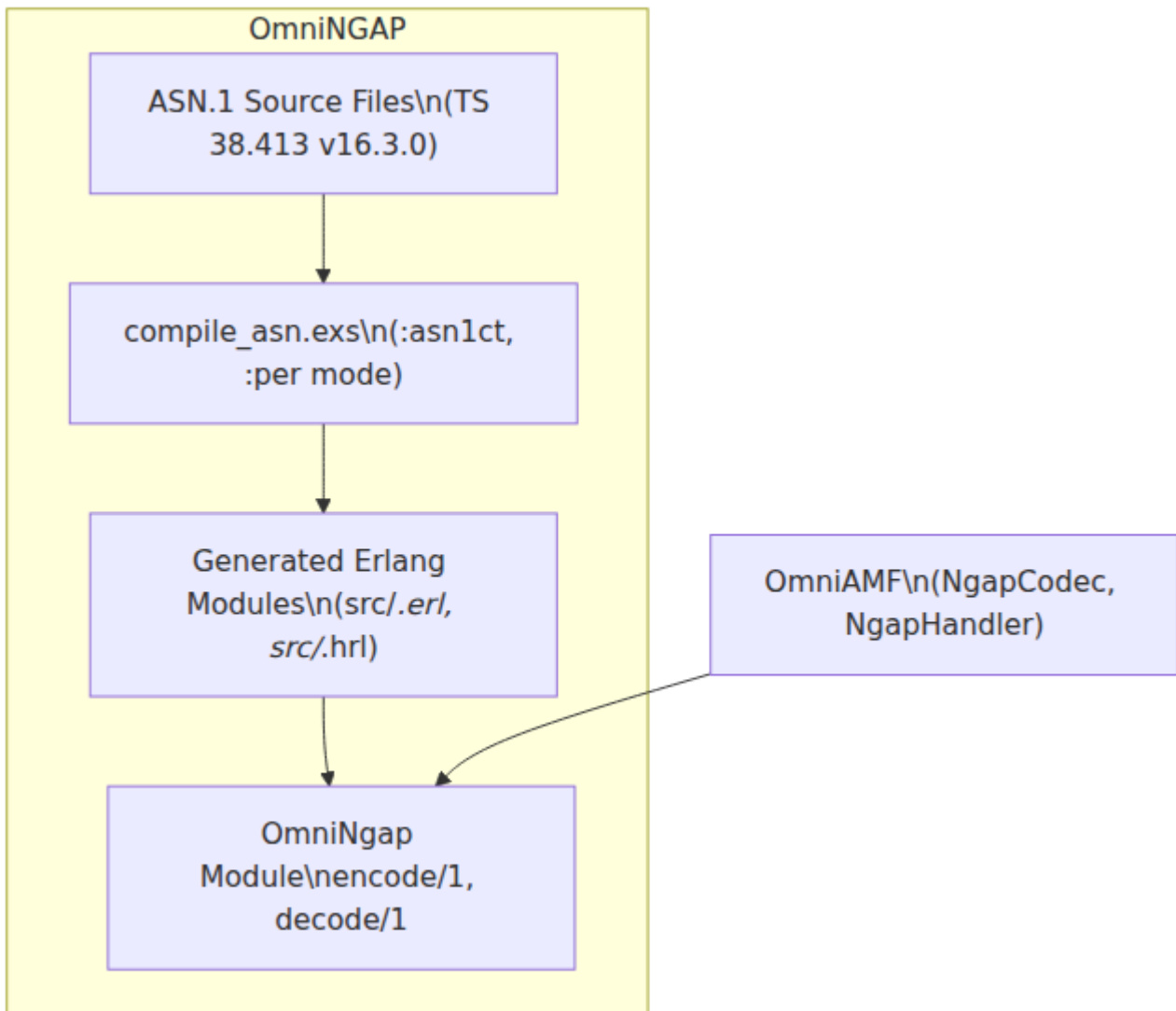


# OmniNGAP Operations

## 1. Component Overview

OmniNGAP is the NGAP ASN.1 codec library for the Omnitouch 5G core. It provides compile-time ASN.1 compilation of the NGAP protocol definitions from 3GPP TS 38.413 and exposes encode/decode functions for NGAP PDUs using Aligned PER (APER) encoding. OmniNGAP is consumed as a dependency by OmniAMF for all NGAP message construction and parsing over the N2/SCTP interface.

OmniNGAP is a library, not a standalone NF. It has no SBI endpoints, no NRF registration, and no runtime configuration. All ASN.1 modules are compiled once at build time using Erlang's `:asn1ct` compiler.



## Compile-Once Approach

The ASN.1 compilation is performed by `compile_asn.exe` during the Mix compile phase. The script:

1. Reads `.asn1` source files from the `asn1/` directory in dependency order.
2. Compiles each module using `:asn1ct.compile/2` with `:per` mode (Aligned PER, as required by TS 38.413).
3. Generates `.erl` and `.hrl` files in `src/`.
4. Copies `.hrl` files to `include/` for dependent projects using `Record.extract(from_lib:)`.
5. Skips recompilation when source files have not changed (mtime comparison).

6. Handles read-only filesystems (e.g., Docker) by falling back to precompiled modules.

## ASN.1 Modules

Module	Description
NGAP-CommonDataTypes	Base ASN.1 types (Criticality, PresenceInformation, etc.)
NGAP-Containers	NGAP container types (ProtocolIE-Container, ProtocolExtensionContainer)
NGAP-Constants	NGAP constants (procedure codes, IE IDs, max values)
NGAP-IEs	NGAP Information Elements (all IE definitions)
NGAP-PDU-Contents	NGAP PDU content structures (InitiatingMessage, SuccessfulOutcome, etc.)
NGAP-PDU-Descriptions	Top-level NGAP-PDU type definition and procedure dispatch

## 2. 3GPP Role and Spec References

Aspect	Reference
NGAP protocol definition	TS 38.413
NGAP transport (SCTP)	TS 38.412
NGAP encoding (Aligned PER)	TS 38.413 Section 9, ITU-T X.691
AMF / gNB interface (N2)	TS 23.501 Section 8.2.5
NGAP procedures	TS 38.413 Section 8

## 3. API Reference

OmniNGAP exposes two functions via the `OmniNgap` module:

### `OmniNgap.decode/1`

Decodes a binary NGAP PDU (APER-encoded) into an Erlang ASN.1 record.

```
{:ok, ngap_pdu} = OmniNgap.decode(binary)
```

Returns `{:ok, record}` on success or `{:error, reason}` on decode failure.

### `OmniNgap.encode/1`

Encodes an Erlang ASN.1 record into a binary NGAP PDU (APER-encoded).

```
{:ok, binary} = OmniNgap.encode(ngap_pdu_record)
```

Returns `{:ok, binary}` on success or `{:error, reason}` on encode failure.

# Erlang Record Access

NGAP IE records can be accessed from Elixir using `Record.extract/2`:

```
require Record
Record.extract("NGAP-PDU", from_lib: "omni_ngap/include/NGAP-PDU-
Descriptions.hrl")
```

## 4. Build and Compilation

### Prerequisites

- Erlang/OTP 27+ (provides `:asn1` application with `:asn1ct` compiler)
- Elixir 1.17+

### Build Steps

```
cd OmniNGAP
mix deps.get
mix compile
```

The `mix compile` alias runs `compile_asn.exs` before standard Elixir compilation. On first build, all six ASN.1 modules are compiled. Subsequent builds skip unchanged modules.

### Dependency Usage

OmniAMF depends on OmniNGAP as a path or git dependency:

```
# In OmniAMF mix.exs
{:omni_ngap, git: "...", branch: "main"}
```

OmniAMF's `NgapCodec` module uses the generated Erlang records and `OmniNgap.encode/1` to construct NGAP messages (NG Setup Response, Initial

Context Setup Request, Handover Request, Paging, etc.).

## **5. NGAP Procedures Supported**

The following NGAP procedures are encoded/decoded by OmniNGAP and used by OmniAMF:

<b>Procedure Code</b>	<b>Name</b>	<b>Direction</b>	<b>Usage</b>
21	NGSetup	gNB -> AMF	gNB connection establishment
14	InitialUEMessage	gNB -> AMF	First NAS message from UE
4	InitialContextSetup	AMF -> gNB	Security context and registration accept
46	DownlinkNASTransport	AMF -> gNB	NAS PDUs to UE
47	UplinkNASTransport	gNB -> AMF	NAS PDUs from UE
24	Paging	AMF -> gNB	UE paging in idle mode
41	UEContextRelease	AMF -> gNB	UE context teardown
1	HandoverPreparation	AMF -> Target gNB	Handover request
0	HandoverResourceAllocation	AMF -> Source gNB	Handover command
25	PathSwitchRequest	gNB -> AMF / AMF	Xn handover path switch

Procedure Code	Name	Direction	Usage
		-> gNB	
29	PDU Session Resource Setup	AMF -> gNB	PDU session resource allocation

## 6. Known Limitations

ID	Area	Description
NGAP-1	ASN.1 version	ASN.1 source files are from TS 38.413 v16.3.0 (Rel-16). Later releases may add new IEs or procedures not covered by this version
NGAP-2	Encoding only	OmniNGAP provides raw encode/decode. Message construction (populating IE values, handling optionality) is the responsibility of the consuming application (OmniAMF NgapCodec)
NGAP-3	No runtime configuration	OmniNGAP is a compile-time library with no runtime configuration or telemetry

## 7. Troubleshooting

### ASN.1 compilation fails

Check that Erlang/OTP includes the `:asn1` application. Run `erl -eval 'application:ensure_all_started(asn1), halt().'` to verify. If compilation fails with a specific module, check the error output from `:asn1ct.compile/2` for syntax issues in the `.asn1` source file.

## Encode/decode returns `{:error, reason}`

The input record or binary does not conform to the NGAP ASN.1 schema.

Common causes:

1. Missing mandatory IEs in the record passed to `encode/1`.
2. Corrupted or truncated binary passed to `decode/1`.
3. IEs with values outside the defined ASN.1 constraints (e.g., integer out of range).

Check the error reason tuple for the specific IE or constraint that failed.

## Read-only filesystem in Docker

The `compile_asn.exs` script detects read-only filesystems and falls back to precompiled modules in `src/` and `include/`. Ensure the Docker image was built with a full compilation pass so that `src/*.erl` and `include/*.hrl` files are present.