §5.3 (eNB forwards NAS to MME; MME responds with SCTP SHUTDOWN).

The three failure types share a common mediation-boundary trigger but arise through two mechanisms: cross-schema value-space failure (Types I–II) and cross-component containment failure (Type III). The taxonomy is not 5G-specific: LTE exposes the same mediation pattern at the RRC→S1AP boundary (silent-corruption finding, §5.2) and the same opaque NAS forwarding through S1AP (containment finding, §5.3). Appendix C gives the schema detail.

3.6. Threat Model

The radio-facing vulnerabilities demonstrated in §5 are reachable before user authentication or before AS security fully protects the relevant payload. We assume a radio-proximate attacker with an SDR and a modified UE stack, with no SIM, credentials, or cooperation from legitimate UEs or the core network.

For Type-I and Type-II value-space failures, the attacker sends a schema-valid RRC message carrying a boundary-divergent value. Triggers are delivered on UL-CCCH/SRB0, or on SRB1 before the AS security mode completes, via a single `RRCSetupRequest`, `RRCConnectionRequest`, or equivalent early uplink message. SRB0 is not associated with a PDCP entity (TS 36.323 §4.2.1), and AS integrity protection is not yet active at this point. For Type-III containment failures, the attacker sends an early NAS-bearing uplink message; the base station forwards the NAS body opaquely toward the core, and the attack targets whether any resulting failure remains confined to the triggering UE or propagates over the shared S1AP/NGAP association. Figure 2 summarizes the topology.

Impact amplification. Radio confidentiality does not protect against either class. For mapped fields, the base station decrypts before re-encoding, so a value valid under the source schema remains invisible until translation. For opaque payloads, transparent forwarding bridges unauthenticated Uu traffic onto the trusted S1/NG link, while the base station amplifies downstream core failures both spatially (across all UEs multiplexed on the shared SCTP association) and temporally (via multi-second SCTP reconnection backoff timers).

4. Methodology

CELLBREAKER operates in three phases, shown in Figure 3: static *boundary analysis* (§4.3), dynamic *boundary-targeted testing* (§4.4), and *multi-tier validation* (§4.5).

4.1. Challenges and Design

CELLBREAKER is built around four obstacles that make translation surface vulnerabilities hard to find, each shaping a phase. First, no single normative artifact enumerates cross-schema consistency constraints across 3GPP working-group boundaries, so Phase 1 enumerates corresponding ASN.1 types across the boundary and computes value-space divergence (§4.3). Second, opaque containers are invisible to enum analysis because NAS PDUs appear as `OCTET STRING`, so Phase 1's second pass enumerates opaque containers and protocol-discriminator bytes used for containment tests (§4.3.2). Third, commercial firmware schema versions are unobservable in deployed devices, so Tier 4 supply-chain source-level verification against the AGPLv3 Radisys BSP grounds opaque-deployment findings in inspectable source (Appendix D). Fourth, silent corruption emits no crash signal, so matched attack/control seeds, packet-level capture, and cross-implementation comparison detect divergence that crash-oriented fuzzing misses (§4.5).

4.2. Operational Model

To operationalize the translation-surface definition in §3.2, CELLBREAKER represents each candidate boundary using three recovered artifacts from the schema pair and the implementing codebase.

Definition 1 (Translation Surface)**.** *For a pair of ASN.1 schemas (S_{src}, S_{dst}) instantiated by a base-station codebase B, the* operational translation surface *is the tuple $\mathcal{T}_B = \langle M, C, F \rangle$, where M is the type correspondence between source and destination schemas extracted from B's cross-boundary code paths; C is the C-level mapping policy applied by B between decoder and encoder, drawn from $\mathcal{P} = \{\text{CAST}, \text{ASSIGN}, \text{SENTINEL}, \text{SWITCH}, \text{ASSERT}\}$; and F is the set of opaque containers, typically `OCTET STRING`s, that B forwards transparently across the boundary.*

The tuple $\mathcal{T}_B$ is operational rather than sound-and-complete: M, C, and F are recovered by schema extraction,

code-pattern matching, and analyst-assisted tracing. They prioritize boundary candidates for testing; they do not certify the absence of translation-surface vulnerabilities.

Operational checks. The three checks used by CELLBREAKER correspond directly to the taxonomy in §3.5. **C1** targets Type-I value-space failures by flagging source values that fall outside the destination schema and then checking whether the implementation policy lets them reach an encoder or a schema-miscalibrated guard. **C2** targets Type-II semantic-mapping failures by flagging numeric indices that both schemas accept but label differently. **C3** identifies Type-III candidates by flagging forwarding edges e that transparently carry opaque UE payloads over a shared S1AP/NGAP association.

4.3. Phase 1: Cross-Schema Boundary Analysis

Phase 1 runs three checks C1–C3, corresponding to the three failure types defined in §3.5 and operationalized in §4.2. The checks share the recovered artifacts (M, C, F) and are combined in the divergence-and-containment funnel (§6.1; per-boundary breakdown in Appendix A).

4.3.1. Value-Range and Semantic-Mapping Checks (C1, C2). Step 1: Extract corresponding types. We identify ASN.1 types in S_{src} that have a counterpart in S_{dst} by matching type names (e.g., RRC `EstablishmentCause` ↔ NGAP `RRCEstablishmentCause`), tracing C-level assignments, and following ITTI message flows.

Step 2: Compute value-space divergence and semantic mismatch. Write values(t) for the set of PER-visible enumeration indices in type t, base and extension values alike; idx(t) for the same index set viewed as the domain of named entries; and name$_S(v)$ for the schema-local label at index v in schema S.

Definition 2 (Value-Space Divergence)**.** *For a pair of corresponding* `ENUMERATED` *types (t_s, t_d), the value-space divergence set is*

$$D(t_s, t_d) = \mathit{values}(t_s) \setminus \mathit{values}(t_d). \tag{1}$$

Check C1 reports values in $D(t_s, t_d)$ that the source schema can encode but the destination schema cannot represent.

Definition 3 (Semantic Mismatch)**.** *For the same pair (t_s, t_d), the semantic-mismatch set is*

$$S(t_s, t_d) = \{\, v \mid v \in \mathit{idx}(t_s) \cap \mathit{idx}(t_d), \; \mathit{name}_{S_{src}}(v) \neq \mathit{name}_{S_{dst}}(v) \,\}. \tag{2}$$

Check C2 reports indices in $S(t_s, t_d)$ that both schemas define but label differently, so that a value survives translation with its meaning changed.

For example, at index 7, name$_{S_{\text{src}}}(7)$ = `spare1` while name$_{S_{\text{dst}}}(7)$ = `mo-ExceptionData`. We focus on enumerations; the approach generalizes to `BIT STRING` size constraints and `INTEGER` ranges (§9). Algorithm 1 lifts both checks to the type level.

Algorithm 1 DIVERGENCEANALYSIS: enumerate divergence and semantic-mismatch candidates for a schema pair instantiated by codebase B.

Require: Schemas $(S_{\text{src}}, S_{\text{dst}})$, codebase B
Ensure: Candidate set P, each entry annotated with its policy and reachability

$P \leftarrow \emptyset$
for all $t_s \in T_{\text{src}}$ with kind(t_s) = `ENUMERATED` **do**
  $t_d \leftarrow M(t_s)$ ▷ normalized-name match in B
  if $t_d = \perp$ **then continue**
  end if
  $D \leftarrow$ values$(t_s) \setminus$ values(t_d) ▷ divergence set
  $S \leftarrow \{v : v \in \text{idx}(t_s) \cap \text{idx}(t_d),$ name$_{S_{\text{src}}}(v) \neq$ name$_{S_{\text{dst}}}(v)\}$ ▷ mismatch set
  for all $v \in D \cup S$ **do**
    $p \leftarrow C(t_s, v)$ ▷ *analyst*: policy lookup in B
    $r \leftarrow$ SCORE(REACH(t_s), p) ▷ *analyst*: Table 2
    $P \leftarrow P \cup \{(t_s, t_d, v, p, r)\}$
  end for
end for
return P

Step 3: Verify intermediary range checks. For each divergent value, we statically check whether the C code between decoder and encoder contains a range check that excludes it. If the range check uses the C enum's `LAST` sentinel (common pattern), we verify whether `LAST` is calibrated to S_{src}, S_{dst}, or neither.

	CAST	ASSIGN	SENTINEL	SWITCH	ASSERT
Pre-auth	**T1**	**T1**	T2	T3	T3
Post-auth	**T1**	T2	T2	T3	T3
Internal	T2	T3	T3	T3	T3

TABLE 2: Analyst rubric SCORE(REACH, p) applied in Algorithm 1. Rows are the procedural reachability of the source type (REACH(t_s)); columns are the C-level mapping policy $C(t_s, v) \in \mathcal{P}$. The policy class is what predicts the failure mode: CAST yields silent corruption, ASSERT causes a process abort, and SWITCH with a sentinel defaults to a safe local rejection (§6). The tiers rank only whether a divergent value survives to downstream encoding, not severity: ASSERT is T3 because the guard filters the value out, and the process abort it raises instead is the availability failure of §5.1, classified dynamically in Phase 2.

Our *enum divergence analyzer* implements the schema-side steps of Algorithm 1 on each codebase's ASN.1-generated C++ headers, materializing M by normalized-name matching and emitting divergence set D and mismatch set S. The C-level mapping policy $C(t_s, v)$ is recovered by analyst-guided static analysis of the decoder-to-encoder call chain, classifying each call site into one of the five elements of $\mathcal{P}$ (Table 2). In OAI and srsRAN, `AssertFatal` aborts the process on guard failure, so the rubric places ASSERT in T3 for downstream filtering while Phase 2 classifies the resulting dynamic outcome as CRASHED (§5.1). We scope this paper to `ENUMERATED` types (§9). The procedural reachability

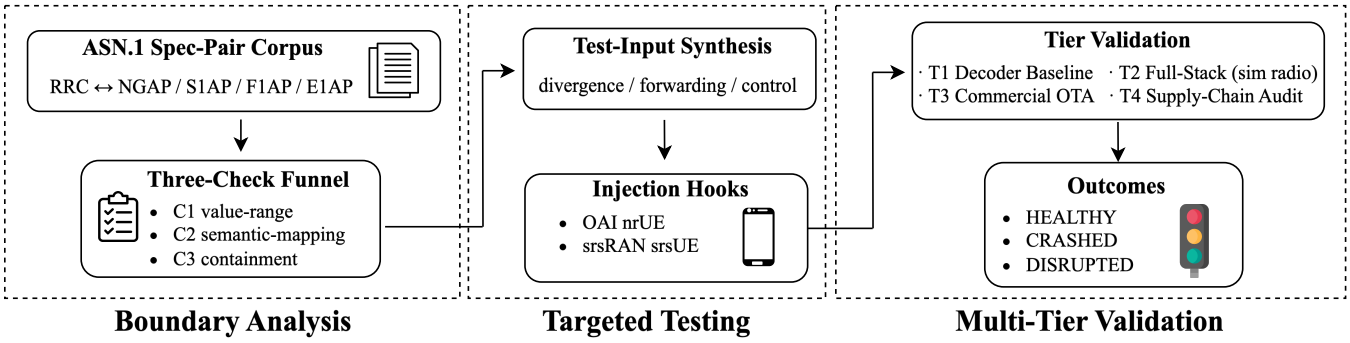


Figure 3. CELLBREAKER methodology pipeline: Phase 1 boundary analysis, Phase 2 boundary-targeted testing, Phase 3 multi-tier validation; re-applied to LTE RRC ↔ S1AP in §4.6.

REACH(t_s) is likewise established by analyst tracing of the codebase call graph: *pre-auth* if t_s is decoded before AS security mode is complete, *post-auth* if after, and *internal* if reachable only from CU/DU-internal handlers. Automation boundary and engineering cost are discussed in §9.

4.3.2. Containment Check (C3). C3 instantiates the Type-III containment condition from §3.5 using the operational artifacts in §4.2.

Definition 4 (Containment Failure Condition)**.** *For an opaque forwarding edge e carrying a container in F, the C3 containment condition holds when*

$$\text{TRANSPARENT}(e) \wedge \text{SHARED}(e) \wedge \neg\,\text{ISOLATION}(e). \quad (3)$$

The three predicates capture the forwarding path and its observed outcome. TS 36.413 §8.6.2.3 and TS 38.413 §8.6.1.2 mandate that eNBs/gNBs forward the NAS-PDU IE "without interpretation," so TRANSPARENT(e) holds by specification. SHARED(e) holds when multiple UEs are multiplexed over the same S1AP/NGAP SCTP association. These two predicates identify the forwarding paths exercised by C3. The final predicate, $\neg$ ISOLATION(e), is evaluated dynamically: it holds when a payload from one UE causes an observable effect on the shared association rather than remaining confined to that UE.

Accordingly, a forwarding edge satisfying TRANSPARENT$(e) \wedge$ SHARED(e) is selected for containment testing, and a Type-III failure is confirmed when dynamic testing additionally establishes $\neg$ ISOLATION(e). For NAS forwarding, we exercise this condition using a well-formed container carrying an out-of-context PD and observe whether the resulting core-side handling affects the shared S1AP/NGAP association.

The C1/C2 enum-divergence sweep across four 5G NR and LTE boundaries narrows the candidate set to 30 HIGH-priority pairs: a filter on testing scope rather than a prediction that any given pair is exploitable (§6.1, Appendix A); C3 enumerates opaque forwarding edges separately for containment testing.

4.4. Phase 2: Boundary-Targeted Testing

For each translation surface point from Phase 1, CELLBREAKER generates a targeted test case and injects it into a running base station.

4.4.1. Test Case Synthesis. Unlike mutation-based fuzzing, CELLBREAKER synthesizes inputs that are *precision-guided* by the boundary analysis:

Divergence test inputs. Start with a structurally valid UL RRC message, modify only the field corresponding to the divergence point (setting it to a value in the divergence set), and encode in UPER. The resulting message passes the RRC decoder and reaches the translation boundary, where it triggers one of three outcomes depending on the C-level mapping policy: a range-check assert that aborts the process, an encoder failure when the destination schema cannot represent the value, or a silent semantic remapping when both schemas accept the index but assign it different meanings.

Forwarding test inputs. For controlled testbeds, construct a NAS PDU with an out-of-context Protocol Discriminator (e.g., 0x03, circuit-switched Call Control, no meaning in EPS/5GS) and embed it in a structurally valid UL-DCCH message. Dynamic validation then evaluates $\neg$ ISOLATION(e) by checking whether downstream handling of the forwarded payload disrupts the shared association.

We supplement divergence-targeted test inputs with three categories of untargeted test inputs: *semantic mutations* (field-level ASN.1 variations), *binary mutations* (bitflip, truncation, byte replacement), and *boundary inputs* (empty, all-zeros, oversized). Across 768 such inputs on the LTE srsENB chain, none reached the translation surface; the one crash was a decoder-side type confusion.

4.4.2. Injection Infrastructure. We implement UL injection hooks in the OAI nrUE (software UE):

- `RRC_REPLACE_UL_CCCH`: replaces `RRCSetupRequest` with attacker-controlled bytes before transmission on SRB0 (pre-auth).
- `RRC_REPLACE_UL_DCCH`: replaces UL-DCCH messages on SRB1, indexed by a position counter `SKIP=`N that selects which of the UE's outgoing UL-DCCH PDUs is replaced.

For 5G NR, the patched nrUE and unmodified gNB run in OAI's rfsimulator mode (TCP-based IQ transport). For LTE, we implement an analogous `RRC_REPLACE_UL_CCCH` hook in srsRAN 4G's srsUE and use the ZMQ radio driver with srsENB and srsEPC. Both setups preserve the encoded protocol bytes delivered to the base station while avoiding RF-channel nondeterminism.

4.4.3. Test Harness Automation. The test-case runner manages the full lifecycle: starting a fresh base station, injecting each test input via a per-input UE restart, monitoring for crashes (process exit, assertion failures, signal-induced termination) and S1AP/SCTP disruptions (for forwarding-gap inputs), classifying results (HEALTHY / CRASHED / DISRUPTED / false-positive), preserving crash artifacts, and logging to CSV with resume support.

4.5. Phase 3: Multi-Tier Validation

We validate boundary-targeted test inputs at four tiers. Tiers 1–3 are *dynamic* testing tiers with increasing deployment realism; Tier 4 is an *orthogonal* static-analysis tier used when the deployment is opaque (commercial firmware without symbols), responding to the third challenge (§4.1).

Tier 1 (Decoder Baseline). Test inputs go directly to the target's ASN.1 decoder under ASAN/UBSan. This tier is a negative control, and its role is to bound what the decode step alone can expose: a translation-surface fault is created after decoding, so it cannot appear here, and a finding at this tier would be a decoder bug rather than a translation one. The campaign and its zero-finding result are reported in §6.2.

Tier 2 (Simulated-Radio Full-Stack). Test inputs are promoted to ZMQ full-stack delivery via a patched UE that completes the lower layers, exercising the full decode-and-handle path without RF hardware in same-stack (OAI UE→OAI gNB) and cross-stack (OAI UE→srsRAN gNB) configurations. Its role is the converse of Tier 1's: it is the first tier at which the mapping and re-encoding steps run, and therefore the first at which a translation-surface fault can appear.

Tier 3 (Commercial OTA): the test input is transmitted over the air by a USRP-driven UE against a Baicells Nova 430H commercial CBRS small cell (Qualcomm SoC, Radisys BSP 9955 Rel 9.3): a direct way to confirm that a finding triggers in a deployed environment with a non-open-source ASN.1 compiler.

Tier 4 (Supply-chain source verification): when a finding manifests on an opaque deployment, we ground it via independently published source from any vendor in the deployment's supply chain. For the Baicells commercial DoS, this reduces to cross-referencing the runtime-observed NAS forwarding path against the AGPLv3 Radisys eNB application layer embedded in the firmware. The source confirms the transparent NAS forwarding path; the absence of a bounded discriminator check is recorded only as defense-in-depth context. Unlike Tier 1–3 (dynamic), this static step makes the source-level evidence *independently inspectable* without firmware or hardware access.

Interpretation. We treat the tier outcomes as evidence weights. Finding the same pattern only in OAI would suggest a vendor-specific implementation bug; finding it across both open-source stacks and a commercial deployment is instead consistent with a boundary-level structural issue, although our sample cannot exclude shared engineering conventions as an alternative explanation. The three codebases mediate this boundary in three different ways (§6.3), so what recurs across them is the unguarded boundary rather than the code that fails to guard it.

4.6. Cross-Generation Validation

To test whether the translation surface is specific to 5G NR or a broader 3GPP architectural property, we apply the same methodology to LTE. The LTE eNB translates between RRC (TS 36.331) and S1AP (TS 36.413) at the Uu→S1 boundary, the LTE analogue of the 5G NR Uu→NG boundary. We run the divergence analyzer on the LTE schema pair, generate LTE-specific divergence and forwarding test inputs, and inject them into both srsRAN 4G's srsENB (Tier 2, simulated radio) and the Baicells Nova 430H (Tier 3, OTA). If the same value-space or containment patterns produce failures in both LTE and 5G NR (across open-source and commercial stacks), the translation boundary is a cross-generation property rather than a single-implementation bug.

5. Findings

We present three translation-surface findings discovered by CELLBREAKER, spanning two protocol generations (5G NR and LTE), two implementation strategies (assertion vs. silent pass-through), and two vendor classes (open-source and commercial): one value-space exploit, one containment exploit, and one wire-level silent-corruption existence proof. Each finding is accompanied by a matched control experiment that isolates the divergent value as the root cause.

5.1. Pre-Authentication gNB Crash (5G NR, Type-I)

5.1.1. Vulnerability. A single 6-byte `RRCSetupRequest` carrying `establishmentCause = 12` (`spare4`) deterministically crashes the OAI 5G NR gNB across our observed trials. The crash occurs at the RRC→NGAP translation boundary: the RRC decoder accepts the value (valid in the 16-value RRC enum), but the RRC-to-NGAP mediation logic cannot map it into the NGAP value space (only 11 values are defined in NGAP TS 38.413 v15.8.0). In OAI, value 12 aborts at a schema-miscalibrated `AssertFatal` guard; value 11 reaches the NGAP APER encoder and fails there. Type-I instantiates the C1 value-range check of §4.3, validated by the NGAP APER encoder unit test described below and by Tier 2 full-stack OAI rfsimulator runs against the 5G NR mainline. OAI is one of the most widely used open-source 5G NR

RAN stacks in O-RAN testbeds, private-5G deployments, and vendor/operator R&D labs; the pre-authentication, cell-wide crash is reachable by any UE that can transmit a PRACH preamble, against any deployment running it.

5.1.2. Crash Path. The desynchronization across the three representations (RRC ASN.1 fixed 16-value enum, C-enum with `LAST=12`, NGAP ASN.1 extensible 11-value enum) is detailed in §3.1; here we trace the crash path. The sole attacker-controlled divergent input is the 6-byte `RRCSetupRequest` (step 1); the subsequent `RRCSetupComplete` (step 4) carries the stored cause to the NGAP boundary.

1) Rogue UE completes RACH and sends `RRCSetupRequest` with `establishmentCause = 12` (`spare4`).
2) The `asn1c` UPER decoder accepts it: valid in the fixed 16-value enum.
3) The handler stores `UE->establishment_cause = 12`.
4) After `RRCSetupComplete`, the C-side range check `cause < NGAP_RRC_CAUSE_LAST` (`LAST = 0x0C` on 2026.w16) **fails** ($12 < 12$ is false) and triggers `AssertFatal`: process abort, entire cell down.

All five unmapped spare values (11–15) crash the gNB: on 2026.w16 value 12 fails the intermediary assertion ($12 < 12$); value 11 passes ($11 < 12$) but later fails the NGAP APER encoder; both share the root cause (RRC value space exceeds NGAP's). The OAI source at `rrc_gNB_NGAP.c:218` (2026.w16) documents the unvalidated assumption (§3.4):

```
/* Assume that cause is coded in the same way in
   RRC and NGap, just check that the value is in
        NGap range */
AssertFatal(UE->establishment_cause <
    NGAP_RRC_CAUSE_LAST, ...);
```

On 2025.w05 (`rrc_gNB_NGAP.c:166`), `LAST = 0x11` and the C enum was non-contiguous (gap 0x0A–0x0F); value 12 passed the range check and crashed at the NGAP APER encoder downstream. The sentinel `NGAP_RRC_CAUSE_LAST` is calibrated to the C enum size, not the NGAP ASN.1 encodable range (0–10). The specification anticipates the input: TS 38.331 §6.2.2 states that "gNB is not expected to reject an `RRCSetupRequest` due to unknown cause value being used by the UE," and sets the same expectation for the three other request messages that carry a cause. The failure is therefore not a strict-rejection policy; the gNB is expected to accept the message, and OAI aborts the process instead.

5.1.3. Control Experiment. We isolate the root cause using an APER encoding unit test that invokes OAI's NGAP ASN.1 encoder for each of the 16 possible `EstablishmentCause` values: values 0–10 encode successfully (root + `notAvailable` extension), while values 11–15 (`spare5...spare1`) fail (exactly the five spares without NGAP mapping) (Appendix B gives the per-value result). Injecting the matched control seed (`cause=3`, `mo-Signalling`) via the identical mechanism produces successful encoding and no crash, proving the divergent value, not the injection method, is the cause.

Properties. Pre-auth (post-RACH only) single 6-byte `RRCSetupRequest`; cell-wide DoS; classified as Type-I; deterministic across two OAI versions 14 months apart via the two paths of §3.4. srsRAN Project does not crash at the RRC→NGAP boundary because its CU-CP uses an explicit `switch/case` with `default: not_available` (703-seed campaign yields zero crashes); we return to cross-boundary value-space variants in srsRAN's CU-internal cause translation in §6.1.

5.2. Silent Semantic Corruption (LTE, Type-II)

5.2.1. Vulnerability. This finding exercises the same translation surface concept on the 4G LTE eNB, which is the non-disaggregated simplification of the gNB shown in Figure 1: RRC and S1AP both reside inside a single eNB process, so the RRC→S1AP boundary becomes an in-process cast rather than an F1AP-mediated CU-CP internal mapping. Injecting `establishmentCause = 7` (`spare1` in LTE RRC) into srsRAN 4G's eNB causes **no crash and no error-level log at default verbosity**. The value is silently forwarded to the MME as `mo-ExceptionData` (a legitimate S1AP extension value for IoT exception-data reporting). The semantic corruption is not reflected in error-level logs or alarms observable in our tested setup.

5.2.2. The LTE Enum Mapping. LTE RRC (TS 36.331) defines `EstablishmentCause` as a fixed 8-value enum (0–7); S1AP (TS 36.413) defines `RRC-Establishment-Cause` as an extensible enum with 5 root values plus 3 extension values. Indices 0–6 align, but index 7 is the divergence point: RRC reserves it as `spare1` while S1AP defines it as `mo-ExceptionData` (IoT exception-data reporting, Rel-13).

5.2.3. Root Cause. srsRAN 4G translates the value via a direct C-style cast at `rrc_ue.cc:567`:

```
s1ap_cause.value =
  (asn1::s1ap::rrc_establishment_cause_opts::
      options)
   establishment_cause.value;
```

No switch/case, no range check, no semantic validation. The cast succeeds for all values 0–7 because both enums have at least 8 entries; the *meaning* of index 7 differs across schemas.

5.2.4. Live Verification and Control. We inject via a patched srsUE through the ZMQ radio driver and capture the eNB's S1AP hex dump. The attack seed (`cause=7`) yields IE `0x82` (extension byte, decoded as `mo-ExceptionData`); the matched control (`cause=3`, identical injection mechanism, schema-valid) yields IE `0x30` (root byte, decoded as `mo-Signalling`). The divergent value is the sole difference.

5.2.5. Security Impact. Impact scope. The corruption is wire-level: every injected frame carries S1AP IE 0x82 (`RRC-Establishment-Cause = mo-ExceptionData`) verbatim to the MME. Neither of our test EPCs (srsEPC, Open5GS) acts on this Rel-13 IE, and a source audit of Open5GS places the stopping point at the core's application boundary: the generated S1AP codec decodes the IE, no MME application path reads the decoded value, and a GTPv2-C *MO Exception Data Counter* field is declared but never populated. The value is not rejected; this implementation wires no consumer for it. The finding is an existence proof of silent semantic corruption at the RRC→S1AP boundary. MME-side exploitation against production EPCs that consume the IE for billing, lawful intercept, or admission control is left to future work (§9).

Properties. Pre-auth single 6-byte `RRCConnectionRequest`; deterministic (C-style cast); silent S1AP signaling corruption with no crash, no error-level log at default verbosity, and no alarm observable in our tested setup; classified as Type-II (wire-level existence proof; MME-side impact deferred, §9).

5.3. Commercial DoS via NAS Forwarding (LTE, Type-III)

5.3.1. Vulnerability. A single 10-byte pre-authentication OTA frame carrying an out-of-context NAS Protocol Discriminator (0x03, circuit-switched Call Control) triggers a 30-second cell-wide outage in our commercial Baicells Nova 430H / Open5GS deployment. The root-cause crash resides in the Open5GS 2.7.0 MME, which terminates its process and emits an SCTP SHUTDOWN upon receiving PD=0x03 (replacing Open5GS with srsEPC on the same Baicells unit causes no teardown). Table 3 records whether the association went down; a separate run observed the core itself, where the payload aborted the MME process in all eight delivered trials, each incrementing the container's restart count, with no abort for the PD 0x07 control and no faults across 57 ordinary attaches on the same stack. The eNB neither crashes nor deviates from the specification: it forwards the payload as TS 36.413 §8.6.2.3 requires (§2.4), and the single SCTP association the MME tears down is the one TS 36.412 §7 mandates per eNB–MME pair. Its one unconstrained contribution is temporal: it waits a fixed 30 s before sending SCTP INIT, which sets the outage duration. Our testbed has one eNB, so the measured outage spans that cell.

5.3.2. Packet-Level Proof. We capture the S1AP traffic between eNB and MME during injection (Figure 4):

The pcap shows the NAS payload with PD=0x03 (`03 41 00 00 00 00 00`) verbatim inside the S1AP `id-NAS-PDU` IE, confirming that the eNB forwarded the NAS payload unchanged; the externally visible failure is transport-association teardown rather than an S1AP-level rejection.

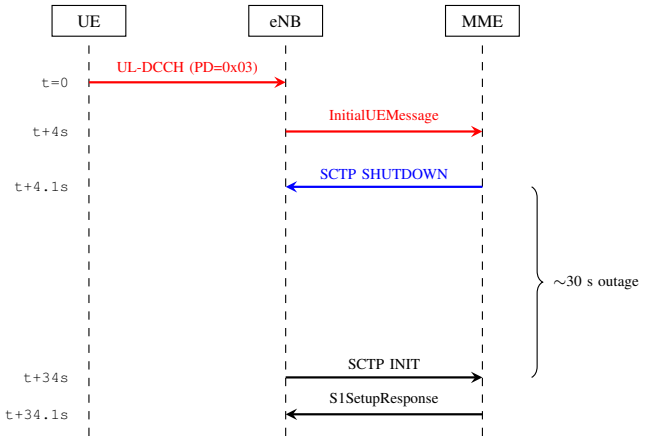


Figure 4. Commercial DoS attack sequence. The eNB forwards the invalid NAS PDU (PD=0x03) to the MME, which tears down the SCTP association. All UE services are disrupted for 30 seconds.

Phase	Trials	Seed	Outcome
A: Reproducibility	5	PD=0x03	all disrupted
B: Control	5	PD=0x07	none disrupted
C: PD variants	6	0x00/0x05/0x0F	all disrupted
Total	**16**	*every attack disrupted; no control did*	

TABLE 3: Controlled reproduction of the commercial DoS against the Baicells unit and the Open5GS core it was deployed with. §5.3 reports the second-core comparison.

5.3.3. Controlled Reproduction. We run a strict controlled experiment in three phases (Table 3, 16 trials total): Phase A repeats the attack seed (NAS PD = 0x03) five times to confirm reproducibility; Phase B substitutes a matched control seed (NAS PD = 0x07, a valid EMM Protocol Discriminator differing from the attack seed by exactly one byte) five times to isolate the divergent byte as the root cause; Phase C sweeps three additional out-of-context Protocol Discriminator values (0x00, 0x05, 0x0F) across six trials to characterize the trigger surface.

The control seed differs by exactly one byte (NAS PD = 0x07, valid EMM), confirming that the out-of-context Protocol Discriminator, rather than the injection mechanism, triggers the S1AP disruption. Outage duration is consistent at 30 ± 1 s across all 11 disruption events of Table 3. The eNB's next SCTP INIT follows the SHUTDOWN by 30 s. Our firmware analysis confirms that this delay is governed by the eNB-side `WR_TMR_MME_SETUP_RSP` timer (`wr_tmr.c:1793`), rather than by core recovery.

Against Open5GS, the association is torn down; substituting srsEPC on the same base station leaves it up, with the payload logged as an Attach Request and the protocol discriminator never validated. Whether the disruption occurs is therefore MME-dependent. The base station's contribution does not change: it carries an unauthenticated container from a single UE onto the association the cell shares.

Properties. Pre-auth 10-byte UL-DCCH NAS PD=0x03 against the Baicells Nova 430H, yielding a 30 s cell-wide

Implementation	Gen.	Forward path	PD check?
Radisys eNB (Baicells)	LTE	`wrUmmS1apBldInitUePdu`	No
srsRAN 4G eNB	LTE	`s1ap::ue::send_initialue...`	No
OAI eNB	LTE	`rrc_eNB_send_S1AP_NAS_FIRST_REQ`	No
srsRAN Project gNB-CU-CP	5G NR	`rrc_setup_procedure.cpp:161`	No
OAI gNB	5G NR	`rrc_forward_ue_nas_message`	No

TABLE 4: NAS Protocol-Discriminator handling in the forwarding path of five base-station implementations. Because verbatim forwarding is specification-mandated, all surveyed stacks expose downstream core parsers directly to unauthenticated, out-of-context radio NAS headers.

S1AP outage per injection; deterministic across the 11 trials of Table 3, with no disruption across 5 matched controls; classified as Type-III.

5.4. Cross-Finding Analysis

Our findings demonstrate two distinct boundary hazards: implementation flaws during cross-schema translation (OAI and srsRAN 4G) and architectural exposure/amplification during spec-compliant transparent forwarding (Baicells/Open5GS). The three findings cover both failure classes defined in §3.5: two are value-space failures, and one is a containment failure. Their validation strength differs, however, with the LTE silent-corruption case supported only by wire-level evidence. Together they span two protocol generations and three implementation strategies. None of the three is reachable by testing that stops at a single schema's decoder (§6.2).

Cross-implementation C3 forwarding condition. The Type-III forwarding condition is not unique to Baicells. Static source audit of four additional base-station implementations finds the same pattern (Table 4): the base station transparently carries the NAS PDU into the `InitialUEMessage` or `UplinkNASTransport` NAS-PDU IE. We runtime-confirm a Type-III failure in the Baicells/Open5GS deployment; for the other four stacks, the audit establishes the forwarding condition only, not a downstream shared-association failure or end-to-end DoS. The absence of a bounded Protocol-Discriminator check is reported only as defense-in-depth context.

The Type-III blind-forward code pattern is thus present in five stacks across three code bases spanning both LTE and 5G NR (Baicells runtime-confirmed; the other four source-confirmed). Because this forwarding is specification-mandated (§2.4), the security implication is not that every stack should parse NAS at the base station, but that transparent forwarding requires explicit containment at the S1AP/NGAP layer; we discuss compliant mitigations in §7.

Shared properties. Table 7 summarizes the three findings along the dimensions that matter for cross-finding assessment: which check of the §4.3 framework they instantiate, what an attacker needs to trigger them, and what evidence tier supports them.

Direction	Strategy	Loss
F1AP → NGAP (srsRAN)	41-entry lookup table	35/41 (**85.4%**)
F1AP → NGAP (OAI)	Hardcoded single value	41/41 (**100%**)
E1AP → NGAP (srsRAN)	34-entry lookup table	23/34 (**67.6%**)
NGAP → F1AP (srsRAN)	Reverse table (empty)	57/57 (**100%**)
NGAP → E1AP (srsRAN)	Reverse table (empty)	57/57 (**100%**)

TABLE 5: CU-internal cause translation losses across srsRAN and OAI. srsRAN rows are measured by directly invoking the compiled converter library over all source values; the OAI row is source-confirmed against OAI's hardcoded mapping (no compiled converter library exists). A source value is counted as lost when translation maps it to `unspecified`, hardcodes it to a single destination, drops it, or has no implemented reverse mapping rather than preserving a semantically equivalent destination cause.

6. Analysis

6.1. Structural Source of Translation Loss: CU-Internal Cause Translation

The three findings in §5.1–5.3 arise from two structural sources: schema divergence for Types I and II, and opaque forwarding onto shared transport for Type III.

F1AP (TS 38.473, Rel-17) defines `CauseRadioNetwork` with 41 values, while NGAP (TS 38.413, Rel-17) defines the same-named field with 57 values. At the numeric-index level, the two schemas are almost entirely misaligned: among the 41 F1AP indices, **only one non-sentinel value** (index 11, `cell_not_available`) shares the same name with NGAP. The remaining indices either denote different causes at the same number (e.g., F1AP index 1 = `rl_fail_rlc`, NGAP index 1 = `txnrelocoverall_expiry`) or require an implementation-specific mapping. Two independent implementations bridge this incompatibility differently, and **both degrade information**:

Structural lossiness. Exercising all source values through srsRAN's compiled converters and source-tracing OAI's hardcoded F1AP→NGAP mapping (Table 5) shows both implementations are lossy: every srsRAN forward direction collapses unmapped causes to `unspecified`, OAI maps all 41 input values to a single destination, and both srsRAN reverse directions are unimplemented; at the converter level, each unmapped F1AP cause is an individually exercisable silent-loss candidate (external reachability depends on the surrounding CU/DU control path; downstream AMF-side exploitation impact is deferred to future work). The lookup-table-with-massive-unspecified pattern recurs in `e1ap_cause_converters.cpp`, evidence that the mechanism generalizes across CU-internal boundaries. Three Type-II instances are detailed in Appendix C.

6.2. Where Existing Approaches Inject

Inside the base station, the uplink path is decode, store, map into the destination schema, re-encode. Type-I and Type-II faults are created at the map step, so an input reaches them only if it enters at or before the decode step. The question is therefore injection position, not coverage volume.

Decoder-only fuzzers stop at the first stage without reaching the map or re-encode steps. Stateful end-to-end campaigns run further but inject elsewhere: RANsacked [2] supplies already-formed S1AP and NGAP at the RAN–core interface from the base station's own position, establishing the association itself with S1Setup or NGSetup, and its systems under test are cores. An injector placed after the map step supplies the translated PDU directly, so the steps that create the fault never run. What the class requires is cross-schema awareness: a value the decoder accepts but the destination encoder cannot represent or reinterprets.

Our six decoder-only harnesses (Appendix E) return no translation-surface finding. That establishes that those harnesses do not reach this class; it does not establish that every stateful fuzzer would miss it, and we make no such claim.

Triaging the 30 HIGH-priority pairs. Our C1/C2 divergence analyzer scans all paired `ENUMERATED` types across four protocol boundaries (with RRC↔NGAP scanned in both srsRAN codebases) in 5G NR and LTE: 87 paired types, 36 (41%) divergent, 30 HIGH-priority, with NGAP↔F1AP contributing 44 paired (15 divergent), consistent with F1AP lagging NGAP's extensions (Appendix A). Across the validated campaign, CELLBREAKER confirms the three findings of §5.1–5.3. Converter-level enumeration additionally identifies 8 structural-lossiness directions across two CU-internal cause-translation surfaces (`F1AP↔NGAP`, `E1AP↔NGAP`) and four enum-cause families, including a cross-category rewrite at NRPPa where all four `CauseTransport` values translate into `misc::unspecified`; we include `F1AP→NRPPa` as a converter-coverage check rather than a main finding. Table 5 summarizes the major CU-internal directions; Table 10 expands them into 8 direction-level cases (plus 4 `F1AP→NRPPa` coverage rows). Together, these exercise every exported converter function in srsRAN's compiled `f1ap_cause_converters.cpp`.

Evidence levels and disposition. *Structural divergence* is established by the analyzer alone; two schemas disagree on a value space. *Implementation correctness loss* adds a deployed converter that drops or rewrites the value, shown by exercising the compiled converter or by tracing the mapping in source. *Demonstrated boundary failure* adds reachability by the §3.6 attacker and an observable crash, wire-level corruption, or loss of containment. Table 9 places all 30 candidates on these levels: seven were exercised, comprising 2 demonstrated boundary failures, 1 demonstrated safe, and 4 converter-level lossiness without demonstrated reachability. The other 23 were not; 6 are core-originated downlink fields outside the attacker model, 14 are dropped, unimplemented, or source-confirmed only, and 3 are matcher artifacts with no translation path to exercise. Three classes sit outside the present instantiation: `BIT STRING`/`INTEGER` divergences (deferred to the type-general extension), cross-protocol `static_cast` between unrelated typedefs (requires a typedef-aware analyzer), and policy classes outside $\mathcal{P}$ (hand-rolled converters with embedded validation, reported as manual-review-required).

6.3. Mediation and Downstream Handling Determine Failure Mode

The same translation-surface abstraction produces different externally visible failures depending on how the mediation path is handled (§1), one failure mode per check. OAI's schema-miscalibrated `AssertFatal` on RRC→NGAP aborts the process on any unmapped value: the Type-I (C1) cell-wide gNB *crash*. srsRAN 4G's unguarded C-style cast on RRC→S1AP instead puts a legitimate value of different meaning on the wire: the Type-II (C2) *silent semantic lossiness* (§5.2). srsRAN Project's CU-internal F1AP→NGAP lookup table with an `unspecified` sentinel exhibits the same converter-level pattern (external reachability deferred to future work, §6.1); a similar absence of local validation in OAI's `encode_ngap_cause()` is currently absence-of-defense evidence rather than an attacker-reachable finding. In the Baicells/Open5GS setup, the eNB's specification-mandated transparent forwarding carries the UE-originated container onto the shared S1AP/SCTP association; Open5GS tears it down, and the eNB's reconnection schedule sets the Type-III (C3) *30-second commercial outage*. A *vertical* (cross-release) dimension and the supply-chain implications of the commercial DoS are deferred to Appendix D.

7. Mitigation

We propose defenses at four layers, each targeting one root cause from §6.1–§6.3; the absence of normative cross-schema mappings is a 3GPP process-level gap rather than a single implementation oversight, and the four layers cover non-substitutable slices of it.

Specification level. The relevant 3GPP working groups should adopt, via Change Requests, normative enum-translation tables for every cross-protocol boundary (e.g., RRC↔NGAP, F1AP↔NGAP, E1AP↔NGAP). Where no semantic mapping exists (e.g., 35/41 F1AP causes), the spec should mandate explicit "unspecified" mapping and define receiver behavior; alternatively, a single ASN.1 module imported by both schemas eliminates definition-level divergence (C1). For opaque NAS forwarding (TS 36.413 §8.6.2.3 and TS 38.413 §8.6.1.2), the specifications should strengthen both error containment and input validation. NAS specifications should require MME/AMF implementations to handle unknown or out-of-context Protocol Discriminators as UE-scoped errors rather than node-level failures. As a defense-in-depth measure, S1AP/NGAP could additionally permit minimal validation of the unencrypted NAS header at the RAN edge to filter clearly invalid protocol types before forwarding.

Implementation level. For mapped fields (Types I–II), every cross-schema assignment should apply a policy from $\mathcal{P} = \{\text{CAST}, \text{ASSIGN}, \text{SENTINEL}, \text{SWITCH}, \text{ASSERT}\}$ that fails closed: a bounded `switch/case` with an explicit `default` mapping to a safe sentinel (`not_available`, `unspecified`) or rejecting the message. srsRAN Project's `switch/case` with `default: not_available` at the RRC→NGAP boundary is a deployed SWITCH+SENTINEL instance: divergent values sink to the sentinel, eliminating the Type-I crash path (§5.1). Unsafe patterns: `static_cast` between unrelated enum types, direct integer assignment, and C-style casts without range checks. For opaque forwarding (Type-III), the containment duty sits with the core: MME/AMF implementations should treat forwarded `NAS-PDU` containers as untrusted radio input, replacing process-aborting assertions with a per-UE `Error Indication` or silent discard. The base station can still cut the amplification by replacing fixed 30-second SCTP reconnection timers with sub-second probes (§5.3).

Tooling level. A static analyzer comparing paired ASN.1 schemas (cf. §4) is automatable as a CI/CD gate for the schema side; the $\mathcal{P}$-set conformance check on the C side requires per-codebase human review to flag unguarded casts and schema-miscalibrated assertions. For opaque NAS forwarding, static call-graph review inventories the transparent `OCTET STRING` paths that carry UE input onto shared S1AP/NGAP associations (§5.3).

Testing level. Reaching Type-I and Type-II failures requires testing that enters at or before RRC decoding and then exercises the cross-schema mapping and re-encoding path with boundary-aware inputs; decoder-only harnesses stop short of that path, and RAN–core-interface campaigns start past it (§6.2). Our methodology (§4) instantiates this approach for RRC↔NGAP and F1AP↔NGAP, and extends to joint RAN–Core testing that injects out-of-context opaque containers over the radio interface to evaluate both core-process survival and RAN-side SCTP recovery latency (§5.3).

Why all four. Spec fixes do not protect base stations or cores on current releases; implementation defaults do not prevent future schema drift; tooling cannot catch C-side cast bugs after the schema check passes (cf. Type-I) or dynamic transport-recovery delays; testing only catches bugs after they ship. Removing any layer leaves residual attack surface.

8. Related Work

8.1. RAN and Infrastructure Fuzzing

RANsacked [2] couples AFL++-driven harnesses with ASNFuzzGen-generated structure-aware mutators from ASN.1 schemas, Berserker [3] applies ASN.1-based RRC fuzzing, ORANClaw [27] fuzzes O-RAN E2 (with 5G-Spector [28] as the O-RAN-compliant Layer-3 attack-detection counterpart), and Trend Micro [29] reports ASN.1 decoder bugs in 5G core. LLFuzz [30] extends OTA fuzzing into baseband lower layers. These approaches predominantly surface *parser and memory-safety bugs*; the translation surface lies *after* the parser. Our Tier 1 decoder-only baseline returns no translation-surface findings, consistent with that boundary (Appendix E). Baseband fuzzers (5Ghoul [4], FirmWire [31], BaseSAFE [32]) target single-component UE-side corruption, orthogonal to the network-side cross-schema mediation studied here.

8.2. Cellular Protocol Testing

LTEFuzz [33] and DoLTEst [6] test *single-specification deviations* in the LTE control plane; the translation surface emerges when two *individually correct* specs collide at their boundary. Raza et al. [34] catalog LTE inter-layer logical-state mismatches; CELLBREAKER adds the schema-pair encoding/enum dimension and Type-III boundary-containment mechanism on top of that lineage. CITesting [35] exercises context-integrity violations within a single NAS/S1AP context rather than crossing the F1AP→NGAP boundary. 5GBaseChecker [8] and DIKEUE [36] target UE-side compliance via finite-state-machine analysis; Hermes [37] synthesizes finite-state machines from 3GPP specifications and model-checks them to surface ambiguities and UE-baseband deviations; ConSET [7] examines intra-schema cross-field semantic constraints on the UE side; the translation surface is the inter-schema dual on the gNB/eNB side. Sni5Gect [38] and 5Gone [9] provide OTA delivery but do not identify which content triggers cross-schema failures; GLaDoS [39] operates at the physical/protocol-DoS layer, not the schema-mapping mechanism studied here.

8.3. Specification-Based and Formal Analysis

BaseSpec [5] and BaseComp [40] compare baseband firmware against spec text; Chen et al. [41] show most security-relevant 3GPP change requests reflect specification design errors; P1 Security [42] documents cross-protocol attack paths. CellularLint [43] applies NLP to 3GPP NAS and Security Architecture specs to surface intra- and cross-document inconsistencies in spec prose, but does not reason about cross-schema value mappings at runtime. Formal protocol-logic analyzers (LTEInspector [44], 5GReasoner [45]) reason over single-protocol state, not cross-schema value mappings.

8.4. Commercial Small-Cell and Interface-Mismatch Work

Prior Baicells work targets web-management vulnerabilities (CISA ICSA-23-061-02 [46]); §5.3 instead shows how a spec-compliant commercial small cell exposes core-network parsers to radio-injected containers and amplifies a core crash into a prolonged cell-wide outage. The translation surface is the cellular instance of the broader interface-mismatch pattern surfaced in differential TLS parsing (Frankencerts [10]), HTTP request smuggling [11], [47], application-layer protocol confusion in TLS (ALPACA [48]), and ASN.1 bugs spanning decades [49], [50], [51], with LangSec [52] as theoretical lineage. The closest conceptual ancestor is Balduzzi et

al. [53] on ICS gateways (explicit translators); cellular base stations are implicit translators within a single binary, the mechanism is cross-schema value-space divergence rather than a parser bug, and the schemas are owned by different 3GPP groups on independent release cycles.

9. Discussion and Limitations

Generalization to Other Protocol Boundaries. The method generalizes along the definition of the surface: wherever a node terminates two independently maintained schemas and re-encodes between them, the same analysis applies. We instantiate it at four boundaries across two generations, and find divergence at each.

For cellular protocols, adding a boundary between schemas the analyzer already parses costs one entry in the pair table. NGAP↔XnAP, held out of the evaluation, returns 64 paired types, 13 divergent and 9 HIGH-priority, demonstrating front-end schema-analysis extensibility. Two further boundaries have the same shape. An eNB terminates S1AP toward the core and X2AP (TS 36.423) toward a peer eNB, a candidate cause-mapping path to trace; their `CauseRadioNetwork` enumerations disagree already at index 0, the semantic mismatch C2 would flag. An ng-eNB-CU terminates NGAP toward the core and W1AP (TS 37.473) toward its DU, the E-UTRA counterpart of the NGAP↔F1AP boundary we evaluate; most indices their `CauseRadioNetwork` enumerations share carry different names, and NGAP defines many causes W1AP cannot represent, so both C2 and C1 would apply. Taking either from a pair-table entry to a runtime finding would additionally need that implementation's ASN.1-generated headers, a traced decoder-to-encoder path, and an implementation-specific injector; that requirement, not the pair-table entry, is what bounds generalization today.

For non-cellular protocols, what is cellular-specific in CELLBREAKER is the instantiation: the schema extractors read ASN.1, the pair table names 3GPP protocols, and the reachability tiers are anchored to AS security mode complete. A port would supply a schema reader for the target representation and a reachability predicate for the target trust boundary. HTTP/2↔HTTP/1.1 is the closest analogue: a gateway decodes one framing and re-emits the other, and request smuggling arises where the two disagree on where a message ends [11]. X.509 poses the same question wherever a component re-encodes certificate fields into another credential representation [10]. We scope the contribution of this paper to the systematization of cross-schema testing in cellular base stations. Within cellular, the approach extends to non-`ENUMERATED` divergences and to runtime V1–V3 validation of the cross-release dimension (Appendix D).

Automation Scope and Analyst-in-the-Loop Stages. Algorithm 1 fully automates schema-side divergence detection across the protocol pair, the step that scales poorly under manual review and accounts for the bulk of candidate enumeration. This automated front-end makes large-scale cross-release and cross-protocol triage tractable, and surfaces the divergences that drive every confirmed crash in our evaluation. Two downstream stages remain manual, and neither is manual by necessity. Reachability tracing is a query over the call graph: walk from each decode site to the re-encode site that consumes the value, and record where the path sits relative to AS security mode complete. Policy classification is a per-codebase review of how each converter guards its inputs, and is partly mechanizable: the guards take a small number of recognizable code shapes, among them an unchecked cast between value spaces, a switch with or without a default, a sentinel comparison, and an assertion, so a pattern pass over the converter body can propose a classification for each call site. What resists automation is the residue, where a guard's adequacy turns on implementation semantics rather than on its shape, and those sites still need review. We have implemented neither, so both stages were carried out by hand in this work. That effort scales with the number of divergences to review, not with schema size: more types without more divergent pairs add no manual work.

Type Coverage of the Divergence Framework. The framework in §4.3 is type-parametric; we instantiate it for `ENUMERATED` divergences because they dominate the cause-translation surface and yield the highest-value HIGH-priority pairs. The template extends to `INTEGER` range and `BIT STRING` layout divergences, and to structural-encoding mismatches (e.g., BCD vs. TBCD for PLMN-Identity), via the same steps. Cross-IE or procedural divergences expressed implicitly in 3GPP prose fall outside the current instantiation; the modular pipeline can be extended via additional extractors.

Dynamic Validation Coverage. Dynamic pair-level validation focuses on the HIGH-priority pairs on the cause-translation surfaces, since these carry the strongest evidence of safety-relevant misinterpretation and are reachable from our testbed. Six deferred pairs are core-originated downlink fields outside the §3.6 UE-side attacker model; fourteen are dropped, unimplemented, or source-confirmed only. CELLBREAKER thus does not claim end-to-end dynamic confirmation on every candidate divergence; static evidence for deferred pairs is preserved through the same priority criteria.

What the Analysis Misses. The pairing step matches normalized type names by containment, and both of its error directions follow from that choice. False positives are visible and removable: three of the 30 HIGH-priority candidates (10%) pair unrelated concepts and are discarded at the analyst stage (§6.2). False negatives are not visible. A source and destination type that correspond semantically but share no name substring is never paired, so its divergence is never enumerated, and we have no ground truth against which to count such cases. Two further classes escape the current instantiation: divergences carried by `static_cast` between unrelated typedefs, which need a typedef-aware analyzer rather than a name matcher, and correspondences stated only in 3GPP prose. Reachability tracing is likewise manual, so a divergence whose decode site is not evident from the call graph can be tiered wrongly. The pipeline is an enumeration aid rather than a decision procedure; each of these gaps is a property of the matcher and the manual

stages, not of the boundary itself. The cross-implementation evidence carries a matching limit: each implementation we examined mediates this boundary in its own way, which shows the boundary is hard to get right rather than that the implementations share a lineage.

10. Conclusion

We formalize the *translation surface*: the base-station mediation boundary where values are mapped across schemas or opaque payloads are forwarded across protocol contexts, which prior cellular fuzzing and compliance testing do not systematically exercise. CELLBREAKER surfaces three boundary failures across LTE and 5G NR: an OAI gNB crash, an srsRAN silent corruption, and a shared-association DoS in a commercial Baicells/Open5GS deployment. Local fixes may address individual symptoms without eliminating the broader class; robust defenses must preserve value-space semantics across schema mappings and contain failures along opaque forwarding paths. Correctness within each protocol context is therefore insufficient; safety must be preserved across the mediation boundary.

Acknowledgments

We thank the anonymous reviewers for their valuable comments. This material is based upon work supported in part by the National Science Foundation (NSF) under Grant No. 2623205, 2234945, 2228617, 2120369, and 2129164. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the NSF.

References

[1] "OpenAirInterface 5G RAN," https://openairinterface.org/, 2024.

[2] N. Bennett, W. Zhu, B. Simon, R. Kennedy, W. Enck, P. Traynor, and K. Butler, "RANsacked: A domain-informed approach for fuzzing LTE and 5G RAN-core interfaces," in *ACM Conference on Computer and Communications Security (CCS)*, 2024.

[3] S. Potnuru and P. K. Nakarmi, "Berserker: ASN.1-based fuzzing of radio resource control protocol for 4G and 5G," in *IEEE International Conference on Wireless and Mobile Computing (WiMob)*, 2021.

[4] M. E. Garbelini, Z. Shang, S. Luo, S. Chattopadhyay, S. Sun, and E. Kurniawan, "5Ghoul: Unleashing chaos on 5G edge devices via stateful multi-layer fuzzing," *IEEE Transactions on Dependable and Secure Computing*, 2025.

[5] E. Kim, D. Kim, C. Park, I. Yun, and Y. Kim, "BaseSpec: Comparative analysis of baseband software and cellular specifications for L3 protocols," in *Network and Distributed System Security Symposium (NDSS)*, 2021.

[6] C. Park, S. Bae, B. Oh, J. Lee, E. Lee, I. Yun, and Y. Kim, "DoLTEst: In-depth downlink negative testing framework for LTE devices," in *USENIX Security Symposium*, 2022.

[7] Q. Huang, X. Wang, W. Guo, G. Gu, and H. Hu, "Semantics over syntax: Uncovering pre-authentication 5G baseband vulnerabilities," in *USENIX Security Symposium*, 2026.

[8] K. Tu, A. Al Ishtiaq, S. M. M. Rashid, Y. Dong, W. Wang, T. Wu, and S. R. Hussain, "Logic gone astray: A security analysis framework for the control plane protocols of 5G basebands," in *USENIX Security Symposium*, 2024.

[9] S. Erni, M. Kotuliak, M. Roeschlin, R. Baker, and S. Capkun, "5Gone: Uplink overshadowing attacks in 5G-SA," *arXiv preprint arXiv:2602.10272*, 2026.

[10] C. Brubaker, S. Jana, B. Ray, S. Khurshid, and V. Shmatikov, "Using Frankencerts for automated adversarial testing of certificate validation in SSL/TLS implementations," in *IEEE Symposium on Security and Privacy (S&P)*, 2014.

[11] J. Kettle, "HTTP desync attacks: Request smuggling reborn," Black Hat USA / DEF CON 27, 2019.

[12] Software Radio Systems, "srsRAN project," https://www.srsran.com/, 2024, open source O-RAN 5G CU/DU.

[13] 3GPP, "3GPP TS 38.331: NR; Radio Resource Control (RRC) Protocol Specification (v17.3.0)," 2023.

[14] ——, "3GPP TS 36.331: Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol Specification," 2024.

[15] ——, "3GPP TS 36.413: Evolved Universal Terrestrial Radio Access Network (E-UTRAN); S1 Application Protocol (S1AP)," 2024.

[16] ——, "3GPP TS 38.413: NG-RAN; NG Application Protocol (NGAP) (v15.8.0)," 2020.

[17] ——, "3GPP TS 23.501: System Architecture for the 5G System (5GS)," 2024.

[18] ——, "3GPP TS 38.401: NG-RAN; Architecture Description," 2024.

[19] ——, "3GPP TS 38.473: NG-RAN; F1 Application Protocol (F1AP)," 2024.

[20] ——, "3GPP TS 38.463: NG-RAN; E1 Application Protocol (E1AP)," 2024.

[21] ITU-T, "ITU-T X.680–X.683: ASN.1 – abstract syntax notation one," 2021.

[22] ——, "ITU-T X.691: ASN.1 encoding rules: Specification of packed encoding rules (PER)," 2021.

[23] L. Walkin, "`asn1c` – the ASN.1 compiler," https://github.com/vlm/asn1c, 2004.

[24] 3GPP, "3GPP TS 24.301: Non-Access-Stratum (NAS) Protocol for EPS; Stage 3 (Rel-16, additional APN rate control for exception data)," 2020.

[25] ——, "3GPP TS 24.501: Non-Access-Stratum (NAS) Protocol for 5G System (5GS); Stage 3," 2024.

[26] ——, "3GPP TS 24.007: Mobile Radio Interface Signalling Layer 3; General Aspects," 2024.

[27] G. Benita, M. E. Garbelini, S. Chattopadhyay, and J. Zhou, "ORAN-Claw: Shredding E2 nodes in O-RAN via structure-aware MitM fuzzing," in *ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, 2026.

[28] H. Wen, P. Porras, V. Yegneswaran, A. Gehani, and Z. Lin, "5G-Spector: An O-RAN compliant layer-3 cellular attack detection service," in *Network and Distributed System Security Symposium (NDSS)*, 2024.

[29] Trend Micro Research, "Attacks on 5G infrastructure from user devices: ASN.1 vulnerabilities in 5G cores," Trend Micro Research, 2023.

[30] T. D. Hoang, T. Oh, C. Park, I. Yun, and Y. Kim, "LLFuzz: An over-the-air dynamic testing framework for cellular baseband lower layers," in *USENIX Security Symposium*, 2025.

[31] G. Hernandez, M. Muench, D. Maier, A. Milburn, S. Park, T. Scharnowski, T. Tucker, P. Traynor, and K. Butler, "FirmWire: Transparent dynamic analysis for cellular baseband firmware," in *Network and Distributed System Security Symposium (NDSS)*, 2022.

[32] D. Maier, L. Seidel, and S. Park, "BaseSAFE: Baseband sanitized fuzzing through emulation," in *ACM Conference on Security and Privacy in Wireless and Mobile Networks (WiSec)*, 2020.

[33] H. Kim, J. Lee, E. Lee, and Y. Kim, "Touching the untouchables: Dynamic security analysis of the LTE control plane," in *IEEE Symposium on Security and Privacy (S&P)*, 2019.

[34] M. T. Raza, F. M. Anwar, and S. Lu, "Exposing LTE security weaknesses at protocol inter-layer, and inter-radio interactions," in *International Conference on Security and Privacy in Communication Networks (SecureComm)*, 2017.

[35] M. Son, K. Kim, B. Oh, C. Park, and Y. Kim, "CITesting: Systematic testing of context integrity violations in LTE core networks," in *ACM Conference on Computer and Communications Security (CCS)*, 2025.

[36] S. R. Hussain, I. Karim, A. Al Ishtiaq, O. Chowdhury, and E. Bertino, "Noncompliance as deviant behavior: An automated black-box noncompliance checker for 4G LTE cellular devices," in *ACM Conference on Computer and Communications Security (CCS)*, 2021.

[37] A. Al Ishtiaq, S. S. S. Das, S. M. M. Rashid, A. Ranjbar, K. Tu, T. Wu, Z. Song, W. Wang, M. Akon, R. Zhang, and S. R. Hussain, "Hermes: Unlocking security analysis of cellular network protocols by synthesizing finite state machines from natural language specifications," in *USENIX Security Symposium*, 2024.

[38] S. Luo, M. E. Garbelini, S. Chattopadhyay, and J. Zhou, "Sni5Gect: A practical approach to inject aNRchy into 5G NR," in *USENIX Security Symposium*, 2025.

[39] S. Erni, M. Kotuliak, R. Baker, I. Martinovic, and S. Capkun, "GLaDoS: Location-aware denial-of-service of cellular networks," in *USENIX Security Symposium*, 2025.

[40] E. Kim, M. W. Baek, C. Park, D. Kim, Y. Kim, and I. Yun, "BASECOMP: A comparative analysis for integrity protection in cellular baseband software," in *USENIX Security Symposium*, 2023.

[41] Y. Chen, D. Tang, Y. Yao, M. Zha, X. Wang, X. Liu, H. Tang, and D. Zhao, "Seeing the forest for the trees: Understanding security hazards in the 3GPP ecosystem through intelligent analysis on change requests," in *USENIX Security Symposium*, 2022.

[42] P1 Security, "Understanding cross-protocol attack paths in 4G and 5G networks," Blog post, P1 Security, https://www.p1sec.com/blog/protocol-stack-diagrams-in-4g-and-5g-networks, 2026.

[43] M. M. Rahman, I. Karim, and E. Bertino, "CellularLint: A systematic approach to identify inconsistent behavior in cellular network specifications," in *USENIX Security Symposium*, 2024.

[44] S. R. Hussain, O. Chowdhury, S. Mehnaz, and E. Bertino, "LTEInspector: A systematic approach for adversarial testing of 4G LTE," in *Network and Distributed System Security Symposium (NDSS)*, 2018.

[45] S. R. Hussain, M. Echeverria, I. Karim, O. Chowdhury, and E. Bertino, "5GReasoner: A property-directed security and privacy analysis framework for 5G cellular network protocol," in *ACM Conference on Computer and Communications Security (CCS)*, 2019.

[46] Cybersecurity and Infrastructure Security Agency, "ICSA-23-061-02: Baicells Nova," https://www.cisa.gov/news-events/ics-advisories/icsa-23-061-02, 2023, pre-authentication command injection in Baicells web GUI.

[47] B. Kallus, P. Anantharaman, M. Locasto, and S. W. Smith, "The HTTP garden: Discovering parsing vulnerabilities in HTTP/1.1 implementations by differential fuzzing of request streams," arXiv preprint arXiv:2405.17737, 2024.

[48] M. Brinkmann, C. Dresen, R. Merget, D. Poddebniak, J. Müller, J. Somorovsky, J. Schwenk, and S. Schinzel, "ALPACA: Application layer protocol confusion – analyzing and mitigating cracks in TLS authentication," in *USENIX Security Symposium*, 2021.

[49] CERT Coordination Center, "Vulnerability note VU#583108: Microsoft ASN.1 library improperly decodes constructed bit strings," https://www.kb.cert.org/vuls/id/583108, Feb. 2004, reported by eEye Digital Security; addressed in Microsoft Security Bulletin MS04-007.

[50] D. Kaminsky, M. L. Patterson, and L. Sassaman, "PKI layer cake: New collision attacks against the global X.509 infrastructure," in *Financial Cryptography and Data Security (FC)*, 2010.

Harness	**Iters**	**exec/s**
OAI `ul_ccch`	4.23B	48.9K
OAI `ul_dcch`	129M	1.5K
OAI `nr_cap`	376M	4.4K
OAI `mrdc_cap`	347M	4.0K
srsRAN `ul_ccch`	11.88B	144K
srsRAN `ul_dcch`	75K	25K
Total	**16.96B**	

TABLE 6: Decoder-only fuzzing: 16.96B iterations, zero translation surface findings.

	Type-I (§5.1)	**Type-II (§5.2)**	**Type-III (§5.3)**
Check (§4.3)	C1 value-range	C2 semantic-mapping	C3 containment
Generation	5G NR	4G LTE	4G LTE
Trigger payload	6-byte UL-CCCH	6-byte UL-CCCH	10-byte UL-DCCH
Pre-auth reach	yes (post-RACH)	yes (post-RACH)	yes (pre-NAS auth, RRC)
Outcome	cell-wide gNB abort	wire-level silent corruption	30 s cell-wide outage
Evidence tiers	APER unit test + Tier 2 dynamic	Tier 2 dynamic + source	Tier 3 OTA + Tier 4 source

TABLE 7: Shared properties of the three translation surface findings.

[51] F. Karimi, X. Xing, X. Gong, and E. Rodionov, "Over the air, under the radar: Attacking and securing the Pixel modem," Black Hat USA, 2023.

[52] LangSec Workshop Series, "LangSec: Language-theoretic security workshops," IEEE Security & Privacy satellite workshops, 2014–present, 2014. [Online]. Available: http://langsec.org/

[53] M. Balduzzi, L. Bongiorni, R. Flores, P. Z. Lin, C. Perine, and R. Vosseler, "Lost in translation: When industrial protocol translation goes wrong," Black Hat USA, 2020.

Appendix A. Quantitative Attack Surface

Table 8 reports the divergence analyzer output across four protocol boundaries (srsRAN 4G and srsRAN Project headers are both scanned at RRC↔NGAP).

Boundary	**Pairs**	**Diverg.**	**HIGH-priority**
LTE RRC↔S1AP	5	5	5
5G NR RRC↔NGAP	7	6	5
5G NR NGAP↔F1AP	44	15	13
5G NR NGAP↔E1AP	31	10	7
Total	**87**	**36** (41%)	**30**

TABLE 8: Translation surface scope: paired enumeration types across protocol boundaries. Pairs = automatically matched; Diverg. = value-space mismatch; HIGH-priority = reachable spare-to-semantic or missing-mapping mismatches (24 of the 30 do not require AMF-side injection; Table 9 gives the disposition of all 30, see §6.2).

Type-III (transparent forwarding) is invisible to the enum-divergence counts in Table 8 and is enumerated separately by the opaque-container pass (§4.3.2).

#	Boundary	Type pair	Evidence
Exercised — demonstrated boundary failure (2)			
1	RRC→S1AP	`EstablishmentCause` → `RRC-Establishment-Cause`	Type-II: `spare1` reaches the wire as `mo-ExceptionData` (§5.2)
7	RRC→NGAP	`EstablishmentCause` → `RRCEstablishmentCause`	OAI, Type-I: unmapped spare aborts the gNB (§5.1)
Exercised — demonstrated safe (1)			
6	RRC→NGAP	`EstablishmentCause` → `RRCEstablishmentCause`	srsRAN Project: bounded `switch`/`default`; unmapped values rejected locally
Exercised — converter-level lossiness, external reachability not demonstrated (4)			
11	NGAP→F1AP	`CauseMisc`	Lookup table, index 4 changes meaning
12	NGAP→F1AP	`CauseRadioNetwork`	57 vs. 41 values; 39 indices differ in meaning
13	NGAP→F1AP	`CauseTransport`	Indices 0–1 transposed
25	NGAP→E1AP	`CauseRadioNetwork`	NGAP superset of the E1AP table; reverse direction unmapped
Not exercised — core-originated downlink, outside the §3.6 attacker model (6)			
2	RRC→S1AP	`ReportInterval` → `M6report-Interval`	MDT configuration; sent by the core, not the UE
3	RRC→S1AP	`LoggingDuration-r10` → `LoggingDuration`	MDT configuration
4	RRC→S1AP	`LoggingInterval-r10` → `LoggingInterval`	MDT configuration
5	RRC→S1AP	`RAT-Type`	Inter-RAT handover, network-initiated; not driven from a UE-side procedure
8	RRC→NGAP	`ReportInterval` → `ExtendedReportIntervalMDT`	MDT configuration
10	RRC→NGAP	`LoggingDuration-r16` → `LoggingDuration`	MDT configuration
Not exercised — dropped, unimplemented, or source-confirmed only (14)			
14	NGAP→F1AP	`NextPagingAreaScope` → `AreaScope`	Extracted, then discarded before F1AP
15	NGAP→F1AP	`EUTRA-Paging-Time-Window` → `NRPaging-Time-Window`	Neither extracted nor forwarded
16	NGAP→F1AP	`PagingCauseIndication` → `PagingCause`	Never extracted from NGAP
17	NGAP→F1AP	`ReportingPeriodicity`	MDT unsupported in CU-CP
18	NGAP→F1AP	`NotificationControl`	Field declared, never populated
19	NGAP→F1AP	`M6report-Interval`	MDT unsupported in CU-CP
20	NGAP→F1AP	`MDT-Activation`	`logged_mdt_only` has no F1AP equivalent
21	NGAP→F1AP	`EventType`	Location reporting is a TODO in CU-CP
22	NGAP→F1AP	`RedCapIndication`	Present in NGAP HO, unused in CU-CP
24	NGAP→E1AP	`CauseMisc`	Same lookup pattern; confirmed in source, converter not exercised
26	NGAP→E1AP	`CauseTransport`	1 of 3 values meaningful; confirmed in source, converter not exercised
28	NGAP→E1AP	`ReportingPeriodicity`	MDT unsupported
29	NGAP→E1AP	`MDT-Activation`	`logged_mdt_only` missing in E1AP
30	NGAP→E1AP	`DirectForwardingPathAvailability` → `DirectFwdPathAvail`	Handover path not translated
Not exercised — matcher artifact, no translation path to exercise (3)			
9	RRC→NGAP	`EUTRA-Q-OffsetRange` → `Range`	dB offset paired with a distance range
23	NGAP→F1AP	`RRCState` → `Service-State`	UE state paired with cell state
27	NGAP→E1AP	`SourceOfUEActivityBehaviour` → `UE-Activity`	Information source paired with activity state

TABLE 9: Disposition of all 30 HIGH-priority candidates. The first three groups were exercised; the last three were not, each for a stated reason. The # column identifies the pair; it is not a ranking.

Boundary	Direction	Values	Outcome / Loss %	Evidence
RRC → NGAP (5G, OAI)	uplink	–	crash (Type-I)	Dynamic
RRC → NGAP (5G, srsRAN Project)	uplink	–	safe	Dynamic
RRC → S1AP (LTE, srsRAN 4G)	uplink	–	silent corruption (Type-II)	Dynamic
RRC → S1AP (LTE, commercial)	uplink	–	DoS (Type-III)	OTA
F1AP → NGAP	forward	41	85.4%	Converter
NGAP → F1AP	reverse	57	100.0%	Converter
E1AP → NGAP	forward	34	67.6%	Converter
NGAP → E1AP	reverse	57	100.0%	Converter
F1AP CauseMisc → NGAP	forward	5	0.0%	Converter
NGAP CauseMisc → F1AP	reverse	6	83.3%	Converter
F1AP CauseTransport → NGAP	forward	4	75.0%	Converter
NGAP CauseTransport → F1AP	reverse	2	0.0%	Converter
F1AP CauseRadioNetwork → NRPPa	forward	41	97.6%	Suppl.
F1AP CauseTransport → NRPPa	forward	4	100% (cross-family)	Suppl.
F1AP CauseProtocol → NRPPa	forward	7	0.0%	Suppl.
F1AP CauseMisc → NRPPa	forward	5	80.0%	Suppl.

TABLE 10: Validation and converter-coverage summary. End-to-end rows characterize attacker reachability; converter-level rows establish structural lossiness from running implementation code but not by themselves attacker reachability; supplementary F1AP→NRPPa rows are converter-coverage checks rather than main findings.

Appendix B.
Type-I Evidence: APER Control and Specification Survey

APER encoding control. Table 11 gives OAI's NGAP encoder results for every `EstablishmentCause` 0–15.

Val.	NR RRC name	APER	Status
0–9	emergency . . . mcs-Priority	OK	Root
10	spare6	OK	Extension
11	spare5	**FAIL**	No mapping
12	**spare4**	**FAIL**	**Trigger**
13–15	spare3 . . . spare1	**FAIL**	No mapping

TABLE 11: APER encoding control for the RRC→NGAP `EstablishmentCause` boundary: values 0–10 encode; 11–15 fail. The boundary matches the 5 spare values without NGAP mapping.

Specification survey. We grepped nine 3GPP specifications spanning the RAN–Core control plane, across ten release versions (Table 12), for normative clauses (*shall*, *must*, *is required to*) mandating cross-schema value-space validation, defensive mapping between independently-defined enumerations, or per-UE containment of opaque forwarded

payloads. No spec contained such a clause; the closest candidates (PDCP §5.8 erroneous-protocol-data discard, RRC §5.7.2 ASN.1-syntax-invalid message handling, NGAP `notAvailable` value definition) are intra-schema or descriptive, not cross-schema mandates.

Spec	Version	Role
TS 36.323	Rel-17	PDCP LTE
TS 36.331	Rel-17	RRC LTE (source schema)
TS 36.413	Rel-17	S1AP LTE (destination schema)
TS 38.331	Rel-18	RRC NR (source schema, latest)
TS 38.413	Rel-17	NGAP 5G (destination schema)
TS 38.413	Rel-18	NGAP 5G (latest)
TS 38.401	Rel-15	NG-RAN architecture
TS 38.473	Rel-17	F1AP (destination schema)
TS 23.501	Rel-17	5G system architecture
TS 33.501	Rel-17	5G security architecture

TABLE 12: 3GPP specifications surveyed for cross-schema validation mandates. None contain such a mandate.

Appendix C. Type-II Evidence: Silent Corruption (LTE Cross-Generation + F1AP-NGAP)

LTE cross-generation detail. Index 7 is `spare1` in LTE RRC (TS 36.331) but `mo-ExceptionData` in S1AP (TS 36.413); the divergence is exercised in §5.2; srsRAN 4G and OAI LTE translate it with unsafe patterns: C-style cast and direct integer assignment, respectively.

We additionally confirmed the Type-II corruption end-to-end on a full LTE control-plane testbed (srsRAN 4G `srsenb` + `srsepc` + `srsue`, four-line patch forcing `establishmentCause` via env var). The `RRC-EstablishmentCause` IE at offset 0x008A of the InitialUEMessage SCTP data chunk differs by exactly one byte (`0x30` vs `0x82`) between the control and attack runs, matching the value predicted by the static eNB enum cast at `rrc_ue.cc:567`. `srsepc`'s MME accepts the corrupted cause without protocol-level error and proceeds with bearer setup; we observed no behavioral differential between runs on this academic-grade EPC. Operational impact is implementation-conditional: a Release-16 MME implementing IoT routing extensions (TS 23.682 §5.13.4) would be expected to differentiate, but `srsepc` does not implement those extensions.

F1AP→NGAP silent corruption. Each of the 35 distinct silent-corruption instances at this cause boundary (§6.1) is a value the DU can legitimately raise and the AMF receives as *unspecified*. Three examples: `release_due_to_pre_emption` (emergency-call preemption silently relabeled, defeating priority accounting); `gnb_cu_cell_capacity_exceeded` (capacity-driven releases indistinguishable from radio failures, defeating load balancing); and `amf_initiated_abnormal_release` (the AMF's own release tagged *unspecified*, erasing the causal chain in network telemetry). With over four-fifths of cause values silently degraded, the AMF's view is structurally incomplete (downstream consumer behavior not characterized here).

Appendix D. Supply-Chain and Cross-Release Evidence

Supply-chain implications. The Radisys TOTALeNodeB BSP behind the commercial DoS (§5.3) is AGPLv3-licensed; source-path strings in the Baicells firmware binary (`radisys/src/enbapp/wr_umm_*`) match the open-source tree, in which `wrUmmS1apBldInitUePdu` is likewise identifiable. On the same Qualcomm SoC family, firmware QRTB_2.12.5 (436Q, 2022-10) carries 362 translation functions, and BLQ_3.0.12 (430H, 2023-02) carries 543 (+50%). Each such function is a point at which a value can cross a schema boundary, so the 50% growth is consistent with attack-surface expansion as 3GPP procedure handlers are added.

Cross-release dimension (preliminary). The translation surface has a second, *vertical* dimension (same schema, different releases), spanning V1 (spare-to-real reallocation), V2 (structural extension) and V3 (`nonCriticalExtension` chain growth). Consistent with the scope declared in §3.5, this paper defers strict V1–V3 runtime exploitation (§9) and reports preliminary evidence only: **87/189** (46%) divergence test vectors produce different decoder behavior between the Release 9 and Release 17 LTE RRC ASN.1 decoders (accepted in one release, rejected, or differently interpreted); and of 84 V1–V3 vectors driven against two same-stack srsRAN 4G builds (`release_22_04`, April 2022, ≈ R15/R16; `release_25_10`, October 2025, ≈ R17; both with AddressSanitizer), only V3 crashed, the 14-byte `I21_reconfig_noncrit` vector hitting the same `sched_ue::set_cfg()` null-deref SIGSEGV in both (a 3.5-year persistent V3 finding), while V1/V2 are rejected by srsRAN's `asn1c`-generated decoder via its `rejected_unknown` safety net, an implementation-specific observation that does not generalize to other stacks.

Appendix E. Decoder-Fuzzing Negative Result

Six libFuzzer+ASan harnesses (four 5G NR, two LTE) ran 16.96B iterations across single-schema decoder boundaries with zero translation-surface findings (Table 6); the srsRAN UL-DCCH harness halted at approximately 75K iterations on a decoder stack-buffer-overflow (a parser-class memory-safety bug, orthogonal to the translation surface). This is consistent with where each established approach injects (Table 1): parser fuzzing (Berserker [3]) stays within one schema, specification compliance (BaseSpec [5], DoLTEst [6]) within one spec while the divergence exists *between* specifications, and state-machine analysis (5GBaseChecker [8]) within one protocol; mutation fuzzing (5Ghoul [4]) may accidentally produce a divergent value but does not systematically enumerate the translation surface. Stateful campaigns such as RANsacked [2] run further but inject already-formed S1AP and NGAP at the RAN–core interface (§6.2), after the steps that create Type-I and Type-II faults.

Ethics Considerations

Testing environment. All open-source testing (OAI, srsRAN) is performed on researcher-owned hardware in isolated RF-simulation environments (OAI rfsimulator, srsRAN ZMQ radio driver). No real radio transmissions occur; no production networks, third-party devices, or real subscribers are affected. The Baicells Nova 430H testing is conducted on a researcher-owned device operating on a CBRS band with SAS authorization inside a shielded enclosure.

Open-source disclosure. The gNB crash vulnerability (§5.1) was reported via OpenAirInterface's confidential issue tracker. OpenAirInterface merged a fix that maps spare RRC establishment causes to NGAP notAvailable, implementing the SWITCH+SENTINEL handling. The LTE silent corruption (§5.2) was reported to the srsRAN maintainers; receipt has been acknowledged, and the issue is under investigation. No fix had been released or merged upstream as of this camera-ready. The core-side process termination reported in §5.3 was disclosed privately to the Open5GS maintainers through the project's security-advisory process and is awaiting triage; the assertion is present in the v2.7.0 source we tested and absent from that path in v2.7.1 and later.

Commercial disclosure. The RAN-amplified DoS behavior (§5.3) was reported through CERT/CC's VINCE process (published by CISA as ICSA-26-272-04 / CVE-2026-96274). The vendor did not respond to CISA's coordination requests, and no fix is planned. In our deployment, the resulting crash itself occurs in Open5GS v2.7.0, where the MME fails to safely handle the forwarded NAS input.

Responsible limitations. Proof-of-concept seeds are disclosed with sufficient detail for independent verification but are not packaged as automated attack tools. The 10-byte payload structure is documented to enable vendor verification and mitigation of the affected handling path. A bounded NAS Protocol-Discriminator check at the eNB is only a defense-in-depth option. We release the divergence analyzer, safe static test harnesses, and non-commercial proof-of-concept seeds. Commercial OTA reproduction artifacts are not released, and we do not release the general UE injection hooks or the campaign orchestration built on them. The artifact does include the one narrow hook needed to reproduce the LTE wire-level differential: a single-field override that sets `establishmentCause` on `RRCConnectionRequest` and can reach no other field.

Generative AI Usage Considerations

Generative AI was used for editorial purposes in this manuscript, and all outputs were inspected by the authors to ensure accuracy and originality. All research contributions are the authors' own.

Appendix F. Meta-Review

The following meta-review was prepared by the program committee for the 2027 IEEE Symposium on Security and Privacy (S&P) as part of the review process as detailed in the call for papers.

F.1. Summary

This paper studies vulnerabilities that arise when LTE and 5G base stations translate messages across protocol schemas. Its key observation is that a value accepted by one protocol may be unrepresentable or may carry a different meaning after translation. The paper presents CELLBREAKER, an analyst-assisted workflow that compares schema value spaces, inspects implementation mappings, generates boundary-targeted inputs, and validates failures across multiple protocol boundaries. The study demonstrates crashes, semantic corruption, and service disruption that conventional single-decoder testing may miss because these failures occur during mapping, re-encoding, or forwarding.

F.2. Scientific Contributions

- Addresses a Long-Known Issue
- Identifies an Impactful Vulnerability
- Provides a Valuable Step Forward in an Established Field

F.3. Reasons for Acceptance

1) Identifies an Impactful Vulnerability: The paper successfully demonstrates how protocol boundary mismatches can bypass single-schema verification
2) Addresses a Long-Known Issue: The paper develops a method to evaluate attacks when base stations translate messages across protocol schemas, which is a long-standing issue.
3) Provides a Valuable Step Forward in an Established Field: The study discovers various vulnerabilities and provides methods to detect more. There is a concrete validation for the example of open-source stacks and commercial over-the-air hardware.

F.4. Noteworthy Concerns

1) The procedure is not fully automated, requiring an analyst for the final decision