

ALMA MATER STUDIORUM · UNIVERSITÀ DI BOLOGNA
CAMPUS DI CESENA

SCUOLA DI SCIENZE
Corso di Laurea Magistrale in Ingegneria e Scienze Informatiche

Implementazione Cloud del piano di controllo della rete 5G

Relazione finale in:

Instradamento e trasporto in Internet

Relatore:
Prof. FRANCO CALLEGATI

Tesi di laurea di:
DANIELE ROSSI

Co-relatori:
CHIARA GRASSELLI
CHIARA CONTOLI
DAVIDE BORSATTI

Terza Sessione di Laurea
Anno accademico: 2019-2020

PAROLE CHIAVE

EPC

5GC

NFV

Cloud Computing

OSM

OpenStack

”Gli uomini che ammiro di più nella nostra società, Marcus, sono quelli che costruiscono ponti, grattacieli e imperi. Ma, in realtà, i più coraggiosi e ammirevoli sono quelli che riescono a costruire l’amore perché non esiste impresa più grande e difficile.”

Joël Dicker

Ai miei genitori, Marina e Paolino, che mi hanno insegnato che anche dal nulla, con amore e sacrificio, è possibile realizzare grandi cose.

Indice

Introduzione	iv
Sommario	vi
1 Network Function Virtualization	1
1.1 Introduzione	1
1.2 NFV framework architetturale	3
1.3 NFV-MANO framework	7
1.3.1 Virtualized Infrastructure Manager	8
1.3.2 VNF Manager	9
1.3.3 NS Catalogue	10
1.3.4 VNF Catalogue	10
1.3.5 NFV Instances repository	10
1.3.6 NFVI Resources repository	10
1.3.7 NFV Orchestrator	10
1.4 Descrittori ETSI-MANO	12
2 Reti di telecomunicazioni cellulari	14
2.1 Introduzione	14
2.2 Seconda generazione: 2G	15
2.3 Terza generazione: 3G	15
2.3.1 3G rete di accesso radio	16
2.3.2 3G core network	17
2.4 Quarta generazione: 4G LTE	17
2.4.1 Evolved Packet Core	19
2.5 Quinta generazione: 5G	30
2.5.1 Software Defined Network	33
2.5.2 Network Slicing	34
2.5.3 Service Function Chaining	36
2.5.4 Multi-access Edge Computing	44
2.5.5 Architettura sistema 5G	49

3	Strumenti software	69
3.1	Open Source MANO	69
3.1.1	Architettura OSM	71
3.1.2	Ciclo di vita dei Network Service	73
3.1.3	Configurazione delle VNF	77
3.2	OpenStack	81
3.3	Open5GS	85
3.3.1	Moduli software Open5GS	86
3.3.2	Configurazione Open5GS	89
3.4	srsLTE	101
3.4.1	Configurazione del simulatore	102
3.4.2	Esecuzione del simulatore	104
3.5	UERANSIM	106
3.5.1	Configurazione del simulatore	106
3.5.2	Esecuzione del simulatore	108
4	Orchestrazione di EPC e 5GC virtualizzati	110
4.1	Ambiente di esecuzione	110
4.2	Deployment manuale di analisi	112
4.3	Deployment automatizzato	114
4.3.1	Architettura di deployment	114
4.3.2	Creazione dell'immagine di Open5GS per OpenStack	117
4.3.3	VNFD e configurazione day 0	117
4.3.4	Configurazione day 1	122
4.3.5	NSD, on-boarding e deployment	128
5	Validazione del lavoro svolto	131
5.1	Analisi del traffico	131
5.1.1	Procedura di collegamento dell'eNodeB alla MME	131
5.1.2	Procedura di collegamento UE senza contesto salvato nell'EPC	132
5.1.3	Procedura di collegamento UE con contesto salvato nell'EPC	150
5.1.4	Procedura di abbattimento del collegamento tra UE ed EPC	151
5.1.5	Procedura di collegamento del gNodeB all'AMF	155
5.1.6	Procedura di registrazione dell'UE al 5GC	156
5.1.7	Procedura di creazione di una PDU Session	172
5.2	Creazione Network Service	181
5.3	Limitazione AMBR per abbonamento 5G	182

Conclusioni e sviluppi futuri	184
A Implementazioni	185
A.1 Descrittori	185
A.1.1 radio_access_vnfd.yaml	185
A.1.2 user_gateway_vnfd.yaml	187
A.1.3 control_gateway_vnfd.yaml	189
A.1.4 auth_functional_block_vnfd.yaml	191
A.1.5 misc_functional_block_vnfd.yaml	193
A.1.6 Open5GS_nsd.yaml	194
A.2 Script my-open5gs-dbctl	196
A.3 Configurazione day 1	197
A.3.1 Codice python per file corrispondenti alle azioni	197
A.3.2 Struttura sgwuupfpgwucharm.py	197
A.3.3 Cambio indirizzi nel file hss.conf	198
A.4 Limitazione AMBR	199
A.4.1 TokenBucket.java	199
Ringraziamenti	200
Bibliografia	202
Libri	202
Articoli	202
Report tecnici	204
Siti	206

Introduzione

Dalla nascita della tecnologia mobile sono trascorsi circa 40 anni e in questo periodo, le capacità delle reti cellulari, si sono evolute a un ritmo estremamente veloce. Tale evoluzione, ha alimentato sia il cambiamento sociale che l'innovazione su scala globale. Le capacità delle reti 3G e successivamente 4G, sono state sviluppate in risposta alle richieste di Internet mobile e hanno portato smartphone e tablet a diventare accessori di uso quotidiano; al giorno d'oggi, è difficile pensare a una vita non connessa. La pervasività dei dispositivi IoT e le richieste di applicazioni emergenti come intelligenza artificiale, realtà mixata e auto a guida autonoma, hanno portato allo sviluppo della prossima generazione di reti cellulari: 5G.

Mentre le reti 3G e 4G sono focalizzate sui servizi voce e dati e sono principalmente associate allo smartphone, il 5G offrirà molto di più, consentendo l'interconnessione di miliardi di dispositivi di qualsiasi tipo. Il 5G non è esclusivamente sinonimo di connettività wireless più veloce, ma promette di trasformare i processi di consumo aziendali e industriali esistenti, guidando la prossima ondata di crescita economica globale.

Tutto ciò comporta numerose sfide: gestire grossi volumi di traffico, garantire una maggiore qualità dell'esperienza utente, gestire dispositivi estremamente eterogenei, offrire un servizio diverso a seconda del tipo di esigenze e garantire una copertura pervasiva. In questo contesto, ci si aspetta che i nuovi paradigmi di progettazione dell'architettura di rete 5G facciano la differenza rispetto alle generazioni precedenti, introducendo importanti cambiamenti per quanto riguarda flessibilità e scalabilità. Attualmente, le reti mobili sono costituite da apparecchi hardware dedicati e appositamente collegati tra loro; la rigidità di quest'architettura rende difficoltosa la raggiunta degli obiettivi sopracitati. Per questo, così come avvenuto nel mondo dell'informatica, anche quello delle telecomunicazioni si trova ad avere a che fare con i concetti di Cloud e servizi. 3GPP¹ ha standardizzato un'architettura orientata ai servizi per il core di rete 5G. Ogni elemento di quest'ultimo è rap-

¹3rd Generation Partnership Project unisce diverse organizzazioni che si occupano degli standard nel mondo delle telecomunicazioni

presentato da un servizio che espone delle API ben definite per l'interazione con gli altri componenti di rete. Quest'architettura si presta ad essere messa in esecuzione su Cloud, l'unico problema è che ancora il livello di maturità di quest'ultimo potrebbe non essere adeguato per supportare tutto ciò.

Più nel dettaglio, la programmabilità di rete rappresenta un elemento chiave per la progettazione, l'implementazione, la distribuzione, la gestione e la manutenzione di apparecchiature e componenti di rete mediante la programmazione del software. Ai concetti di Cloud e servizi si affiancano quelli di Software Defined Network, Network Function Virtualization e Multi-access Edge Computing.

Tramite Software Defined Network, il piano di controllo è separato dal piano dati. Le funzionalità di controllo sono incapsulate in un'entità, controller SDN, logicamente separata da quella che si occupa dell'inoltro dei dati.

Tramite Network Function Virtualization è possibile disaccoppiare una funzionalità di rete dall'hardware sottostante. In particolare, prevede l'utilizzo delle tecnologie di virtualizzazione al fine di virtualizzare i componenti di rete in modo che possano essere eseguiti su hardware di comodità. NFV abilita il Network Slicing permettendo di costruire reti logiche differenti sulla stessa infrastruttura comune.

Tramite Multi-access Edge Computing le capacità di calcolo e archiviazione e i servizi vengono portati il più vicino possibile all'utente: nella parte edge del Cloud.

La combinazione di tutti questi paradigmi dovrebbe portare il 5G a raggiungere gli obiettivi definiti e a essere più di un aggiornamento; è una trasformazione radicale dei principali segmenti dell'economia.

Sommario

Il presente elaborato ha l'obiettivo di virtualizzare e orchestrare i componenti del 5GC ed EPC utilizzando l'approccio NFV. In particolare, lo scenario progettato prevede di mettere in esecuzione e configurare dinamicamente i componenti dei core tramite il framework ETSI NFV Management and Orchestration.

Per l'implementazione, Open Source MANO, OSM, viene utilizzato come piattaforma di gestione e orchestrazione e OpenStack come piattaforma Cloud e VIM. I componenti dei core di rete sono implementati da Open5GS in conformità con la release 16 di 3GPP; per connettersi ai core verranno utilizzati due simulatori: UERANSIM e srsLTE. Il primo, simula la RAN e l'UE che permettono di collegarsi al 5GC, mentre il secondo quelli che permettono di collegarsi all'EPC.

In una prima fase, i componenti verranno messi in esecuzione ognuno su una macchina virtuale differente e configurati manualmente. Questo permetterà facilmente di catturare e analizzare il traffico generato dalle procedure di collegamento, stabilimento della connessione e disconnessione. Una volta fatto ciò, verrà realizzato lo scenario precedentemente proposto.

Il documento è strutturato in cinque capitoli.

Il **Capitolo 1** introduce il paradigma Network Function Virtualization fornendo una panoramica del framework architetturale proposto da ETSI. Particolare attenzione sarà posta sulla parte di gestione e orchestrazione e sui template (descrittori) che questa utilizza per la gestione del ciclo di vita di funzioni e servizi di rete virtualizzati.

Nel **Capitolo 2** verranno trattate le reti cellulari. Dopo una breve introduzione delle prime generazioni, la trattazione si concentrerà sulla quarta e la quinta. Verranno descritti gli elementi che compongono i core di rete e per il 5G, i pilastri tecnologici su cui si fonda.

Il **Capitolo 3** introduce a fondo gli strumenti che verranno utilizzati per la realizzazione della parte sperimentale.

Il **Capitolo 4** entra nel merito dello scenario da realizzare illustrando le scelte architetturali effettuate e i principali dettagli implementativi.

Nel **Capitolo 5** sono contenuti i risultati dell'analisi del traffico e dei test effettuati per valutare le funzionalità dei core.

Capitolo 1

Network Function Virtualization

Questo capitolo tratterà la virtualizzazione delle funzioni di rete: Network Function Virtualization. Dopo aver spiegato cos'è NFV e i vantaggi che comporta, entrerà nel merito dell'architettura NFV di riferimento ponendo particolare attenzione sulla componente di gestione e orchestrazione.

1.1 Introduzione

Le reti dei diversi operatori sono costituite da una grande varietà di apparati hardware proprietari che svolgono diverse funzioni di rete, **NF**. Questi sono connessi tra loro in maniera tale da fornire la macro-funzionalità o il servizio richiesto. Le relazioni che sussistono tra le diverse funzionalità di rete sono statiche e possono essere modellate tramite un grafo, **NF Forwarding Graph** o tramite un set, **NF Set Construct** [26]. Questa soluzione, presenta diversi problemi:

- occupazione di spazio
- costi energetici
- investimento di capitale (Capex)
- necessità di skills differenti per configurare, integrare e mantenere i diversi dispositivi
- rapida obsolescenza
- scarsa flessibilità e scalabilità.

Per far fronte a questo problema, nell'ultimo decennio sono stati proposti approcci basati su software che garantiscono una migliore scalabilità e flessibilità a un costo inferiore. Network Function Virtualization, **NFV** è un paradigma dell'architettura di rete che va in questa direzione e sfrutta le tecnologie di virtualizzazione per cambiare il modo in cui i servizi di rete possono essere progettati, implementati e gestiti.

Invece di utilizzare dispositivi hardware specializzati, NFV propone l'implementazione di funzioni di rete come entità software che possono essere eseguite su commodity hardware. Si fa riferimento a queste mediante il termine funzioni di rete virtualizzate, **VNF** e possono essere distribuite su una o più macchine virtuali in esecuzione su uno o più server fisici. Uno dei più grandi cambiamenti introdotti da NFV è che la virtualizzazione abilita l'utilizzo di metodi dinamici, anziché statici, per la costruzione e la gestione di grafi o set che modellano le relazioni tra le diverse funzioni di rete (in questo caso VNF). Come mostrato in [1.1], NFV non prevede blocchi proprietari monolitici, ma elementi software flessibili che possono essere organizzati, riconfigurati e migrati dinamicamente per adattare i servizi di rete a seconda delle necessità. In tal modo, NFV facilita il ridimensionamento delle VNF e l'istanziatura di nuovi componenti dove necessario senza richiedere investimenti in nuove apparecchiature dedicate.

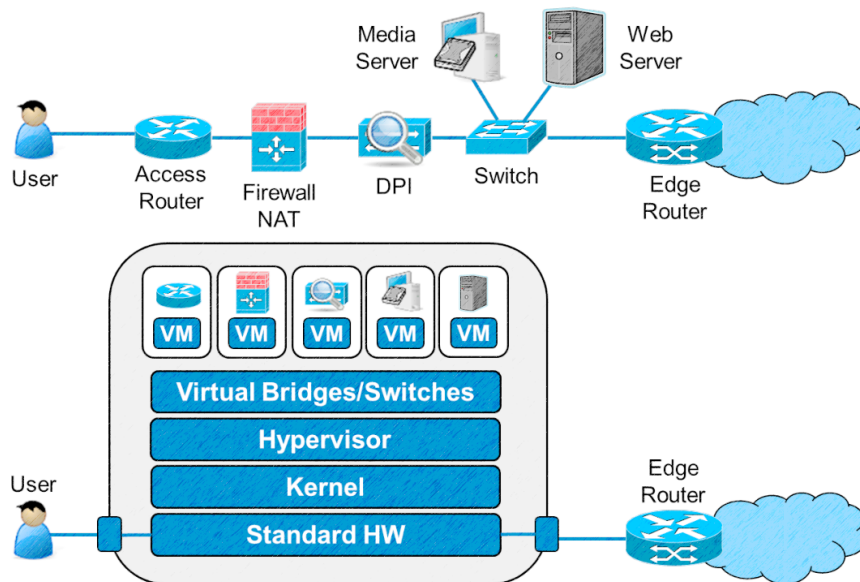


Figura 1.1: Comparazione semplificata di un approccio tradizionale e un approccio NFV

NFV applica al mondo delle reti l'approccio Cloud Computing al fine di renderle scalabili e adattabili. Inoltre, l'utilizzo del Cloud apre la strada a nuovi modelli di business basati sul concetto di *As A Service, AAS* in cui l'infrastruttura fisica, su cui girano le diverse VNF, può essere condivisa e fornita da terzi.

I vantaggi dell'approccio NFV rispetto a quello tradizionale sono innumerevoli; tra questi figurano [38]:

- riduzione dei costi
- aumento della velocità del time to market minimizzando la durata del ciclo di innovazione degli operatori di rete
- scalabilità e flessibilità dei servizi
- fornitura servizi di rete rapida senza necessità di recarsi in loco per installare nuove apparecchiature HW
- ottimizzazione della configurazione e/o topologia della rete in tempo reale sulla base del traffico e/o dei servizi richiesti
- ottimizzazione dei consumi e dei carichi di lavoro; sfruttando la virtualizzazione, è possibile concentrare il carico di lavoro su un numero di server ridotto durante i momenti di calma (ad esempio, dopo mezzanotte) ponendo i restanti in modalità risparmio energetico o spegnendoli.

In sostanza, i servizi di rete, **NS**, in futuro si comporranno di NF non virtualizzate e/o virtualizzate e sarà fondamentale garantire interoperabilità tra queste. Il focus principale sarà sviluppare un'architettura che supporti la diversità delle NF garantendo: affidabilità, disponibilità, manutenibilità, sicurezza e performance.

Di seguito verrà trattato nel dettaglio il framework architetturale NFV e la componente di gestione e orchestrazione di questo.

1.2 NFV framework architetturale

Ad alto livello, il framework NFV si compone di tre elementi principali [1.2]:

- **Virtualised Network Functions.** Implementazione software delle funzioni di rete in grado di essere eseguite sopra ad una NFVI.
- **NFV Infrastructure, NFVI.** Astrae le diverse risorse fisiche e gestisce la modalità di virtualizzazione di queste.

- **NFV Management and Orchestration.** Implementa l'orchestrazione e la gestione delle risorse hardware e/o software che supportano la NFVI e la gestione del ciclo di vita delle VNF.

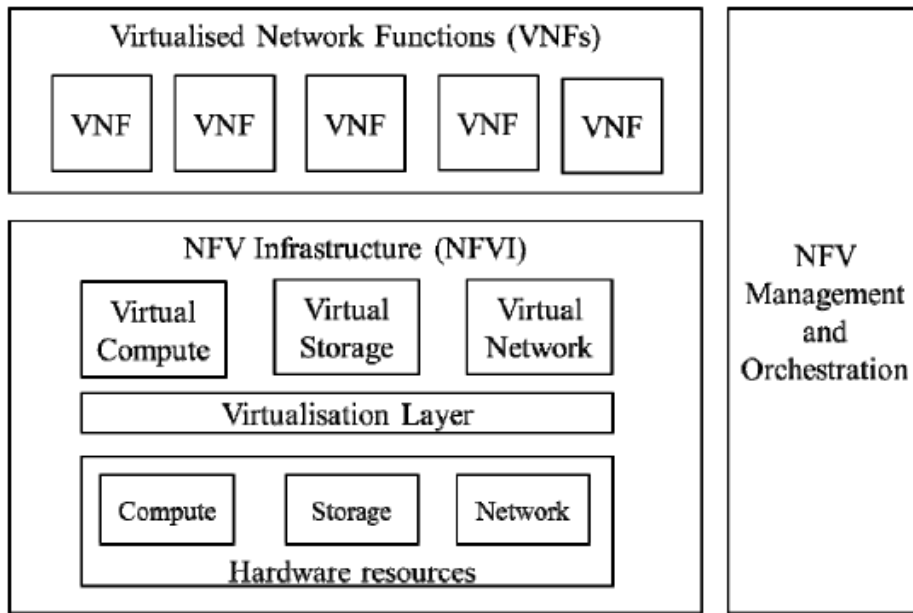


Figura 1.2: NFV framework architetturale ad alto livello di astrazione

Il framework permette di creare e gestire dinamicamente le istanze VNF e le relazioni tra queste in termini di dati, controllo, gestione, dipendenze e altre caratteristiche. Queste relazioni possono essere modellate tramite:

- **VNF Forwarding Graph, VNF-FG.** Racchiude i casi in cui le relazioni tra le VNF sono specificate; ad esempio, concatenazione di funzioni di rete (firewall, NAT, load balancer) sul percorso verso un web server.
- **VNF Set.** Racchiude i casi in cui le relazioni tra le VNF non sono specificate. Ad esempio, virtualizzazione di gateway indipendenti.

In questa trattazione considereremo il primo modello in cui la sequenza delle VNF può essere decisa sia a tempo di design che di esecuzione.

Ad un maggiore livello di dettaglio, il framework architetturale NFV può essere suddiviso in blocchi funzionali e interfacce logiche tra questi [1.3].

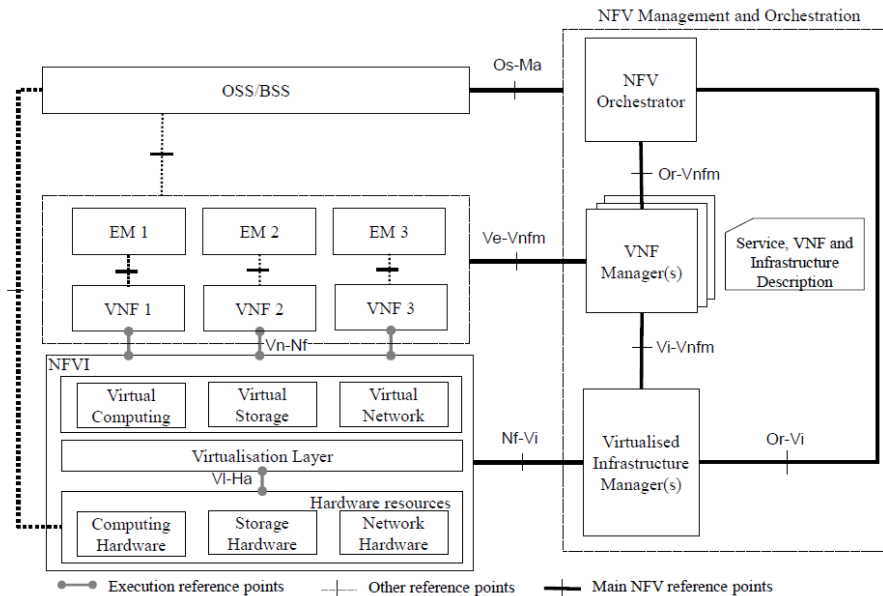


Figura 1.3: NFV framework architetturale

Di seguito verrà fornita una breve descrizione dei blocchi funzionali raffigurati in [1.3].

Le Virtualised Network Functions, **VNFs**, come già detto, corrispondono alla versione software virtualizzata delle tradizionali funzionalità di rete svolte da appositi apparati hardware. Queste funzionalità di rete possono essere: *middle-boxes* come firewall e NAT, elementi di rete in generale (come server DHCP) oppure elementi appartenenti al core delle reti mobili come MME, SGW... nel caso dell'EPC o AMF, SMF... nel caso di 5GC. Il comportamento e le interfacce di una funzionalità di rete sono indipendenti dal fatto che questa sia virtualizzata o meno. Una funzionalità di rete è ulteriormente decomposta in componenti interni che possono essere eseguiti su diverse macchine virtuali.

Gli Element Management, **EM**, sono i sistemi che forniscono funzionalità di gestione a una o più VNFs. Tali funzionalità possono essere riassunte con l'acronimo FCAPS: Fault, Configuration, Accounting, Performance e Security.

La NFV Infrastructure, **NFVI**, è l'insieme dei componenti hardware e software che costituiscono l'ambiente in cui le VNF sono distribuite, eseguite e gestite. La NFVI può essere geograficamente distribuita in diversi NFVI Points-of-Presence (NFVI-PoPs) e la rete utilizzata per l'interconnessione fa parte di essa. Dal punto di vista delle VNF la distribuzione è trasparente. La NFVI comprende:

- l'insieme delle risorse hardware di rete, computazione e memorizzazione che vengono messe a disposizione delle VNFs attraverso il layer di virtualizzazione. L'hardware dedicato alla computazione è di comodità (commodity hardware). Le risorse di memorizzazione possono essere condivise (ad esempio memorie NAS: Network Attached Storage) oppure locali ai singoli server. Le risorse di rete comprendono sia le funzioni di commutazione, svolte tipicamente da switch e router, sia i collegamenti cablati e wireless.
- il layer di virtualizzazione che astrae le risorse hardware permettendo il disaccoppiamento delle VNFs da queste. Questo disaccoppiamento garantisce la portabilità delle VNFs su infrastrutture dotate di hardware diverso e garantisce una gestione del ciclo di vita indipendente dall'infrastruttura utilizzata. Il livello di astrazione si ottiene spesso utilizzando hypervisor e, nel caso di risorse di rete, reti virtuali, VLAN, oppure overlay network¹.

Il **Virtualised Infrastructure Manager** controlla e gestisce la virtualizzazione delle risorse fisiche di rete, computazione e memorizzazione di cui è responsabile e l'interazione delle VNFs con queste. Per quanto riguarda la gestione delle risorse si occupa di fare l'inventario di quelle disponibili e dedicabili all'infrastruttura NFV e di allocarle alle diverse macchine. Oltre a ciò, effettua operazioni di monitoraggio e controllo, ad esempio, collezionando informazioni relative ai fallimenti o analizzando le cause che portano a problemi di performance nell'infrastruttura NFV. È possibile eseguire e distribuire più istanze di Virtualised Infrastructure Manager.

L'**NFV Orchestrator**, NFVO, è responsabile della gestione e orchestrazione della NFVI e delle risorse software. Inoltre, si occupa di istanziare i diversi servizi di rete sulla NFVI.

Il **VNF Manager**, VNFM, gestisce il ciclo di vita di una o più VNF ed esegue operazioni di creazione dell'istanza, aggiornamento, interrogazione, ridimensionamento, terminazione, eccetera. Come nel caso di VIM, è possibile eseguire e distribuire più VNF Manager.

Il **Service, VNF and Infrastructure Description** è una base di dati che fornisce informazioni in merito ai template per la creazione delle VNF, al VNF-FG, ai servizi di rete e alla NFVI.

Gli **Operations and Business Support Systems**, OSS/BSS, sono sistemi e applicazioni di gestione utilizzate dai fornitori di servizi e dagli

¹Rete di calcolatori costruita su un'altra rete. I nodi possono essere connessi fra loro tramite collegamenti logici o virtuali, ciascuno dei quali corrisponde ad un percorso nella rete sottostante.

operatori di telecomunicazioni per supportare la fornitura di diversi servizi end-to-end (ad esempio, gestione degli ordini o dei pagamenti).

1.3 NFV-MANO framework

L'adozione di NFV ci garantisce un'architettura di rete scalabile e flessibile, ma introduce nuove sfide da affrontare. I nuovi servizi di rete end-to-end possono richiedere il collegamento di funzioni di rete sia virtualizzate, VNF, sia fisiche, PNF, al fine di fornire la funzionalità globale richiesta. Inoltre, le VNF devono: comunicare con l'infrastruttura NFV sottostante, essere allocate in maniera location-aware al fine di realizzare il servizio previsto e essere dotate delle risorse hardware necessarie. Oltre a ciò, meccanismi di monitoraggio e controllo devono essere adottati per rilevare gli eventuali guasti e recuperare da questi ultimi. Tutti questi problemi sono stati tenuti in considerazione nella progettazione del framework NFV Management and Orchestration la cui architettura è descritta in [28].

L'immagine [1.4] mostra ad un maggior livello di dettaglio le entità funzionali che compongono il framework NFV-MANO e le interfacce logiche sia tra esse che con il resto del framework.

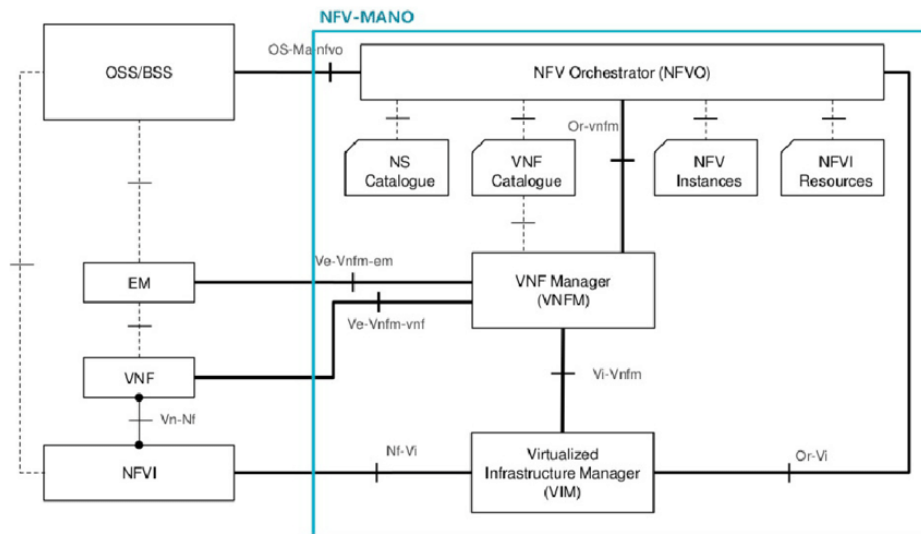


Figura 1.4: NFV-MANO framework architetturale

Il framework possiede un'architettura a più livelli in cui ciascuna entità è responsabile di attività di gestione e orchestrazione differenti e collabora con le altre per gestire e orchestrare l'intera infrastruttura e l'esecuzione distribuita dei vari servizi. Ogni blocco offre i propri servizi agli altri in maniera

trasparente tramite apposite API, questo garantisce un livello di indipendenza che consente la distribuzione delle diverse componenti del framework. Le principali funzionalità di MANO sono:

- gestire e orchestrare la NFVI. Tale attività comprende la coordinazione delle risorse virtuali e fisiche, il mantenimento dell'inventario delle risorse disponibili, l'allocazione e il rilascio dinamico delle risorse virtuali e la gestione dei guasti e dei problemi di performance di tali risorse.
- gestire e orchestrare le VNFs. Tale attività, oltre alla fornitura delle funzionalità FCAPS, prevede il mantenimento delle risorse virtuali associate alle VNFs per tutto il loro ciclo di vita. Le risorse possono essere assegnate o scalate dinamicamente in base a informazioni di runtime, ad esempio collezionando dati che ci permettano di capire come la funzione di rete sta performando sulla base di KPI² definiti.
- gestire e orchestrare i servizi di rete. Tale attività comprende la coordinazione delle VNFs che compongono i servizi, l'orchestrazione della NFVI in modo che sia possibile eseguire i servizi stessi e la gestione dell'intero ciclo di vita di questi.

Le sottosezioni successive tratteranno nel dettaglio i diversi blocchi funzionali che compongono il framework.

1.3.1 Virtualized Infrastructure Manager

Il VIM è responsabile del controllo e della gestione delle risorse di rete, elaborazione e memorizzazione della NFVI. In particolare, queste risorse possono essere tutte quelle di NFVI-PoP o un loro sottoinsieme oppure appartenere a NFVI-PoP differenti. Più dettagliatamente, le funzionalità offerte da VIM sono:

- Orchestrare l'allocazione, l'aggiornamento e il rilascio delle risorse NFVI e l'ottimizzazione del loro utilizzo. Per fare ciò è necessario mantenere il mapping tra risorse virtuali e risorse fisiche; in particolare, VIM utilizza un registro delle allocazioni delle risorse virtuali su quelle fisiche.
- Gestire il VNF-FG creando e mantenendo link virtuali, sotto reti, reti virtuali e porte per l'interconnessione delle VNFs. Ciò richiede inoltre di gestire le policy di sicurezza per permettere o meno al traffico di fluire attraverso la rete e implementare meccanismi di controllo di questo.

²Key Performance Indicators rappresenta l'insieme degli indicatori che permettono di misurare le prestazioni di una determinata attività o processo

- Gestire le immagini software delle VNFs, come richiesto da altri blocchi funzionali del framework (ad esempio, NFVO), al fine di razionalizzare l’allocazione delle risorse virtuali. Prima di poter aggiungere un’immagine al repository di quelle gestite è necessario che VIM esegua uno step di validazione di questa.
- Collezionare informazioni relative alle performance e ai guasti delle risorse hardware (rete, elaborazione, memorizzazione), software (hypervisor) e virtuali (VMs). Le informazioni relative a queste ultime dovranno poi essere inoltrate ai blocchi di livello superiore.

VIM interagisce con i livelli superiori attraverso i punti di interconnessione, $Vi-Vnfm$ e $Or-Vi$, che trasportano le richieste di prenotazione e/o allocazione delle risorse inviate rispettivamente dal VNF Manager e dal VNF Orchestrator. Questi ultimi, ricevono dal VIM, tramite i punti di interconnessione, le informazioni in merito alle performance e agli errori.

Il punto di riferimento etichettato come $Nf-Vi$ viene invece utilizzato dal VIM per eseguire l’assegnazione di risorse virtualizzate in risposta alle richieste di allocazione e per scambiare altre informazioni sulle prestazioni e sullo stato con la NFVI.

1.3.2 VNF Manager

Il VNF Manager è il responsabile della gestione del ciclo di vita di istanze VNF. Un VNF Manager può gestire una o più istanze di uno o più tipi. Un VNF Manager offre un insieme di funzioni di base utili per gran parte delle VNF, per la gestione di VNF che richiedono una funzionalità peculiare è possibile specificare questa in un apposito pacchetto di estensione, *VNF packet*.

La messa in esecuzione e il comportamento operativo di ogni VNF sono descritti in un template chiamato Virtualised Network Function Descriptor, **VNFD** che descrive accuratamente i requisiti necessari per realizzare la VNF a cui è associato ed è memorizzato nel VNF Catalogue [1.3.4]. Per assegnare le risorse della NFVI ad una VNF si fa riferimento al descrittore associato a questa, tenendo però in considerazione requisiti e vincoli specificati nella richiesta di esecuzione che potrebbero sovrascriverli.

Il VNFM non interagisce solo con le VNF, tramite il punto di riferimento $Ve-Vnfm-vnf$, ma può anche cooperare con i sistemi EM attraverso il punto di riferimento $Ve-Vnfm-em$ al fine di configurare le VNF e monitorare prestazioni e utilizzo delle risorse.

1.3.3 NS Catalogue

L'NS Catalogue è il repository che elenca tutti i servizi di rete integrati e supporta la creazione e la gestione dei templates di esecuzione (Network Service Descriptor, **NSD**, Virtual Link Descriptor, **VLD** e VNF Forwarding Graph Descriptor, **VNFFGD**) che specificano i requisiti necessari per la realizzazione di ciascun servizio.

1.3.4 VNF Catalogue

Il VNF Catalogue è il repository che contiene tutti i VNF Packages integrati e supporta la creazione e la gestione dei VNFD. L'NFV Orchestrator e l'NFV Manager possono interrogare il repository per ottenere informazioni (ad esempio, ottenere un VNFD).

1.3.5 NFV Instances repository

L'NFV Instances repository mantiene le informazioni di tutte le istanze delle VNF e dei servizi di rete, NS. Ogni istanza corrisponde rispettivamente ad un record VNF o NS, questi ultimi vengono aggiornati durante tutto il ciclo di vita dell'istanza a loro associata. Questo permette all'NFV Orchestrator e all'NFV Manager di avere una visione consistente delle istanze attive e delle relazioni tra loro.

1.3.6 NFVI Resources repository

L'NFVI Resources repository mantiene informazioni in merito alle risorse NFVI disponibili, riservate o allocate. Tale repository svolge un ruolo fondamentale nel supportare le attività di governance e orchestrazione dell'NFV Orchestrator consentendo a quest'ultimo di visualizzare le risorse utilizzate da una VNF o un NS (ad esempio, visualizzare il numero di VMs utilizzate da una VNF in un determinato punto del suo ciclo di vita).

1.3.7 NFV Orchestrator

L'NFV Orchestrator ha due responsabilità principali:

- l'orchestrazione di risorse, che fornisce una gestione globale delle risorse NFVI condivise tra più VIM;
- l'orchestrazione di Network Service, che consiste nel creare istanze di NS e gestirne il ciclo di vita.

Per semplicità, le due funzioni sono state inglobate in un unico componente funzionale che utilizza due basi di dati: NFV Instances e NFV Resources. In realtà, queste due funzioni potrebbero essere suddivise in due componenti distinti a giovamento della modularità. In tal scenario, l'NFVO utilizzerebbe la funzionalità di orchestrazione dei Network Service per coordinare le varie VNF che collaborando assieme realizzano il servizio; a sua volta, tale funzionalità, utilizzerebbe i servizi esposti dal VNF Manager e dalla funzione di orchestrazione delle risorse per gestire il ciclo di vita delle VNF e allocare o scalare le risorse associate ad esse. Nel dettaglio, alcune delle funzionalità offerte dall'NFVO tramite l'orchestrazione di Network Service sono:

- Gestione dei template per l'esecuzione dei Network Service (NSDs) e dei VNF Packages. Durante l'integrazione di NS, e delle VNF che lo compongono, è richiesto uno step di validazione al fine di verificare la correttezza del template per l'esecuzione (VNFD e NSD) e la presenza di tutte le informazioni necessarie per l'esecuzione del servizio.
- Creazione di istanze di NS e gestione del loro ciclo di vita. Per quanto riguarda l'istanziamento di un NS, l'orchestratore ottiene informazioni dall'NSD, che descrive le caratteristiche e i requisiti di ciascun NS e le interconnessioni logiche tra le VNF che lo costituiscono. In base a queste decide le modalità di coordinamento delle VNF, dove eseguire le diverse istanze e orchestra l'eventuale connessione con le PNF necessarie. Una volta prese le decisioni, queste vengono inviate a VNF Manager e VIM tramite gli opportuni punti di riferimento. Questi si occuperanno dell'istanziamento e della configurazione delle VNF assegnandoli le risorse richieste.
- Gestione della creazione di istanze di VNF in collaborazione con il VNF Manager.
- Validazione e autorizzazione delle richieste di risorse NFVI da parte di un VNF Manager, siccome queste potrebbero impattare sul NS.
- Gestione dell'integrità e della visibilità delle istanze dei NS durante tutto il loro ciclo di vita e delle relazioni di queste con le istanze delle VNF. Per quest'ultima funzionalità si avvale del NFV Instances repository.

Mentre alcune funzionalità offerte tramite l'orchestrazione di risorse sono:

- Validazione e autorizzazione delle richieste di risorse NFVI da parte di un VNF Manager, siccome queste potrebbero impattare la modalità di allocazione delle risorse.

- Gestione delle relazioni tra istanze VNF e risorse NVFI allocate tramite l'utilizzo del NFVI Resources repository e delle informazioni fornite dai VIM.
- Gestione delle policy di accesso alle risorse da parte delle istanze di NS e VNF (ad esempio, specifica di policy per il controllo degli accessi o di regole di accesso basate sulla posizione geografica).
- Collezionare informazioni in merito all'utilizzo delle risorse da parte delle diverse istanze.

Le funzionalità di entrambi i componenti sono esposte tramite apposite interfacce utilizzabili sia dai blocchi funzionali di NFV-MANO sia da entità esterne autorizzate (ad esempio, sistemi OSS/BSS).

1.4 Descrittori ETSI-MANO

Come detto in precedenza, un servizio di rete, NS, è costituito da un insieme di funzioni di rete concatenate tra loro che cooperano per fornire la funzionalità finale. Utilizzando NFV è possibile che le diverse funzioni siano virtuali, VNF, o fisiche, PNF.

Ogni servizio di rete può essere descritto in maniera astratta da un VNF Forwarding Graph, VNF-FG, i cui nodi rappresentano le funzioni di rete che costituiscono il servizio e gli archi bi-direzionali le relazioni tra queste. In [1.5] è raffigurato un esempio di un servizio di rete end-to-end composto da VNF; l'infrastruttura e i collegamenti fisici sottostanti sono nascosti dal livello di virtualizzazione. I provider di servizi possono quindi definire più Network Forwarding Paths (NFP) per descrivere la direzione dei flussi di traffico attraverso gli elementi del servizio di rete. Al fine di supportare la messa in esecuzione di un servizio di rete, lo standard NFV definisce un insieme di template, chiamati descrittori MANO, che contengono le informazioni necessarie ai blocchi funzionali del framework NFV-MANO per gestire il ciclo di vita delle VNF e del NS:

- il **Network Service Descriptor**, NSD, è il template di più alto livello che descrive i requisiti e il comportamento operativo di un servizio di rete e fa riferimento agli altri tipi di descrittori per fornire informazioni sulla topologia e sulle funzioni di rete costituenti;
- il **Virtualized Network Function Descriptor**, VNFD, definisce il comportamento e la configurazione di una singola VNF, i suoi punti di connessione interni ed esterni, le Virtual Deployment Unit per eseguire

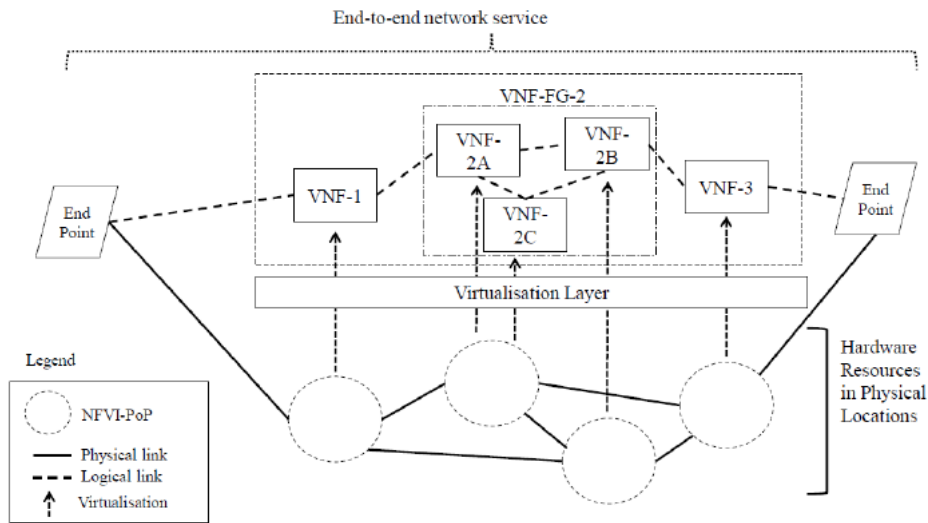


Figura 1.5: Esempio di un servizio di rete composto da VNF e descritto da un VNF-FG innestato

ciascuno dei componenti VNF su una VM (con requisiti in termini di risorse di archiviazione, calcolo e rete), i requisiti in termini di prestazioni e la sua topologia interna. Questo template è usato in primo luogo per istanziare la VNF, per la gestione del ciclo di vita e per configurazioni future;

- il **Physical Network Function Descriptor**, PNFD, fornisce informazioni sulle modalità di connessione con una PNF e i requisiti che deve avere il collegamento virtuale usato per l'interconnessione;
- il **VNF Forwarding Graph Descriptor**, VNFFGD, è il template che descrive il VNF-FG, ovvero la topologia o parte di essa di un NS, facendo riferimento alle VNF, PNF e i collegamenti virtuali tra esse;
- il **Virtual Link Descriptor**, VLD, descrive i collegamenti tra VNF, PNF e gli endpoint del NS, riportando anche i parametri associati e le risorse richieste;

Capitolo 2

Reti di telecomunicazioni cellulari

Questo capitolo tratterà l'evoluzione delle reti cellulari dalla prima generazione alla quinta. Per ogni generazione verranno introdotti gli aspetti più rilevanti ponendo particolare attenzione al 4G e al 5G.

2.1 Introduzione

Prima di entrare nel merito della rete 5G è opportuno introdurre l'evoluzione delle reti cellulari dalla loro nascita fino al giorno d'oggi; per fare ciò ci si avvarrà di quanto riportato in [35], [49] e [40].

Nel 1980, l'Europa riconobbe la necessità di un sistema digitale continentale in grado di sostituire i sistemi telefonici cellulari nazionali analogici e incompatibili tra loro. Questa necessità fu soddisfatta tramite lo standard GSM; l'installazione delle tecnologie GSM avvenne a partire dagli anni '90 e da allora ha rivestito un ruolo centrale nella telefonia cellulare di tutto il mondo.

È possibile classificare le tecnologie cellulari in "generazioni"; le più datate sono state progettate esclusivamente per il traffico telefonico. I sistemi di prima generazione, 1G, trasmettevano i dati in maniera analogica ed erano pensati esclusivamente per le comunicazioni audio. Tali sistemi, sono stati sostituiti da quelli di seconda generazione, 2G, digitali. Anche questi ultimi vennero progettati per la fonia, ma più avanti furono estesi, 2.5G (GPRS), per supportare il traffico dati. I sistemi 3G supportano sia la voce che i dati con un'enfasi maggiore sulla capacità di trasporto di questi ultimi e sulle prestazioni dei collegamenti di accesso radio. Vi sono poi i sistemi 4G che possiedono un nucleo interamente basato su IP e trasportano sia voce che

dati a velocità di megabit. Terminiamo questa trattazione con i sistemi 5G che hanno come primo obiettivo quello di connettere in maniera completa la nostra società introducendo innumerevoli benefici e il cui nucleo è costruito da servizi invocabili tramite apposite API.

Di seguito vengono evidenziati gli aspetti più rilevanti delle diverse "generazioni".

2.2 Seconda generazione: 2G

L'architettura semplificata dei sistemi 2G è riportata nell'immagine [2.1]. Un'area geografica è suddivisa in celle ognuna delle quali contiene una Base Transceiver Station, **BTS**, che scambia dati con le stazioni mobili (abbonati mobili) della cella. Per praticità, spesso una BTS è posizionata nell'intersezione tra le celle in modo che una sola stazione, dotata di più antenne direzionali, riesca a coprire tre celle. Per l'interfaccia aerea lo standard GSM 2G utilizza una combinazione di FDM¹/TDM² in cui il canale è suddiviso in sotto-bande di frequenza all'interno delle quali il tempo è suddiviso in frame e slot (date F bande di frequenza e T slot, il canale è in grado di supportare F*T dispositivi contemporaneamente). Un Base Station Controller, **BSC**, serve diverse BTS e ha il compito di allocare canali radio agli utenti di queste eseguendo la procedura di *paging*, che consente di localizzare la cella in cui l'utente mobile risiede e di gestire l'*handoff* ovvero il cambiamento di BTS da parte dell'utente. Il BSC, assieme alle BTS che controlla, compone il Base Station System, **BSS**. Un Mobile Switching Center, **MSC**, ha il compito di gestire diversi BSC e gioca un ruolo fondamentale nell'identificazione e autorizzazione degli utenti (ad esempio determinando se un utente ha il permesso di collegarsi alla rete cellulare), nello stabilire o terminare una chiamata e nell'*handoff*. Un operatore possiede diversi MSC collegati alla rete telefonica pubblica tramite **Gateway MSC**.

2.3 Terza generazione: 3G

La miniaturizzazione e l'evoluzione dei dispositivi mobili hanno scaturito negli utenti la volontà di utilizzarli per leggere mail, vedere video in streaming, utilizzare servizi di navigazione, eccetera. Per questo motivo, è necessario

¹Suddivisione del canale in sotto-bande di frequenza ognuna delle quali può essere assegnata ad un diverso dispositivo mobile

²Suddivisione del tempo in frame ulteriormente ripartiti in slot, ad ogni dispositivo mobile è assegnato un particolare slot di un frame

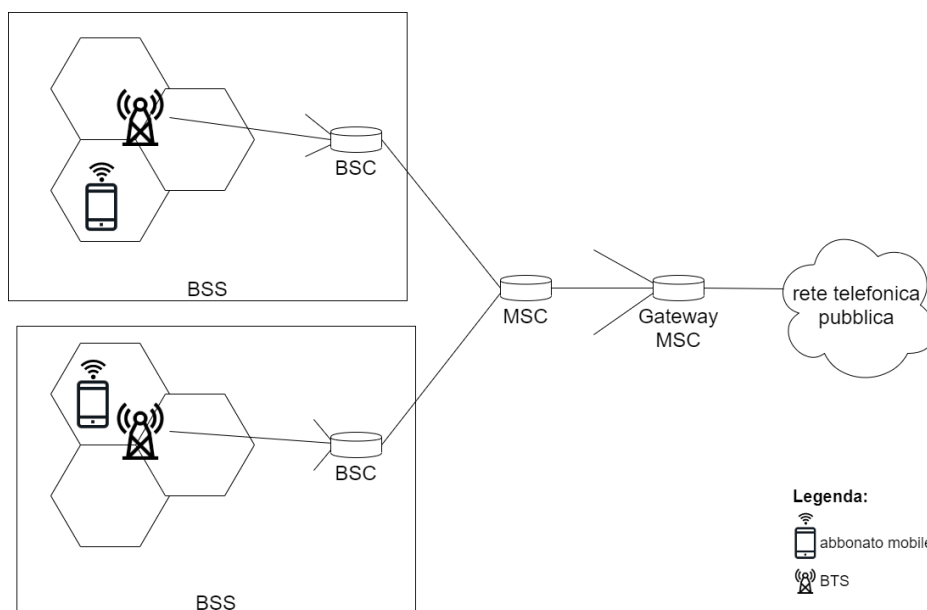


Figura 2.1: Architettura semplificata rete cellulare GSM 2G

che i dispositivi mobili siano in grado di eseguire l'intero stack TCP/IP e collegarsi a Internet attraverso la rete cellulare.

Il mondo delle reti cellulari è costituito da un insieme elevato di standard in continua evoluzione e competizione tra loro. Per complicare ancora di più le cose, non esiste un ente ufficiale di riferimento che fissi i requisiti delle varie generazioni. In questa trattazione ci focalizzeremo sullo standard 3G UMTS sviluppato dal 3GPP. L'immagine [2.2] mostra l'architettura semplificata dei sistemi 3G.

2.3.1 3G rete di accesso radio

La rete di accesso radio è il primo hop dal punto di vista dell'utente. Le antenne radio (BTS nell'architettura 2G) nel gergo UMTS sono denominate **NodeB** e sono collegate ai Radio Network Controller, **RNC**. L'RNC controlla tipicamente più NodeB ed è connesso sia alla rete voce tramite un MSC sia ad Internet tramite SGSN. Rispetto al 2G, per l'interfaccia aerea si utilizza una tecnica CDMA³ nota come Direct Sequence Wideband CDMA anziché lo schema GSM FDM/TDM.

³Code Division Multiple Access appartiene alla famiglia di protocolli di suddivisione del canale come FDM e TDM ed è il protocollo di accesso al canale condiviso più diffuso nelle reti wireless e cellulare.

2.3.2 3G core network

La core network connette le diverse reti di accesso radio ad Internet. Per il suo sviluppo si è deciso di lasciare intatto il nucleo GSM per la voce, aggiungendo in parallelo un nucleo dedicato alla trasmissione dei dati. Quest'ultimo nucleo è caratterizzato da due tipi di nodi: Serving GPRS Support Nodes, **SGSN** e Gateway GPRS Support Nodes, **GGSN**. L'SGSN ha il compito di inoltrare i datagrammi a/da i nodi mobili nella rete di accesso radio a cui esso è collegato. Inoltre, interagisce con l'MSC per fornire i servizi di autenticazione, handoff e ottenere le informazioni in merito alla cella di appartenenza di ciascun nodo mobile. Il GGSN agisce come un gateway connettendo diversi SGSN a Internet.

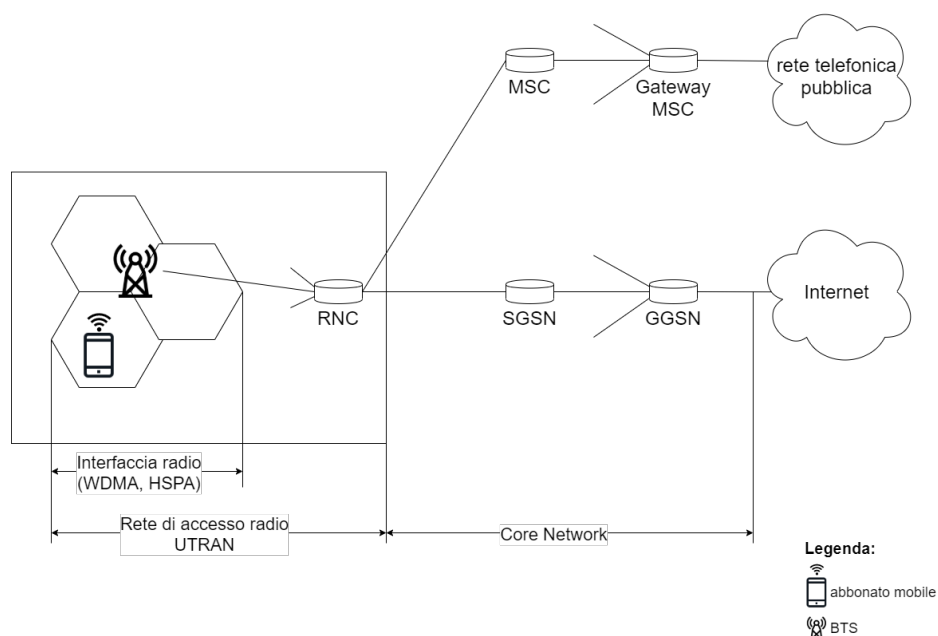


Figura 2.2: Architettura semplificata rete cellulare UMTS 3G

2.4 Quarta generazione: 4G LTE

Nonostante la continua evoluzione, lo standard UMTS possiede un insieme di limitazioni di progettazioni insuperabili. Per questo motivo, 3GPP ha deciso di riprogettare sia la rete di accesso radio, che prende il nome di Evolved-UTRAN, che la rete core, che prende il nome di Evolved Packet Core, **EPC**. Il risultato della riprogettazione prende il nome di Long-Term Evolution, **LTE** e i cambiamenti più significativi rispetto a UMTS sono:

- utilizzo di un'interfaccia aerea, Orthogonal Frequency Division Multiplexing, **OFDM**, completamente diversa da WCDMA;
- supporto a tecnologie di accesso differenti. Il nucleo di rete fornisce supporto e interoperabilità per diverse Radio Access Technologies, **RAT**: E-UTRAN (RAN di LTE e LTE-Advanced), UTRAN (RAN di UMTS), il GERAN (RAN di GSM/GPRS) e tecnologie di accesso non 3GPP come WiMAX, WLAN o reti fisse. Le tecnologie di accesso non 3GPP possono essere trusted o untrusted.
- Architettura di rete unificata (non c'è un percorso esclusivamente dedicato ai dati e uno alla voce) e interamente basata su IP;
- separazione del piano dati dal piano di controllo. Ciò garantisce agli operatori di rete una maggiore flessibilità nella gestione della latenza del piano dati e una scalabilità indipendente tra i due piani. Inoltre, muovendo l'EPC in Cloud, la separazione, permette di collocare le entità che si occupano di gestire il traffico del piano utente nella parte edge vicina alla rete radio in modo da diminuire sensibilmente la latenza. Le entità che si occupano dei segnali di controllo possono essere nella parte di Cloud centralizzata. Tuttavia, va precisato che nell'architettura EPC originale il disaccoppiamento viene realizzato solo in una certa misura e i gateway di rete principali eseguono ancora sia le funzionalità del piano utente che del piano di controllo. La divisione completa tra i due, Control and User Plane Separation, **CUPS**, è stata realizzata a partire dalla Release 14 [4], [14].

La figura [2.3] mostra l'architettura dell'Evolved Packet System, **EPS**⁴ [37]. La parte centrale rappresenta l'EPC che si occupa di gestire la qualità di servizio, QoS, la mobilità e le policy di accesso. Le diverse RAN sono collegate all'EPC tramite il Serving Gateway, **S-GW**, per quanto riguarda la gestione del traffico dati utente e tramite la Mobility Management Entity, **MME**, per quanto riguarda i dati di controllo. Le RAN non 3GPP trusted sono collegate direttamente al Packet Data Network Gateway, **PDN-GW**, mentre quelle non trusted sono collegate all'Evolved Packet Data Gateway, **ePDG**, che fornisce principalmente funzionalità di sicurezza come il tunneling IPsec. Il **PDN-GW** è il componente architetturale comune per le entità sopra evidenziate ed è il punto di entrata e uscita del traffico da e verso gli utenti mobili.

⁴Termine 3GPP che si riferisce a un sistema end-to-end completo, ovvero la combinazione di User Equipment (UE), RAN e la stessa Packet Core Network (EPC).

Il Policy and Charging Rules Functions, **PCRF**, è responsabile di fornire le informazioni di QoS al PDN-GW. Queste informazioni possono includere regole di tariffazione, regole di controllo del flusso e priorità del traffico.

I registri Home Subscriber Service, **HSS**, e Authentication, Authorization, Accounting, **AAA**, si occupano della registrazione dell'utente all'EPC e ne mantengono le informazioni.

L'entità Access Network Discovery and Selection Function, **ANDSF**, si occupa di individuare la rete di accesso dello User Equipment⁵, **UE**, e gestisce la politica di mobilità.

Le componenti e le interfacce logiche principali dell'EPC sono trattati in maniera più approfondita nella sezione [2.4.1].

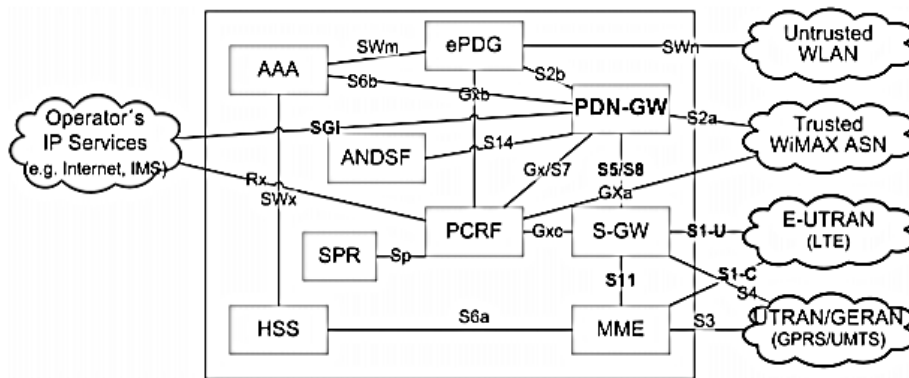


Figura 2.3: Architettura Evolved Packet System

2.4.1 Evolved Packet Core

La figura [2.4] mostra in maniera semplificata le componenti e le interfacce logiche principali dell'EPC differenziando il piano di controllo da quello utente. Ovviamente, la rete di un operatore include diverse entità che svolgono le funzioni di rete evidenziate con il fine di bilanciare il carico complessivo e di garantire tolleranza ai guasti.

Mobility Management Entity

È il nodo di rete responsabile di tutti gli scambi dei segnali di controllo tra le stazioni base e la rete centrale e tra gli utenti e la rete centrale. Nelle reti

⁵Qualsiasi device usato dall'utente per comunicare

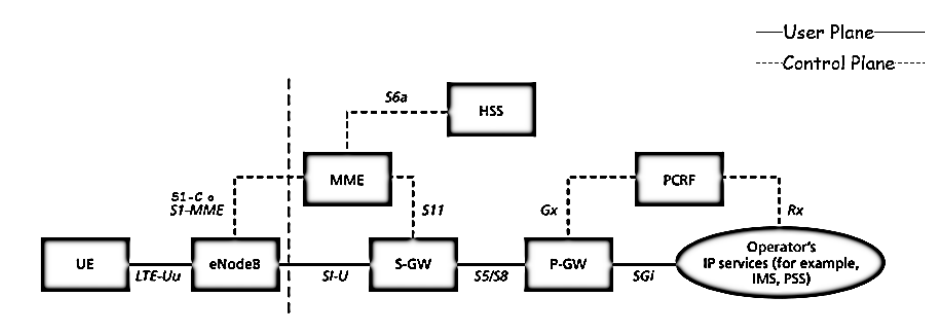


Figura 2.4: Architettura semplificata Evolved Packet Core

di grandi dimensioni, di solito ci sono molte MME per far fronte alla quantità di segnalazioni e a causa della ridondanza delle stazioni radio. Queste segnalazioni non hanno nulla a che fare con l'interfaccia aerea. L'insieme dei protocolli per la trasmissione dei segnali di controllo tra UE e MME prende il nome di Non-Access Stratum, **NAS**. Ad un alto livello di astrazione, la MME ha due funzionalità principali: gestire i bearers e gestire le connessioni.

Nelle reti a commutazione di circuito gli utenti ricevono una porzione fissa della larghezza di banda della rete. Diversamente nelle reti a pacchetto gli utenti ricevono un bearer con una certa quality of service (QoS) che spazia da una larghezza di banda minima garantita fino a un servizio best efforts senza nessuna garanzia. Un bearer quindi può essere visto come un canale logico con un livello di QoS associato. Gestire i bearers significa creare e mantenere i diversi canali logici e garantire la QoS associata. La gestione della connessione è correlata alla creazione della connessione fisica e alla sicurezza tra la rete core e l'UE.

Più nello specifico le funzionalità della MME sono:

- **Autenticazione.** Quando un utente abbonato si collega per la prima volta alla rete LTE, l'eNode-B comunica con la MME sull'interfaccia S1 e agevola lo scambio delle informazioni di autenticazione tra il dispositivo mobile e la MME. Questo richiede quindi le informazioni di autenticazione all'HSS, che verrà descritto in seguito e autentica l'utente. Una volta fatto ciò, se la procedura di autenticazione va a buon fine, inoltra le chiavi di crittografia all'eNode-B in modo che questa possa comunicare con l'UE in maniera crittografata sull'interfaccia aerea.
- **Istituzione dei bearers.** La MME comunica con altri componenti della rete per la creazione di un bearer tra l'eNode-B e il PDN-GW. Se sono presenti più istanze di quest'ultimo è compito della MME scegliere quale usare.

- **Gestione della mobilità NAS.** Nel caso di inattività prolungata del dispositivo mobile (dai 10 ai 30 secondi nella pratica), la connessione dell'interfaccia aerea e le risorse allocate nella rete radio vengono rilasciate (UE passa in stato idle). Il dispositivo mobile è libero di spostarsi tra diverse celle della stessa Tracking Area, **TA**⁶, senza notificare la rete in modo da non sovraccaricare quest'ultima e risparmiare batteria. Nel caso in cui arrivino pacchetti dati da Internet indirizzati al dispositivo, la MME deve inviare dei messaggi di paging a tutti gli eNode-B appartenenti alla TA del dispositivo mobile, quest'ultimo risponderà poi al paging al fine di ristabilire il bearer.
- **Supporto all'handoff.** Nel caso in cui non sia disponibile l'interfaccia che connette tra loro i diversi eNode-B, X2, la MME aiuta a inoltrare i messaggi di handoff tra i due eNode-B coinvolti. Nel caso in cui, dopo la procedura di handoff, cambi il PDN-GW di riferimento è compito della MME ristabilire il tunnel IP (bearer) tra questo e l'utente.
- **Interoperabilità con altre reti radio.** Quando un dispositivo mobile raggiunge il limite dell'area di copertura LTE, l'eNode-B può decidere di "spostare" il dispositivo mobile su una rete GSM o UMTS oppure istruire quest'ultimo a eseguire un cambio di cella in modo che possa "spostarsi" su una cella più adatta. In entrambi i casi, la MME è l'entità che si occupa della gestione complessiva della procedura e comunica con le componenti della rete GSM o UMTS.
- **Supporto per SMS e voce.** Siccome la rete LTE è interamente basata su IP, sono necessarie alcune funzionalità aggiuntive per supportare i servizi tradizionali, SMS e chiamate vocali, della core network GSM o UMTS a commutazione di circuito. Per quanto riguarda gli SMS, la specifica [5] introduce la funzionalità di SMS su SGs. Come mostrato in figura [2.5], l'interfaccia SGs ha il compito di inoltrare i messaggi SMS tra un MSC appartenente alla core network GSM o UMTS e la MME. Dalla MME, l'SMS viene incapsulato in un messaggio di segnalazione NAS, incapsulato poi in un messaggio di segnalazione RRC dall'eNode-B e consegnato al dispositivo mobile. Sui messaggi generati da quest'ultimo viene eseguita la procedura inversa. L'immagine [2.6] mostra lo stack dei protocolli dell'interfaccia aerea; da questa si evince che non è necessaria nessuna applicazione di livello superiore basata su IP per inviare e ricevere messaggi. Per quanto riguarda la voce, il sistema progettato prende il nome di Voice over LTE, **VoLTE** e si basa sul

⁶Raggruppamento logico di celle nelle reti LTE

protocollo SIP⁷. Se un utente lascia l'area di copertura LTE mentre è in corso una chiamata, basata su IP, questa deve essere convertita al volo in una chiamata a commutazione di circuito e renderizzata sulla rete GSM o UMTS. Questa procedura prende il nome di Circuit-Switched Fallback, **CSFB**.

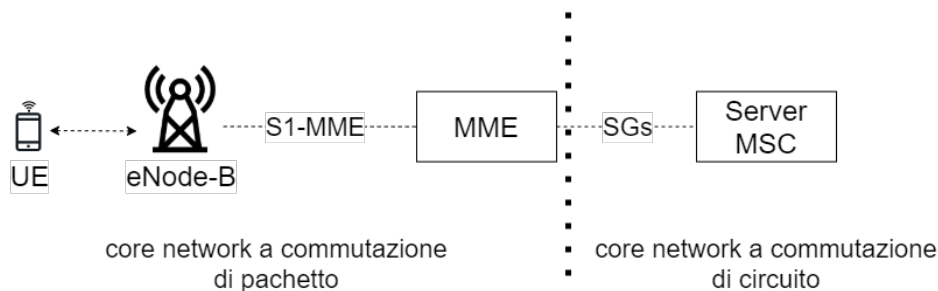


Figura 2.5: Interconnessione SGs per SMS su LTE

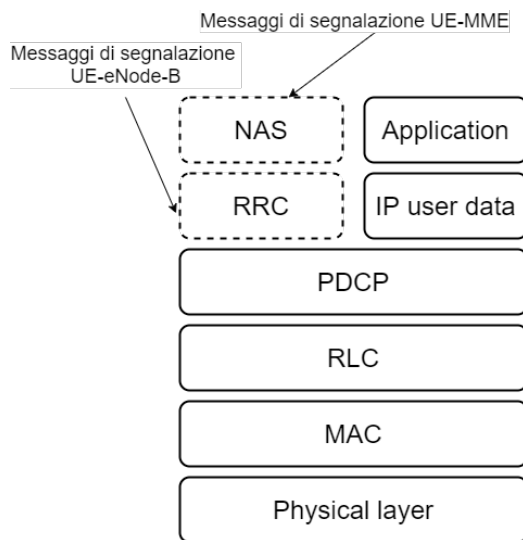


Figura 2.6: Stack protocolli interfaccia aerea LTE

Home Subscriber Server

È responsabile della memorizzazione dei dati per il profilo del cliente e fornisce alla MME i dati necessari per l'autenticazione di questo. Contiene anche informazioni relative al PDN-GW a cui l'utente può connettersi. Oltre a ciò,

⁷Protocollo di segnalazione utilizzato per iniziare, modificare o terminare sessioni fra uno o più partecipanti.

memorizza la MME presso cui l'utente è attualmente collegato. I parametri utente più importanti in HSS sono:

- International Mobile Subscriber Identity, **IMSI**, che identifica in modo univoco un utente abbonato. L'IMSI include implicitamente il Mobile Country Code, **MCC** e il Mobile Network Code **MNC** e viene utilizzato quando l'utente è in roaming per trovare la sua rete domestica ed accedere all'HSS. Per poter fare ciò, una copia dell'IMSI è memorizzata nella SIM dell'abbonato.
- Informazioni di autenticazione utilizzate per l'autenticazione dell'abbonato e per la generazione di chiavi crittografiche.
- Informazioni in merito ai servizi a commutazione di circuito come il numero di telefono dell'utente, denominato Mobile Subscriber Integrated Services Digital Network, **MSISDN**, e i servizi che l'utente può utilizzare come SMS, inoltro di chiamata e così via. Mentre il MSISDN è utilizzato per alcuni scopi in LTE, gli altri valori sono principalmente di interesse mentre l'utente è connesso a GSM o UMTS.
- Informazioni in merito ai servizi a commutazione di pacchetto come i nomi dei punti di accesso, **APN**, che l'abbonato è autorizzato a utilizzare.
- l'ID del MSC attualmente in servizio in modo che le chiamate, a commutazione di circuito, in entrata e gli SMS possano essere instradati correttamente.
- l'ID di SGSN o MME per poterli aggiornare nel caso in cui il profilo utente venga aggiornato.

Serving Gateway

Il Serving Gateway, **S-GW**, è uno dei due gateway che elaborano il traffico del piano dati. Ogni UE è solitamente servita da un SGW unico alla volta. L'SGW:

- è responsabile dello scambio del traffico tra P-GW e 4G RAN
- è il nodo di appoggio della mobilità del piano dati quando l'UE cambia eNode-B
- bufferizza i pacchetti downlink destinati agli UE inattivi (ovvero in stato di idle)
- svolge le funzioni di fatturazione e intercettazione del traffico.

Packet Data Network Gateway

Il PDN-GW interconnette l'EPC con reti IP esterne (ad esempio, Internet). Per questo motivo, è il componente utilizzato per eseguire operazioni di filtraggio e di ispezione del traffico. Il PDN-GW è anche responsabile dell'assegnazione degli indirizzi IP ai dispositivi mobili. Quando un dispositivo mobile si connette alla rete, l'eNode-B contatta la MME, come descritto in precedenza, al fine di autenticare l'utente. La MME quindi, autentica l'abbonato e richiede un indirizzo IP per il dispositivo al PDN-GW. Oltre a ciò, svolge un ruolo importante per gli scenari di roaming internazionale. Per far sì che un utente possa accedere senza interruzioni a Internet anche quando è all'estero, le interfacce di roaming collegano le principali reti, LTE, UMTS e GPRS⁸, di diversi operatori di rete. In questo modo, la rete straniera può contattare la rete domestica di un utente al fine di autenticare quest'ultimo. Quando viene stabilito un bearer, ad esempio, per l'accesso ad Internet viene creato un tunnel GTP [2.4.1] tra l'S-GW della rete ospitante e il PDN-GW della rete domestica. Questa tecnica è denominata **home routing** [2.7]. Lo svantaggio è che i dati dell'utente vengono ritrasportati nella rete domestica per poter essere inviati su Internet; per questo, gli standard specificano un'alternativa denominata **local breakout** in cui la connessione ad Internet è stabilita direttamente dal PDN-GW della rete ospitante.

Policy and Charging Rules Functions

Il PCRF è l'entità che gestisce le policy e le funzionalità di addebito e decide come devono essere trattati i pacchetti. È l'elemento chiave del framework Policy and Charging (PCC), che include anche la Policy and Charging Enforcement Function, PCEF, e fornisce:

- un controllo dinamico, basato sul flusso dati, delle policy/funzionalità di addebito
- una gestione a grana fine della QoS per ciascun servizio offerto

Il PCEF fa parte del PGW ed è utilizzato dal PCRF per applicare le politiche e le regole di tariffazione e per controllare le risorse associate ai vari bearer.

Le principali funzioni del PCC sono le seguenti:

- **Gating control.** Il PCRF decide se i pacchetti appartenenti ai flussi di traffico associati a un determinato servizio devono essere bloccati

⁸General Packet Radio Service è una delle tecnologie di telefonia mobile cellulare. Viene definita 2.5G a delineare il fatto che è una via di mezzo tra GSM (2G) e UMTS (3G)

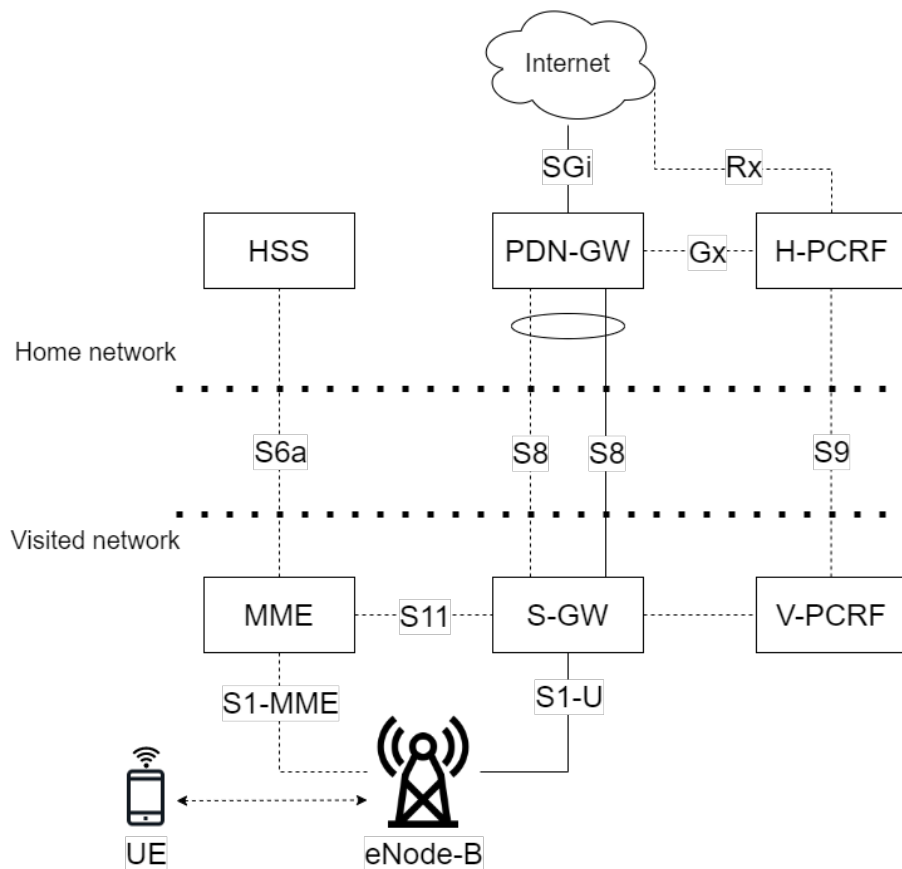


Figura 2.7: Scenario semplificato roaming LTE tramite home routing

o meno e invia le sue decisioni al PCEF che gestisce i pacchetti di conseguenza. Il PCRF può prendere decisioni sulla base di informazioni ed eventi di sessione segnalati da funzioni applicative esterne.

- **Controllo QoS.** Il PCRF fornisce al PCEF i parametri QoS associati ai flussi di traffico (ad esempio, bit rate, latenza massima). Il PCEF applica quindi le regole QoS configurando i bearer appropriati.
- **Controllo della tariffazione online e offline.** Il PCRF decide se la tariffazione online o offline deve essere applicata per una determinata sessione di servizio. Il PCEF raccoglie i dati di tariffazione e interagisce con i sistemi adibiti a tale attività, OCS e OFCS. Inoltre, il PCEF può eseguire misurazioni sui flussi IP quando specificato dal PCRF.

Il PCRF fornisce le sue decisioni sotto forma di regole PCC, che contengono le informazioni utilizzate dal PCEF e dai sistemi di tariffazione. Le regole

PCC contengono i modelli SDF (Service Data Flow) utilizzati dal PCEF per identificare i pacchetti IP appartenenti a una determinata sessione di servizio sulla base di indirizzi di origine e destinazione, numeri di porta e protocollo. Le regole PCC contengono anche i parametri QoS che devono essere applicati.

Come mostrato nella figura [2.7] in un contesto di roaming internazionale, tramite home routing, abbiamo due entità che svolgono la funzionalità di PCRF: H-PCRF e V-PCRF. Se la V-PCRF determina, sulla base dell'identità dell'utente, che questo è in roaming, stabilisce una connessione di controllo con H-PCRF per una corretta gestione delle policy e delle funzionalità di addebito [11].

Interfacce logiche

La figura [2.4] mostra, oltre ai componenti logici dell'EPC, le principali interfacce per il collegamento di questi. Le funzionalità di queste ultime sono introdotte e descritte in diversi standard 3GPP [10], [6].

- **S1-MME**. Utilizzata per lo scambio dei segnali di controllo tra MME e un eNode-B posizionato nella E-UTRAN.[24.301]
- **S1-U**. Utilizzata per il tunneling del traffico del piano utente tra eNode-B e S-GW e per supportare le procedure di handoff.
- **S6a**. Utilizzata per interconnettere MME e HSS al fine di scambiare informazioni utili all'autenticazione e autorizzazione dell'utente.
- **S11**. Utilizzata per la segnalazione di controllo tra MME e S-GW.
- **S5**. Utilizzata per il tunneling del traffico del piano utente e per lo scambio di segnali del piano di controllo al fine di gestire la sessione tra S-GW e PDN-GW. Nel caso di roaming, come abbiamo già visto, si utilizza come etichetta **S8**.
- **SGi**. Utilizzata per interconnettere il PDN-GW con la Packet Data Network. Questa può essere una rete pubblica o privata di un operatore o una rete intra-operatore.
- **Gx**. Utilizzata per inviare le regole PCC dal PCRF al PCEF, situato nel PDN-GW.
- **Rx**. Utilizzata dalle Application Functions esterne per inviare al PCRF report in merito ai servizi.

Nel caso di più MME, l'interconnessione tra questi avviene tramite l'interfaccia logica **S10**.

Protocolli

GPRS Tunneling Protocol (GTP) Il protocollo GTP è stato introdotto nello standard GSM, più dettagliatamente a partire da GPRS, per soddisfare esigenze specifiche come la mobilità, gestione dei bearer e per effettuare il tunneling dei pacchetti IP attraverso il core della rete.

Un path GTP viene identificato in ogni nodo da: indirizzo IP e un numero di porta UDP ed è utilizzato per multiplexare i tunnel GTP (ovvero ad un path possono essere associati più tunnel) [7]. Un tunnel GTP è identificato in ogni nodo da: Tunnel Endpoint Identifier, **TEID**, indirizzo IP e un numero di porta UDP. In una comunicazione tra due nodi, il trasmittente deve utilizzare lo stesso valore di TEID assegnato dal ricevente. Il valore TEID deve essere assegnato in modo non prevedibile per le interfacce PGW S5, S8, S2a, S2b. Un tunnel può essere GTP-C, dedicato al piano di controllo e GTP-U dedicato al piano utente. I due sotto protocolli sono associati a delle versioni; in questa trattazione ci concentreremo su GTPv2-C, introdotto durante lo sviluppo dell'EPS per migliorare la gestione di bearers e fa riferimento a [2], e GTPv1-U che fa riferimento a [8].

I tunnel vengono creati, gestiti e rilasciati tramite messaggi GTP-C. I criteri per stabilire quando tunnel uguali o differenti devono essere usati variano a seconda delle interfacce. Per il piano di controllo, per ogni endpoint:

- Il TEID-C deve essere univoco per connessione con PDN-GW sulle interfacce S2a, S2b, S5 e S8 basate su GTP. Lo stesso tunnel sarà condiviso per i messaggi di controllo relativi a tutti i bearers associati alla connessione.
- Ci deve essere solo una coppia di TEID-C per UE su ciascuna delle interfacce S3, S10, S16 e N26. Lo stesso tunnel sarà condiviso per i messaggi di controllo relativi allo stesso terminale.
- Ci deve essere solo una coppia di TEID-C per UE sulle interfacce S11 e S4. Lo stesso tunnel sarà condiviso per i messaggi di controllo relativi allo stesso terminale.

GTP-U viene utilizzato per l'incapsulamento e il tunneling del traffico dati degli utenti. Quando un UE invia pacchetti IP verso un PDN esterno (ad esempio Internet) [2.12a] questi seguono il seguente percorso:

1. **UE** → **eNB**. L'UE invia un pacchetto IP all'indirizzo di destinazione 74.125.71.104 (www.google.com), simile a questo [2.8], sul canale radio.



Figura 2.8: Pacchetto IP tra UE ed eNB

2. **eNB** → **S-GW**. L'eNB aggiunge al pacchetto ricevuto un header **GTP tunnel** che consiste in 3 header singoli: header IP, header UDP, header GTP. Il pacchetto derivante è [2.9].



Figura 2.9: Pacchetto GTP tra eNB e S-GW

3. **S-GW** → **PDN-GW**. L'S-GW modifica l'header GTP e l'header IP più esterno [2.10].



Figura 2.10: Pacchetto GTP tra S-GW e PDN-GW

4. **PDN-GW** → **PDN esterno**. Il PDN-GW rimuove gli header più esterni e trasmette il pacchetto IP originale verso la destinazione.

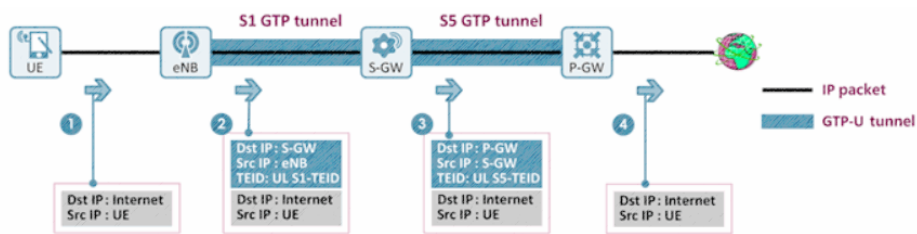


Figura 2.11: Pacchetto GTP tra PDN-GW e PDN

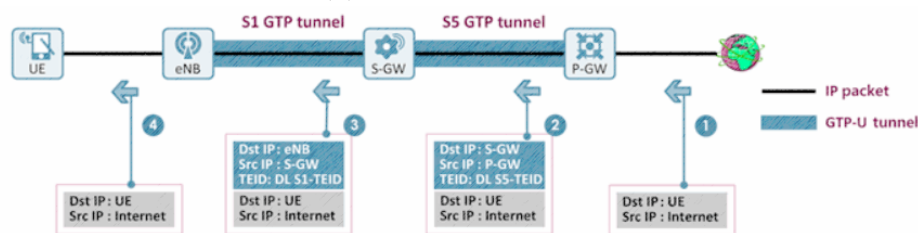
Per i pacchetti provenienti da Internet verso l'UE si applica il procedimento contrario [2.12b].

Diameter Il protocollo Diameter è stato originariamente progettato per Autenticazione, Autorizzazione e Accounting ed è il successore del protocollo RADIUS. Diameter è stato progettato per sopperire a diverse carenze di RADIUS e migliorare alcuni aspetti: gestisce in maniera migliore gli errori, presenta una consegna più affidabile dei messaggi, migliora la sicurezza, le unità informative trasportate possono essere di dimensione maggiore, eccetera. La versione base di Diameter è standardizzata in [20].

All'interno di un EPS il protocollo è presente in diverse interfacce, tra cui:



(a) Da UE a Internet



(b) Da Internet a UE

Figura 2.12: GTP tunneling e incapsulamento

- **s6a.** Al fine di trasportare i dati dell'utente mobile, trasportare le informazioni relative alla posizione di questo, autorizzare l'utente consentendogli di accedere all'EPS e trasportare le informazioni di autenticazione.
- **Gx.** Mette in comunicazione il PCRF con il PDN-GW, in particolare il PCEF, permettendo lo scambio delle regole PCC.

Per trasportare i messaggi tra due nodi, Diameter si appoggia a **TCP** o Stream Control Transport Protocol, **SCTP**. Siccome entrambi sono orientati alla connessione, è necessario instaurare quest'ultima prima di poter comunicare. I messaggi possono essere protetti mediante l'utilizzo di protocolli crittografici come TLS.

S1-AP Il protocollo S1-AP è stato progettato per una sola interfaccia, S1-MME, da cui trae il nome. È il protocollo a supporto di tutte le procedure tra eNode-B e MME e, in maniera trasparente, di tutte le procedure tra UE e MME o gli altri nodi dell'EPC.

Il protocollo di basa su *procedure elementari* che consistono in un messaggio iniziale seguito da una risposta. Le diverse procedure elementari sono indipendenti tra loro e possono essere concatenate in maniera flessibile per costruire procedure complesse.

Non-Access Stratum (NAS) Il protocollo NAS è utilizzato tra UE e MME per gestire le procedure di mobilità, EPS Mobility Management (EMM), e di gestione delle sessioni, EPS Session Management (ESM).

Le procedure EMM sono usate per tenere traccia dell'UE, autenticarlo, fornire le chiavi di sicurezza, controllo dell'integrità e cifratura. Inoltre, tali procedure possono fornire alla rete dati legati alla capacità di risorse dell'UE e a quest'ultimo informazioni legate ai servizi di rete. Le procedure ESM sono usate per la gestione dei bearer: instaurazione, modifica, abbattimento.

Un messaggio NAS contiene un campo di 4 bit, **protocol discriminator**, che indica se il messaggio è legato a una procedura EMM o ESM e un campo, **message identity**, che costituisce l'identificativo del messaggio. Nel caso di procedure MME il messaggio contiene un campo aggiuntivo, **security header**, se l'integrità del messaggio è protetta e/o cifrata. D'altro canto, anche messaggi EMS contengono campi aggiuntivi: **EPS bearer identity** che costituisce l'identificativo del bearer di interesse e **Protocol Transaction Identifier** che identifica una specifica tipologia di messaggio NAS tra UE e MME.

CUPS

Il design architetturale dell'EPC ha permesso di separare le funzioni principali del piano di controllo nella MME. Tuttavia, SGW e PGW eseguono molte funzioni del piano di controllo durante la creazione delle sessioni. Per questo motivo, a partire dalla versione 14 di 3GPP i due componenti sono stati divisi in due parti: una dedicata alla gestione delle funzioni del piano dati e una a quelle del piano di controllo. Quindi, SGW si suddivide in SGW-C e SGW-U e in maniera analoga PGW in PGW-C e PGW-U [2.13].

2.5 Quinta generazione: 5G

Il 4G è stato progettato per migliorare la capacità, velocità, latenza dei dati utente e l'utilizzo dello spettro di banda rispetto al 3G. Il 5G, non è solo un'evoluzione della tecnologia a banda larga, ma costituisce uno degli elementi fondamentali per la digitalizzazione del futuro e supporterà la trasformazione dei processi di tutti i settori economici. Offrirà nuovi servizi agli utenti consumatori, ma anche a figure aziendali quali: industrie verticali⁹, fornitori

⁹Un mercato verticale è un mercato in cui i venditori offrono beni e servizi specifici per un settore, un commercio, una professione o un altro gruppo di clienti con esigenze specializzate. Le industrie verticali operano in questi mercati. Esempi di industrie verticali sono: l'automotive, produttori di stampanti 3D, sviluppatori di sistemi ERP...

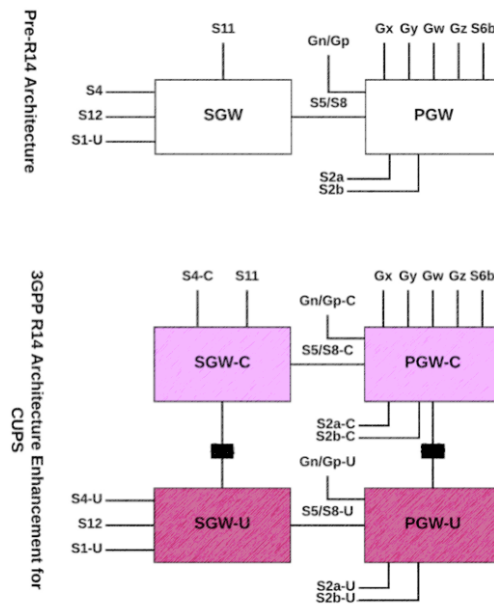


Figura 2.13: EPC Control and User Plane Separation

di servizi e proprietari o fornitori di infrastrutture di rete.

In primo luogo, garantirà la continuità dell'esperienza utente in situazioni difficili: aree densamente popolate, aree rurali, alta mobilità. Il sistema 5G fornirà l'accesso all'utente ovunque esso si trovi e selezionerà in maniera trasparente la tecnologia di accesso migliore. Questa scelta non si baserà solo sul throughput, ma verrà presa a seconda della natura del servizio. Ad esempio, per quanto riguarda i giochi online una bassa latenza è preferibile ad un alto throughput.

Il 5G rivoluzionerà anche il mondo dell'Internet of Things fornendo l'infrastruttura di rete per connettere in Internet un numero enorme di sensori e attuatori e supportando questi tramite un consumo energetico molto basso in modo da poter risparmiare batteria.

In aggiunta, servizi *mission critical* diventeranno realizzabili nativamente tramite l'infrastruttura 5G senza la necessità di creare reti specifiche per ragioni di affidabilità e sicurezza. Coprirà anche nuovi servizi che richiedono una reattività in tempo reale come i servizi Vehicle-to-Vehicle o Vehicle-to-Road, aprendo la strada all'auto a guida autonoma, all'automazione di fabbrica o ai servizi sanitari a distanza. L'immagine [2.14] fornisce una misura numerica dei miglioramenti dell'infrastruttura 5G rispetto a quella 4G negli scenari introdotti.

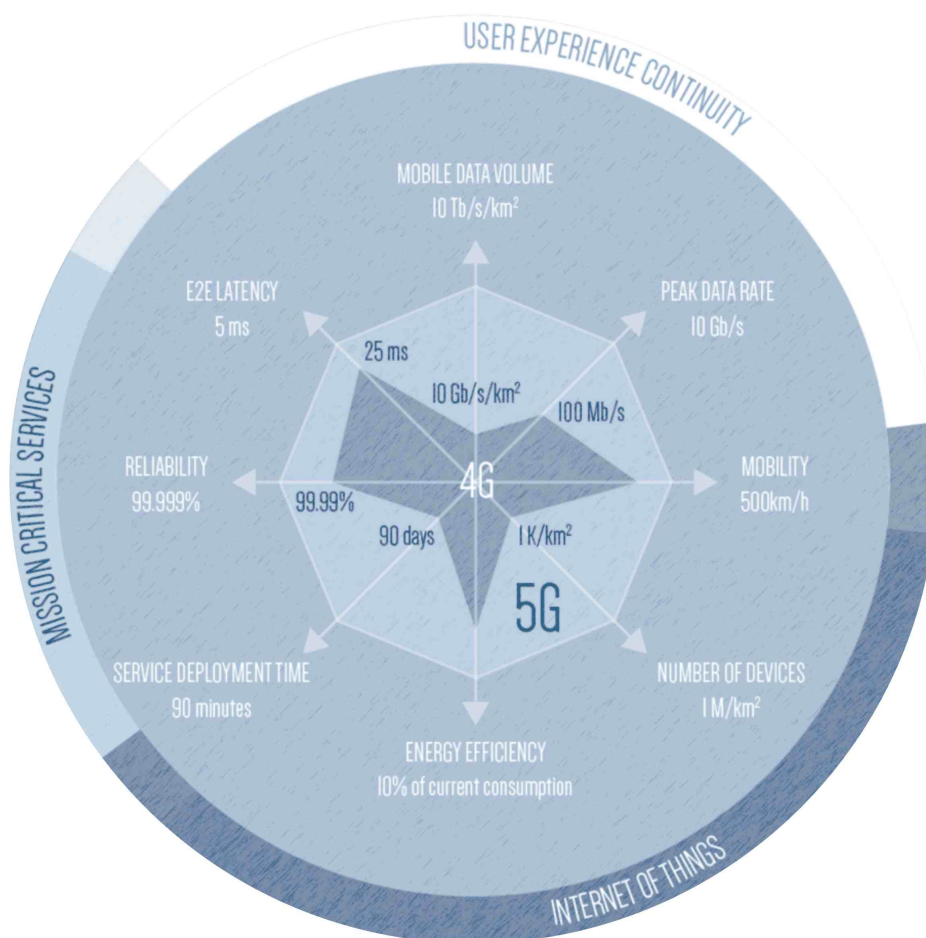


Figura 2.14: Capacità dell'infrastruttura 5G rispetto a quella 4G

In conclusione, il 5G sarà in grado di soddisfare le esigenze di diversi settori e diversi casi d'uso aventi requisiti più disparati: throughput, latenza, consumo energetico, alta mobilità, eccetera. In [17] è possibile trovare un'ampia lista dei requisiti che l'infrastruttura sarà in grado di soddisfare accompagnati da un insieme di KPI per valutarne la bontà di soddisfacimento.

Un fattore abilitante per ottenere la flessibilità e la scalabilità necessarie all'infrastruttura per soddisfare i diversi requisiti è la programmabilità di rete. Questa permette di progettare, implementare, distribuire, gestire e mantenere le diverse componenti della rete mediante la programmazione software. La flessibilità del mezzo software permette la riprogettazione dell'architettura e dei servizi di rete, l'ottimizzazione dei costi e dei processi e la gestione automatizzata della rete. La programmabilità di rete pone l'attenzione sui concetti di: Network Function Virtualization, **NFV**, trattato nel capitolo precedente [1], Software Defined Network, **SDN** e Multi-access

Edge Computing, **MEC** considerati i pilastri tecnologici del 5G [18]. **SDN** e **NFV** aprono nuovi orizzonti anche per quanto riguarda il Service Function Chaining, **SFC**. Quest'ultimo è un meccanismo che consente di definire e istanziare un insieme ordinato di servizi di rete e successivamente indirizzare il traffico attraverso questi. SFC offre possibilità interessanti per diversi domini, tra cui il **Network Slicing**, che consente all'infrastruttura di rete di fornire servizi personalizzati, dedicati e logicamente isolati in base alle differenti esigenze dei vari settori verticali.

2.5.1 Software Defined Network

SDN introduce la programmabilità dinamica dei nodi di rete che si occupano dell'inoltro dei dati. Per fare ciò propone di separare il piano di controllo da quello dati in modo da centralizzare il controllo di tutti i flussi dati e dei percorsi che essi intraprendono all'interno della rete. Nell'architettura SDN quindi le entità che si occupano del controllo della rete sono separate dall'hardware che effettua l'inoltro dei dati (ad esempio, switch, router, access point wireless) [39]. L'architettura prevede:

- introduzione di un'entità software, il controller, che svolge le attività del piano di controllo;
- un set di OpenAPI sia per quanto riguarda il piano di controllo che il piano dati. Il piano di controllo utilizza le API del piano dati per dirigere gli apparati di rete. Le API del piano di controllo sono messe a disposizione per realizzare in maniera semplice applicazioni di rete personalizzate.

Il controller SDN, prende tutte le decisioni e inoltra le regole definite ai nodi del piano dati. Questi ultimi sono entità molto semplici dotate di scarsa intelligenza, che implementano tecniche d'inoltro rapido dei dati.

In [2.15] è mostrata l'architettura SDN che si compone di tre livelli. Il livello più basso, **infrastructure layer** o **data plane**, rappresenta l'hardware che si occupa dell'inoltro dei dati e del raccoglimento di informazioni a fine statistico. Un livello sopra, si trova il **control layer** o **control plane**, che è responsabile della gestione e dell'orchestrazione del livello sottostante. Per fare ciò, utilizza le informazioni raccolte dal data plane e definisce opportune regole. I controller SDN si posizionano a questo livello e, come detto in precedenza, comunicano con il livello sottostante tramite apposite API denominate *southbound API*. Esistono diversi protocolli per le interfacce southbound: Forwarding and Control Element Separation (ForCES), SoftRouter, OpenFlow; quest'ultimo è il più comune e utilizzato motivo per

cui è considerato standard de facto. L'**application layer** contiene le applicazioni di rete in grado di introdurre nuove funzionalità: ingegneria del traffico, bilanciamento del carico di server applicativi, controllo degli accessi, sicurezza, eccetera. Questo livello possiede una visione astratta e globale della rete fornitagli dal livello di controllo ed utilizza le *northbound API* per guidare quest'ultimo.

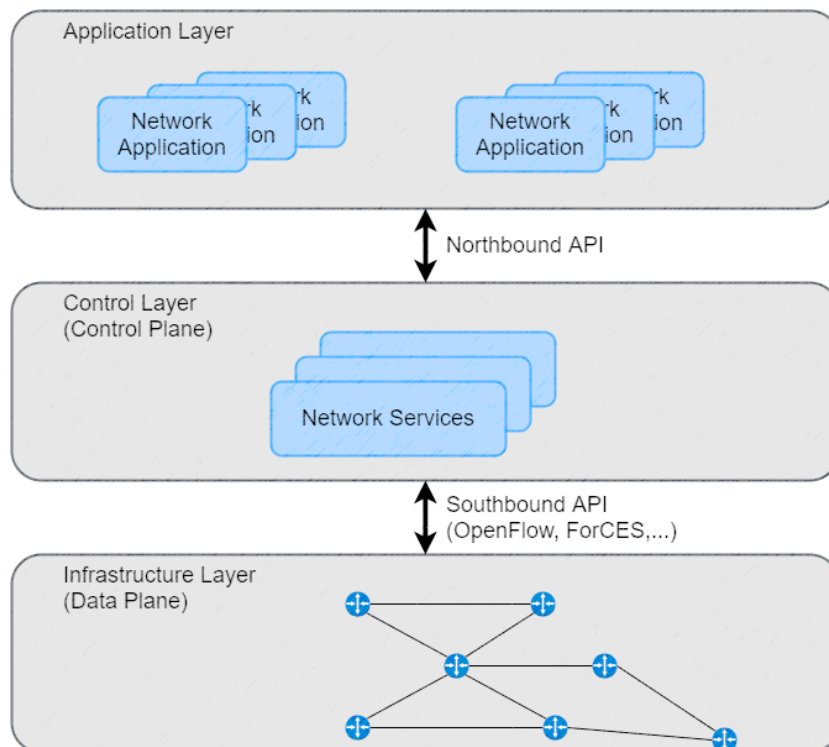


Figura 2.15: Architettura Software Defined Network a tre livelli [19]

2.5.2 Network Slicing

Il network slicing consiste nel partizionamento di una rete fisica in diverse reti virtuali; ogni rete può essere personalizzata e ottimizzata per un tipo specifico di applicazione o utente. Sfruttando il Cloud Computing, le tecnologie di virtualizzazione e i concetti introdotti nelle due sezioni precedenti, le risorse della rete fisica possono essere gestite dinamicamente ed efficientemente per creare reti logiche ad hoc [55]. Tutto ciò, avviene in maniera trasparente dal punto di vista dell'utente finale.

Un network slice è composto da un insieme di funzioni di rete che cooperano tra loro per soddisfare lo specifico caso d'uso. Oltre alla possibilità

di personalizzazione, gli slice, sfruttando la flessibilità della virtualizzazione, possono essere riconfigurati nel caso in cui i requisiti da soddisfare mutino.

L'immagine [2.16] mostra la differenza tra un'infrastruttura di rete unica per casi d'uso con requisiti diversi e l'introduzione di network slice aventi caratteristiche differenti: larghezza di banda, bassa latenza e basso consumo capaci di soddisfare al meglio i bisogni di ciascun caso d'uso.

In [30] è proposta l'adozione di Generic Slice Template, **GST**, ovvero un insieme di attributi: throughput supportato, API fornite, funzionalità supportate, eccetera, per caratterizzare ogni slice. All'interno del GST i vari attributi, identificati da appositi nomi, sono associati a delle unità di misura. L'associazione di ciascun attributo a un valore o a un range di valori avviene nel Network Slice Type, **NEST**. Quest'ultimo serve per la creazione del Network Slice Template, **NST**, che contiene le funzioni di rete, la loro configurazione e le loro interconnessioni utili per fornire le funzionalità necessarie e soddisfare i livelli di servizio in base alle informazioni contenute nel NEST. Dal NST vengono poi create le Network Slice Instances, **NSIs**, ovvero l'insieme delle istanze di funzionalità e delle risorse di rete, elaborazione e storage che costituiscono lo slice. Per la costruzione del NST, oltre alle informazioni del NEST, sono tenute in considerazione quelle legate all'infrastruttura sottostante che ospiterà lo slice.

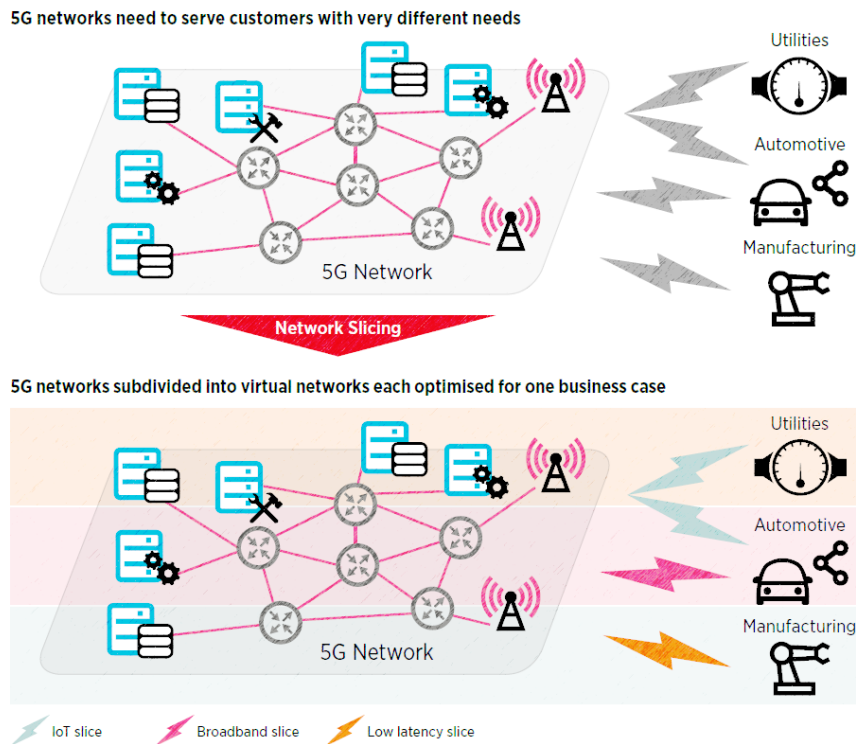


Figura 2.16: Network slicing per diversi casi d'uso [31]

Nella sottosezione successiva verrà introdotta l'integrazione di NFV e SDN per realizzare il Service Function Chaining e verrà analizzato come questo può essere impiegato per la creazione di slice di rete.

2.5.3 Service Function Chaining

Service Function Chaining, **SFC**, è un meccanismo che consente di individuare una lista di funzioni di rete (ad esempio, firewall o NAT) e collegarle assieme per formare una sorta di catena virtuale che prende il nome di Service Function Chain, **SFC**.

Quando arriva una richiesta per un servizio di rete, le funzioni necessarie per eseguire tale servizio devono essere individuate e i percorsi tra queste devono essere determinati; questi ultimi, spesso, devono avere una grandezza di banda garantita. In questo processo ci sono diversi requisiti legati alle performance come un tempo breve per creare l'istanza del servizio che risponde alla richiesta dell'utente e un elevato utilizzo dei dispositivi di rete.

Come detto in precedenza, nelle reti tradizionali tutte le funzioni di rete sono implementate da appositi dispositivi. Quando si ha la necessità di

gestire innumerevoli utenti e servizi questo causa problemi di mantenimento dell'infrastruttura in termini di lavoro necessario, tempo e costi. Inoltre, le funzioni di inoltro del traffico sono spesso incorporate nell'hardware e questo comporta inflessibilità nella scelta dei percorsi tra i nodi incaricati di svolgere le diverse funzioni di rete e di conseguenza l'incapacità della rete di adattarsi dinamicamente. NFV punta a risolvere il problema dei dispositivi di rete dedicati tramite l'utilizzo della virtualizzazione, mentre SDN, grazie alla separazione tra piano di controllo e piano dati, permette agli amministratori di rete di programmare il comportamento di questa tramite opportune interfacce (le northbound interfaces). Per questo motivo, la combinazione di NFV e SDN abilita alla creazione di SFC in maniera efficiente e flessibile senza il bisogno di configurazioni completamente manuali. In generale, la procedura che va dall'arrivo della richiesta di un servizio all'inizializzazione della SFC si compone di tre step:

1. generazione di una catena logica di funzioni in base alla richiesta di servizio;
2. determinazioni di nodi VNF per le funzioni individuate al punto precedente, e dei collegamenti tra questi;
3. installazione di regole di forwarding nel piano dati che hanno il compito di dirigere il traffico.

In [2.17] sono rappresentati due casi d'uso con requisiti differenti che possiedono una catena di servizi diversa a seconda delle esigenze.

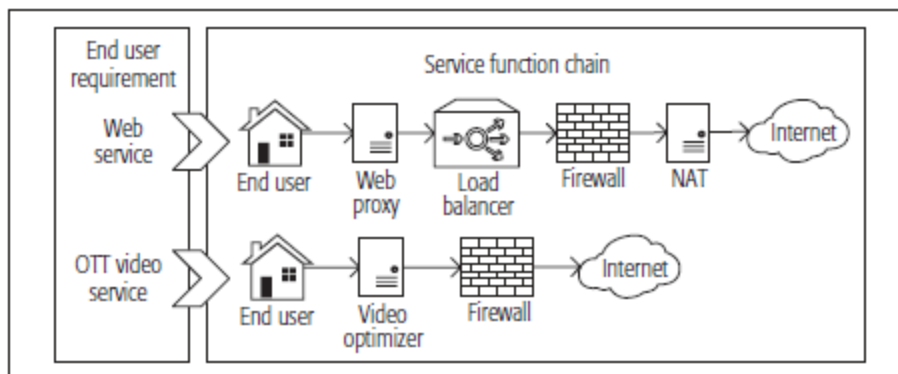


Figura 2.17: Scenari SFC [54]

L'integrazione di NFV e SDN solleva diversi problemi più o meno tecnici che devono essere risolti, ad esempio:

- posizionamento delle risorse SDN che costituiscono il data plane
- posizionamento del controller SDN
- posizionamento delle applicazioni SDN
- interazione tra le entità coinvolte per la fornitura di VNFs (Element Manager, VNF Manager, controller SDN e applicazioni SDN).

Per quanto riguarda il posizionamento delle risorse gli scenari possibili sono [2.18]:

- switch o router fisici
- switch o router virtuali
- switch SDN software based che sfrutta la scheda di rete di un server
- switch o router implementati come VNF

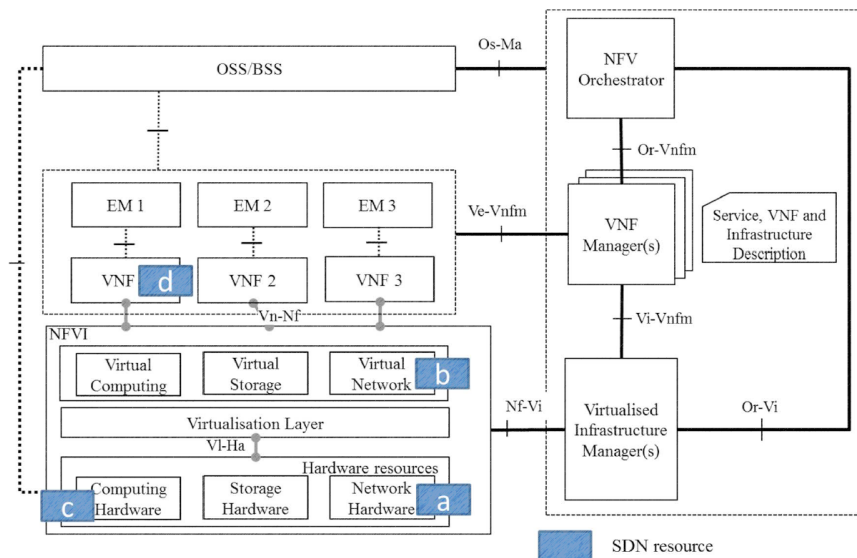


Figura 2.18: Possibili posizioni delle risorse SDN all'interno dell'architettura NFV

Gli scenari possibili per il controller SDN sono [2.19]:

1. le funzionalità del controller SDN sono integrate con quelle del VIM
2. il controller SDN è virtualizzato sotto forma di VNF

3. il controller è posizionato all'interno della NFVI. È il classico caso di controller usato per la connettività di rete all'interno della NFVI;
4. il controller può fare parte dell'OSS ed appartiene al tenant. Tale controller può essere collegato a un altro, che opera a livello di NFVI, in modo da costruire una gerarchia.
5. Il controller può essere una funzione di rete fisica, PNF, ovvero implementato in un apposito apparato hardware.

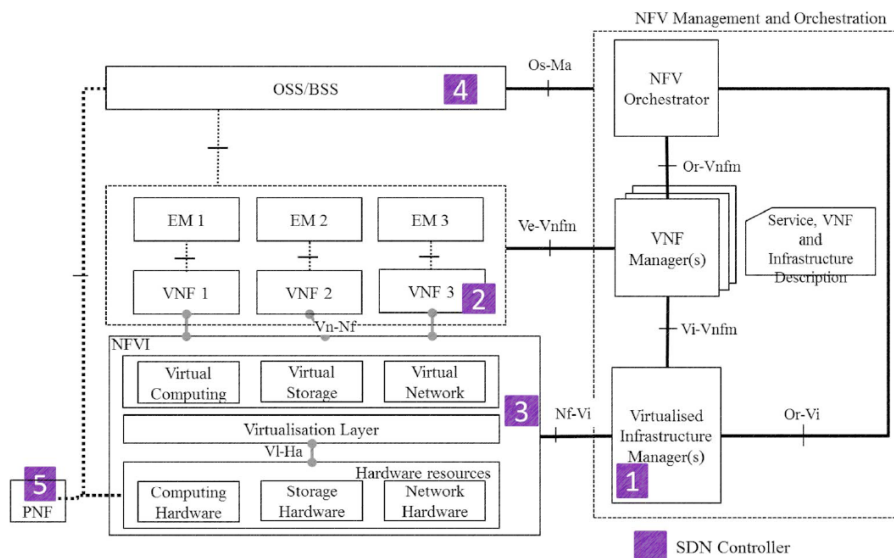


Figura 2.19: Possibili posizioni del controller SDN all'interno dell'architettura NFV

L'ultima entità da considerare sono le applicazioni SDN; anche in questo caso gli scenari possibili sono diversi e sono raffigurati in [2.20]

Il controller SDN possiede diverse interfacce per poter comunicare con: applicazioni SDN, risorse di rete SDN, altri controller SDN e l'NFV Orchestrator. In particolare, le diverse interfacce sono:

- **Application Control Interface.** Permette al controller di comunicare con le applicazioni e a queste ultime di richiedere risorse di rete.
- **Resource Control Interface.** Permette al controller di gestire le risorse di rete.

- **Orchestration Interface.** Controller e orchestratore, tramite questa interfaccia, possono scambiarsi diverse informazioni, ad esempio, quelle riguardanti la topologia di rete. In ottica SFC questa informazione è utile al primo per stabilire i percorsi tra i servizi e al secondo per poter distribuire i questi ultimi all'interno della NFVI.
- **Controller-Controller Interface.** Utili ai diversi controller, allo stesso livello gerarchico o meno, per potersi coordinare.

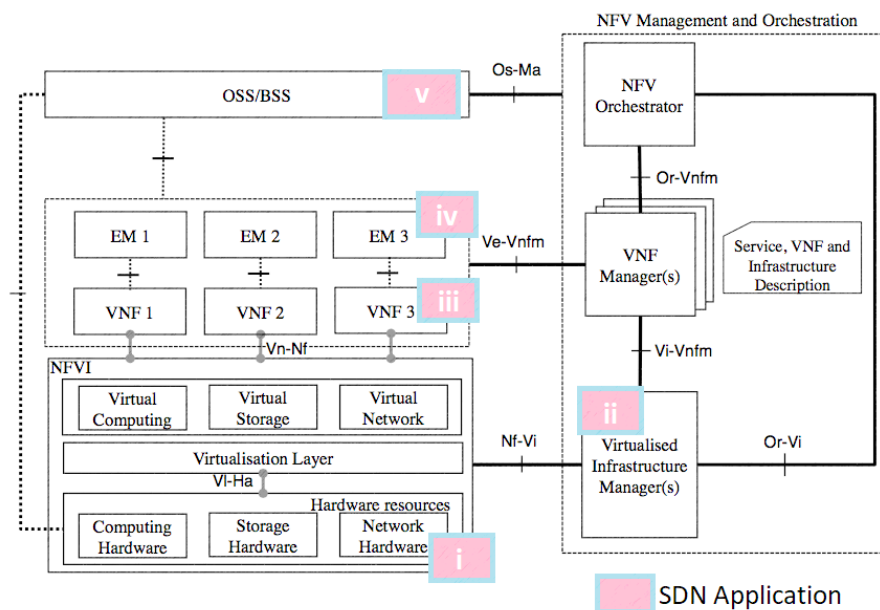


Figura 2.20: Possibili posizioni delle applicazioni SDN all'interno dell'architettura NFV

In [27] è proposto uno scenario in cui più controller SDN sono posizionati all'interno della NFVI e comunicano con i VIM tramite l'interfaccia logica *Nf-Vi* [2.21]. Si tratta dello scenario evidenziato in precedenza in cui i controller SDN sono usati per collegare diversi NVFI-PoP e formano un ulteriore livello di astrazione sotto i VIM che nasconde le risorse sottostanti e i dettagli tecnici utilizzati per fornire i servizi di connettività. Tramite l'utilizzo di SDN è possibile costruire in maniera dinamica il VNF Forwarding Graph, VNF-FC, che modella le connessioni tra le diverse VNF appartenenti ad un servizio di rete. La procedura per fare ciò si compone di tre passi:

1. **Incapsulamento:** il traffico fluisce attraverso un classificatore SDN, che può essere implementato come VNF, che lo incapsula aggiungendo-

gli un header. Questo header sarà indispensabile per stabilire il percorso del traffico e costruire la catena di servizi.

2. **Inoltro:** Il controller SDN svolge il ruolo di Service Function Forwarder, SFF, inoltrando il traffico in accordo con le informazioni contenute nell'header.
3. **Decapsulamento:** un classificatore SDN si occupa di rimuovere l'header dal traffico.

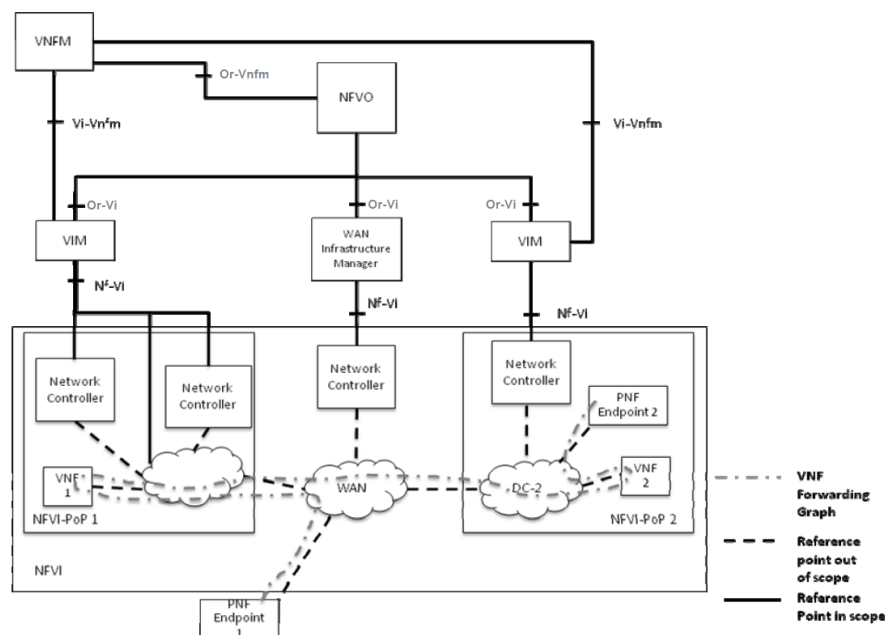


Figura 2.21: Controller SDN all'interno della NFVI per fornire connettività tra più PoP

In [33] è proposta un'architettura SFC che sfrutta NFV e SDN 2.22 i cui componenti sono:

- **Service function chain applications.** Sono applicazioni utilizzate dai tenant¹⁰ che registrano i requisiti di rete di questi ultimi e li traducono al controller SDN e agli NFV manager.
- **Orchestrator.** È l'entità che si occupa di orchestrare e gestire l'infrastruttura NFV e le risorse software.

¹⁰In ambiente Cloud, tenant (inquilino) individua un'organizzazione o un utente che, tramite un apposito piano tariffario, utilizza un set di risorse.

- **SDN Controller.** È l'entità centralizzata che possiede una visione generale della rete e si occupa di gestire i percorsi tra gli elementi di rete in base ai requisiti forniti dalle service function chain application.
- **NFV Manager.** È responsabile della gestione del ciclo di vita delle funzioni di rete virtualizzate ed esegue le operazioni di installazione, aggiornamento, query, ridimensionamento e terminazione.
- **Network elements.** Rappresentano le risorse di rete che possono essere hardware (ad esempio, un apparato che svolge il ruolo di firewall) oppure software (ad esempio, un componente che svolge il ruolo di load balancer eseguito in Cloud).

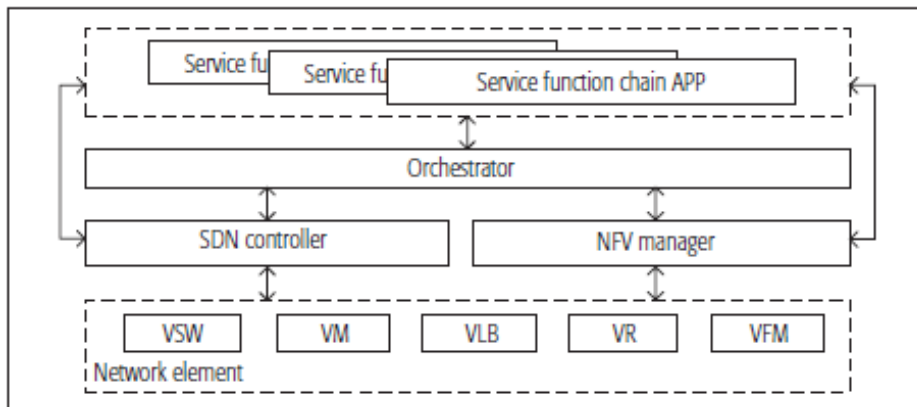


Figura 2.22: Architettura SFC che sfrutta NFV e SDN proposta da IETF

In [54] viene proposto un framework funzionale basato sull'architettura precedentemente proposta [2.23]. In questo caso, il controller SDN è integrato nel VIM.

Il **management plane** riceve le richieste di fornitura dei servizi effettuate tramite appositi programmi applicativi; una richiesta contiene informazioni come le funzioni di rete richieste e la larghezza di banda. Ricevuta una richiesta, questa viene inoltrata all'**SFC orchestrator**. Il modulo di **service modeling** si occupa di tradurre la richiesta in una descrizione SFC, che contiene le funzionalità da istanziare e le connessioni tra queste, e di consegnare questa all'**SFC resource allocation**. Questo modulo in base al grafo logico delle funzionalità nella descrizione seleziona i nodi VNF da utilizzare e i percorsi tra questi tramite appositi algoritmi.

Il **control plane** è formato da due componenti fondamentali: **NFV manager** e **SDN controller**. La prima tramite i moduli **VNF lifecycle management** e **VNF node monitoring** si occupa, rispettivamente, di gestire

il ciclo di vita delle VNF e di monitorare e reportare lo stato di esecuzione dei nodi che ospitano le funzionalità virtuali. La seconda, tramite i moduli di **traffic classification** e **traffic forwarding**, si occupa della definizione delle policy SFC che dovranno essere applicate dal data plane e, tramite il **network monitoring**, di monitorare lo stato della rete e fare reporting al cliente. Una policy SFC consiste in un insieme di regole da applicare, a livello di data plane, al traffico. Ogni regola contiene un insieme di campi che indicano a quale SFC il traffico appartiene e un insieme di azioni da applicare su tali campi.

Il **data plane** implementa le policy SFC ed effettua l'inoltro dei pacchetti sulla base delle decisioni prese dal control plane. Si basa su un'infrastruttura Cloud per fornire, tramite virtualizzazione, funzioni di rete. Queste ultime possono essere implementate anche in hardware.

Confrontando tale framework con ETSI MANO, possiamo affermare che i componenti di orchestrazione hanno funzionalità simili, così come i componenti NFVM e VNFM che si occupano entrambi di gestire il ciclo di vita delle funzioni di rete virtuali. Il controller SDN è integrato nel VIM e per questo svolge anche i compiti di quest'ultimo (ad esempio, il monitoraggio dei nodi).

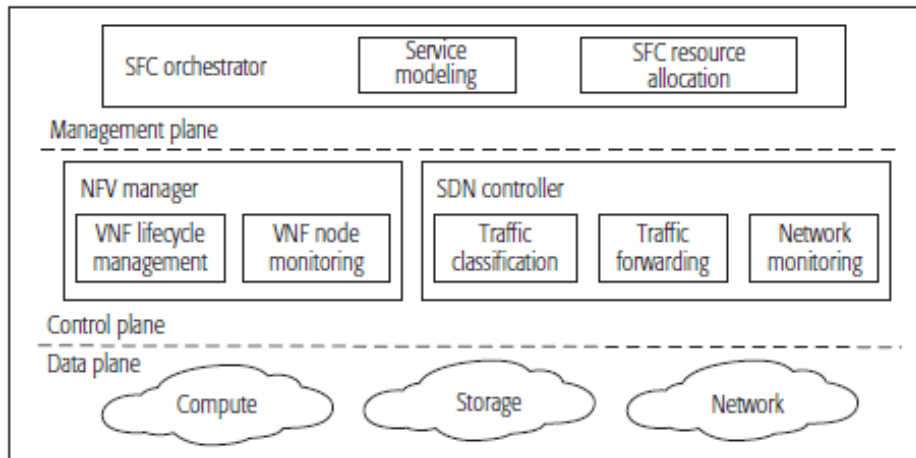


Figura 2.23: Framework funzionale SFC

Come detto in precedenza, SFC offre possibilità interessanti per diversi domini, tra cui il Network Slicing. Prendendo come riferimento l'architettura raffigurata in [2.21] è possibile definire uno scenario in cui abbiamo più tenant che eseguono diverse slice di rete. Ogni tenant ha accesso ad alcune macchine virtuali della NFVI. In questo caso, abbiamo 3 tipi di controller SDN:

1. quello a livello di tenant, che in maniera dinamica effettuerà il concatenamento tra le varie VNFs che compongono un servizio di rete gestito dal NFVO.
2. quello a livello di NFVI-PoP che gestisce le risorse all'interno del PoP stesso. Il controller a livello di tenant ha una visione astratta delle risorse che gli viene fornita da questo livello.
3. quello a livello di WAN che gestisce la connettività SDN in tale dominio.

Ogni tenant gestisce gli slice del proprio dominio attraverso il proprio NFVO che coordina le risorse in diversi PoP. In questa maniera è possibile creare slice di rete senza soluzione di continuità dal dominio del data center a quello WAN per il trasporto end to end nella rete 5G.

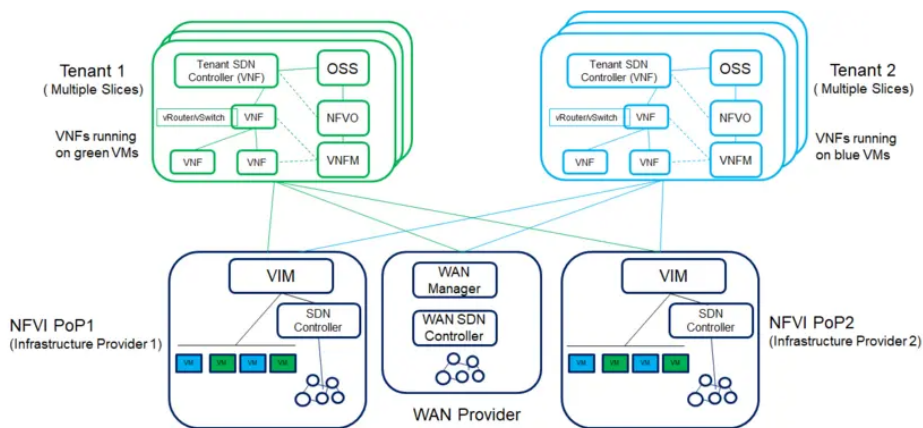


Figura 2.24: Network slicing tramite SFC

2.5.4 Multi-access Edge Computing

L'edge computing permette di hostare le applicazioni sulla parte edge della rete, il più vicino possibile agli utilizzatori e ai dati generati dalle applicazioni. L'edge computing è riconosciuto come uno dei pilastri chiave per soddisfare i KPI del 5G, soprattutto per quanto riguarda la bassa latenza e l'efficienza della larghezza di banda [24].

ETSI ISG MEC (Industry Specification Group for Multi-access Edge Computing) è la sede degli standard tecnici per l'edge computing. In [2.25] sono raffigurati gli elementi funzionali e i punti di riferimento tra essi dell'architettura di un sistema MEC [25].

- ricevere record DNS dal MEC platform manager e configurare DNS server/proxy in accordo con queste;
- ospitare servizi MEC.

Le **MEC application** vengono eseguite come macchine virtuali sull'infrastruttura di virtualizzazione fornita dall'host MEC e possono interagire con la piattaforma MEC per utilizzare e fornire servizi MEC. Possono avere un certo numero di regole e requisiti associati, come risorse richieste, latenza massima, servizi necessari o utili, eccetera. Questi requisiti sono convalidati dal MEC system level management e possono essere assegnati a valori predefiniti se mancanti.

Il **MEC system level management** è composto dal **Multi-access edge orchestrator**, dall'**Operations Support System, OSS** e dal **User application lifecycle management proxy**.

Il primo, è la parte principale del sistema ed è responsabile di:

- mantenere una visione del sistema MEC basata su MEC host, risorse disponibili, servizi MEC disponibili e topologia;
- on-boarding dei pacchetti delle applicazioni, controllo dell'integrità e dell'autenticità di questi, convalida delle regole e dei requisiti dell'applicazione e, se necessario, l'adeguamento degli stessi per conformarsi alle politiche dell'operatore. Inoltre, mantiene un registro dei pacchetti on-boarded e prepara i **virtualization infrastructure manager** alla gestione delle applicazioni;
- selezionare i MEC host appropriati su cui istanziare le applicazioni in base ai vincoli, come latenza, risorse e servizi disponibili;
- avviare l'istanziamento e la terminazione delle applicazioni;
- avviare il riposizionamento delle applicazioni quando supportato e se necessario.

Il secondo, fa riferimento all'OSS dell'operatore. Riceve di istanziamento o terminazione di applicazioni tramite il **CFS portal** o le **device applications** e decide se autorizzarle o meno. Le richieste autorizzate sono inoltrate al **Multi-access edge orchestrator** per essere ulteriormente processate. Se supportato, l'OSS riceve anche richieste dalle **device applications** per il trasferimento delle applicazioni tra Cloud esterni e il sistema MEC.

Il terzo consente alle **device applications** di richiedere l'on-boarding, l'istanziamento, la terminazione delle applicazioni utente e se supportato il riposizionamento di queste all'interno o all'esterno del sistema MEC. Permette

anche di informare le **device applications** in merito allo stato delle applicazioni utente. Un'applicazione utente è un'applicazione MEC istanziata su un sistema MEC a seguito di una richiesta effettuata da un utente tramite **device application**.

Il **MEC host level management** è composto dal **MEC platform manager** e dal **virtualization infrastructure manager**.

Il primo, è responsabile delle seguenti funzionalità:

- gestire il ciclo di vita delle applicazioni e informare il **multi-access edge orchestrator** in merito agli eventi applicativi rilevanti;
- fornire funzionalità di gestione alla **MEC platform**
- gestire le regole e i requisiti delle applicazioni comprese le autorizzazioni del servizio, le regole del traffico, la configurazione DNS e la risoluzione dei conflitti;
- ricevere dal **virtualization infrastructure manager** i report dei guasti delle risorse virtualizzate e della misurazione performance e conservarli per elaborazioni successive.

Il secondo, è responsabile di:

- allocare, gestire e rilasciare le risorse virtuali della **virtualized infrastructure**;
- preparare la **virtualized infrastructure** a eseguire un'immagine software
- collezionare le informazioni in merito alle performance e agli errori delle risorse virtualizzate
- quando supportato, effettuare il riposizionamento delle applicazioni.

Le **device applications** sono applicazioni che girano su un device utente e hanno la capacità di interagire con il sistema MEC tramite un **user application lifecycle management proxy**.

Il **customer facing service portal** permette ai clienti dell'operatore di selezionare e ordinare un insieme di applicazioni MEC che soddisfano le loro esigenze e ottenere informazioni in merito al livello di servizio fornito da queste.

MEC e NFV sono concetti complementari: l'architettura MEC è stata progettata in modo da consentire diverse modalità di esecuzione e distribuzione di un sistema MEC. Un gruppo dedicato, ETSI GR MEC 017, fornisce

un'analisi dettagliata sulla messa in esecuzione di un sistema MEC in un ambiente NFV. In [2.26] è mostrata l'architettura di un sistema MEC eseguito in ambiente NFV. Le differenze dall'architettura discussa precedentemente sono:

- la **MEC platform** e le **MEC applications** sono eseguite come VNF
- la **virtualization infrastructure** è eseguita come una NFVI e gestita da un VIM
- il **MEC platform manager**, **MEPM** è sostituito dal **MEC platform manager - NFV**, **MEPM-V** e delega la gestione del ciclo di vita delle VNF a uno o più **VNFM**.
- Il **MEC orchestrator**, **MEO** è sostituito dal **MEC application orchestrator**, **MEAO** che si affida al **NFVO** per l'orchestrazione delle risorse e degli insiemi di applicazioni MEC, implementate come VNF, che assieme formano un servizio di rete NFV.

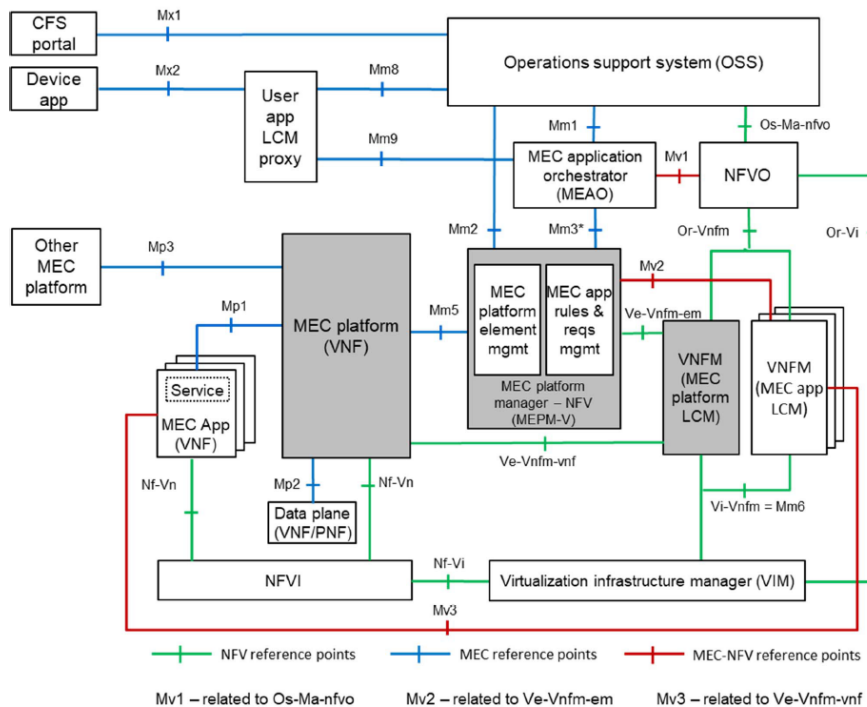


Figura 2.26: Architettura sistema Multi-access Edge Computing eseguito in ambiente NFV

2.5.5 Architettura sistema 5G

Nelle sottosezioni precedenti sono state introdotte le tecnologie alla base delle reti mobili di quinta generazione. Questa sottosezione tratterà l'architettura di tali sistemi e le modalità con cui questa sfrutta le tecnologie sopra descritte.

L'architettura di un sistema 5G, mostrata in [2.27], si compone di una rete di accesso, AN, e di una core network, 5GC. La rete di accesso può essere suddivisa in:

- New Generation Radio Access Network, NG-RAN, che usa l'interfaccia radio del sistema 5G;
- non-3GPP Access Network (ad esempio, WiFi, xDSL, eccetera).

Come le generazioni precedenti, la rete 5G collega gli UE a data network esterne (ad esempio, Internet). La connessione logica tra UE e data network prende il nome di **PDU session**. A livello di trasporto, una PDU session è composta da una sequenza di tunnel all'interno del 5GC e da uno o più radio bearers sull'interfaccia radio. Una PDU session è molto simile a un bearer EPS [2.4.1] e si differenzia per la modalità di gestione della QoS e per il fatto che può trasportare, oltre a pacchetti IP, frame Ethernet o non strutturati abilitando una comunicazione di livello 2 tra gruppi di UE. Nei sistemi 5G il modello per la gestione della QoS è basato sul concetto di QoS Flow. Sono supportati QoS Flow che richiedono un bit rate garantito, GBR QoS Flows, o meno, Non-GBR QoS Flows.

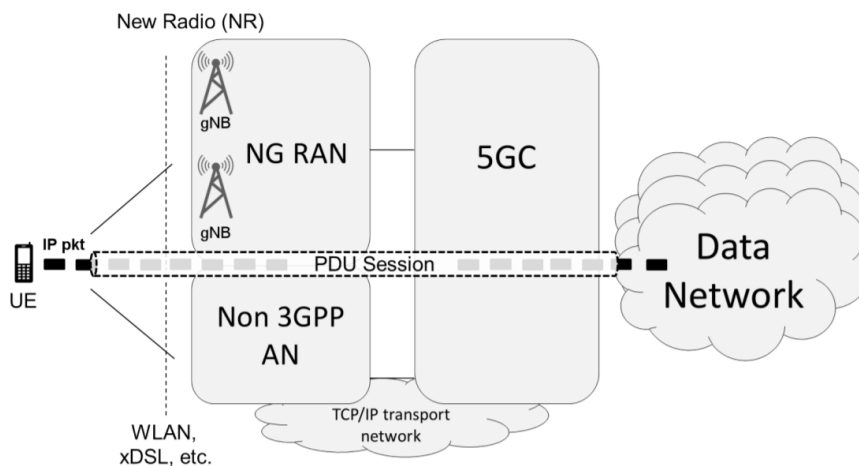


Figura 2.27: Architettura sistema 5G [16]

Il QoS Flow è l'unità minima che permette di differenziare la QoS all'interno di una PDU Session. Per identificare un flusso si utilizza un QoS Flow ID, QFI; il traffico del piano dati che riceve lo stesso QFI all'interno di una sessione è inoltrato nello stesso modo.

I QoS Flow sono controllati dalla SMF [2.5.5] e possono essere preconfigurati, stabiliti attraverso la procedura di PDU Session Establishment e modificati attraverso la procedura di PDU Session Modification.

Il core della rete 5G è stato progettato attorno al concetto di servizio: i blocchi funzionali del core sono implementati come servizi interrogabili tramite API standard. In questo modo, l'architettura può sfruttare le tecnologie NFV e SDN e i relativi vantaggi. Alcuni principi fondamentali di design dell'architettura sono [15]:

- separazione delle funzionalità del piano utente da quelle del piano di controllo in modo da consentire una scalabilità indipendente e una messa in esecuzione flessibile;
- modularizzazione del design delle funzioni di rete
- minimizzare le dipendenze tra la rete di accesso e la rete core
- supportare l'esposizione delle funzionalità della rete.

L'architettura del 5GC può essere rappresentata in due modi: il primo è basato sui reference point [2.28], il secondo sui servizi [2.29]. Nel primo caso vengono definiti i blocchi funzionali e le interfacce logiche tra essi. Questo comporta svantaggi legati alla scarsa flessibilità e scalabilità evidenziati nelle sezioni precedenti. Per questo motivo, tale architettura, è stata evoluta in una Service Oriented Architecture, SOA, in cui i blocchi funzionali comunicano tra loro mediante apposite API. Ulteriore novità è l'introduzione del NRF che è l'entità che svolge il ruolo di registro dei servizi permettendone la localizzazione.

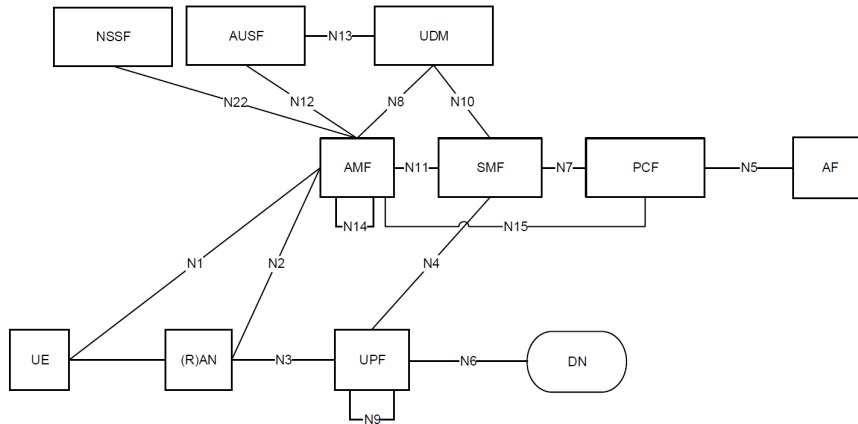


Figura 2.28: Architettura della rete core 5G basata sui punti di riferimento

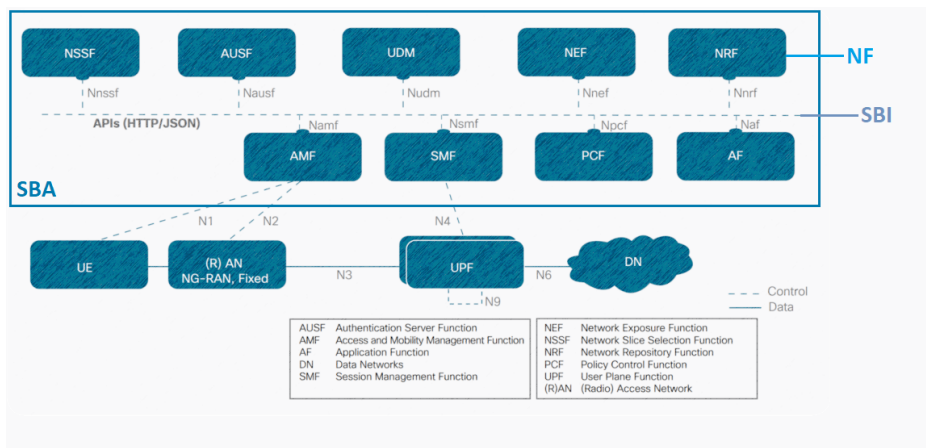


Figura 2.29: Service Based Architecture della rete core 5G

Di seguito verranno introdotte le funzionalità di ciascun blocco dell'architettura a servizi.

Network Functions SBA 5GC

Access and Mobility Function. L'AMF, svolge gran parte delle funzioni svolte dalla MME nell'EPC [2.4.1]. In particolare, è responsabile solo della gestione della connessione e della mobilità; tutti i messaggi relativi alla gestione delle sessioni vengono inoltrati alla Session Management Function, SMF.

Siccome una rete mobile può includere più istanze di AMF, si utilizza, al fine del riconoscimento, un **Globally Unique AMF Identifier**, GUAMI.

Nel dettaglio, i compiti svolti dall'AMF sono:

- **gestione della registrazione.** Un UE deve essere registrato per poter usufruire dei servizi offerti dalla rete e per la gestione della mobilità e della raggiungibilità. Una volta fatto ciò, passa dallo stato RM-DEREGISTERED a RM-REGISTERED e viene creato un contesto all'interno del core di rete contenente tutte le informazioni associate ad esso. La registrazione viene effettuata alla prima connessione, periodicamente affinché l'UE rimanga disponibile, a fronte di uno spostamento tra AMF e per aggiornare o rinegoziare le capacità associate all'UE.
- **Gestione della connessione.** Instaurazione e abbattimento della connessione per le segnalazioni di controllo con l'UE (interfaccia N1).
- **Gestione della raggiungibilità.** Si assicura che l'UE sia sempre raggiungibile, ad esempio eseguendo il paging. Nel caso in cui l'UE sia in stato di idle, CM-Idle, come nel EPC se un UE rimane inattivo per un determinato lasso di tempo la connessione dell'interfaccia aerea e le risorse allocate ad esso vengono rilasciate, la procedura avvia il processo di instaurazione della connessione con l'AMF facendolo passare in stato CM-Connected.
- **Gestione della mobilità.** Mantiene conoscenza sulla posizione dell'utente nella rete. Come detto in precedenza l'UE esegue delle registrazioni periodiche che funzionano come keep-alive che attestano la presenza dell'utente nel sistema (non è uscito dal raggio di copertura e il dispositivo che utilizza per collegarsi sta funzionando).
- Trasporta in maniera trasparente i messaggi di gestione della sessione tra UE e SMF.
- Alloca bearer ID per l'interconnessione con l'Evolved Packet System.
- Gestisce le segnalazioni NAS garantendone integrità e cifratura.

L'AMF che serve un UE si trova nel PLMN in cui esso è situato: nel Visited PLMN quando l'UE è in roaming, nell'Home PLMN quando non lo è.

Session Management Function. La SMF esegue le funzioni di controllo delle sessioni svolte, nell'architettura 4G, da MME, SGW-C e PGW-C. Nel dettaglio:

- alloca indirizzi IP agli UE

- guida le segnalazioni NAS dedicate alla gestione delle sessioni
- invia le regole legate alle QoS alla RAN tramite l'AMF
- controlla le PDU Session
- Seleziona e controlla l'UPF per il routing del traffico. La funzione di selezione UPF abilita il Mobile Edge Computing (MEC) selezionando un UPF vicino al bordo della rete.

In base alla configurazione, quando un UE è in roaming, una PDU Session può essere controllata dalla SMF della VPLMN o da quello della HPLMN.

Unified Data Repository. L'UDR, fornisce la possibilità di archiviare e recuperare informazioni in merito alle sottoscrizioni, per l'UDM, alle policy, per il PCF, e ai dati strutturati esposti, per il NEF.

Nel caso del roaming l'UDR nella Home PLMN e l'UDR nella Visited PLMN possono essere utilizzati per servire un UE. In particolare, i dati sulle sottoscrizioni sono contenuti nell'UDR all'interno della HPLMN, mentre i dati relativi alle policy e quelli esposti possono essere memorizzati in quello della VPLMN.

User Plane Function. Il sistema 5G si affida all'UPF, per la gestione del traffico appartenente al piano utente. Le funzionalità offerte dall'UPF possono essere programmate dalla SMF attraverso l'interfaccia logica N4. Gli UPF possono essere distribuiti geograficamente o centralizzati. Le specifiche 3GPP non pongono restrizioni al numero di UPF (UPF a cascata) utilizzati per servire una PDU session. Un'istanza UPF offre le seguenti funzionalità:

- instradamento del traffico sulla base delle regole definite dalla SMF
- applicazione della QoS
- monitoraggio utilizzo di risorse e report di questi alla SMF tramite l'interfaccia N4
- segnalazione di eventi alla SMF in base a diverse condizioni di attivazione
- buffering e inoltro dei dati per garantirne l'integrità durante l'handoff.

Policy Control Function. Il PCF fornisce un framework unico per la definizione di qualsiasi tipo di policy all'interno della rete e per fornire queste alle diverse funzioni di rete, NF, del piano di controllo a seconda delle

necessità [21]. Nel caso in cui la NF da servire sia la SMF, le policy che gli vengono fornite dal PCF possono corrispondere a:

- proprietà di addebito associate a una PDU Session
- le diverse regole di QoS e/o tariffazione e/o inoltro del traffico da applicare ai diversi flussi di servizio all'interno della PDU Session.

Per quanto riguarda l'AMF, le policy inviate dal PCF possono essere raggruppate in due categorie:

- policy inviate direttamente all'AMF che possono corrispondere a limitazioni sulla tracking area dell'UE o a priorità legate ai tipi di accesso che questo può utilizzare;
- policy inviate all'UE attraverso l'AMF. Le Access Network Discovery and Selection Policy, ANDSP, sono usate dall'UE per selezionare un punto di accesso non-3GPP; le User Equipment Route Selection Policy, URSP, sono utilizzate dall'UE per determinare come instradare il traffico in uscita. Il traffico può essere instradato verso una PDU Session stabilita, può essere inoltrato su un accesso non-3GPP al di fuori di una PDU Session o può attivare la creazione di una nuova PDU Session ad esso dedicata.

Per la definizione delle policy, il PCF deve recuperare e tenere in considerazione informazioni in merito a:

- abbonamento dell'utente o requisiti dell'applicazione specifica. Queste possono essere ottenute dall'UDR sia per singolo utente abbonato, sia per gruppi di utenti, sia per tutti gli utenti di un partner di roaming.
- Condizioni di rete (ad esempio, informazioni sul tipo di accesso che l'UE sta attualmente utilizzando).

Network Exposure Function. Il NEF rappresenta il mezzo mediante il quale domini esterni possono accedere alle funzionalità e ai dati del piano di controllo del 5GC ed esporre dati strutturati all'interno di questo. I client del NEF potrebbero essere AF esterne, eventualmente impiegate da operatori di terze parti o NF interne del piano di controllo. La piattaforma Multi-Access Edge Computing (MEC) è un esempio di client NEF. Le funzionalità del NEF possono essere categorizzate in questo modo:

- **esposizione sicura.** Fornisce la possibilità di autenticare e autorizzare le entità appartenenti ai domini esterni a cui sono esposti i dati e le funzionalità della rete.

- **Gestione dell'esposizione di dati strutturati.** Il NEF può memorizzare, in maniera strutturata, i dati che gli vengono forniti da altre NF all'interno dell'UDR. In seguito, i dati memorizzati possono essere forniti ad altre NF.
- **Mapping di concetti.** Consiste nella traduzione dei concetti usati nel dominio dell'AF a concetti usati nel dominio del 5GC.

Network Repository Function, L'NRF, è un registro di servizi e consente la localizzazione delle diverse NF che compongono il 5GC. Le funzionalità offerte includono: gestione dei profili e delle istanze NF e rilevamento dei servizi delle NF. La prima funzionalità richiede il mantenimento, da parte dell'NRF, di informazioni relative a ciascuna istanza di NF, come il suo indirizzo, i servizi supportati e gli attributi che definiscono il dominio in cui questi servizi possono essere utilizzati (identificatori dei network slice e PLMN ID). La seconda funzionalità implementa i metodi che consentono alle NF di interrogare l'NRF per poter localizzare una o più NF che forniscono uno o più servizi.

Ogni PLMN deve eseguire il proprio NRF, lo stesso discorso vale per tutte le porzioni separate dal resto della rete per via del network slicing.

Unified Data Management. L'UDM è una funzione di controllo che fornisce l'accesso ai supporti di memorizzazione dati. In particolare, consente di:

- gestire i dati degli abbonati e autorizzare gli accessi
- gestire e memorizzare l'identificatore degli utenti
- autenticare gli utenti.

La gestione degli abbonati richiede di accedere alle informazioni relative agli abbonamenti contenute nell'UDR, fornire i dati rilevanti alle NF che servono l'UE (AMF, SMF) e aggiornare, nell'UDR, la posizione dell'UE e l'identità dei nodi di servizio (ad esempio l'AMF). L'UDM è sempre situata nell'H-PLMN dell'UE.

Authentication Server Function. L'AUSF, è responsabile delle procedure di autenticazione e scambio chiavi per autenticare reciprocamente UE e rete. L'AUSF è strettamente legato all'UDM e si trova sempre nella HPLMN dell'abbonato.

Application Function. L'AF, non fa parte del core 5G ma può interagire con esso per vari scopi, ad esempio per: accedere ai dati e alle funzionalità tramite NEF, influenzare il modo in cui il traffico relativo alle applicazioni dovrebbe essere instradato o informare il core sulle esigenze di QoS delle applicazioni. Sebbene esterni, alcuni AF possono essere considerati dall'operatore affidabili e quindi autorizzati ad accedere alle funzioni principali del 5G direttamente, senza utilizzare il NEF.

Network Slice Selection Function. La NSSF seleziona i network slice da utilizzare per servire un UE. Inoltre, determina l'insieme di istanze AMF che possono essere utilizzate per l'UE. Ogni PLMN gestisce il proprio NSSF, che ha una panoramica di tutti i network slice.

Sebbene i nomi delle componenti siano diverse rispetto a quelli dell'EPC 4G, alcune NF della SBA ricoprono i ruoli di alcune componenti dell'EPC. In [2.30] è raffigurata l'evoluzione e il mapping tra le componenti principali dell'EPC e del 5GC.

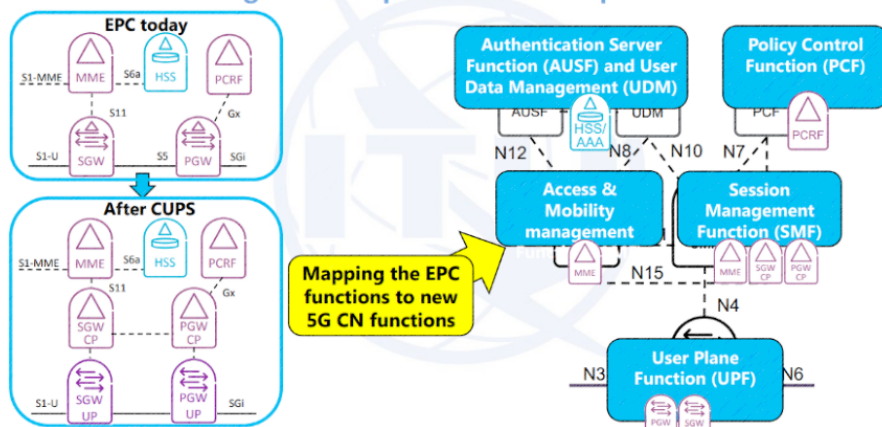


Figura 2.30: Evoluzione dell'EPC 4G nel 5GC

Architettura network slicing

Il network slicing richiede che l'architettura di rete sia in grado di isolare un insieme di istanze di funzioni di rete, NF, che formeranno lo slice e che sia in grado di selezionare gli slice opportuni.

Nei sistemi 5G il piano utente e il piano di controllo sono separati e il primo è confinato in una singola NF: l'UPF. Un UE può collegarsi simultaneamente a più UPF e per questo motivo è possibile assegnare ad ogni slice

un UPF dedicato. In questo modo, tutto il traffico del piano utente può essere confinato all'interno di uno slice.

L'AMF e la SMF sono separate, questo permette di confinare un'istanza SMF in ciascun slice, mentre l'AMF può essere comune a più slice. Naturalmente UE differenti possono essere assegnati ad AMF diversi.

I dati relativi agli abbonati gestiti dall'UDM sono separati in dati legati all'accesso, alla mobilità e alle sessioni. Questa suddivisione rispetta quella tra AMF e SMF e consente, se desiderato, di suddividere le funzionalità dell'UDM negli stessi slice di questi.

Il PCF può essere comune a tutti gli slice, specifico per alcuni slice o entrambi. Questo significa che le policy comuni a tutti gli slice saranno contenute in un'istanza condivisa, mentre le policy peculiari di uno slice saranno contenute in un'istanza dedicata.

Tutte le funzioni di rete fino a ora citate possono essere assegnate a uno slice senza esserne consapevoli. Questa è la proprietà architetturale chiave dello slicing: consente la multi-tenancy senza rendere le funzioni di rete consapevoli di ciò. Le uniche funzioni consapevoli dello slicing sono NEF, NRF e NSSF.

Il NEF può essere una NF specifica per un network slice, ma è sempre necessaria un'istanza comune, a conoscenza dei diversi slice, in grado di determinare dove risiedono i NEF specifici.

La NRF è in grado di selezionare per ciascun slice le istanze di NF corrette; in questo modo, la SBA è in grado di fornire servizi differenti o diverse versioni di un servizio in network slice differenti.

La NSSF può supportare l'AMF nella selezione della NSSAI e consente alla rete di controllare quali slice vengono utilizzati per le PDU Session dell'UE.

In [2.31] sono raffigurati tre slice differenti: i primi due, condividono un'istanza dell'AMF, mentre il terzo ne possiede una dedicata. Gli UE assegnati all'AMF in alto sono in grado di accedere esclusivamente ai servizi offerti dagli slice #1 e #2, mentre quelli assegnati all'AMF in basso accedono esclusivamente ai servizi offerti dallo slice #3. Ogni slice contiene un'istanza dedicata di SMF, UPF e PCF. I PCF specifici controllano le policy relative alle SMF, mentre il PCF comune controlla le policy relative agli AMF e agli UE.

La selezione degli slice da usare per le diverse UE e le sue PDU Session deve essere effettuata dall'operatore in maniera consistente con l'abbonamento dell'utente. Il 5GC fornisce alle applicazioni dell'UE, che accedono alla rete, delle regole che contengono il/i Data Network Names, DNN¹¹ e le

¹¹Nei sistemi 5G il DNN è l'equivalente dell'APN nei sistemi 4G

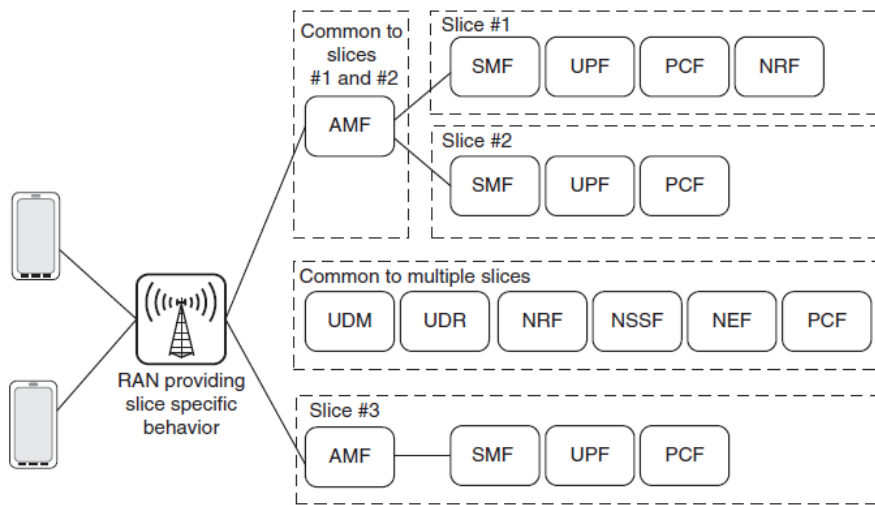


Figura 2.31: Network slicing in un sistema 5G [21]

informazioni sugli slice. La selezione dell'AMF consono per servire un UE dovrebbe considerare gli slice che l'UE potrebbe dover usare. Siccome la selezione dell'AMF iniziale viene eseguita dall'AN, che non è a conoscenza dei dati di sottoscrizione, 3GPP ha deciso di utilizzare un paradigma di selezione degli slice assistito dall'UE. Alla prima registrazione, la rete fornisce all'UE la NSSAI; dopodiché l'UE fornisce, ad ogni accesso, questa informazione all'AN abilitando questa alla selezione dell'AMF. La NSSAI è un insieme di S-NSSAI ognuna delle quali si compone di:

- **Slice Service Type**, SST, che indica le caratteristiche dello slice
- **Slice Differentiator**, SD, ulteriore parametro di differenziazione tra slice diversi che possiedono lo stesso SST. Lo Slice Differentiator può essere usato per differenziare slice di diversi tenant.

In [2.32] è raffigurata la procedura, semplificata, di registrazione di un UE e selezione degli slice:

1. l'UE fornisce la NSSAI come parte della richiesta della registrazione. Questo parametro è stato fornito all'UE al primo accesso alla PLMN (è un parametro configurato in quest'ultima).
2. La RAN seleziona un AMF in base alla NSSAI contenuta nella richiesta e inoltra questa all'AMF selezionato.

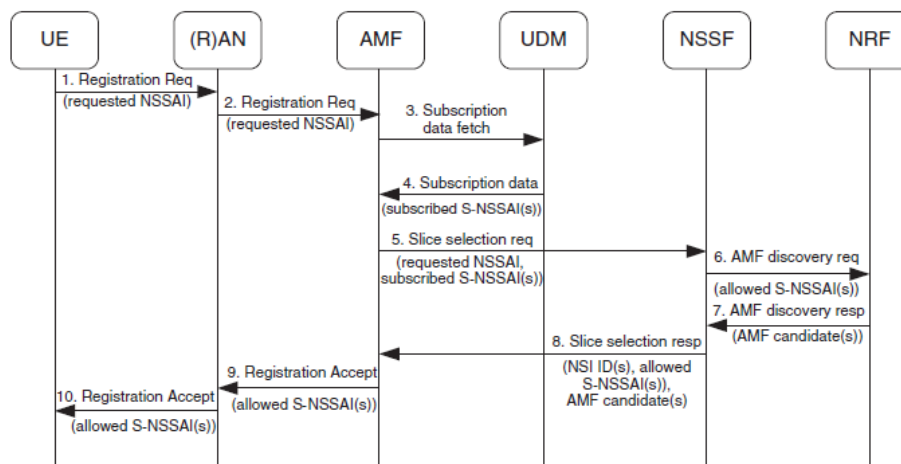


Figura 2.32: Procedura semplificata di selezione di network slice [21]

3. L'AMF ottiene i dati legati all'abbonamento dell'utente dall'UDM e fornisce a quest'ultimo anche la NSSAI richiesta.
4. L'UDM restituisce i dati dell'abbonamento e le S-NSSAI associate a questo e appartenenti alla NSSAI fornito.
5. L'AMF contatta l'NSSF per ottenere una o più Network Slice Instance, NSI, conformi alle S-NSSAI forniti al passo precedente. Oltre a ciò, viene richiesto l'insieme degli AMF in grado di servire tali istanze.
6. La NSSF richiede alla NRF le AMF adatte fornendole le S-NSSAI.
7. NRF invia gli AMF adatti all'NSSF.
8. La NSSF fornisce all'AMF l'id della NSI, le S-NSSAI permesse e le istanze AMF candidate.
9. L'AMF invia l'accettazione della registrazione alla RAN, includendo le S-NSSAI consentite. Nel caso in cui sia necessario selezionare un'AMF diversa, l'AMF che elabora la richiesta di registrazione la reindirizza a un'AMF che appartiene al set delle candidate.
10. La RAN invia l'accettazione della registrazione all'UE. Oltre a ciò, viene fornito all'UE l'elenco delle S-NSSAI consentite che userà nelle procedure successive come NSSAI.

Protocolli

Protocolli del piano di controllo tra 5G-AN e 5G Core L'interfaccia N2 o NG-C tra l'AN e il 5GC (nel dettaglio, l'AMF), è stata definita seguendo un principio fondamentale: tipi differenti di reti di accesso si collegano al 5GC tramite un unico protocollo del piano di controllo, l'**NG Application Protocol**, NGAP. I servizi forniti dal NGAP si dividono in due macro categorie: non associati agli UE e associati agli UE.

I primi, sono servizi legati alla gestione dell'interfaccia N2 come la configurazione o il reset. Tramite questi servizi l'AN e l'AMF sono in grado di scambiarsi informazioni di configurazione in merito a:

- Tracking Areas, PLMN ID e slices (S-NSSAI) supportati dall'AN
- identificatore dell'AMF e slices (S-NSSAI) che esso supporta
- carico di lavoro dell'AMF.

I secondi, sono associati ad un singolo UE e includono:

- procedure relative al trasporto NAS tra l'UE e l'AMF
- procedure legate alla gestione del contesto dell'UE; ad esempio, fornitura all'AN del materiale di sicurezza per rendere sicure le operazioni con l'UE
- procedure relative alle risorse delle sessioni PDU
- procedure relative alla gestione dell'handoff.

Lo stack del protocollo NGAP è raffigurato in [2.33].

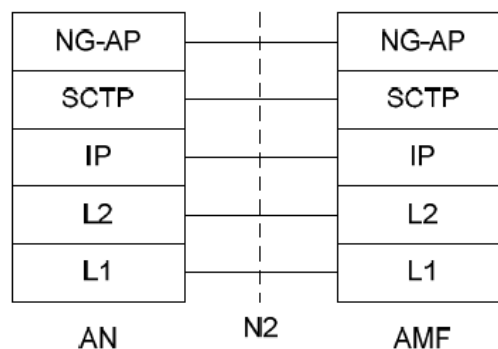


Figura 2.33: Stack protocollo NGAP

Le informazioni legate alla gestione delle sessioni sono scambiate tra SMF e AN avvalendosi dell'AMF come relay trasparente [2.34].

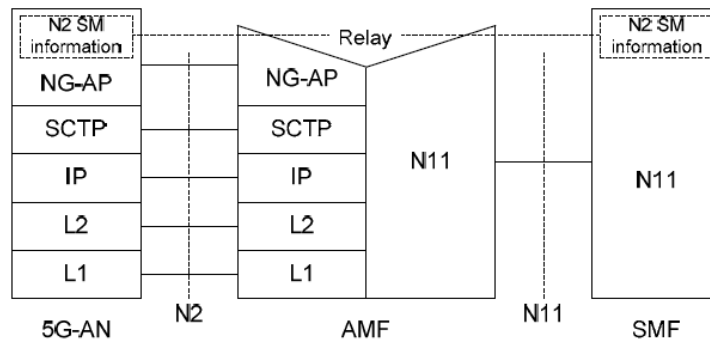


Figura 2.34: Relay AMF per informazioni di sessione tra AN e SMF

Protocolli del piano di controllo tra UE e 5G Core Una singola connessione di segnalazione NAS N1 viene utilizzata per ciascun accesso a cui è collegato l'UE (sia 3GPP che non-3GPP) [2.35]. Il punto di terminazione di tale connessione è l'AMF e gli scopi per cui viene usata sono molteplici: gestione delle registrazioni, gestione delle connessioni, trasporto dei messaggi di sessione, consegna di SMS (gestita dal componente SMSF tramite approccio SMS over NAS). In particolare, il protocollo NAS sull'interfaccia logica N1 si divide in due componenti: NAS-Mobility Management, NAS-MM, e NAS-Session Management, NAS-SM.

La prima si occupa di:

- procedure di gestione della registrazione e della connessione tra UE e AMF;
- trasmissione di altri tipi di informazioni NAS (ad esempio, NAS SM) per conto di altre NF;
- fornire una connessione NAS sicura (protezione dell'integrità, cifratura) tra l'UE e il 5GC, includendo la protezione di payload legati ad altre NF (ad esempio SMS). Al fine di garantire la sicurezza dei messaggi NAS si utilizza il contesto di sicurezza stabilito tra AMF e UE dopo l'autenticazione di quest'ultimo. A tal fine, l'AMF riceve una Master Key dall'AUSF dalla quale deriva il necessario per proteggere i messaggi NAS.

La seconda supporta l'instaurazione delle PDU Session per il traffico del piano utente, la modifica e il rilascio di queste. I messaggi SM sono trasferiti dal NAS-MM in questa maniera:

- in trasmissione, il livello NAS-MM crea un messaggio NAS-MM includendo il payload del messaggio SM e informazioni per l'inoltro della

richiesta. Infine, nel NAS security header viene aggiunta un'indicazione sul fatto che il messaggio trasporta una segnalazione SM.

- In ricezione, il livello NAS-MM elabora la parte MM del messaggio eseguendo il controllo dell'integrità tramite il security header. Dopodiché, se l'header contiene l'indicazione in merito al trasporto di una segnalazione SM, questa viene inoltrata all'entità opportuna utilizzando le informazioni per l'inoltro.

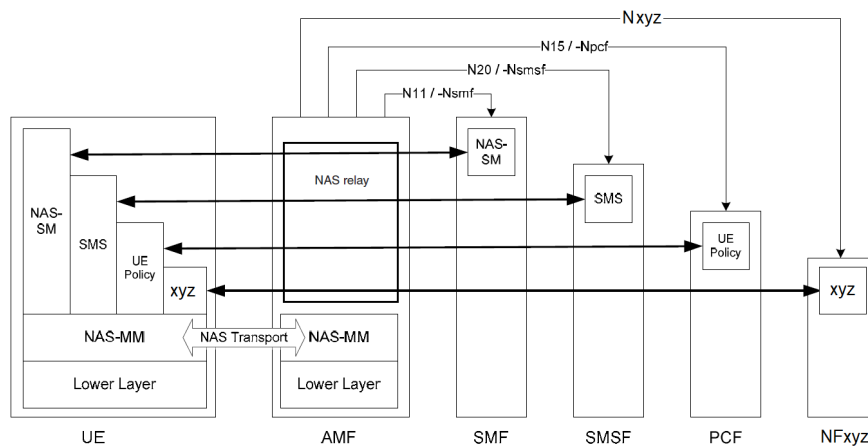


Figura 2.35: Trasporto NAS per SM, SMS e altri servizi

Protocolli del piano di controllo tra NFs e 5G Core Lo stack dei protocolli per le Service Based Infrastructure, SBI, è raffigurato in [2.36] e prevede l'uso di HTTP/2 più JSON come protocollo di serializzazione del livello applicativo, TCP come protocollo a livello di trasporto e IP per il livello di rete. Per la sicurezza a livello di trasporto, tutte le NF 3GPP devono supportare TLS. Le NF, per l'interazione, espongono API REST (RESTful quando possibile) definite tramite l'IDL¹² OpenAPI 3.0.0. Se una NF non registra alcun numero di porta sull'NRF, deve essere pronta a ricevere connessioni sui numeri di porta predefiniti (ad esempio, la porta TCP 80 per *http* e la TCP 443 per *https*).

Protocolli del piano di controllo per l'interfaccia N4 tra SMF e UPF Tutte le istanze UPF appartenenti ad una PDU Session sono controllate dalla SMF attraverso l'interfaccia N4. Quest'ultima non appartiene alle SBI,

¹²Interface Definition Language

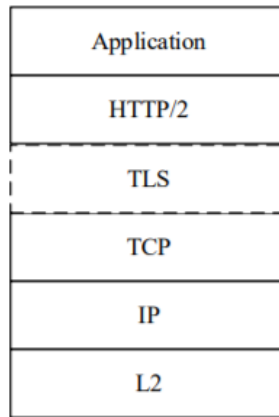


Figura 2.36: Stack dei protocolli SBI [3]

motivo per cui l'UPF non deve offrire i propri servizi attraverso API REST. Come conseguenza della programmabilità delle istanze UPF da parte della SMF, le funzionalità e le policy implementate in ognuna di esse possono essere diverse e dipendono dal caso d'uso specifico e dalla topologia della PDU Session. La SMF è in grado di controllare l'UPF tramite sessioni PFCP. Questo protocollo è utilizzato anche nelle interfacce Sxa (tra SGW-C e SGW-U) e Sxb (tra PGW-C e PGW-U) per l'implementazione del CUPS nell'EPC [2.37].

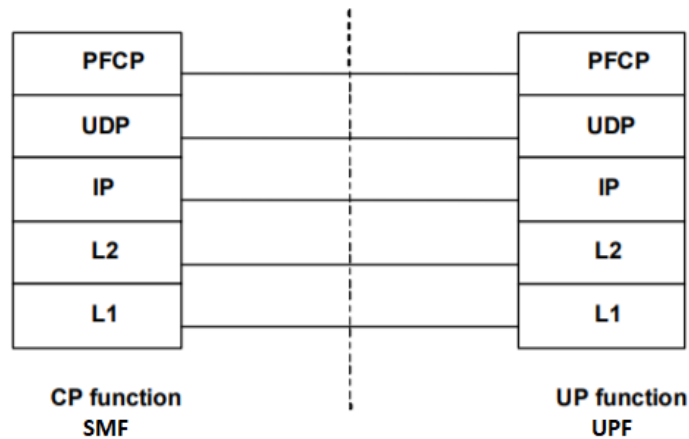


Figura 2.37: Stack dei protocolli del piano di controllo per l'interfaccia N4 [9]

Protocolli del piano utente In [2.38] è raffigurato lo stack dei protocolli per il trasporto dei dati del piano utente relativo ad una PDU Session. Lo strato PDU corrisponde alle PDU scambiate tra l'UE e la Data Network attraverso la sessione: nel caso di PDU Session IPv6 saranno pacchetti IPv6, nel caso di PDU Session Ethernet saranno frame Ethernet e così via. Lo strato GTP-U fornisce supporto al multiplexing del traffico di diverse PDU Session incapsulando le diverse PDU in maniera simile a quanto descritto in [2.4.1].

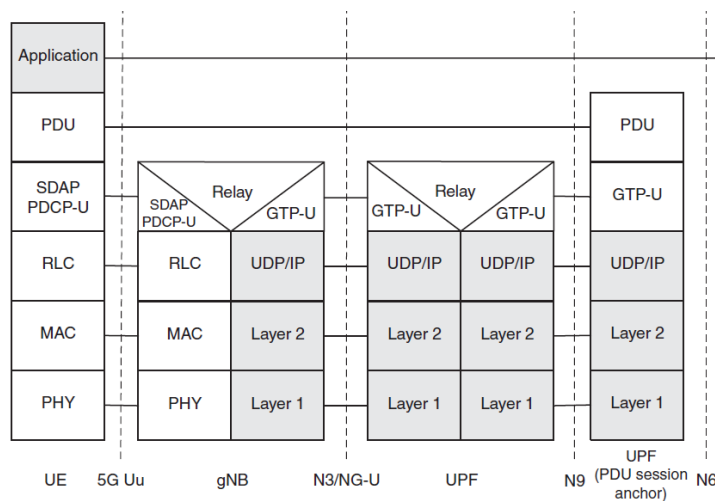


Figura 2.38: Stack dei protocolli del piano utente [21]

Interoperabilità con 4G Per supportare un servizio senza soluzione di continuità tra 5GS ed EPS, è necessaria un'architettura per l'interoperabilità [2.39]:

- un UPF comune che serve come ancora per l'IP dell'UE in modo che questo possa essere preservato;
- una funzione che svolge il ruolo combinato di SMF e PGW-C. In questo caso, SMF e UPF, sono considerati rispettivamente PGW-C e PGW-U dal punto di vista di un S-GW;
- un database degli abbonamenti comune che svolge il ruolo di HSS e UDM/UDR;
- un'interfaccia di segnalazione, N26, per il trasferimento del contesto associato all'UE tra MME e AMF.

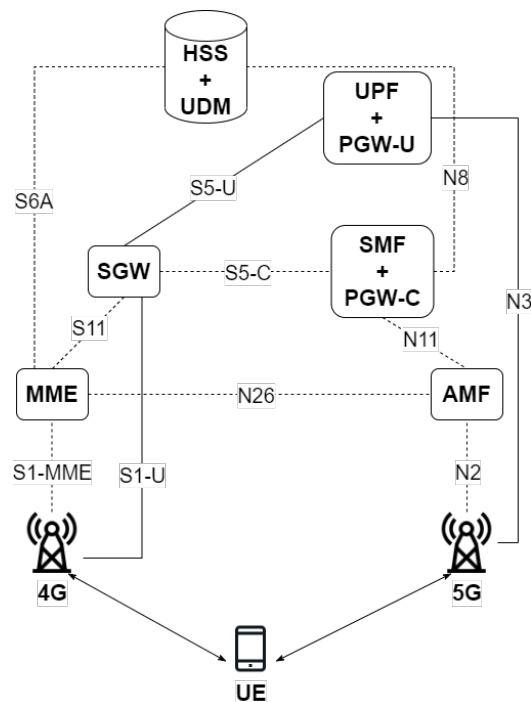


Figura 2.39: Architettura per l'interoperabilità 5GS ed EPS

In aggiunta, è possibile avere due tipi di UE:

- *single registration mode UE* [2.40]. L'UE ha un solo stato NAS attivo che può essere associato al 5GC o all'EPC e mantiene un singolo stato di registrazione. Per poter riutilizzare il contesto di sicurezza 5G precedente, quando si riconnette al 5GC, l'UE mantiene il 5G-GUTI¹³ e il contesto di sicurezza nativo al passaggio da 5GC a EPS.
- *Dual registration mode UE* [2.41]. In questa modalità, l'UE mantiene due stati NAS indipendentemente. Può essere registrato solo al 5GC, solo all'EPC o al 5GC ed EPC contemporaneamente.

L'interfaccia N26 può essere presente o meno. Nel primo caso, le procedure di interoperabilità si possono avvalere dell'interfaccia per scambiare lo stato di Mobility Management e Session Management e il contesto di sicurezza tra rete sorgente e rete di destinazione. Il supporto di tale interfaccia permette di fornire un servizio senza soluzione di continuità.

¹³Globally Unique Temporary ID identifica il dispositivo mobile all'interno della rete LTE. Si compone di Globally Unique Mobility Management Entity Identifier (GUMMEI), che identifica la rete, e M-TMSI, che identifica il dispositivo.

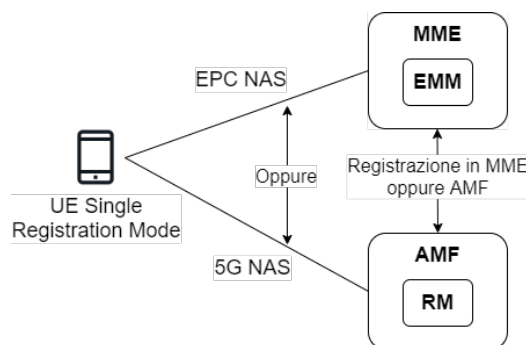


Figura 2.40: Single registration mode UE

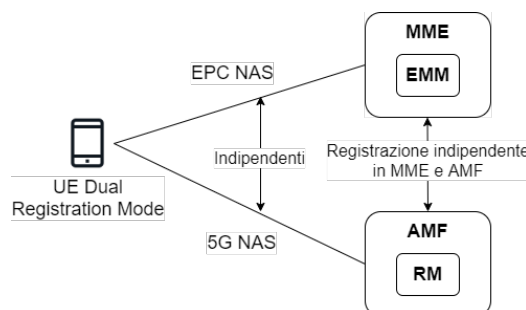


Figura 2.41: Dual registration mode UE

Quando l'UE supporta la modalità di registrazione singola e la rete supporta l'interoperabilità con l'interfaccia N26, si applicano i seguenti principi:

- in caso di mobilità in modalità idle da 5GC a EPC, l'UE esegue la procedura di **Tracking Area Update**, TAU, con il 4G-GUTI mappato a partire dal 5G-GUTI. Se l'UE ha stabilito una sessione PDU oppure se l'UE o l'EPC supportano la modalità di "collegamento senza connettività PDN"¹⁴, la MME recupera il contesto MM e SM dell'UE dal 5GC. In caso contrario, l'UE esegue una procedura di collegamento alla rete di destinazione. Nel caso di mobilità in modalità connessa dal 5GC all'EPC, viene eseguito l'handoff tra sistemi.
- In caso di mobilità in modalità idle da EPC a 5GC, l'UE esegue la procedura di registrazione tramite il 5G-GUTI nativo e il 5G-GUTI mappato a partire dal 4G-GUTI. Se, tramite il GUTI nativo, l'AMF è in grado di recuperare il contesto di sicurezza, allora utilizza quest'ultimo

¹⁴Funzionalità introdotta a partire dalla release 13 per consentire agli UE che supportano le ottimizzazioni Cellular Internet of Things di rimanere collegate senza connettività PDN

anziché quello recuperato dall'EPC. Il contesto MM e SM è recuperato dall'EPC e di ciò se ne occupano, rispettivamente, l'AMF e la SMF. Anche in questo caso, lo scenario di mobilità in modalità connessa prevede l'handoff tra sistemi.

Qualora l'interfaccia N26 non fosse presente, la continuità dell'indirizzo IP a fronte della mobilità tra sistemi è fornita all'UE memorizzando e recuperando l'indirizzo del componente SMF+PGW-C e le informazioni APN/DDN tramite l'UDM+HSS.

Quando l'UE supporta la modalità di registrazione singola e la rete supporta l'interoperabilità senza l'interfaccia N26, si applicano i seguenti principi:

- in caso di mobilità tra 5GC ed EPC, l'UE può eseguire la procedura di TAU tramite il 4G-GUTI mappato a partire dal 5G-GUTI. La MME determina che il nodo sorgente è un AMF e rifiuta il TAU con l'indicazione *"Handover PDN Connection Setup Support"*. Sulla base di questa indicazione, l'UE può eseguire la connessione all'EPC con l'indicazione di *"handover"* nel messaggio di richiesta. Successivamente, sposta tutte le sue PDU Session dal 5GC all'EPC utilizzando la procedura di creazione della connessione PDN con il flag *"handover"* impostato.
- In caso di mobilità tra EPC e 5GC, l'UE esegue la procedura di registrazione, al 5GC, di tipo *"aggiornamento registrazione a fronte di mobilità"* con il 5G-GUTI mappato a partire dal 4G-GUTI. L'AMF determina che il nodo di origine è un MME, ma procede come se la registrazione fosse di tipo *"registrazione iniziale"* e invia un messaggio di accettazione all'UE che include l'indicazione *"Handover PDU Session Setup support"*. Sulla base di questa indicazione, l'UE può successivamente spostare tutte le sue connessioni PDN dall'EPC al 5GC utilizzando la procedura di creazione della PDU Session con l'indicazione *"Existing PDU Sessions"*.

Roaming In un contesto di roaming, l'accesso dell'UE è servito dalla **Visited Public Land Mobile Network**, VPLMN, che è diversa dalla sua **Home Public Land Mobile Network**, HPLMN. Per una PDU Session, l'accesso alla DN può avvenire nella HPLMN e si parla di modalità **home routed** [2.42], HR, o nella VPLMN e si parla di modalità **local breakout** [2.43], LBO.

Nel primo caso, l'HPLMN mantiene il controllo sulla connettività N6 della PDU Session e le relative policy; la VPLMN non deve essere al corrente dei servizi offerti dalla PDU Session, ma si limita ad assicurare il giusto grado

di QoS, supportare gli handoff, eccetera. Nel secondo caso invece la VPLMN ha il controllo totale sulla PDU Session.

Diverse PDU Session appartenenti alla stessa UE possono essere stabilite in modalità differenti (LBO o HR). I dati di sottoscrizione indicano alla VPLMN sulla base di DNN e S-NSSAI in quale modalità può essere stabilita la PDU Session corrispondente. L'HPLMN può consentire l'utilizzo di LBO per alcuni VPLMN (ad esempio, per le VPLMN appartenenti allo stesso gruppo dell'operatore che gestisce la HPLMN), mentre per altri consentire solo l'HR.

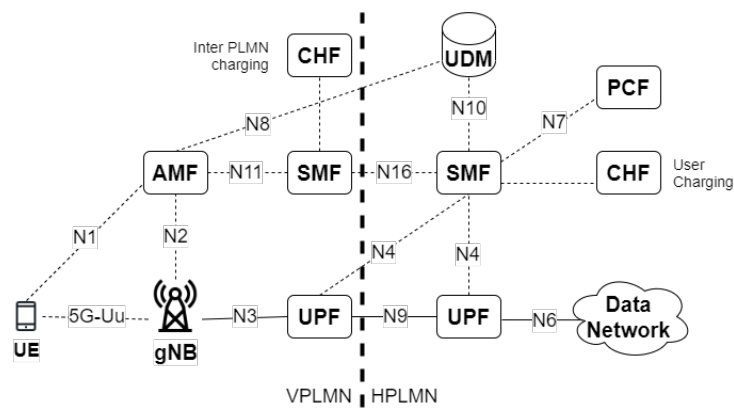


Figura 2.42: Modalità roaming home routed

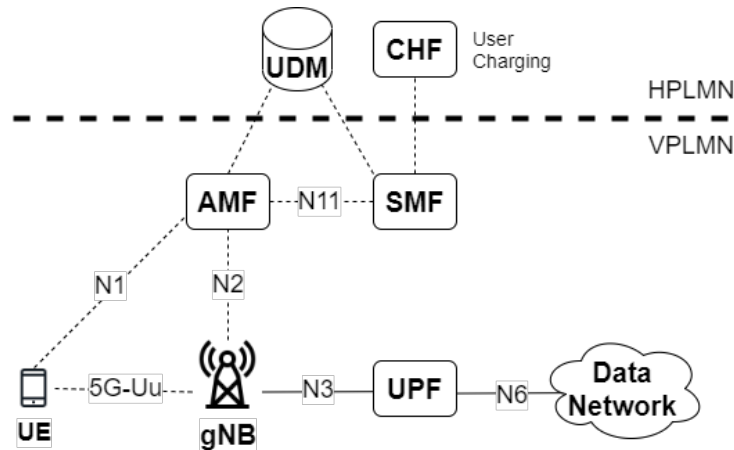


Figura 2.43: Modalità roaming local breakout

Capitolo 3

Strumenti software

Nei capitoli precedenti sono stati introdotti gli aspetti teorici rilevanti per comprendere i capitoli seguenti che descriveranno la messa in esecuzione della rete core 4G e 5G, rispettivamente EPC e 5GC, virtualizzata. In particolare, questo capitolo introdurrà gli strumenti software necessari per raggiungere tale obiettivo.

3.1 Open Source MANO

Open Source Mano, OSM, è un'iniziativa coordinata da ETSI per sviluppare uno stack software, open-source, MANO (Management and Orchestration) allineato con ETSI NFV.

L'obiettivo principale di OSM è fornire la possibilità di creare reti su richiesta ("Network as a Service" o NaaS) che poi verranno sfruttate da terzi. In tal senso, OSM funziona come un orchestratore di servizi e fornisce la possibilità di creare questi ultimi, controllarne l'intero ciclo di vita ed apportare modifiche tramite API e monitorarne lo stato globale.

In [3.1] viene contestualizzato OSM all'interno dell'architettura ESTI NFV [1]. È importante notare che, anche se nella documentazione standard ETSI il framework NFV MANO include anche il Virtualized Infrastructure Manager, nelle implementazioni reali quest'ultimo è solitamente accoppiato con la NFVI. Ciò è legato al fatto che al giorno d'oggi esiste una grande varietà di sistemi VIM, ampiamente distribuiti e utilizzati. È il caso di OpenStack, un sistema di infrastruttura Cloud open-source, che verrà descritto in seguito, ma anche di diverse soluzioni proprietarie come Amazon Web Services (AWS), VMware, vCD eccetera.

I punti di riferimento *Or-Vi* e *Vi-Vnfm* tra VIM e, rispettivamente, NFVO e VNFM sono genericamente chiamati Northbound interface, NBI. A

seconda delle implementazioni, il VIM può controllare direttamente l'infrastruttura o interagire con altri software di gestione dell'infrastruttura. L'interfaccia tra il VIM o gli elementi software specializzati e l'infrastruttura sottostante viene solitamente denominata Southbound Interface, SBI.

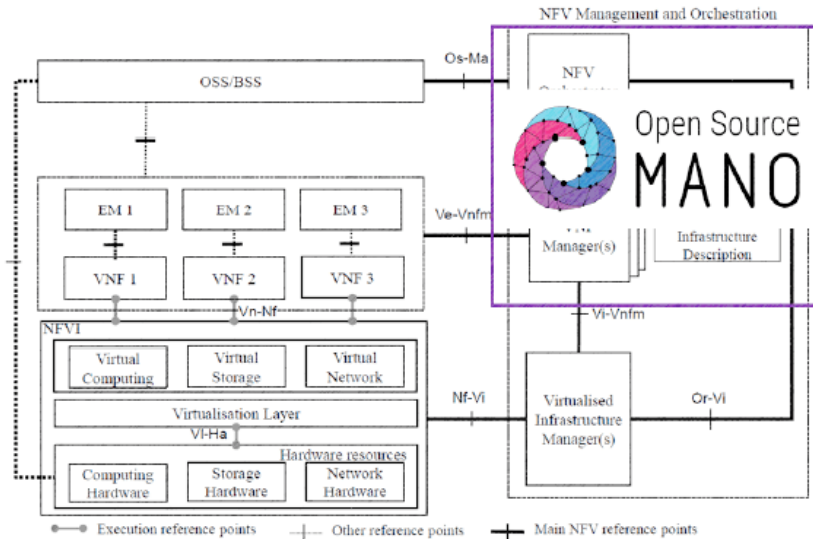


Figura 3.1: OSM contestualizzato nell'architettura ETSI NFV

OSM è dotato di un Information Model, IM, preciso, allineato con ETSI NFV, che permette di modellare e automatizzare l'intero ciclo di vita di funzioni di rete virtuali, fisiche o ibride, servizi di rete, NS e segmenti di rete, NSI, a partire dalla loro distribuzione (day 0), configurazione (day 1) e monitoraggio quotidiano (day 2). Come accennato in precedenza, l'IM è completamente indipendente dall'infrastruttura; in questo modo lo stesso modello può essere utilizzato per istanziare un dato elemento (ad esempio VNF) in un'ampia varietà di tipi di VIM. Inoltre, la NBI fornisce tutte le astrazioni necessarie che garantiscono il controllo completo del funzionamento dei NS/NSI evitando di esporre dettagli non necessari in merito a come questi sono costituiti. In [3.2] è schematizzato l'utilizzo delle NBI per istanziare, configurare e gestire NF, NS e/o NSI. In particolare, tramite l'IM viene descritta la topologia di una NF, di un NS e/o di un NSI e le procedure di configurazione e gestione del ciclo di vita. Tramite la NBI del sistema MANO, le informazioni in merito alla topologia giungono ad un componente, VIM/SDN Connector, che si occupa di istanziare le VNF sulla NFVI tramite le NBI del VIM. Le informazioni in merito alla configurazione e alla gestione giungono ai charms che si occupano di ciò. I charms sono collezioni di script

e metadati che automatizzano e rendono ripetibile la configurazione e la gestione delle VNF. Nel dettaglio, in OSM il componente che si occupa di ciò è Juju [3.1.3].

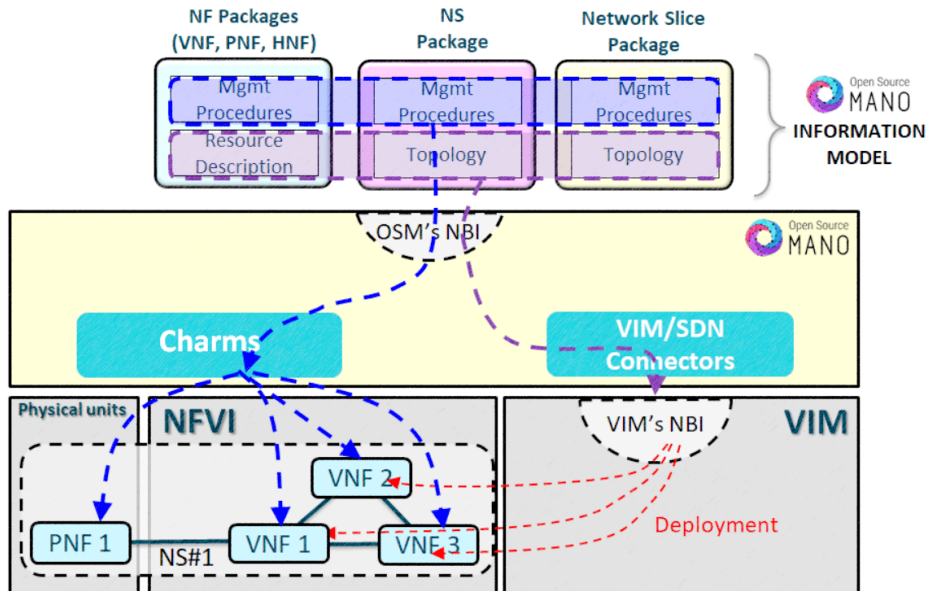


Figura 3.2: Utilizzo NBI per istanziare, configurare e gestire NF, NS e/o NSI [48]

3.1.1 Architettura OSM

Sin dal principio, OSM è stato progettato per garantire modularità e adottando i principi di progettazione di applicazioni Cloud. A partire dalla release 4 è stato introdotto un bus di messaggi per favorire la comunicazione asincrona tra i componenti. Questo rende OSM più aperto e di conseguenza aumenta la facilità di integrazione di nuovi moduli. In [3.3] è mostrata, ad alto livello, l'architettura sopra citata che si compone di:

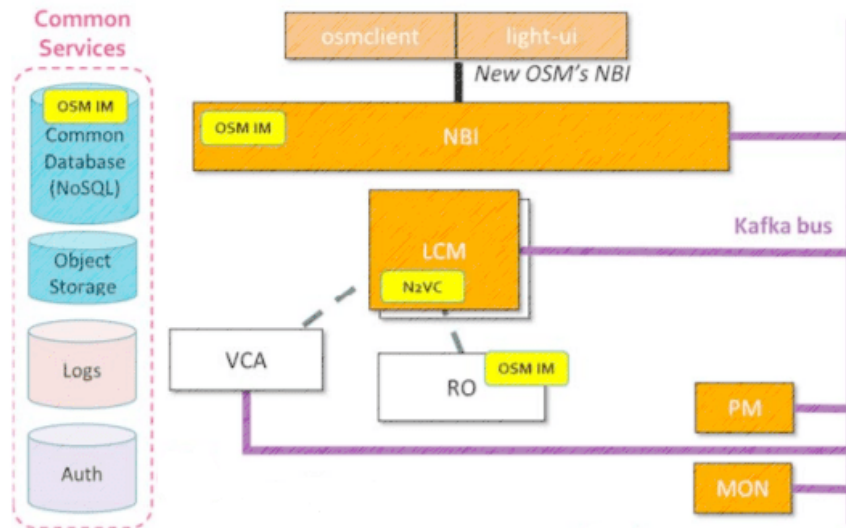


Figura 3.3: OSM architettura di alto livello [34]

- Kafka bus, per la comunicazione asincrona a scambio di messaggi.
- l'NBI, introdotta in precedenza, che consente agli utenti di interagire con la piattaforma tramite CLI (osm client) e tramite GUI (light-ui). Inoltre, espone i servizi offerti da OSM a sistemi OSS/BSS esterni.
- Il Life Cycle Manager, LCM, che è il componente responsabile delle funzionalità di orchestrazione dei servizi di rete e della gestione del loro ciclo di vita. L'LCM interagisce con il modulo RO per la messa in esecuzione del NS, con il modulo VCA per la configurazione delle VNF e con il modulo MON per l'esportazione delle metriche e la configurazione degli allarmi.
- Il Resource Orchestrator, RO, che si occupa dell'orchestrazione della NFVI e dell'allocazione delle risorse. Per svolgere i propri compiti interagisce con uno o più VIM e/o controller SDN.
- Il VNF Configuration and Abstraction, VCA, agisce da intermediario tra il VNFM, implementato da OSM e i diversi tipi di VNF; offre gli strumenti necessari per configurare le diverse VNF.
- Il modulo di monitoraggio, MON, che supporta la gestione dei guasti e delle performance, il monitoraggio degli allarmi e la collezione delle

metriche sia dell'infrastruttura sia delle VNF. Le metriche collezionate vengono poi esposte e consumate dal Prometheus Time Series Data Base, TSDB.

- Il modulo di Policy management, POL, che insieme al modulo MON permette di gestire e attivare allarmi sulla base di eventi e metriche dell'infrastruttura e delle VNF. Possono essere definiti allarmi per la scalabilità automatica delle VNF sulla base del raggiungimento di certe soglie da parte delle metriche.
- Network Service to VNF Communication, N2VC, che rappresenta il Juju controller incaricato di gestire la configurazione dinamica delle VNF e degli indicatori.
- DB e supporti di memorizzazione condivisi.

3.1.2 Ciclo di vita dei Network Service

Il ciclo di vita di un Network Service e tutte le operazioni coinvolte in esso possono essere raggruppate nelle seguenti fasi:

0. modellazione
1. on-boarding
2. creazione NS (day 0 e day 1)
3. operazioni su NS (day 2)
4. terminazione NS.

0. Modellazione

L'IM di OSM fornisce i meccanismi per la progettazione del comportamento di un NS a partire dalla descrizione della topologia fino a quella delle operazioni del suo ciclo di vita; oltre a ciò, permette di includere il codice relativo alle automazioni. La composizione di più NS costituisce un Network slice; per supportare la modellazione e la distribuzione di quest'ultimo, l'IM fornisce un modello specifico denominato Network Slice Template, NST. Un NST è generalmente una composizione di diversi NSD e VLD [1.4] che interconnettono i NS in base a una topologia specifica. Slice differenti possono condividere gli stessi NS. I NS sono a loro volta composti da Network Functions (VNF, PNF o HNF) aventi comportamenti differenti. Anche in questo

caso l'IM fornisce un mezzo per poter descrivere la topologia interna, le risorse richieste e le operazioni del ciclo di vita delle VNF. In particolare, la progettazione di servizi e funzioni di rete avviene, rispettivamente, tramite l'uso di NS Packages e NF Packages. Un package non è altro che una cartella compressa, *.tar.gz* che a seconda del tipo, NS o NF, contiene diversi file e sottocartelle. Nel dettaglio, un NF Package contiene:

- una cartella *charms*. Al suo interno troviamo tutto il necessario per la configurazione dinamica e il monitoring della NF (day 1 e day 2);
- una cartella *cloud-init*. Contiene i file necessari per l'istanziamento iniziale delle immagini Cloud su cui eseguirà la NF;
- cartelle *icons*, *images* e *scripts*;
- file *README* per documentare il pacchetto;
- file *descriptor*, *VNFD*.

Il VNFD è scritto in YAML¹ e contiene tutti i dati necessari per modellare la NF. Nel dettaglio:

- informazioni di identificazione della NF come *id*, *nome*, *nome breve*, *logo*, eccetera;
- un'interfaccia di management utilizzata per configurare ed eseguire azioni sulla NF;
- punti di connessione esterni che sono esposti dal VNF verso il NS per abilitare l'interconnessione tra funzioni diverse;
- link virtuali interni usati per connettere le diverse VDU all'interno della VNF. Queste connessioni non vengono esposte verso l'esterno e vengono mappate dal VIM in reti virtuali;
- una o più Virtual Deployment Unit, VDU, che corrispondono alle macchine virtuali che eseguiranno componenti della NF. Per ogni VDU vengono poi specificati alcuni dati di identificazione, le interfacce che possono essere collegate a punti di connessione esterni o link virtuali interni, i requisiti in termini di risorse di calcolo e di archiviazione e l'immagine del software per la VM. Inoltre, è possibile specificare il file *cloud-init* da eseguire al primo avvio per configurare l'istanza.

¹Linguaggio di serializzazione dati human-readable

- Profili IP, che consentono di definire parametri per le reti virtuali che interconnettono le VDU come ad esempio indirizzi di rete, parametri DHCP e server DNS;
- metriche e parametri usati per le operazioni di day 0, day 1 e day 2.

Un NS Package contiene pressoché le stesse cartelle e file di un NF Package; in particolare, sono assenti le cartelle images e cloud-init, ma sono presenti le cartelle ns_config e vnf_config. Anche in questo caso, la modellazione del NS avviene tramite un file descriptor, NSD, scritto in YAML, che contiene:

- informazioni di identificazione del NS come id, nome, nome breve, logo, eccetera;
- riferimento ai VNFD che modellano le NF che compongono il servizio;
- uno o più descrittori VNFFG [1.4];
- punti di connessione esterni che, in maniera simile a quanto avviene per le NF, possono essere utilizzati da un NST per interconnettere tra loro diversi NS;
- VLD [1.4] che descrivono i collegamenti per l'interconnessione dei punti di connessione esterni delle VNF;
- profili IP che consentono di definire parametri per le reti virtuali che interconnettono le VNF come ad esempio indirizzi di rete, parametri DHCP e server DNS;
- metriche e parametri usati per le operazioni di day 0, day 1 e day 2.

La modellazione separata di NF e NS presenta diversi vantaggi:

- la figura che modella il NS (ad esempio, il fornitore di servizio) può astrarre dalla struttura interna delle NF concentrandosi esclusivamente sulle proprietà esterne che esse forniscono;
- riutilizzo delle NF in più NS senza un lavoro di modellazione extra;
- miglior gestione della complessità delle diverse componenti.

1. On-boarding

Una volta che i packages sono pronti, tramite la CLI di OSM possono essere validati e caricati nel sistema in modo che possano fungere da template per la creazione, successivamente, di funzioni e servizi di rete. Il caricamento dei packages può avvenire anche tramite GUI. Questa fase prende il nome di on-boarding. In OSM, l'unità minima che può essere messa in esecuzione è il NS: non è possibile eseguire una NF singolarmente. Per ogni NS devono essere aggiunti prima tutti i packages delle NF che lo compongono e poi il package del servizio stesso.

2. Creazione NS

Una volta che i pacchetti NS e NF sono stati correttamente inseriti in OSM, possono essere usati come modelli per la creazione del NS. A tempo di creazione dell'istanza possono essere forniti parametri aggiuntivi a NS e alle NF che lo compongono; ciò va a beneficio della riusabilità dei NS e delle NF. In questa fase, OSM interagisce con il VIM per allocare le risorse della NFVI alle VNF che compongono il NS. Questa fase può essere suddivisa in due sottofasi:

- la *day 0*, in cui vengono istanziate le VDU appartenenti alle diverse VNF. Le VDU sono configurate inizialmente a tempo di boot tramite cloud-init. Un esempio di configurazione iniziale tramite cloud-init è l'abilitazione del servizio ssh in modo che la VDU sia gestibile dall'esterno una volta creata.
- La *day 1*, in cui è possibile configurare dinamicamente le diverse VNF tramite Juju [3.1.3].

3. Operazioni su NS

Questa è la fase di *day 2* in cui si monitorano le istanze delle VNF associate al NS che sono attive e forniscono il servizio desiderato. In questa fase vengono gestite le richieste dei client, effettuate tramite apposite API, che possono riguardare il servizio di rete (ad esempio, azioni per aumentare o diminuire le risorse, per sospendere o riabilitare il servizio, eccetera) o le funzionalità offerte da questo (ad esempio, aggiunta di un nuovo utente). Altre azioni da gestire sono quelle invocate internamente dal ciclo di controllo definito nei packages NF e/o NS. Tipicamente queste azioni sono generate quando alcuni parametri che vengono monitorati superano certe soglie (ad esempio, definizione di politiche di scale-out automatico).

4. Terminazione NS

Questa è la fase conclusiva del ciclo di vita di un NS. L'istanza NS viene terminata insieme a tutte le istanze VNF associate. Il VIM riceve la richiesta di rilasciare tutte le risorse virtuali allocate.

3.1.3 Configurazione delle VNF

Come detto in precedenza, OSM permette di automatizzare la configurazione e riconfigurazione delle VNF. Per fare ciò utilizza due strumenti software principali, entrambi sviluppati da Canonical: cloud-init e Juju. Il primo si occupa delle configurazioni eseguite nella fase di day 0 il secondo di quelle eseguite nelle fasi di day 1 e day 2.

Cloud-init

Le immagini Cloud sono modelli di sistemi operativi e tutte le istanze che condividono la stessa immagine sono identiche. Cloud-init permette di inizializzare ciascuna istanza con delle informazioni peculiari che consentono, ad esempio, di configurare le interfacce di rete e i dispositivi di archiviazione, abilitare l'accesso mediante SSH tramite password o tramite chiavi ed eventualmente generare queste ultime. Sebbene cloud-init fosse pensato inizialmente per funzionare con Ubuntu, attualmente è disponibile per la maggior parte dei principali sistemi operativi Linux e FreeBSD.

Juju

Juju è un Operator Lifecycle Manager, OLM. Gli operators sono un approccio di automazione reso popolare in Kubernetes; un container operator si occupa di gestire gli altri container che compongono il sistema. Questo pattern prevede, quindi, di utilizzare software di più alto livello per controllare software di più basso livello; in questo modo, lavorando ad un livello di astrazione più alto, la gestione del sistema in esecuzione risulta più semplice.

Oltre ai servizi di base per la gestione del ciclo di vita degli operator: installazione, aggiornamento, configurazione e rimozione, Juju fornisce servizi di integrazione che permettono di combinarli in maniera dichiarativa e automatizzata. Juju porta il pattern operator di Kubernetes su Windows e Linux, permettendo la gestione di app in ambienti non containerizzati.

Il software di un operator insieme ai metadati dell'applicazione che controlla e degli ambienti su cui può essere utilizzato costituisce un pacchetto denominato **charm**. Nel caso in cui un charm venga messo in esecuzione su

Kubernetes verrà installato su un container, mentre negli ambienti tradizionali, come server bare metal o un'istanza su Cloud pubblico, verrà installato sulla macchina in una directory particolare.

L'ecosistema Juju è particolarmente grande e ricco di concetti; di seguito verranno introdotti quelli principali.

Cloud Identifica una risorsa che fornisce delle istanze su cui è possibile eseguire delle application units. Ciò include Cloud pubblici come Amazon Web Services, Google Compute Engine e Microsoft Azure, nonché Cloud privati basati su OpenStack. Juju può utilizzare anche ambienti che di per sé non sono Cloud, ma che può trattare come tali, è il caso LXD². Nei paragrafi successivi, il concetto di Cloud farà riferimento a questa nozione.

Controller È l'istanza Cloud iniziale e rappresenta un nodo centrale di gestione che ha il compito di mantenere lo stato di tutti i modelli, applicazioni e macchine in esecuzione sul Cloud scelto e di interagire con il Juju client (CLI). Il controller è diverso a seconda della piattaforma Cloud scelta.

Modello Un modello è uno spazio di lavoro che permette di astrarre dall'infrastruttura sottostante e di pensare ad ogni componente di quest'ultima come una singola entità. Applicazioni, spazio di archiviazione, risorse di rete, eccetera sono contenuti in modelli Juju. Sostanzialmente, un modello incapsula un numero indefinito di macchine su cui è possibile, in maniera trasparente, eseguire più applicazioni [3.4]. Ogni modello è gestito da un singolo controller e ciò permette al client di avere un unico punto di contatto con l'intero sistema. Il controller stesso esegue all'interno di un modello di cui è l'unica istanza, eccezion fatta per il caso in cui sia abilitata *l'alta disponibilità del controller*; in questo caso, più controller eseguiranno all'interno dello stesso modello.

²LXD è un gestore di container basati su Linux, LXC

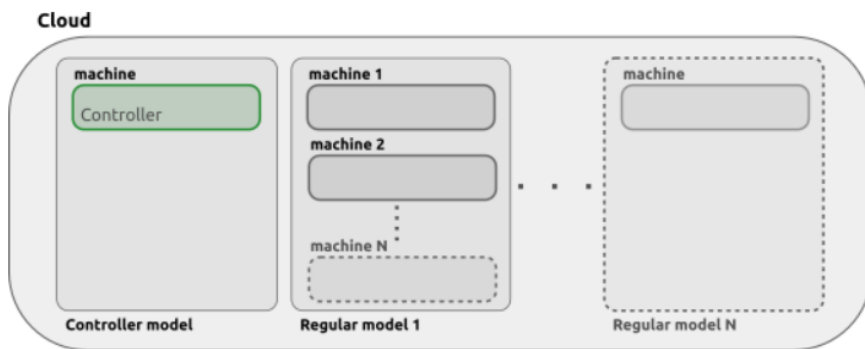


Figura 3.4: Modelli Juju [22]

Macchina Una macchina Juju è un'istanza Cloud che tipicamente ospita una singola application unit o un controller; per differenziare i due casi, le macchine che ospitano le application unit verranno definite regolari. In [3.5] è mostrata una macchina regolare su cui è eseguito un charm e un container LXD che esegue un charm. Nella figura è possibile notare che una macchina regolare esegue due tipi di agente. Un agente Juju è un software che può lavorare a livello macchina, agente macchina, o a livello di application unit, agente unità. Il primo, crea e controlla i container e gli agenti unità che sono in esecuzione sulla macchina, questi ultimi, sono responsabili delle attività relative al charm. In generale, tutti gli agenti tengono traccia dei cambiamenti di stato, rispondono a questi e trasmettono le informazioni in merito al controller. Lo stato di un modello viene creato dalla comunicazione tra un controller e tutti gli agenti in esecuzione su quel modello.

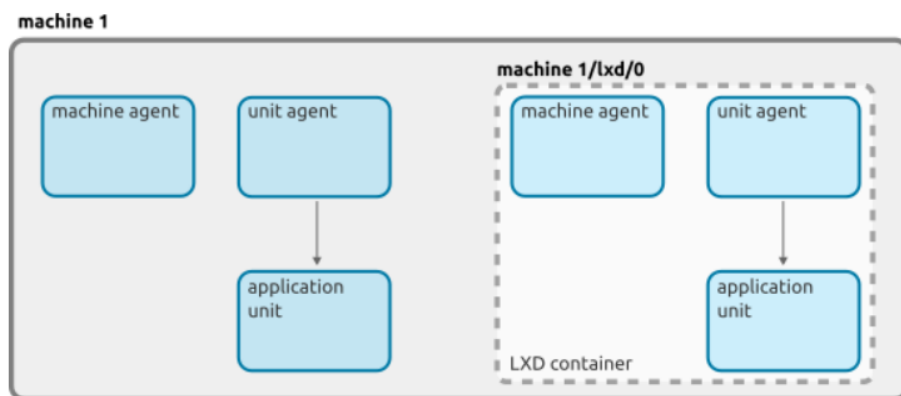


Figura 3.5: Macchina Juju [22]

Applicazione e application unit Un'applicazione si compone di una o più unità che rappresentano del codice in esecuzione. Diverse unità eseguono su macchine diverse condividendo lo stesso charm, le stesse relazioni e la stessa configurazione fornita dall'utente. Ad esempio, è possibile lanciare un servizio su tre unità, in tre macchine distinte, in modo da implementare un set di repliche e garantire tolleranza ai guasti. Tra le varie unità che compongono un'applicazione ve n'è una che funge da leader, ed è la fonte autorevole in merito allo stato e alla configurazione della macchina. Ogni applicazione possiede un solo leader.

Azioni Le azioni sono script che vengono attivati tramite il client ed eseguiti su un'unità. Sono descritte all'interno dei singoli charm attraverso un file *actions.yaml*. Ogni azione può, opzionalmente, prendere in input uno o più parametri.

End-point, interfacce e relazione Un end-point è un punto di collegamento esposto da un'applicazione per il collegamento con un'altra. È definito da tre proprietà: nome, ruolo e interfaccia. Un'interfaccia è il protocollo di comunicazione utilizzato in una relazione tra le applicazioni. Gli end-point di due applicazioni aventi ruolo compatibile e la stessa interfaccia, possono essere collegati formando una relazione.

Juju in OSM Come detto più volte in precedenza, Juju è utilizzato in OSM per la configurazione dinamica in fase day 1 e day 2. OSM supporta una versione limitata di charm: proxy charms e native charms. La differenza tra i due è che, nel primo caso il controller e i charms sono in esecuzione sulla stessa macchina, mentre nel secondo i charms eseguono sulle macchine che ospitano la VNF [3.6]. I native charms sono stati introdotti recentemente e sono ad uno stato di sviluppo primordiale, per questo motivo non verranno approfonditi ulteriormente.

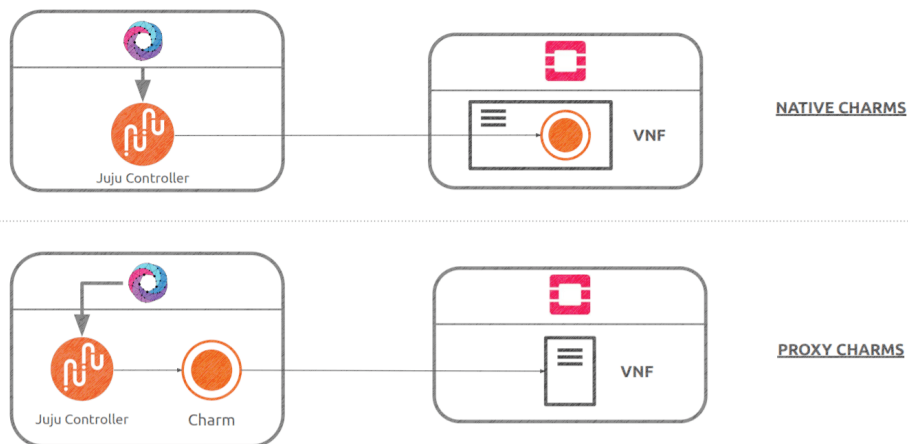


Figura 3.6: Differenza tra proxy charms e native charms [43]

L'uso dei proxy charm prevede che le configurazioni da eseguire sulle VNF vengano mappate in azioni Juju che verranno poi eseguite dal charm stesso. La configurazione della VNF può avvenire in diversi modi, ad esempio via SSH o tramite API REST. Il charm implementa un pattern reattivo che gli permette, non solo di rispondere all'invocazione di certe azioni, ma anche a eventi come il cambio di una configurazione; in questo modo è possibile riconfigurare a caldo una VNF. Più dettagliatamente, un proxy charm si compone di diversi layer ognuno dei quali implementa una certa funzionalità:

- **layer:basic.** Importa il framework reattivo e contiene il codice di base necessario per il funzionamento degli altri livelli.
- **layer:vnfproxy.** Importa le funzioni richieste per eseguire azioni nella VNF tramite SSH.
- **layer:restapi.** Importa le funzioni richieste per eseguire azioni nella VNF tramite API REST.
- **layer:netconf.** Importa le funzioni richieste per eseguire azioni nella VNF tramite Netconf.

3.2 OpenStack

Come detto in precedenza, tra le tante soluzioni utilizzabili come VIM in OSM è presente OpenStack. OpenStack, lanciato nel 2010 da NASA e Rack-space, oggi è una delle piattaforme Cloud IaaS open-source più utilizzata nel mondo. È una raccolta di progetti software open-source che le aziende

o i fornitori di servizi Cloud possono utilizzare per configurare ed eseguire la propria infrastruttura di computing e storage. Le risorse fisiche sottostanti, tramite tecnologie di virtualizzazione, vengono astratte e distribuite on-demand. La figura [3.7] mostra una vista semplificata dell'architettura di OpenStack. Le API supportano l'accesso ai servizi e il controllo delle risorse offerte agli utenti e alle applicazioni. In particolare, per la gestione delle risorse, OpenStack mette a disposizione di utenti e amministratori una GUI e una CLI; quest'ultima permette di esercitare un controllo più preciso sulle specifiche per l'allocazione delle risorse e la gestione delle istanze.

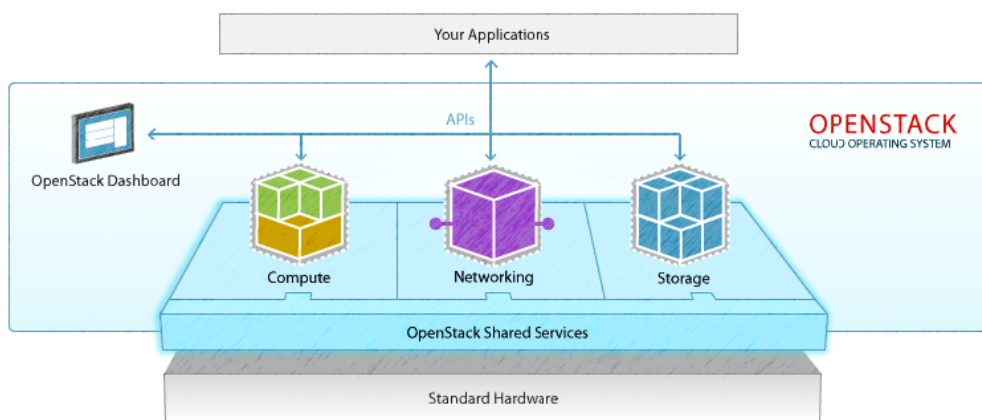


Figura 3.7: Architettura ad alto livello di OpenStack [42]

Per garantire modularità e flessibilità in modo da adattarsi ai bisogni dei singoli utenti, OpenStack implementa un'architettura a servizi [3.8]; ogni servizio è svolto da un insieme preciso di componenti. Di seguito verranno presentati i componenti principali.

Keystone È il componente che fornisce il servizio d'identità e integra funzioni per l'autenticazione, la catalogazione e la conseguente scoperta dei servizi e le politiche per registrare e gestire diversi progetti e utenti. Offre numerose modalità di autenticazione come nome utente/password e un sistema basato token [36]. Inoltre, è possibile integrarlo con un directory back-end esistente come Lightweight Directory Access Protocol, LDAP, per implementare meccanismi di Role Based Access Control, RBAC. È il primo servizio con cui l'utente interagisce al fine di autenticarsi e poter accedere agli altri servizi. In aggiunta, viene utilizzato dagli altri componenti per le verifiche in merito all'identità degli utenti e per la localizzazione degli altri servizi all'interno della piattaforma.

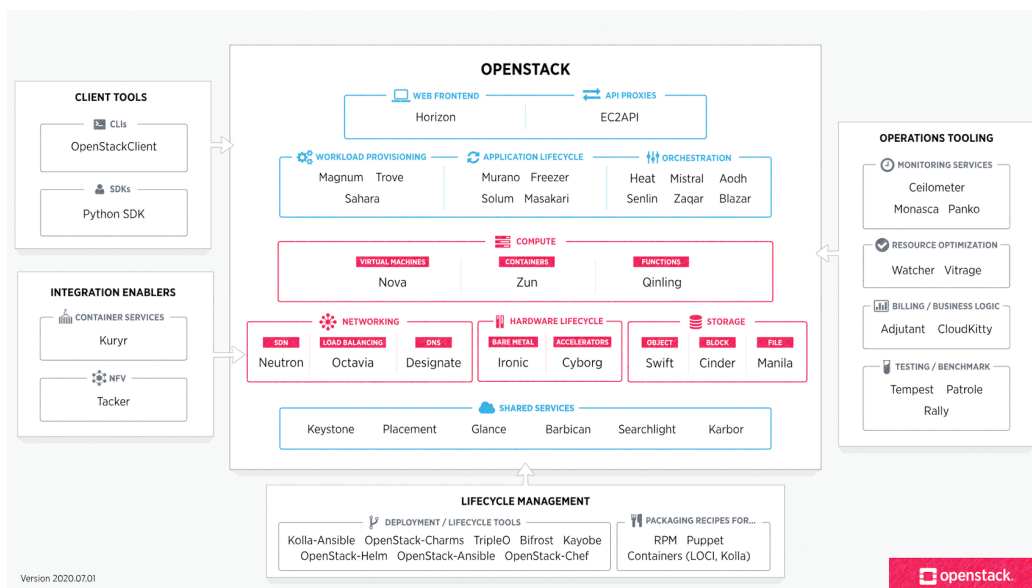


Figura 3.8: Architettura a servizi di OpenStack [42]

Swift Swift è un sistema di archiviazione ad oggetti; sistemi di questo tipo gestiscono e memorizzano i dati come oggetti, a differenza di altre architetture di archiviazione come i file system che gestiscono i dati come gerarchia di file e l'archiviazione a blocchi che gestisce i dati come blocchi all'interno di settori e tracce. Ogni oggetto, in genere, include i dati stessi, una quantità variabile di metadati e un identificatore univoco globale. Il sistema di memorizzazione offerto da Swift è distribuito e sacrifica la consistenza immediata, adottando un modello di eventual consistency, per garantire scalabilità e tolleranza ai guasti.

Glance Glance è il componente che funge da registro di immagini memorizzando i metadati relativi a queste. In altre parole, gestisce la scoperta, la registrazione e la consegna delle immagini virtuali e si coordina con i sistemi di archiviazione di OpenStack, come Swift, per memorizzarle.

Cinder Cinder si occupa di fornire un servizio di archiviazione a blocchi. Si occupa della creazione, allocazione e deallocazione dei volumi alle/dalle istanze. Tramite apposite API, gli utenti possono creare e gestire i volumi e grazie alla virtualizzazione e al livello di astrazione introdotto da Cinder non devono preoccuparsi di dove questi verranno memorizzati.

Nova È uno dei primi componenti del core di OpenStack e anche uno dei più complicati a livello architetturale. È composto da più processi, ognuno dei quali svolge funzioni diverse. L'interfaccia rivolta all'utente espone delle API REST per l'interazione con questo, mentre internamente i processi comunicano tramite un meccanismo a scambio di messaggi RPC fornito dalla libreria **oslo.messaging**, un'astrazione sopra alle code di messaggi. I diversi processi che compongono Nova e le interazioni tra loro e con altri componenti di OpenStack sono raffigurati in [3.9].

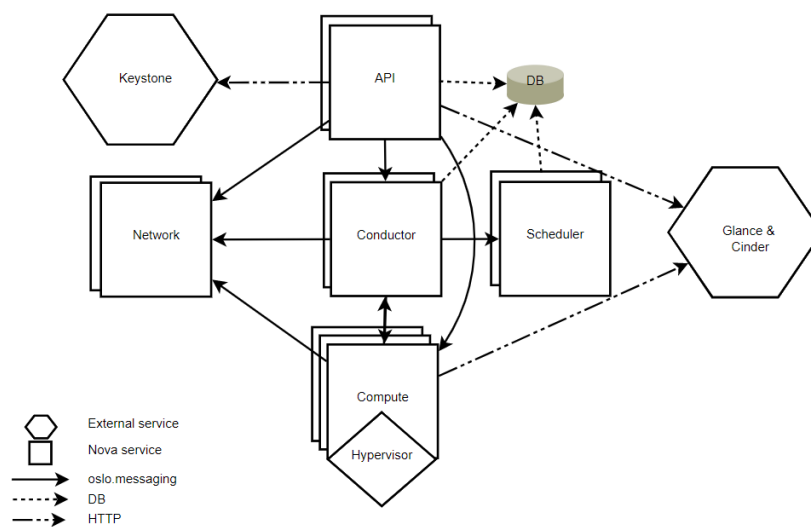


Figura 3.9: Processi di Nova e interazioni tra loro e con gli altri componenti di OpenStack

- **nova-API:** è il componente di interazione con l'utente descritto in precedenza;
- **nova-compute:** ospita e gestisce le istanze VM comunicando con l'hypervisor sottostante;
- **nova-scheduler:** coordina tutti i servizi e determina il posizionamento delle risorse richieste;
- **nova-network:** accetta i messaggi di richiesta ed esegue attività per configurare le reti. Attualmente, è possibile di scegliere di installare e configurare nova-network o utilizzare il servizio OpenStack networking, Neutron.
- **DB:** memorizza gli stati a tempo di build e di esecuzione dell'infrastruttura Cloud;

- **nova-conductor:** è un ulteriore livello di astrazione che permette ai processi di compute di non accedere al DB. Le possibili operazioni su quest'ultimo vengono implementate ed offerte dal conductor. Il vantaggio più grande derivante da questo accentramento è la miglior gestione della sicurezza.

Neutron Neutron offre un servizio Network as a Service (NaaS), consentendo agli utenti di creare reti su cui collegare le Virtual Network Interfaces delle proprie istanze virtuali. In maniera simile a quanto visto per gli altri processi, Neutron disaccoppia la vista logica della rete dalla vista fisica, fornendo delle API per definire, gestire e connettere le reti virtuali. I concetti principali della visione logica della rete sono:

- **virtual network:** rappresenta un livello 2 di rete virtualizzato e isolato, può essere considerato come uno switch logico;
- **virtual subnet:** rappresenta l'insieme di indirizzi IPv4 o IPv6 che possono essere assegnati alle VM su una determinata virtual network;
- **virtual ports:** rappresentano le porte dello switch logico su una determinata rete che possono essere collegate alle interfacce delle VM.

Inoltre, Neutron offre un supporto alla multi-tenancy isolando completamente le risorse assegnate a diversi utenti anche se condividono gli stessi nodi fisici. Ciò, è ottenuto mediante l'uso di tecniche come VLAN tagging e/o il tunneling per isolare il traffico tra le istanze virtuali indipendentemente dal fatto che siano co-locate o in esecuzione in nodi separati del cluster.

Horizon Horizon è la dashboard web che consente agli utenti e agli amministratori di controllare tutti i componenti della piattaforma OpenStack. Non fa altro che avviare azioni tramite chiamate alle API offerte dai vari componenti e visualizzare le informazioni ricevute come risposta.

3.3 Open5GS

Open5GS è un'implementazione open-source, supportata da NextEPC, del core di rete 5G e 4G, rispettivamente 5GC e EPC, in conformità alla Release 16 3GPP [41]. Ogni componente dei core implementati è incapsulato in moduli software che comunicano tra loro tramite le interfacce e i protocolli presentati in [2.4] e [2.5]. Di default, i componenti dei core utilizzano indirizzi di loopback, 127.0.0.0/8, per comunicare tra loro, ma è possibile suddividerli in macchine differenti previa apposita configurazione. È un progetto in

corso di sviluppo; per questo, alcune funzionalità non sono ancora state implementate, ad esempio non sono presenti: gestione roaming, chiamate di emergenza, Narrowband Internet of Things ³, eccetera.

Di seguito verranno introdotti i diversi moduli e gli aspetti più rilevanti al fine della loro configurazione.

3.3.1 Moduli software Open5GS

I moduli che compongono Open5GS sono dodici e possono essere installati, in diverse distribuzioni Linux, direttamente come pacchetti o compilati a partire dal codice sorgente. Una volta installati, i moduli vengono lanciati come demoni e sono gestibili tramite il Linux system and service manager, *systemd*.

La topologia di Open5GS è mostrata in [3.10]; come si evince dall'immagine, è implementata la separazione tra piano utente e piano di controllo, CUPS [2.4.1]. Inoltre, i core di rete sono interoperabili e per fare ciò, come già accennato in [2.5.5], l'SMF del 5GC è integrato con il PGW-C dell'EPC e l'UPF con il PGW-U.

Il comportamento di ciascun modulo è configurabile tramite un apposito file YAML che deve essere modificato a seconda della distribuzione dei componenti e degli indirizzi IP e porte utilizzati.

Affinché l'utente possa collegarsi ad uno dei due core, è necessario registrare i dati dell'abbonamento (che userà nell'UE) nel Subscriber Database, che corrisponde a un'istanza MongoDB⁴. Per fare ciò è possibile procedere in tre modi:

- inserire manualmente i dati nel DB collegandosi alla macchina in cui è in esecuzione l'istanza e utilizzando la Mongo shell;
- utilizzare lo script `open5gs-dbctl`; nel caso in cui Open5GS sia stato installato come pacchetto, tale script viene posizionato nella cartella `/usr/bin` ed è direttamente utilizzabile mediante il comando `open5gs-dbctl`. Lo script fornisce un insieme limitato di comandi eseguibili; pertanto, potrebbe non essere adeguato alle operazioni da svolgere.
- Installare la WebUI di Open5GS. Una volta installata è possibile accedervi navigando all'indirizzo `http://localhost:3000` ed effettuando l'ac-

³Standard di tecnologia radio, Low Power Wide Area Network, sviluppato da 3GPP per consentire la comunicazione di un'ampia gamma di dispositivi e servizi cellulari a fronte di un basso consumo energetico

⁴DBMS NoSQL orientato ai documenti

cesso tramite username, *admin* e password, *1423* sarà possibile inserire i dati.

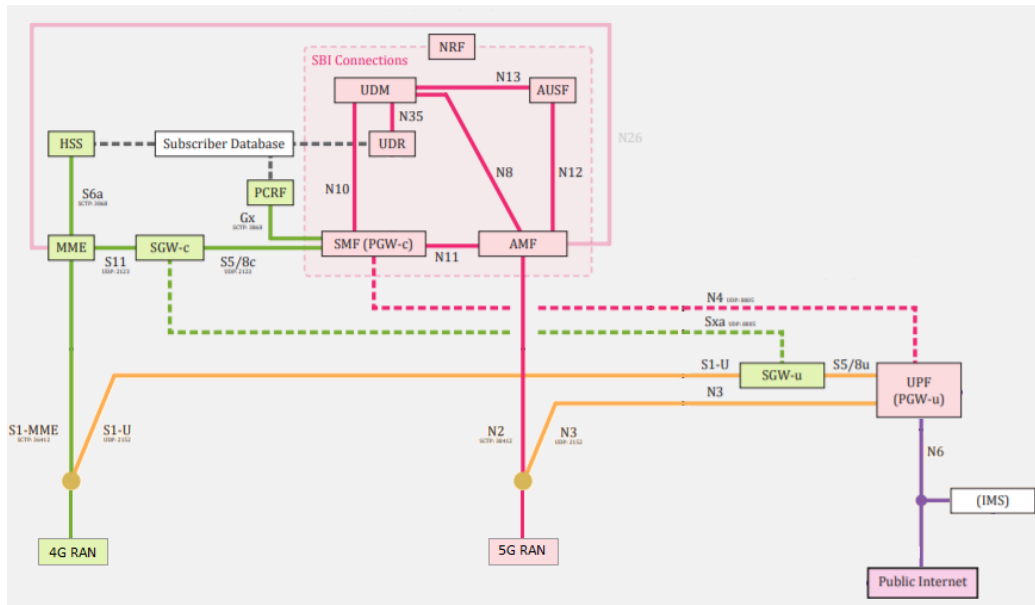


Figura 3.10: Moduli Open5GS e connessioni tra loro

Tramite WebUI è possibile aggiungere profili e abbonati [3.11]. Un abbonato può essere associato a un profilo; quest'ultimo svolge il ruolo di template. I dati principali di un abbonato sono:

- l'IMSI, che identifica l'abbonato;
- la chiave dell'abbonato, K, e la chiave dell'operatore, OP/OPc. Queste sono due chiavi segrete che vengono utilizzate per derivare tutte le altre chiavi secondarie e i vettori di autenticazione per supportare le procedure di autenticazione, cifratura del traffico e protezione dell'integrità. Poiché il valore OP è lo stesso per tutti gli abbonati/SIM, è potenzialmente possibile, tramite spoofing, falsificare tutte le SIM. Per questo motivo, a partire dall'OP e da K si può generare l'OPc utilizzando l'algoritmo RijndaelEncrypt specifico della SIM; una volta codificato, dall'OPc non è più possibile risalire all'OP. In questo modo, nel caso di furto dell'OPc, solo una SIM può essere falsificata.

In aggiunta, ogni abbonamento include le impostazioni per gli APN disponibili per l'utente[3.12]. La sezione adibita a tali impostazioni include il nome

dell'APN e i parametri di QoS del bearer. Un valore di QCI uguale a 9 sta a indicare un livello di qualità di tipo best-effort.

Edit Subscriber

Subscriber Configuration

IMSI*

Subscriber Key (K)*
465B5CE8 B199B49F AA5F0A2E E238A6BC

Authentication Management Field (AMF)*
8000

USIM Type
OPc

Operator Key (OPc/OP)*
E8ED289D EBA952E4 283B54E8 8E6183CA

UE-AMBR Downlink (Kbps)*
1024000

UE-AMBR Uplink (Kbps)*
1024000

APN Configurations

CANCEL SAVE

Figura 3.11: Aggiunta dati abbonato tramite WebUI di Open5GS

Create Subscriber

APN Configurations

Access Point Name (APN)*
internet

Type*
IPv4v6

QoS Class Identifier (QCI)*
 1 2 3 4 5 6 7 8 9 65 66 69 70

ARP Priority Level (1-15)*
8

Capability*
Disabled

Vulnerability*
Disabled

APN-AMBR Downlink (Kbps)*
1024000

APN-AMBR Uplink (Kbps)*
1024000

UE IPv4 Address

UE IPv6 Address

CANCEL SAVE

Figura 3.12: Aggiunta dati APN per abbonato tramite WebUI di Open5GS

3.3.2 Configurazione Open5GS

Come detto in precedenza, la configurazione dei vari componenti è contenuta in file *YAML*. I componenti che utilizzano il protocollo Diameter possiedono un ulteriore file di configurazione, *.conf*. I file *YAML*, una volta installato Open5GS, sono contenuti in */etc/Open5GS*, mentre i *conf* in */etc/freeDiameter*. Nelle sezioni successive verranno descritte le diverse configurazioni per ciascun componente. Si utilizzerà il termine interfaccia in riferimento al concetto di reference point 3GPP.

Configurazione MME

Il demone, *open5gs-mmed*, richiede due file di configurazione: */etc/open5gs/mme.yaml* e */etc/freeDiameter/mme.conf*. Il primo [3.1] permette di:

- definire l'IP ed eventualmente la porta dell'interfaccia di ascolto S1-MME (linee 6 e 7)
- definire l'IP ed eventualmente la porta dell'interfaccia di ascolto S11 (linee 9 e 10)
- definire l'IP ed eventualmente la porta dell'interfaccia S11 su cui è in ascolto l'SGW-C (linee 31 e 32)
- definire l'IP ed eventualmente la porta dell'interfaccia S5/8c su cui è in ascolto l'SMF (linee da 36 a 38)
- definire Global Unique MME ID, GUMMEI, e Tracking Area ID, TAI (linee da 11 a 21)
- definire il nome della rete e gli algoritmi di sicurezza supportati (linee da 22 a 26).

Il GUMMEI identifica una MME all'interno dell'EPC e si compone di:

- Public Land Mobile Network Identifier, PLMN ID, identifica una rete mobile ed è composto da Mobile Country Code, MCC e da Mobile Network Code, MNC; il primo è un numero a tre cifre che identifica il paese e il secondo è un numero a due o tre cifre che indica la rete dell'operatore in quel paese;
- MME Identifier, MMEI, che a sua volta è composto da MME Group ID, MMEGI, che indica l'ID del gruppo, all'interno della PLMN, a cui appartiene la MME e da MME Code, MMEC, che identifica univocamente la MME all'interno del gruppo.

Il TAI identifica globalmente una certa tracking area nella rete. È composto da un PLMN ID che identifica la rete mobile e da un TAC (Tracking Area Code), assegnato dagli operatori, per identificare in modo univoco una tracking area nella propria rete mobile. La MME comunicherà solo con eNodeB situati nella stessa tracking area e all'interno della stessa PLMN.

```
1 logger:
2   file: /var/log/open5gs/mme.log
3 mme:
4   freeDiameter: /etc/freeDiameter/mme.conf
5   slap:
6     - addr: S1MME_LISTEN_IP #default 127.0.0.2
7       #port: 36412
8   gtpc:
9     - addr: S11_LISTEN_IP #default 127.0.0.2
10      #port: 2123
11  gummei:
12    plmn_id:
13      mcc: MCC #default 901
14      mnc: MNC #default 70
15      mme_gid: MMEGID #default 2
16      mme_code: MMECODE #default 1
17  tai:
18    plmn_id:
19      mcc: MCC #default 901
20      mnc: MNC #default 70
21      tac: TAC #default 1
22  security:
23    integrity_order : [ EIA2, EIA1, EIA0 ]
24    ciphering_order : [ EEA0, EEA1, EEA2 ]
25  network_name:
26    full: Open5GS
27  mme_name: open5gs-mme0
28
29  sgwc:
30    gtpc:
31      - addr: SGWC_S11_IP #default 127.0.0.3
32        #port: 2123
33  smf:
34    gtpc:
35      - addr:
```

```

36         - SMF_S58C_IP #default 127.0.0.4
37         - ::1
38         #port: 2123
39 ...

```

Listing 3.1: File di configurazione mme.yaml

Il secondo file [3.2], permette di specificare l'IP ed eventualmente la porta dell'interfaccia di ascolto S6a (linee 3 e 4) e le stesse informazioni per la medesima interfaccia su cui ascolta l'HSS (linea 7).

```

1 Identity = "mme.localdomain";
2 Realm = "localdomain";
3 #Port = 3868;
4 ListenOn = "MME_S6a_IP_ADDRESS"; #default 127.0.0.2
5 ...
6 ConnectPeer = "hss.localdomain" { ConnectTo =
7 "HSS_S6a_IP_ADDRESS"; No_TLS; /*Port = 3868;*/ };
8 #default ConnectTo = "127.0.0.8"

```

Listing 3.2: File di configurazione mme.conf

Configurazione AMF

Il demone, *open5gs-amfd*, richiede un unico file di configurazione: */etc/open5gs/amf.yaml*. Il contenuto di questo file [3.3] è simile a quello visto per la MME e permette di:

- definire l'IP ed eventualmente la porta dell'interfaccia di ascolto SBI (linee 5 e 6). Come detto in precedenza, il 5GC implementa un'architettura a servizi; ogni componente ascolta su un'interfaccia SBI e possiede il riferimento all'interfaccia SBI dell'NRF. Tale componente è utile per scoprire la posizione degli altri servizi.
- Definire l'IP ed eventualmente la porta dell'interfaccia di ascolto N2 (linee 8 e 9);
- definire il Global Unique AMF ID, GUAMI, e il Tracking Area ID, TAI (linee da 9 a 20). Il GUAMI è molto simile al GUMMEI; ciò che cambia è l'identificatore del componente. In particolare, la *region* identifica un insieme di AMF, mentre il *set* identifica l'AMF all'interno della *region*.
- Definire gli allowed NSSAI relativi alla PLMN (linee da 22 a 28). Un NSSAI si compone di diversi S-NSSAI a loro volta composti da Standardized Slice/Service Type, SST e dallo Slice Differentiator, SD. Il

primo fa riferimento al comportamento previsto del network slice in termini di funzionalità e servizi; il secondo è un campo opzionale per distinguere più slice di rete dello stesso tipo. I valori standardizzati per SST sono rappresentati in [3.1].

- Definire il nome della rete e gli algoritmi di sicurezza supportati (linee da 29 a 33);
- definire l'IP ed eventualmente la porta dell'interfaccia SBI su cui è in ascolto l'NRF (linee da 39 a 41).

```

1 logger:
2   file: /var/log/open5gs/amf.log
3 amf:
4   sbi:
5     - addr: N8N11N12_LISTEN_IP #default 127.0.0.5
6       port: 7777
7   ngap:
8     - addr: N2_LISTEN_IP #default 127.0.0.5
9       #port: 38412
10  guami:
11    - plmn_id:
12      mcc: MCC #default 901
13      mnc: MNC #default 70
14    amf_id:
15      region: REGION #default 2
16      set: SET #default 1
17  tai:
18    - plmn_id:
19      mcc: MCC #default 901
20      mnc: MNC #default 70
21      tac: TAC #default 1
22  plmn_support:
23    - plmn_id:
24      mcc: MCC #default 901
25      mnc: MNC #default 70
26    s_nssai:
27      - sst: 1
28        #- sd: 010000
29  security:
30    integrity_order : [ NIA2, NIA1, NIA0 ]

```

```

31     ciphering_order : [ NEA0, NEA1, NEA2 ]
32     network_name:
33         full: Open5GS
34     amf_name: open5gs-amf0
35
36 nrf:
37     sbi:
38         - addr:
39             - NRF_SBI_IP #default 127.0.0.10
40             - ::1
41             port: 7777
42
43 ...

```

Listing 3.3: File di configurazione amf.yaml

Slice/Service Type	Valore
enhanced Mobile Broadband (eMBB)	1
Ultra-Reliable Low Latency Communications (URLLC)	2
Massive IoT (MIoT)	3
Vehicle-to-everything (V2X)	4

Tabella 3.1: Possibili valori per il campo SST

Configurazione SGW-C

Anche in questo caso, l'unico file di configurazione è */etc/open5gs/sgwc.yaml* [3.4] e permette di definire gli IP ed eventualmente le porte per le interfacce di ascolto S11 e Sxa (linee da 6 a 10). Quest'ultima serve per consentire all'SGW-C di controllare l'instradamento eseguito dall'SGW-U; pertanto, è necessario definire l'IP ed eventualmente la porta per l'interfaccia Sxa su cui ascolta l'SGW-U (linee 14 e 15).

```

1 logger:
2     file: /var/log/open5gs/sgwc.log
3
4 sgwc:
5     gtpc:
6         - addr: S11_LISTEN_IP #default 127.0.0.3
7             #port: 2123
8     pfcp:

```

```

 9         - addr: Sxa_LISTEN_IP #default 127.0.0.3
10           #port: 8805
11
12 sgwu:
13     pfcf:
14         - addr: SGWU_Sxa_IP #default 127.0.0.6
15           #port: 8805
16 ...

```

Listing 3.4: File di configurazione sgwc.yaml

Configurazione SGW-U

Il file `/etc/open5gs/sgwu.yaml` [3.5] permette di definire gli IP ed eventualmente le porte per le interfacce di ascolto S1-U, S5/8u e Sxa (linee da 6 a 10). Inoltre, consente di definire l'IP ed eventualmente la porta per l'interfaccia Sxa su cui ascolta l'SGW-C (linee 14 e 15).

```

1 logger:
2     file: /var/log/open5gs/sgwu.log
3
4 sgwu:
5     gtpu:
6         - addr: S1uS5S8u_LISTEN_IP #default 127.0.0.6
7           #port: 2152
8     pfcf:
9         - addr: Sxa_LISTEN_IP #default 127.0.0.6
10          #port: 8805
11
12 sgwc:
13     pfcf:
14         - addr: SGWC_Sxa_IP #default 127.0.0.3
15           #port: 8805
16 ...

```

Listing 3.5: File di configurazione sgwu.yaml

Configurazione SMF+PGW-C

Il componente che integra SMF e PGW-C possiede due file di configurazione: `/etc/open5gs/smf.yaml` e `/etc/freeDiameter/smf.conf`. Il primo [3.6], peremette di:

- definire l'IP ed eventualmente la porta dell'interfaccia di ascolto SBI (linee 7 e 8);
- definire l'IP ed eventualmente la porta delle interfacce di ascolto S5/8c e N11 (linee da 10 a 12);
- definire l'IP ed eventualmente la porta dell'interfaccia di ascolto N4 (linee da 14 a 16); tale interfaccia è equivalente alla Sxa vista per l'SGW-C e permette all'SMF+PGW-C di controllare l'instradamento dell'UPF+PGW-U.
- Definire i server DNS e il pool di indirizzi, IPv4 ed eventualmente IPv6, da assegnare agli UE che richiedono la connessione alla rete (linee da 18 a 22). È possibile assegnare pool diversi ad APN differenti e specificare per questi ultimi un'interfaccia TUN/TAP⁵ diversa da quella di default: ogstun.
- Definire una dimensione massima di trasmissione, in Byte, dei pacchetti (linea 23). Il valore 1400 serve per evitare la frammentazione di questi nella rete backbone tra UE e PGW-U e/o attraverso il reference point SGi, tra PGW-U e PDN, quando alcuni dei collegamenti non supportano pacchetti più grandi di 1500 ottetti.
- Definire l'IP ed eventualmente la porta dell'interfaccia SBI su cui è in ascolto l'NRF (linee da 26 a 30);
- definire l'IP ed eventualmente la porta dell'interfaccia N4 su cui è in ascolto l'UPF (linee da 33 a 35);

```

1 logger:
2     file: /var/log/open5gs/smf.log
3
4 smf:
5     freeDiameter: /etc/freeDiameter/smf.conf
6     sbi:
7         - addr: N7N10N11_LISTEN_IP #default 127.0.0.4
8           port: 7777
9     gtpc:
10        - addr: S5S8cN11_LISTEN_IP #default 127.0.0.4
11        - addr: ::1

```

⁵TUN e TAP sono driver che permettono la creazione di periferiche virtuali. Queste ultime consentono ai programmi nello spazio utente di ricevere e trasmettere pacchetti.

```

12         #port: 2123
13     pfcf:
14         - addr: N4_LISTEN_IP #default 127.0.0.4
15         - addr: ::1
16         #port: 8805
17     pdn:
18         - addr: 10.45.0.1/16
19         - addr: cafe::1/64
20     dns:
21         - 8.8.8.8
22         - 8.8.4.4
23     mtu: 1400
24
25     nrf:
26         sbi:
27             - addr:
28                 - NRF_SBI_IP #default 127.0.0.10
29                 - ::1
30             port: 7777
31
32     upf:
33         pfcf:
34             - addr: UPF_N4_IP #default 127.0.0.7
35             #port: 8805
36     ...

```

Listing 3.6: File di configurazione smf.yaml

Il secondo file [3.7], */etc/freeDiameter/smf.conf*, permette di definire l'IP ed eventualmente la porta su cui l'SMF è in ascolto per l'interfaccia Gx (linee 3 e 4) e di fornire a quest'ultimo l'IP ed eventualmente la porta su cui è in ascolto il PCRF per la medesima interfaccia (linea 7).

```

1 Identity = "smf.localdomain";
2 Realm = "localdomain";
3 #Port = 3868;
4 ListenOn = "SMF_Gx_IP_ADDRESS"; #default 127.0.0.4
5 ...
6 ConnectPeer = "pcrf.localdomain" { ConnectTo =
7 "PCRF_Gx_IP_ADDRESS"; No_TLS; /*Port = 3868;*/ };
8 #default ConnectTo = "127.0.0.9"

```

Listing 3.7: File di configurazione smf.conf

Configurazione UPF+PGW-U

Il componente che integra UPF e PGW-U possiede esclusivamente il file di configurazione `/etc/open5gs/upf.yaml` [3.8]. Tramite questo, è possibile definire gli IP ed eventualmente le porte per le interfacce di ascolto N3, S5/8u e N4 (linee da 6 a 10). Inoltre, consente di specificare l'IP ed eventualmente la porta per l'interfaccia N4 su cui ascolta il PGW-C (linee 17 e 18). Come si evince dalle linee 11, 12 e 13 è necessario riportare la configurazione del/dei pool di indirizzi per la PDN già contenuta nel SMF+PGW-C. Questo perché lo standard 3GPP prevede che gli IP possano essere assegnati sia dall'SMF+PGW-C che dall'UPF+PGW-U. L'idea originale dello sviluppatore di Open5GS era quella di inserire tale configurazione solo nell'SMF+PGW-C per poi inviare tali regole all'UPF+PGW-U tramite protocollo PFCP. Tuttavia, il messaggio PFCP non contiene informazioni sul dispositivo. In un ambiente con più APN in cui l'UPF ha più tunnel attivi (ad esempio 2), è necessario configurare più dispositivi TUN/TAP (esempio ogstun e ogstun2) e ciò non potrebbe essere fatto per il motivo sopra riportato.

```
1 logger:
2     file: /var/log/open5gs/upf.log
3
4 upf:
5     pfcpc:
6         - addr: N4_LISTEN_IP #default 127.0.0.7
7           #port: 8805
8     gtpu:
9         - addr: S5S8uN3_LISTEN_IP #default 127.0.0.7
10        #port: 2152
11    pdn:
12        - addr: 10.45.0.1/16
13        - addr: cafe::1/64
14
15 smf:
16     pfcpc:
17         - addr: SMF_N4_IP #default 127.0.0.4
18           #port: 8805
19 ...
```

Listing 3.8: File di configurazione `upf.yaml`

Configurazione HSS

L'HSS possiede due file di configurazione: */etc/open5gs/hss.yaml* e */etc/freeDiameter/hss.conf*. Nel primo, è sufficiente impostare correttamente l'URI dell'istanza MongoDB che memorizza i dati degli abbonati (linea 1) [3.9]. Il secondo, permette di specificare l'IP ed eventualmente la porta dell'interfaccia di ascolto S6a (linee 3 e 4) e le stesse informazioni per la medesima interfaccia su cui ascolta la MME (linea 7) [3.10].

```
1 db_uri: mongodb://DB_INSTANCE_IP/open5gs
2 #default DB_INSTANCE_IP=localhost
3
4 logger:
5   file: /var/log/open5gs/hss.log
6
7 hss:
8   freeDiameter: etc/freeDiameter/hss.conf
9 ...
```

Listing 3.9: File di configurazione hss.yaml

```
1 Identity = "hss.localdomain";
2 Realm = "localdomain";
3 #Port = 3868;
4 ListenOn = "HSS_S6a_IP_ADDRESS"; #default 127.0.0.8
5 ...
6 ConnectPeer = "mme.localdomain" { ConnectTo =
7 "MME_S6a_IP_ADDRESS"; No_TLS; /*Port = 3868;*/ };
8 #default ConnectTo = "127.0.0.2"
```

Listing 3.10: File di configurazione hss.conf

Configurazione PCRF

Anche il PCRF possiede due file di configurazione: */etc/open5gs/pcrf.yaml* e */etc/freeDiameter/pcrf.conf*. Come nel caso del modulo HSS, nel primo file l'URI del database deve essere impostato correttamente (linea 1) [3.11]. Infatti, il PCRF ha bisogno di accedere alle informazioni memorizzate per fornire il servizio e le funzionalità di controllo della QoS dei bearers. Il secondo file, permette di definire l'IP ed eventualmente la porta su cui il PCRF è in ascolto per l'interfaccia Gx (linee 3 e 4) e di fornire a quest'ultimo l'IP ed eventualmente la porta su cui è in ascolto l'SMF per la medesima interfaccia (linea 7) [3.12].

```

1 db_uri: mongodb://DB_INSTANCE_IP/open5gs
2 #default DB_INSTANCE_IP=localhost
3
4 logger:
5     file: /var/log/open5gs/pcrf.log
6
7 pcrf:
8     freeDiameter: etc/freeDiameter/pcrf.conf
9 ...

```

Listing 3.11: File di configurazione pcrf.yaml

```

1 Identity = "pcrf.localdomain";
2 Realm = "localdomain";
3 #Port = 3868;
4 ListenOn = "PCRF_Gx_IP_ADDRESS"; #default 127.0.0.9
5 ...
6 ConnectPeer = "smf.localdomain" { ConnectTo =
7 "SMF_Gx_IP_ADDRESS"; No_TLS; /*Port = 3868;*/ };
8 #default ConnectTo = "127.0.0.4"

```

Listing 3.12: File di configurazione pcrf.conf

Configurazione UDM

L'unico file di configurazione per il componente, */etc/open5gs/udm.yaml*, permette di definire l'IP ed eventualmente la porta dell'interfaccia di ascolto SBI (linee 5 e 6) e di fornirgli l'IP ed eventualmente la porta su cui è in ascolto l'NRF per la medesima interfaccia (linee da 10 a 13) [3.13].

```

1 logger:
2     file: /var/log/open5gs/udm.log
3 udm:
4     sbi:
5         - addr: N10N35N8N13_LISTEN_IP #default 127.0.0.12
6           port: 7777
7
8     nrf:
9         sbi:
10            - addr:
11              - NRF_SBI_IP #default 127.0.0.10

```

```

12         - ::1
13         port: 7777
14     ...

```

Listing 3.13: File di configurazione udm.yaml

Configurazione UDR

Il file di configurazione */etc/open5gs/udr.yaml* permette di fare la stessa cosa del file visto per l'UDM. Inoltre, è necessario configurare l'URI dell'istanza MongoDB (linea 1) visto che l'UDR può essere visto come un livello di astrazione aggiuntivo tra il DB e l'UDM [3.14].

```

1 db_uri: mongodb://DB_INSTANCE_IP/open5gs
2 #default DB_INSTANCE_IP=localhost
3
4 logger:
5     file: /var/log/open5gs/udr.log
6
7 udr:
8     sbi:
9         - addr: NRF_LISTEN_IP #default 127.0.0.13
10         port: 7777
11
12 nrf:
13     sbi:
14         - addr:
15             - NRF_SBI_IP #default 127.0.0.10
16             - ::1
17         port: 7777
18     ...

```

Listing 3.14: File di configurazione udr.yaml

Configurazione AUSF

Il file di configurazione */etc/open5gs/ausf.yaml* permette di fare la stessa cosa del file visto per l'UDM [3.15].

```

1 logger:
2     file: /var/log/open5gs/ausf.log
3

```

```

4 ausf:
5     sbi:
6         - addr: N13N12_LISTEN_IP #default 127.0.0.11
7           port: 7777
8
9 nrf:
10    sbi:
11        - addr:
12            - NRF_SBI_IP #default 127.0.0.10
13            - ::1
14          port: 7777
15
16 ...

```

Listing 3.15: File di configurazione ausf.yaml

Configurazione NRF

L'unica funzione dell'NRF è quella di svolgere il compito di registro di servizi; per questo, il file di configurazione */etc/open5gs/nrf.yaml* permette esclusivamente di configurare l'IP ed eventualmente la porta di ascolto per la SBI su cui il componente riceverà le richieste.

```

1 logger:
2     file: /var/log/open5gs/nrf.log
3
4 nrf:
5     sbi:
6         addr:
7             - NRF_SBI_IP #default 127.0.0.10
8             - ::1
9         port: 7777
10 ...

```

Listing 3.16: File di configurazione nrf.yaml

3.4 srsLTE

srsLTE [45] è una suite di software LTE gratuita e open source sviluppata da SRS e include:

- srsUE - un'applicazione SDR che svolge il ruolo di UE;

- srsENB - un'applicazione SDR che svolge il ruolo di eNodeB;
- srsEPC - un'implementazione del core LTE con MME, HSS, SGW e PGW;
- un set altamente modulare di librerie comuni per livelli PHY, MAC, RLC, PDCP, RRC, NAS, S1AP e GW.

Questo elaborato non userà srsEPC, ma solo srsUE e srsENB.

srsLTE supporta diversi dispositivi hardware SDR. In alternativa, è possibile sostituire il collegamento radio tra eNodeB e UE con un meccanismo che permette di scambiare dati I/Q⁶ in banda base su un trasporto alternativo. A tale scopo, srsLTE supporta un driver RF basato su ZeroMQ [23] che agisce come un tunnel di trasmissione e ricezione per lo scambio di campioni IQ su TCP o IPC.

3.4.1 Configurazione del simulatore

Una volta installati srsLTE e il driver seguendo quanto descritto in [45] e [23], è necessario adattare i file `~/config/srslte/enb.conf` e `~/config/srslte/ue.conf` alla propria architettura di rete. Nel primo [3.17], è necessario:

- settare i campi MCC e MNC con lo stesso valore impostato nella MME (linee 3 e 4); il campo TAC non è configurabile, srsLTE usa TAC=7; è importante che nel file di configurazione della MME sia impostato lo stesso valore altrimenti il collegamento tra eNodeB e MME non verrà instaurato.
- Settare l'indirizzo dell'interfaccia S1-MME su cui ascolta la MME (linea 5);
- settare gli indirizzi locali per le interfacce S1U e S1-MME (linee 6 e 7).

Oltre a ciò, è possibile:

- abilitare o disabilitare la cattura dei pacchetti MAC tra UE ed eNodeB (linea 13) e opzionalmente anche quella dei pacchetti sull'interfaccia S1-MME (linea 15).
- Selezionare il livello di logging tra debug, info, warning, error, none (linea 19);

⁶I segnali I/Q, o dati I/Q, sono un elemento fondamentale dei sistemi di comunicazione in radiofrequenza

- abilitare o disabilitare srsGUI [44]. Quest'ultima è una libreria grafica open source per SDR. La libreria fornisce una serie di grafici utili per rappresentare numeri reali e complessi. L'installazione di srsGUI deve precedere quella di srsLTE.

```

1 [enb]
2 enb_id = 0x19B
3 mcc = MCC
4 mnc = MNC
5 mme_addr = MMESIMME_IP
6 gtp_bind_addr = S1U_LISTEN_IP
7 slc_bind_addr = SIMME_LISTEN_IP
8 n_prb = 50
9
10 ...
11
12 [pcap]
13 enable = false
14 filename = /tmp/enb.pcap
15 slap_enable = false
16 slap_filename = /tmp/enb_slap.pcap
17
18 [log]
19 all_level = info
20 all_hex_limit = 32
21 filename = /tmp/enb.log
22 file_max_size = -1
23
24 [gui]
25 enable = false
26
27 ...

```

Listing 3.17: File di configurazione enb.conf

Nel secondo file [3.18], è necessario settare i parametri dell'abbonato uguali ai valori contenuti nell'istanza MongoDB che memorizza i dati degli abbonamenti (linee da 18 a 21). È possibile applicare le stesse configurazioni descritte per il file enb.conf per quanto riguarda log, pcap e gui.

```

1 [pcap]
2 enable = false

```

```

3 filename = /tmp/ue.pcap
4 nas_enable = false
5 nas_filename = /tmp/nas.pcap
6
7 [log]
8 all_level = info
9 phy_lib_level = none
10 all_hex_limit = 32
11 filename = /tmp/ue.log
12 file_max_size = -1
13
14 [usim]
15 mode = soft
16 algo = milenage
17 opc = OPC_KEY
18 k    = SUB_KEY
19 imsi = IMSI
20 imei = IMEI
21
22 [gui]
23 enable = false
24
25 ...

```

Listing 3.18: File di configurazione ue.conf

3.4.2 Esecuzione del simulatore

Prima di avviare i componenti di rete LTE su una singola macchina, è necessario assicurarsi che UE ed EPC siano in namespace di rete diversi. Questo perché sia EPC che UE condivideranno la stessa configurazione di rete e, poiché l'UE riceve un indirizzo IP dalla sottorete dell'EPC, il kernel Linux ignorerebbe le interfacce TUN durante l'instradamento del traffico tra le estremità. Pertanto, è necessario creare uno spazio dei nomi di rete separato che l'UE utilizzerà per creare la sua interfaccia TUN. Questo problema riguarda solo UE ed EPC in quanto sono gli unici endpoint IP nella rete e devono comunicare sullo stack TCP/IP. Per creare un nuovo namespace di rete per l'UE è possibile eseguire il comando:

```
sudo ip netns add ue1
```

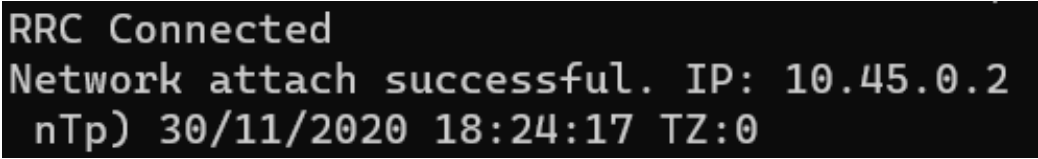
È importante specificare che, se i due componenti sono su macchine distinte, non c'è bisogno di creare alcun namespace.

Una volta configurati i due file ed eventualmente aver creato il namespace di rete per l'UE, è possibile eseguire quest'ultimo e l'eNodeB sulla stessa macchina. Per fare ciò, è necessario spostarsi nella cartella che contiene la build del progetto ed eseguire i comandi:

```
sudo ./srsenb/src/srsenb --rf.device_name=zmq --rf.device_args=
"fail_on_disconnect=true,tx_port=tcp://*:2000,rx_port=
tcp://localhost:2001,id=enb,base_srate=23.04e6"

# Omettere --gw.netns=ue1 se EPC e UE sono eseguiti su
# macchine distinte
sudo ./srsue/src/srsue --rf.device_name=zmq --rf.device_args=
"tx_port=tcp://*:2001,rx_port=tcp://localhost:2000,id=ue,
base_srate=23.04e6" --gw.netns=ue1
```

Se la procedura è andata a buon fine l'UE stamperà a video l'IP assegnatogli dall'EPC [3.13].



```
RRC Connected
Network attach successful. IP: 10.45.0.2
nTp) 30/11/2020 18:24:17 TZ:0
```

Figura 3.13: Procedura di collegamento UE andata a buon fine.

A questo punto è possibile generare traffico sul collegamento di downlink (dall'EPC verso l'UE) e di uplink (dall'UE verso l'EPC) tramite i rispettivi comandi:

```
ping 10.45.0.2
#In generale: ping <UE_IP>

sudo ip netns exec ue1 ping 10.45.0.1
#In generale: sudo ip netns exec ue1 ping <TUN_INTERFACE_IP>
```

Nel caso in cui UE ed EPC siano in esecuzione su macchine distinte, il primo comando di ping deve essere eseguito sulla macchina dell'UE, mentre sulla macchina che ospita il PGW o PGW-U, nel caso di CUPS, è necessario eseguire:

```
sudo ping 10.45.0.1
#In generale: sudo ping <TUN_INTERFACE_IP>
```

3.5 UERANSIM

UERANSIM rappresenta lo stato dell'arte per quanto riguarda i simulatori di UE e RAN (gNodeB) del 5GC. Lo scopo del progetto è quello di fornire uno strumento per testare e studiare il core di rete 5G [46]. Per questo motivo, il simulatore supporta nativamente alcuni dei più famosi software open-source per la creazione di un core 5G, come Open5GS e free5GC.

3.5.1 Configurazione del simulatore

Dopo aver effettuato la build del progetto seguendo quanto descritto in [52] è necessario modificare il file *UERANSIM/config/profile.yaml* specificando il core 5G che si intende usare. In [3.19] è mostrata una configurazione di esempio che prevede l'utilizzo di Open5GS.

```
1 # Default possible profiles are:
2 # - custom
3 # - open5gs
4 # - free5gc
5 # - havel-san
6 # You can also create unlimited number of custom
7 # profiles by creating a folder for them.
8 selected-profile: 'open5gs'
```

Listing 3.19: File di configurazione profile.yaml

In maniera simile a quanto avviene per srsLTE, è necessario configurare due file; la posizione di questi varia a seconda del core scelto; supponendo di utilizzare Open5GS: *UERANSIM/config/open5gs/gnb.yaml* e *UERANSIM/config/open5gs/ue.yaml*. Nel primo [3.20], è necessario:

- settare i campi MCC, MNC e TAC con lo stesso valore impostato nell'AMF (linee 2, 9 e 10);
- Settare l'indirizzo dell'interfaccia N2 su cui ascolta l'AMF (linea 13);
- settare l'indirizzo locale per le interfacce N2 e N3 (linea 5).

A differenza del simulatore precedente, è possibile impostare il TAC, ma non è possibile avere due interfacce distinte per i reference point N2 e N3. Se non sono state cambiate le porte nei file di configurazione del core di rete, è possibile lasciare quelle di default; in caso contrario, è necessario modificarle in accordo con i valori impostati nel core. Lo stesso discorso vale per le informazioni sugli slice di rete.

```

1 gnbId: 1
2 tac: TAC
3 nci: '0000000100'
4
5 host: N2N3LISTEN_IP
6 gtpPort: 2152
7
8 plmn:
9   mcc: MCC
10  mnc: MNC
11
12 amfConfigs:
13   - host: AMF_N2_IP
14     port: 38412
15
16 nssais:
17   - sst: '0x01'
18     sd: '0x010203'
19
20 ignoreStreamIds: true

```

Listing 3.20: File di configurazione gnb.yaml

La configurazione del file [3.21], è pressoché identica a quella di srsLTE: il SUPI è l'equivalente dell'IMSI e anziché l'OPc è necessario fornire l'OP; è importante che il valore contenuto in MongoDB sia marcato come OP e non OPc. Oltre a ciò, è necessario specificare i valori di MCC e MNC. L'Authentication Management Field, AMF, è un numero a 4 cifre esadecimale utilizzato nella procedura AKA [5.1.6] e deve avere lo stesso valore presente in MongoDB (8000 è il valore di default).

```

1 key: SUB_KEY
2 op: OP_KEY
3 amf: '8000'
4 imei: IMEI
5 supi: SUPI
6 plmn:
7   mcc: MCC
8   mnc: MNC
9
10 smsOverNasSupported: true
11 dnn: 'internet'

```

```

12
13 requestedNssai:
14   - sst:
15     value: 1
16   sd:
17     hex: '010203'

```

Listing 3.21: File di configurazione ue.yaml

3.5.2 Esecuzione del simulatore

Per eseguire il simulatore è necessario utilizzare lo script *UERANSIM/run.sh*; fatto ciò è possibile scegliere diverse opzioni:

1. initial-registration
2. periodic-registration
3. de-registration
4. pdu-session-establishment.

Dopo aver stabilito una sessione PDU con il core di rete, è possibile reindirizzare il traffico di un'applicazione sul tunnel GTP-U creato generando traffico in downlink e in uplink [51] [53]. Per fare ciò, è necessario configurare un'interfaccia TUN/TAP:

```

# Crea un dispositivo TUN/TAP
sudo ip tuntap add name uesimtun mode tun
# Associa l'IP dell'UE al dispositivo. L'IP assegnato all'UE è
# visualizzabile nel file UERANSIM/logs/ue-imsi-xxxxxxxxxxxxxxxx.log
sudo ip addr add {IP} dev uesimtun
# Avvia il dispositivo
sudo ip link set uesimtun up
# -- CONFIGURAZIONE ROUTING --
sudo nano /etc/iproute2/rt_tables
# Inserire '1453 uesimtable' in fondo al file
sudo ip rule add from {IP} table uesimtable
sudo ip route add default dev uesimtun table uesimtable

```

Dopodiché reindirizzare il traffico attraverso questa:

```
sudo UERANSIM/build/tun-agent
sudo UERANSIM/build/ue-binder.sh {IP} {COMMAND} {ARGS}
# UERANSIM/build/ue-binder.sh 10.45.0.2 curl google.com
# UERANSIM/build/ue-binder.sh 10.45.0.2 firefox
```

Capitolo 4

Orchestrazione di EPC e 5GC virtualizzati

In questo capitolo verrà descritta la parte principale dell'attività svolta con l'obiettivo di orchestrare, studiare e testare i core di rete: EPC e 5GC. L'implementazione proposta sfrutta tutti gli strumenti descritti nei capitoli precedenti, in particolare:

- Open Source MANO viene utilizzato come piattaforma di orchestrazione;
- OpenStack viene utilizzato come piattaforma Cloud e come VIM;
- Open5GS implementa i core EPC e 5GC;
- srsLTE e UERANSIM sono i simulatori di UE e RAN per testare rispettivamente EPC e 5GC;
- Cloud-init e Juju supportano rispettivamente la configurazione day 0 e day 1. La parte di monitoraggio (day 2) non viene trattata dall'elaborato.

4.1 Ambiente di esecuzione

Due piattaforme OpenStack sono in esecuzione su quattro server rack dotati ciascuno di 62 GB di RAM, 2 CPU Intel Xeon E5-2640 v4 aventi 10 core e 20 thread e Ubuntu 18.04 come sistema operativo. Le macchine fisiche e le piattaforme OpenStack hanno connettività diretta con tutte le reti raffigurate in [\[4.1\]](#).

Nelle piattaforme OpenStack sono state predisposte delle reti, corrispondenti a quelle fisiche, logicamente condivise tra tutti i tenant in modo da avere connettività tra i cluster e l'esterno:

- **external** network 10.250.0.0/16;
- **transportNet** network 10.102.0.0/16 (corrisponde alla Data network 1);
- **mgmtNet** network 10.15.0.0/16;

Siccome tutte le reti hanno il DHCP abilitato per l'assegnazione automatica degli IP, opportuni pool di indirizzi sono stati configurati in modo da non generare conflitti tra V.M appartenenti a cluster differenti e tra V.M e macchine fisiche.

Le modalità di utilizzo di queste reti verranno descritte nelle sezioni successive.

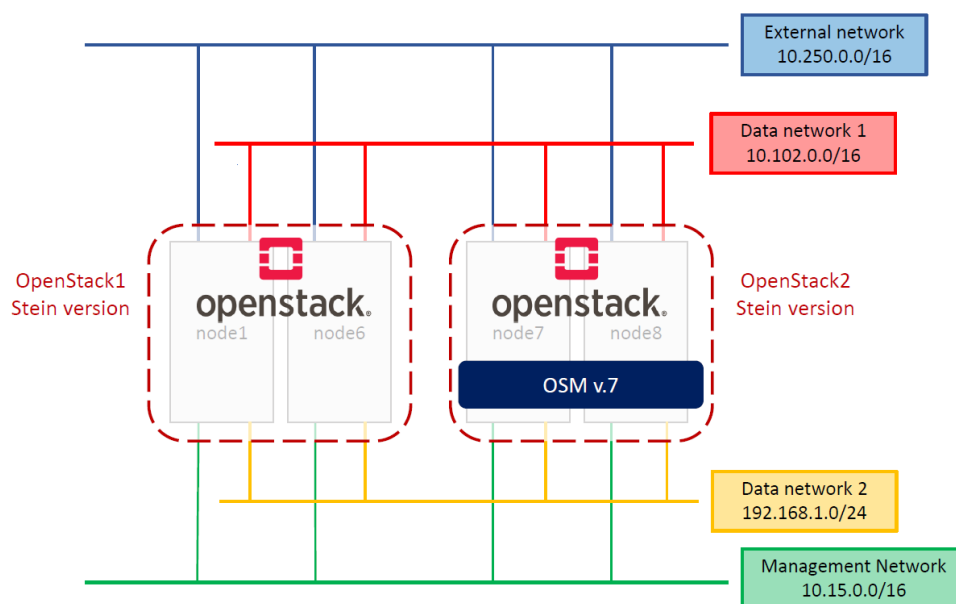


Figura 4.1: Cluster OpenStack

OSM e Juju sono in esecuzione su una V.M sull'istanza OpenStack2. L'IP di tale macchina sulla mgmtNet è 10.15.0.100.

4.2 Deployment manuale di analisi

In fase preliminare, al fine di acquisire dimestichezza con le diverse configurazioni, capire il grado di modularità dei componenti dei core di rete e analizzare il traffico tra questi, il deployment è avvenuto in maniera manuale. In particolare, ogni componente dei core di rete 4G e 5G è stato eseguito su una macchina virtuale dedicata con 1 CPU virtuale, 2 GB di RAM e 20 GB di disco. Visto il numero di V.M da creare, 12, per distribuire il carico sono state utilizzate entrambe le istanze di OpenStack. Tutte le V.M sono state collegate alla stessa rete, *external*, e i file di configurazione presentati in [3.3], [3.4] e [3.5] sono stati modificati in maniera tale che i diversi componenti potessero comunicare. L'architettura di deployment è mostrata in [4.2]. Le interfacce della SBI sono evidenziate in blu; la porta di ascolto è la TCP 7777, per default, mentre la porta usata per trasmettere è scelta dinamicamente.

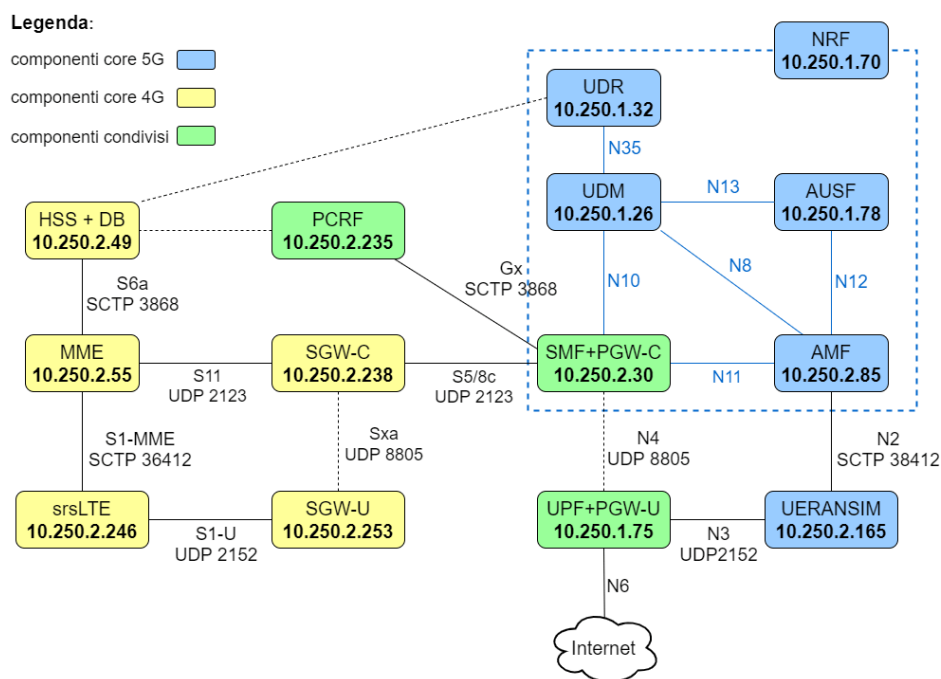


Figura 4.2: Architettura deployment manuale di analisi

Ogni V.M possiede un gruppo di protezione di default che permette l'uscita di qualsiasi tipo di traffico IPv4 e IPv6 su qualsiasi porta e l'ingresso di traffico UDP, TCP e ICMP IPv4 sempre su qualsiasi porta. Affinché le componenti collegate tramite interfacce basate sul protocollo SCTP (S1-MME, S6a....) possano comunicare, è necessario aggiungere tra i gruppi di

protezione della V.M una regola che consenta al traffico SCTP in ingresso di entrare. Per fare ciò è necessario creare un nuovo gruppo di protezione su OpenStack, *enable_sctp*, e aggiungergli la regola raffigurata in [4.3]. Tutte le V.M che possiederanno, tra i gruppi di protezione, *enable_sctp* lasceranno passare il traffico SCTP in ingresso. Per consentire all'UE di navigare, è necessario abilitare il port forwarding e il natting, nella macchina che ospita UPF+PGW-U, della classe dei possibili indirizzi assegnabili all'UE:

```
sudo sh -c "echo 1 > /proc/sys/net/ipv4/ip_forward"
sudo iptables -t nat -A POSTROUTING -s 10.45.0.0/16 ! -o ogstun
-j MASQUERADE
#In generale: sudo iptables -t nat -A POSTROUTING -s {IP_CLASS} !
# -o ogstun -j MASQUERADE
```

The screenshot shows a web interface for adding a firewall rule. The form fields are as follows:

- Regola:** Altro protocollo
- Descrizione:** (empty text area)
- Direzione:** Ingress
- Protocollo IP:** 132
- Remoto:** Gruppo di sicurezza
- Gruppo di sicurezza:** enable_sctp (corrente)
- Tipo etere:** IPv4

The 'Descrizione:' section contains the following text:

Le regole definiscono quale traffico è consentito alle istanze assegnate al gruppo di sicurezza. Una regola del gruppo di sicurezza consiste di tre parti principali.

Regola: È possibile specificare il template di regola desiderato o utilizzare regole personalizzate. Le opzioni sono Regola TCP personalizzata, Regola UDP personalizzata o Regola ICMP personalizzata.

Apri porta/intervallo porte: Per le regole TCP e UDP è possibile scegliere di aprire una porta singola o una serie di porte. Selezionando l'opzione "Serie di porte" all'utente verrà fornito lo spazio per entrambe le porte iniziale e finale della serie. Invece per le regole ICMP specificare un tipo e un codice ICMP negli spazi forniti.

Remoto: È necessario specificare l'origine del traffico da consentire tramite questa regola. L'utente può eseguire questa specifica nella forma di un blocco di indirizzo IP (CIDR) o tramite un gruppo di origine (Gruppo di sicurezza). Selezionando un gruppo di sicurezza come origine, sarà possibile per tutte le altre istanze di quel gruppo di sicurezza accedere a qualsiasi altra istanza in base a questa regola.

Figura 4.3: Regola per abilitare il traffico SCTP in ingresso

Questo deployment rende più semplice la cattura e l'analisi del traffico. In particolare, tramite `tcpdump`¹ è possibile catturare i pacchetti che fluiscono attraverso ciascuna macchina e analizzare le diverse procedure: collegamento dell'eNodeB alla MME, collegamento dell'UE all'EPC senza e con contesto memorizzato, abbattimento del collegamento tra UE ed EPC, collegamento

¹Tool a riga di comando per il debugging delle reti.

del gNodeB all'AMF, registrazione dell'UE al 5GC e collegamento dell'UE al 5GC. Per il core 5G non è stata analizzata la procedura di abbattimento del collegamento tra UE e 5GC in quanto non ancora implementata in UERANSIM. Il risultato dell'analisi è contenuto in [5.1].

4.3 Deployment automatizzato

4.3.1 Architettura di deployment

L'architettura pensata per il deployment automatizzato è raffigurata in [4.4]. In particolare, si è scelto di: raggruppare il più possibile tra loro le entità che svolgono un compito comune, tenere separati i gateway del piano di controllo da quelli del piano dati, suddividere il traffico tra reti differenti. Il core verrà messo in esecuzione su una singola istanza di OpenStack: OpenStack2.

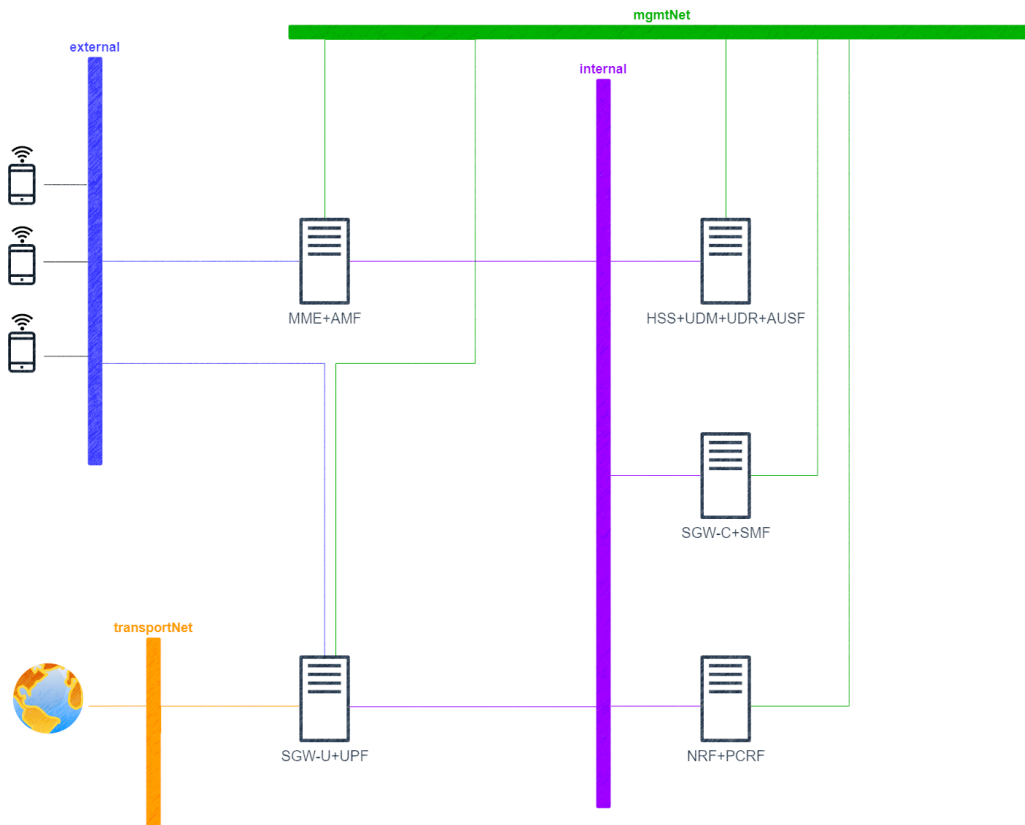


Figura 4.4: Architettura ad alto livello deployment automatizzato

Nel dettaglio, le funzioni aggregate tra loro sono:

- accesso al core - MME e AMF;
- gateway per il traffico di controllo - SGWC e SMF (PGWC);
- gateway per il traffico utente - SGWU e UPF (PGWU);
- autenticazione - HSS, UDM, UDR, AUSF
- service discovery e gestione tariffazione - NRF e PCRF.

Le reti previste sono:

- external. Trasporta il traffico SCTP delle interfacce S1-MME e N2 e il traffico UDP delle interfacce S1-U e N3;
- transportNet. Trasporta il traffico dell'interfaccia N6 da/verso la PDN, in questo caso Internet;
- mgmNet. Garantisce l'accesso alle macchine, per la loro gestione, da parte del fornitore dell'infrastruttura e/o del servizio;
- internal. Trasporta il traffico delle restanti interfacce che interconnettono i diversi componenti.

Lo scenario proposto è puramente pensato per studiare e comprendere le possibilità offerte dai vari strumenti e non è paragonabile ad uno scenario reale. Tuttavia, nonostante la sua semplicità, non preclude la possibilità di ulteriori sviluppi futuri come:

- differenziare la rete di management per il fornitore dell'infrastruttura e quello del servizio;
- posizionare i gateway del piano utente sull'istanza OpenStack1 in modo da simulare la messa in esecuzione in un edge datacenter più vicino alla RAN;
- utilizzare una rete dedicata per il traffico PFCP tra gateway di controllo e gateway utente.

A partire dall'architettura proposta, è stata ricavata quella raffigurata in [4.5] che introduce i concetti di OSM. Ogni entità vista nell'architettura precedente corrisponde ad una VNF a singola VDU; per questo motivo le interfacce di quest'ultima (quadrantini gialli in figura) possiedono un mapping diretto e univoco con i connection point della VNF (quadrantini rossi in figura). In ciascuna VDU sono in esecuzione i diversi servizi del core di rete. Una

scelta alternativa è quella di creare VDU differenti, all'interno della stessa VNF, ognuna delle quali esegue un solo componente del core. Un vantaggio di questa soluzione è quello di poter dimensionare in maniera diversa e più adeguata, le risorse computazionali dedicate a ciascuna entità del core; per contro aumenta la complessità dei descrittori VNFD. Per il caso di studio in esame è sufficiente utilizzare una singola VDU.

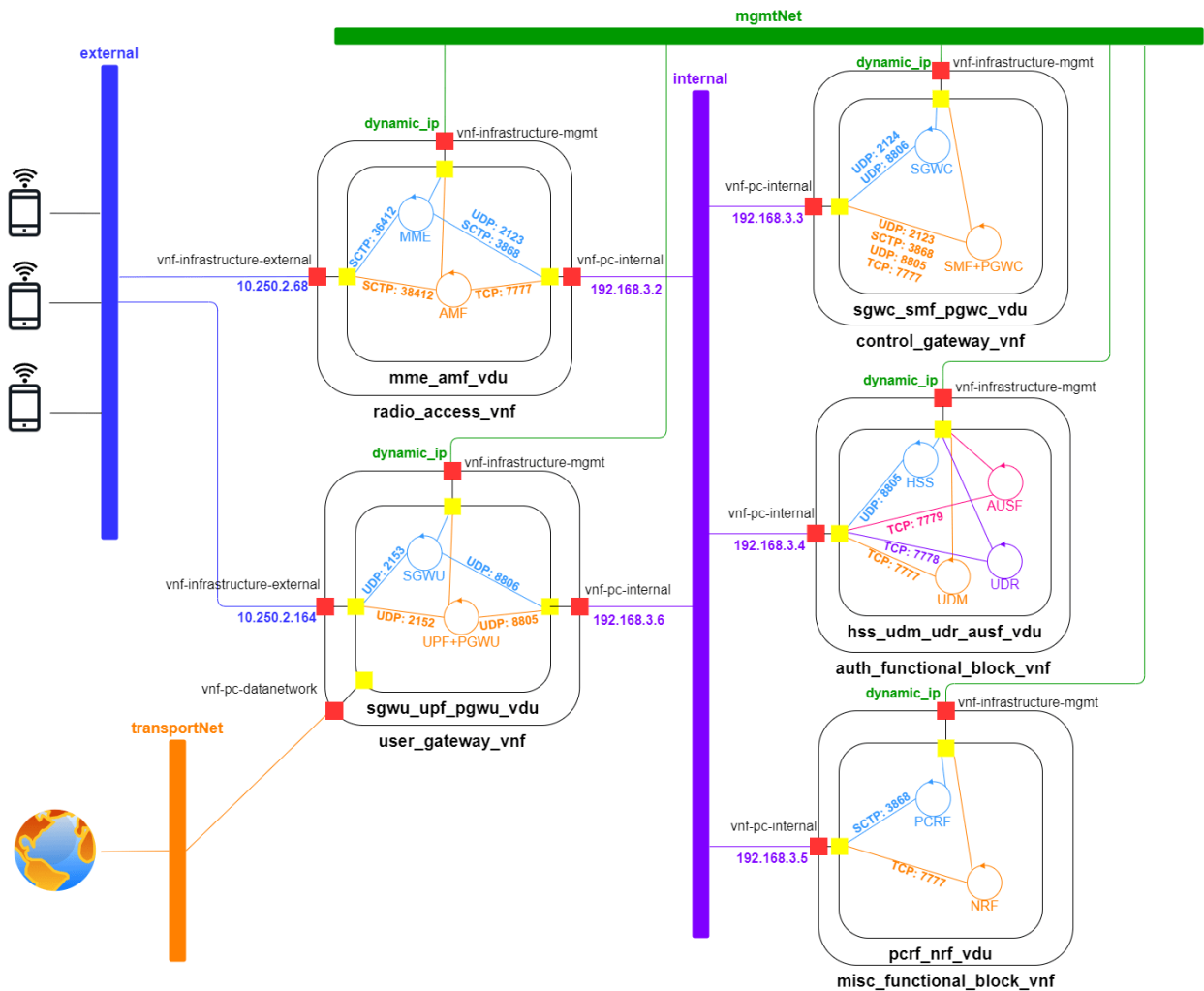


Figura 4.5: Architettura dettagliata deployment automatizzato

È importante notare che più servizi condividono la stessa interfaccia di rete; per questo motivo, al fine di evitare conflitti, sono state cambiate alcune porte di ascolto di default: le TCP per il reference point SBI di UDR e AUSF in *hss_udm_udr_ausf_vdu*, le UDP per i reference point S11 e Sxa di SGW-C in *sgwc_smf_pgwc_vdu*, le UDP per i reference point Sxa e S1-U di SGW-U in

sgwu_upf_pgwu_vdu. Un'alternativa più complessa è quella di aggiungere più interfacce sulla stessa rete.

Gli indirizzi IP sulle reti *external* e *internal* sono statici in quanto devono essere conosciuti a priori per la configurazione di day 1.

4.3.2 Creazione dell'immagine di Open5GS per Open-Stack

L'immagine per il deployment è stata creata a partire da una Ubuntu 18.04. Una volta istanziata una V.M con quest'ultima, è stato installato il software necessario: Open5GS e yq. Quest'ultimo serve per processare file YAML tramite CLI e verrà utilizzato per la configurazione di day 1 [47]. Il metodo di installazione utilizzato per yq è quello basato sui pacchetti Debian; l'installazione tramite pacchetti snap richiede un workaround per poter modificare i file tramite *sudo*. Per poter automatizzare l'inserimento degli utenti abbonati nell'istanza MongoDB, è necessario modificare lo script *open5gs-dbctl* in modo da poter scegliere se il codice operatore inserito è di tipo OP oppure OPc e specificare l'IP del PGW-U nelle impostazioni dell'APN. Per evitare che un aggiornamento di Open5GS sovrascriva lo script, è necessario crearne uno nuovo, *my-open5gs-dbctl*, a partire dall'originale. Lo script creato è contenuto in */usr/bin* in modo da poter usare direttamente il comando *textitmy-open5gs-dbctl*. Il codice della funzionalità aggiunta è contenuto in [A.2].

Infine, è buona pratica disabilitare l'avvio automatico di tutti i demoni di Open5GS e MongoDB; in questo modo, dopo la configurazione di day 1 in ciascuna macchina saranno attivi solo i demoni dei servizi da essa forniti.

4.3.3 VNFD e configurazione day 0

Come si evince dalla figura [4.5] è necessario implementare cinque descrittori VNFD e un descrittore NSD per seguire l'intero deployment. Di seguito non verranno introdotti tutti i descrittori implementati, che sono contenuti in [A.1], ma verrà discusso solamente il VNFD associato alla VNF *user_gateway_vnf* e il NSD. Come descritto in [3.1.2], i descrittori sono contenuti in pacchetti; la struttura del pacchetto e il template del descrittore possono essere generati automaticamente tramite l'apposito strumento offerto da OSM [50].

La parte iniziale del descrittore contiene le informazioni necessarie per la sua identificazione in modo da poterlo referenziare nel NSD.

1 `vnfd:vnfd-catalog:`

```

2     vnfd:
3     -   id: user_gateway_vnfd
4         name: user_gateway_vnfd
5         short-name: user_gateway_vnfd
6         description: Generated by OSM package generator
7         vendor: OSM
8         version: '1.0'
9         ...

```

La parte sottostante specifica l'interfaccia di management che, non solo permette al fornitore dell'infrastruttura e/o del servizio di accedere alla macchina per gestirla, ma consente ciò anche a Juju per le successive configurazioni di day 1 e day 2. Oltre a ciò, vengono specificati i connection point esposti al NS per l'interconnessione con le altre VNF. Ogni connection point può contenere il campo **"port-security-enabled: false"** per disabilitare il filtraggio dei pacchetti.

```

1     mgmt-interface:
2         cp: vnf-infrastructure-mgmt
3     connection-point:
4     -   name: vnf-infrastructure-mgmt
5         id: vnf-infrastructure-mgmt
6         type: VPORT
7     -   name: vnf-infrastructure-external
8         id: vnf-infrastructure-external
9         type: VPORT
10    -   name: vnf-pc-internal
11        id: vnf-pc-internal
12        type: VPORT
13    -   name: vnf-pc-datanetwork
14        id: vnf-pc-datanetwork
15        type: VPORT
16    ...

```

La VNF è composta da una sola VDU di cui è necessario specificare: i parametri identificativi (linee da 2 a 5), il flavour, ovvero le risorse computazionali e di memorizzazione da assegnare alla macchina (linee da 8 a 10), l'immagine che deve essere presente tra le immagini memorizzate in OpenStack (linea 12), il nome del file di cloud init (linea 13) e le interfacce di rete (linee da 18 a 37). Queste ultime sono associate ai connection point precedentemente dichiarati. Come già detto in precedenza, nello scenario preso in esame non sono presenti connection point interni alle VNF, ma solo

esterni. Le linee 14 e 15 abilitano l'utilizzo di un supporto di configurazione su OpenStack; nel caso in esame, l'assenza di tali fa sì che il file di cloud-init non venga utilizzato.

```
1      vdu:
2      -   id: sgwu_upf_pgwu_vdu
3          name: sgwu_upf_pgwu_vdu
4          description: sgwu_upf_pgwu_vdu
5          count: 1
6
7      vm-flavor:
8          vcpu-count: 1
9          memory-mb: 2048
10         storage-gb: 20
11
12         image: 'Open5GS'
13         cloud-init-file: cloud-config
14         supplemental-boot-data:
15             boot-data-drive: true
16
17         interface:
18         -   name: eth0
19             type: EXTERNAL
20             virtual-interface:
21                 type: PARAVIRT
22             external-connection-point-ref:
23                 vnf-infrastructure-mgmt
24         -   name: eth1
25             type: EXTERNAL
26             virtual-interface:
27                 type: PARAVIRT
28             external-connection-point-ref:
29                 vnf-infrastructure-external
30         -   name: eth2
31             type: EXTERNAL
32             virtual-interface:
33                 type: PARAVIRT
34             external-connection-point-ref:
35                 vnf-pc-internal
36         -   name: eth3
37             type: EXTERNAL
```

```

38         virtual-interface:
39             type: PARAVIRT
40         external-connection-point-ref:
41         vnf-pc-datanetwork
42         ...

```

Il file cloud-init si occupa di associare una password all'utente di default *ubuntu*, abilitare l'accesso ssh con password e specificare le interfacce di rete della macchina. Il numero di queste è strettamente legato al numero di quelle specificate nel descrittore. Per semplicità, la password è hardcoded, ma può essere parametrizzata e passata a tempo di creazione. Nel caso in esame, i cloud-init delle varie VNF differiscono solo per il numero di interfacce specificate.

```

#cloud-config
manage_etc_hosts: true
password: ubuntu
chpasswd: { expire: False }
ssh_pwauth: True
network:
  version: 2
  ethernets:
    ens3:
      dhcp4: true
      set-name: ens3
    ens4:
      dhcp4: true
      set-name: ens4
    ens5:
      dhcp4: true
      set-name: ens5
    ens6:
      dhcp4: true
      set-name: ens6

```

Infine, se è prevista la fase di day 1 il descrittore conterrà una sezione dedicata in cui vengono specificate le configurazioni da eseguire e i relativi parametri. La primitiva config è obbligatoria e serve per passare a Juju i parametri per la connessione ssh; in particolare: nome utente, password e ip sulla rete di management della macchina. Questo parametro è fornito automaticamente da OSM. Le primitive successive, numero di sequenza 2 e 3, specificano la necessità di configurare i servizi del core e forniscono in input indirizzi e porte

per i vari reference point. I parametri in questo caso sono hardcoded, ma è possibile fornirli con semplicità a tempo di inizializzazione. Le ultime due primitive, come si evince dal nome, sottolineano la necessità di configurare il natting e avviare i servizi precedentemente configurati.

```
1 vnf-configuration:
2   initial-config-primitive:
3     - seq: '1'
4       name: config
5       parameter:
6         - name: ssh-hostname
7           value: <rw_mgmt_ip>
8         - name: ssh-username
9           value: ubuntu
10        - name: ssh-password
11          value: ubuntu
12     - seq: '2'
13       name: configure-sgwu
14       parameter:
15         - name: gtpu-ip
16           data-type: STRING
17           value: '192.168.3.6'
18         - name: gtpu-port
19           data-type: STRING
20           value: '2153'
21         # ... Altri parametri ...
22     - seq: '3'
23       name: configure-upf-pgwu
24       parameter:
25         - name: gtpu-ip
26           data-type: STRING
27           value: '192.168.3.6'
28         # ... Altri parametri ...
29     - seq: '4'
30       name: add-nat-rule
31     - seq: '5'
32       name: start-services
33
34   juju:
35     charm: sgwuupfpgwucharm
```

4.3.4 Configurazione day 1

Al fine di automatizzare completamente il deployment è necessario: configurare, in ogni file `.yaml` e `.conf`, indirizzi e porte per ciascun reference point, settare il TAC nella MME a 7 affinché sia possibile collegarsi mediante srsLTE, registrare un abbonamento nel DB fornendone i dati, abilitare port forwarding e natting, nella macchina che ospita UPF+PGW-U della classe dei possibili indirizzi assegnabili all'UE e avviare i demoni dei componenti Open5GS opportuni. Anche in questo caso non è possibile introdurre in maniera completa ed esaustiva il codice; si cercheranno di descriverne gli aspetti fondamentali. Per prima cosa è necessario installare Juju:

```
snap install charm --classic
```

Una volta fatto ciò, è possibile creare il charm tramite il comando:

```
charm create <charmname>
```

A questo punto, verrà creata una cartella associata al charm contenente diverse sottocartelle e file. Questi ultimi vanno opportunamente modificati; di seguito è mostrato un esempio che fa riferimento alla VNF precedente, `user_gateway_vnf`.

1. Modificare il file `layer.yaml` includendo il layer di base e quello per collegarsi alla VNF tramite ssh;

```
1 includes:
2   - layer:basic
3   - layer:vnfproxy
```

2. modificare il file `metadata.yaml`;

```
1 name: sgwuupfpgwucharm
2 summary: Configuration of sgwu and upf+pgwu
   ↳ components
3 maintainer: Daniele Rossi <daniele.rossi18@studio.
   ↳ unibo.it>
4 description: |
5   Configuration of sgwu and upf+pgwu components
6 tags:
7   - misc
8   - osm
9   - vnf
10  - nfv
```

```

11 series:
12   - trusty
13   - xenial
14 subordinate: false

```

3. creare e modificare il file actions.yaml. Come detto in precedenza, le varie configurazioni da eseguire sulla VNF vengono mappate in azioni Juju. Quindi, in questo file sono contenute le configurazioni da eseguire e per ciascuna sono specificati i parametri in input. Queste configurazioni sono un soprainsieme di quelle inserite nel descrittore [4.3.3], in quanto viene fornita la possibilità di configurare un ampio set di parametri, anche se non è detto che ci sia la necessità di configurarli tutti.

```

1 configure-sgwu:
2   description: Configures sgwu.yaml file
3   params:
4     gtpu-ip:
5       description: IP for S1-U, S5/8u (SERVER)
6       type: string
7       default: 0.0.0.0
8     gtpu-port:
9       description: Port for S1-U, S5/8u (SERVER)
10      type: string
11      default: '-1'
12      # ... Altri parametri ...
13
14 configure-upf-pgwu:
15   description: Configures upf.yaml file
16   params:
17     gtpu-ip:
18       description: IP for S5/8u, N3 (SERVER)
19       type: string
20       default: 0.0.0.0
21     gtpu-port:
22       description: Port for S5/8u, N3 (SERVER)
23       type: string
24       default: '-1'
25     # ... Altri parametri ...
26
27 add-nat-rule:

```

```

28   description: Enables IPv4 forwarding and add a
        ↪ NAT rule for UEs ip subnet
29
30 start-services:
31   description: Starts sgwc and smf+pgwc services

```

4. Creare la cartella `actions` e, dentro a questa, un file eseguibile per ciascuna azione (ogni file deve avere lo stesso nome dell'azione a cui è associato) contenente il codice in [A.3.1]. A ciascun file devono essere dati i permessi di esecuzione tramite `chmod`.
5. Implementare le funzioni di configurazione nel file `reactive/sgwuupfpg-wucharm.py` la cui struttura è mostrata in [A.3.2]. Gli aspetti salienti dell'implementazione sono contenuti in [4.3.4].
6. Creare la build del charm tramite il comando `sudo -E charm build` e copiare l'output nella cartella `charms` del pacchetto associato alla VNF.

Implementazione azioni

Tramite `charms.sshproxy._run(cmd)` è possibile eseguire un comando bash sulla macchina da configurare. Le principali azioni da eseguire sono elencate all'inizio della sezione e verranno approfondite in seguito.

Modifica indirizzi e porte nei file YAML Al fine di modificare i file YAML si è scelto di utilizzare `yq`, precedentemente introdotto, anziché il comando Unix `sed` utilizzato nella guida OSM [32]. Questa scelta è dovuta ad una questione di flessibilità. Prendiamo come esempio il listato sottostante e supponiamo di voler cambiare tramite `sed` l'indirizzo a riga 5.

```

1 sgwu:
2   gtpu:
3     - addr: 127.0.0.6
4   pfcp:
5     - addr: 127.0.0.6
6 sgwc:
7   pfcp:
8     - addr: 127.0.0.3

```

Il comando più banale, `sed 's/127.0.0.6/192.168.3.6/'`, non funzionerebbe perché cambierebbe anche l'indirizzo a riga 3. Una possibile soluzione è quella di utilizzare i range; in tal caso, il comando diventerebbe

```
sed '/pfcp:\/,- addr:/s/127.0.0.6/192.168.3.6/'
```

Questa soluzione è strettamente legata sia alla struttura dello YAML che all'IP di default, 127.0.0.6. Ad esempio, se l'IP di default diventa 127.0.0.7 il comando non funziona più; allo stesso modo, se ho una lista anziché un indirizzo succede questo.

Utilizzando la versione estesa delle regex sed tramite il flag -r o -E, è possibile specificare un'espressione regolare che descriva un generico indirizzo IP. In questo caso verrebbe cambiato anche l'indirizzo a riga 9.

```
sed -r '/pfcp:\/,- addr:/s/(\b[0-9]{1,3}\.){3}[0-9]{1,3}\b/192.168.3.6/'
```

Per questi motivi si è scelto di utilizzare yq e legare la configurazione esclusivamente alla struttura dello YAML. Cambiare l'indirizzo a riga 5 si traduce in un comando di questo tipo:

```
yq sgwu.pfcp[0].addr "192.168.3.6"
```

La struttura più comune nel file YAML di configurazione è quella sopra riportata, in cui si ha il componente a livello 0 e il protocollo a livello 1 che contiene una lista di elementi, tipicamente indirizzo e porta. Vi sono casi in cui, il protocollo contiene solo l'indirizzo:

```
1 smf:
2   pfcp:
3     addr: 127.0.0.3
```

oppure l'indirizzo è a sua volta una lista, ad esempio:

```
1 nrf:
2   sbi:
3     - addr:
4       - 127.0.0.10
5       - ::1
6     port: 7777
```

In questi casi, i rispettivi comandi yq diventerebbero:

```
yq smf.pfcp.addr "192.168.3.3"
yq smf.pfcp[0].addr[0] "192.168.3.5"
```

Per il cambio della porta, invece, è sufficiente un comando del tipo:

```
yq <componente>.<protocollo>[0].port "<portaScelta>"
```

Generalizzando, nel listato sottostante è riportato il codice che permette di cambiare indirizzo e porta nei file YAML di configurazione. Tale codice, fa riferimento alla configurazione in [4.3.4], ma è facilmente applicabile a tutti i casi visti in precedenza. Non è riportata la gestione delle eccezioni per una questione di semplicità.

```

1  ...
2
3  changeAddressInYaml = 'sudo yq w {filePath} {component}.{protocol}{
    ↪ arrayElementOrNothingProtocol}.addr{arrayElementOrNothingAddr} "{
    ↪ address}" -i'
4  changePortInYaml = 'sudo yq w {filePath} {component}.{protocol}[0].port "{
    ↪ port}" -i'
5
6  @when("actions.configure-sgwu")
7  def configure_sgwu():
8
9      # (component, protocol) : (address, isProtocolAList, isAddressAList,
    ↪ port)
10     componentConfiguration = {
11         ("sgwu", "gtpu"): (function_get("gtpu-ip"), True, False,
    ↪ function_get("gtpu-port")),
12         ("sgwu", "pfcg"): (function_get("pfcg-ip"), True, False,
    ↪ function_get("pfcg-port")),
13         ("sgwc", "pfcg"): (function_get("sgwc-ip"), True, False,
    ↪ function_get("sgwc-port"))
14     }
15     filePath = "/etc/open5gs/sgwu.yaml"
16     arrayElementOrNothingProtocol = "[0]" if addressInfo[1] else ""
17     arrayElementOrNothingAddr = "[0]" if addressInfo[2] else ""
18
19     for componentAndProtocol, addressInfo in fieldToAddressInfo.items():
20         charms.sshproxy._run(
21             changeAddressInYaml.format(
22                 filePath=confFilePath,
23                 component=componentAndProtocol[0],
24                 protocol=componentAndProtocol[1],
25                 arrayElementOrNothingProtocol=arrayElementOrNothingProtocol,
26                 arrayElementOrNothingAddr=arrayElementOrNothingAddr,
27                 address=addressInfo[0],
28             )
29         )
30
31     if int(addressInfo[3]) > 0:
32         charms.sshproxy._run(
33             changePortInYaml.format(
34                 filePath=confFilePath,
35                 component=componentAndProtocol[0],
36                 protocol=componentAndProtocol[1],
37                 port=addressInfo[3],
38             )
39         )
40
41     remove_flag("actions.configure-sgwu")
42
43     ...

```


Le linee 3 e 4 rappresentano i comandi discussi in precedenza sotto forma di stringa parametrizzabile.

Le linee dalla 9 alla 13 sono gli elementi di una mappa che rappresentano le configurazioni da effettuare. Ad ogni coppia (componente, protocollo) sono associati l'indirizzo IP, due booleani che indicano, rispettivamente, se il protocollo e l'indirizzo sono una lista di elementi e la porta. Indirizzo e porta sono parametri e possono essere recuperati tramite la funzione `function_get(<nome_parametro>)`. Il `foreach` scorre ogni elemento della mappa e per ciascuno viene eseguito il cambio di indirizzo, linee dalla 20 alla 29, e nel caso in cui nel descrittore sia stata specificata la porta, il cambio di questa. Se nel descrittore non è stato passato nessun valore per la porta questa avrà valore -1 [3].

La modifica del TAC nella configurazione della MME è effettuata tramite lo stesso comando e simile a quanto appena trattato. Per questo motivo non verrà mostrato il codice per fare ciò.

Modifica indirizzi nei file conf Per la modifica degli indirizzi nei file `.conf` si utilizza il comando `sed`. Ricordiamo che la struttura delle linee da modificare è la seguente:

```
1 # File hss.conf
2 ListenOn = "127.0.0.8";
3 ConnectPeer = "mme.localdomain" { ConnectTo =
  "127.0.0.2"; No-TLS; };
```

La sostituzione può avvenire tramite il comando:

```
sed 's/127.0.0.8/192.168.3.4/'
```

oppure, in maniera più generale, tramite la versione estesa delle regex `sed`:

```
sed -i -r 's/^ListenOn = "(\\b[0-9]{1,3}\\.){3}[0-9]{1,3}\\b"/
ListenOn = "192.168.3.4"/'
```

Prendendo come riferimento la versione semplice del comando, il codice python che implementa l'azione è contenuto in [A.3.3].

Aggiunta abbonamento a MongoDB Per aggiungere un abbonato basta utilizzare lo script citato in precedenza, `my-open5gs-dbctl`. Il codice python di aggiunta di un abbonato è contenuto in [A.3.3]

Abilitazione port forwarding e natting In questo caso, è sufficiente eseguire due volte il comando `charms.sshproxy._run` passandogli come parametri i comandi contenuti in [4.2].

Avvio demoni dei componenti Open5GS In maniera simile al caso precedente, è sufficiente passare a `charms.sshproxy._run` il comando `sudo systemctl start <demone1> <demone2>`

Ad esempio:

```
sudo systemctl start open5gs-sgwud open5gs-upfd
```

4.3.5 NSD, on-boarding e deployment

Una volta che i pacchetti delle VNF sono pronti è necessario preparare quello associato al NS. Anche in questo caso è possibile utilizzare il tool OSM per generare automaticamente la struttura del pacchetto e del descrittore.

La parte iniziale del descrittore contiene le informazioni identificative e la lista delle VNF che lo compongono:

```
1 nsd:nsd-catalog:
2   nsd:
3     - id: Open5GS_nsd
4     name: Open5GS_nsd
5     short-name: Open5GS_nsd
6     description: Generated by OSM package generator
7     vendor: OSM
8     version: '1.0'
9
10    constituent-vnfd:
11      - member-vnf-index: 1
12        vnfd-id-ref: radio_access_vnfd
13      # ... Altri riferimenti ...
14      - member-vnf-index: 5
15        vnfd-id-ref: misc_functional_block_vnfd
16      ...
```

A questo punto è possibile specificare come sono connesse le VNF appartenenti al servizio tramite i Virtual Link Descriptor. Ogni VLD contiene la lista dei connection point, esposti dalle VNF, che collega. Inoltre, i VLD possono essere associati a una rete già esistente all'interno di OpenStack (ad esempio, i VLD `mgmt_net`, `external_net` e `transport_net`) oppure no. In quest'ultimo caso, viene generata automaticamente una rete virtuale interna (ad esempio, il VLD `internal_net`). Per poter fissare gli indirizzi sulla rete interna è necessario associarle un profilo IP (linea 17) che deve essere opportunamente definito (linee da 26 a 32).

```
1 vld:
```

```

2     -   id: mgmt_net
3         name: mgmt_net
4         short-name: mgmt_net
5         type: ELAN
6         mgmt-network: 'true'
7         vim-network-name: mgmtNet
8         vnfd-connection-point-ref:
9         -   member-vnf-index-ref: 1
10            vnfd-id-ref: radio_access_vnfd
11            vnfd-connection-point-ref: vnf-
12                ↔ infrastructure-mgmt
13            ...
14         # ... VLD di external_net e transport_net ...
15     -   id: internal_net
16         name: internal_net
17         ip-profile-ref: ip-internal
18         short-name: internal_net
19         type: ELAN
20         vnfd-connection-point-ref:
21         -   member-vnf-index-ref: 1
22            vnfd-id-ref: radio_access_vnfd
23            vnfd-connection-point-ref: vnf-pc-internal
24            ip-address: 192.168.3.2
25         ...
26 ip-profiles:
27 -   ip-profile-params:
28     dhcp-params:
29         enabled: 'true'
30         ip-version: ipv4
31         subnet-address: 192.168.3.0/24
32     name: ip-internal

```

Completata la stesura del NSD, è possibile comprimere tutte le cartelle in formato *.tar.gz* ed effettuare l'on-boarding tramite CLI o WebGUI. Nel primo caso è necessario eseguire i comandi:

```

osm vnfd-create <VNF_packet_name>.tar.gz
osm nsd-create <NSD_packet_name>.tar.gz

```

Nel secondo caso è sufficiente collegarsi alla pagina web, spostarsi nella sezione relativa alle VNF e trascinare i vari pacchetti; dopodiché è possibile

caricare il pacchetto del NS nell'apposita sezione.

Terminato l'on-boarding, prima di poter avviare la creazione dell'istanza del NS, è necessario aggiungere in OSM un account VIM. Per aggiungere un account è necessario specificare:

- un nome identificativo
- il tipo di infrastruttura (OpenStack nel caso in esame)
- l'URL del VIM
- l'username e la password dell'utente VIM
- il tenant che si vuole utilizzare

Siccome sono stati specificati degli IP statici su reti condivise (rete *external*) è necessario che l'account abbia i permessi di amministratore per il tenant scelto. Anche in questo caso è possibile utilizzare CLI o GUI.

```
osm vim-create --name openstack2 --user danieleRossi
--password <password> --auth_url http://10.15.2.1:5000/v3
--tenant danieleRossi --account_type openstack
```

Infine, sempre tramite GUI o CLI, è possibile avviare la creazione dell'istanza del NS.

```
osm ns-create --ns_name Open5GS --nsd_name Open5GS_nsd
--vim_account [VIM_ACCOUNT]
```

Capitolo 5

Validazione del lavoro svolto

In questo capitolo verranno discusse le principali analisi e validazioni per verificare la bontà del lavoro svolto.

5.1 Analisi del traffico

5.1.1 Procedura di collegamento dell'eNodeB alla MME

L'eNodeB stabilisce un collegamento con la MME inviandogli un messaggio di **S1SetupRequest**. In questo messaggio specifica il suo id globale, che si compone del suo ID più quello della PLMN a cui appartiene, il suo nome, le tracking area supportate e il default paging cycle [5.1]. Quest'ultimo parametro indica il tempo, misurato in frame radio, dopo il quale l'UE si sveglierà dalla modalità idle per leggere i messaggi di paging.

No.	Time	Source	Destination	Protocol	Length	Info
53	7.087554	10.250.2.246	10.250.2.55	S1AP	114	S1SetupRequest


```

> Frame 53: 114 bytes on wire (912 bits), 114 bytes captured (912 bits)
> Ethernet II, Src: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9)
> Internet Protocol Version 4, Src: 10.250.2.246, Dst: 10.250.2.55
> Stream Control Transmission Protocol, Src Port: 52590 (52590), Dst Port: 36412 (36412)
v S1 Application Protocol
  v S1AP-PDU: initiatingMessage (0)
    v initiatingMessage
      procedureCode: id-S1Setup (17)
      criticality: reject (0)
      v value
        v S1SetupRequest
          v protocolIEs: 4 items
            > Item 0: id-Global-ENB-ID
            > Item 1: id-eNBname
            > Item 2: id-SupportedTAs
            > Item 3: id-DefaultPagingDRX
          v SupportedTAs: 1 item
            v Item 0
              v SupportedTAs-Item
                tAC: 7 (0x0007)
                v broadcastPLMNs: 1 item
                  v Item 0
                    PLMNidentity: 00f110
                    Mobile Country Code (MCC): Unknown (1)
                    Mobile Network Code (MNC): Unknown (01)
          v Global-ENB-ID
            pLMNidentity: 00f110
            Mobile Country Code (MCC): Unknown (1)
            Mobile Network Code (MNC): Unknown (01)
            eNB-ID: macroENB-ID (0)
  
```

Figura 5.1: S1SetupRequest inviata dall'eNodeB alla MME

La MME risponde con un messaggio di **S1SetupResponse** specificando il suo nome, la lista delle PLMN e delle MME servite e la sua capacità [5.2]. La capacità relativa di una MME è un valore compreso tra 0 e 255, che consente all'eNodeB di bilanciare il carico tra le MME nello stesso pool.

No.	Time	Source	Destination	Protocol	Length	Info
55	7.088398	10.250.2.55	10.250.2.246	S1AP	110	S1SetupResponse


```

> Frame 55: 110 bytes on wire (880 bits), 110 bytes captured (880 bits)
> Ethernet II, Src: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9), Dst: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e)
> Internet Protocol Version 4, Src: 10.250.2.55, Dst: 10.250.2.246
> Stream Control Transmission Protocol, Src Port: 36412 (36412), Dst Port: 52590 (52590)
v S1 Application Protocol
  v S1AP-PDU: successfulOutcome (1)
    v successfulOutcome
      procedureCode: id-S1Setup (17)
      criticality: reject (0)
      v value
        v S1SetupResponse
          v protocolIEs: 3 items
            > Item 0: id-MMEname
            > Item 1: id-ServedGUMMEIs
            > Item 2: id-RelativeMMECapacity
          v ServedGUMMEIs: 1 item
            v Item 0
              v ServedGUMMEIsItem
                > servedPLMNs: 1 item
                > servedGroupIDs: 1 item
                > servedMMECs: 1 item
  
```

Figura 5.2: S1SetupResponse inviata dalla MME all'eNodeB

5.1.2 Procedura di collegamento UE senza contesto salvato nell'EPC

Per rendere più chiara la trattazione, il flusso dei messaggi è suddiviso in 4 parti. Di seguito verrà fornita una descrizione per ciascun messaggio e

verranno mostrate le catture dei pacchetti di rete.

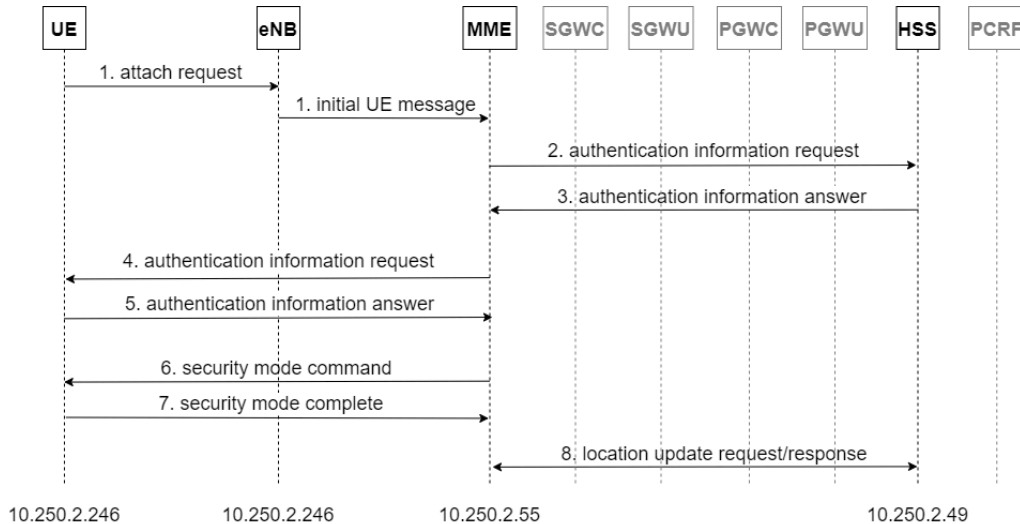


Figura 5.3: Flusso dati per la procedura di collegamento dell'UE senza contesto salvato. Parte 1.

1. L'UE invia all'eNodeB un messaggio di tipo **attach request** contenente diversi parametri, tra cui l'**EPS Mobile Identity** che può essere un GUTI, IMSI o IMEI. In particolare, se l'UE si è già connesso alla rete e possiede un GUTI valido, fornirà quest'ultimo; in caso contrario fornirà il suo identificatore, ovvero l'IMSI. L'IMEI è utilizzato per instaurare bearer di emergenza. Nel nostro caso, essendo la prima registrazione l'UE fornisce l'IMSI [5.4]. Ricevuto il messaggio dell'UE, l'eNodeB, nel caso di registrazione iniziale, selezionerà una MME a cui inoltrare la richiesta.
2. A questo punto inizia la procedura di **Authentication and Key Agreement**, AKA, che ha lo scopo di autenticare l'utente e creare le chiavi per la cifratura e l'integrità dei messaggi. L'AKA si basa su una gerarchia di chiavi; in primo luogo due chiavi, K e AMF, sono condivise tra L'USIM in possesso dell'UE e la parte di rete che si occupa dell'autenticazione. Da tale chiave l'HSS è in grado di ricavarne altre due CK e IK; in seguito vedremo come tali chiavi verranno utilizzate. Tramite il protocollo diameter la MME invia all'HSS una *Authentication Information Request* includendo l'IMSI dell'utente da autenticare e la Serving Network identity, SN id (MCC+MNC). Quest'ultimo parametro verrà usato dall'HSS per computare una chiave K_{ASME} utile

per derivare le chiavi che verranno poi usate per garantire integrità e cifratura [5.5].

```

No.      Time      Source      Destination      Protocol      Length  Info
-----  -
125 37.806807  10.250.2.246  10.250.2.55      S1AP/NAS-EPS  130     InitialUEMessage, Attach request, PDN connectivity request
> Frame 125: 130 bytes on wire (1040 bits), 130 bytes captured (1040 bits)
> Ethernet II, Src: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9)
> Internet Protocol Version 4, Src: 10.250.2.246, Dst: 10.250.2.55
> Stream Control Transmission Protocol, Src Port: 52590 (52590), Dst Port: 36412 (36412)
√ S1 Application Protocol
  √ S1AP-PDU: initiatingMessage (0)
    √ initiatingMessage
      procedureCode: id-initialUEMessage (12)
      criticality: ignore (1)
      √ value
        √ InitialUEMessage
          √ protocolIEs: 5 items
            > Item 0: id-eNB-UE-S1AP-ID
            > Item 1: id-NAS-PDU
            > Item 2: id-TAI
            > Item 3: id-EUTRAN-CGI
            > Item 4: id-RRC-Establishment-Cause
          √ EPS mobile identity
            Length: 8
            ... 1... = Odd/even indication: Odd number of identity digits
            ... .001 = Type of identity: IMSI (1)
            IMSI: 001010123456789
            √ [Association IMSI: 001010123456789]
              Mobile Country Code (MCC): Unknown (1)
              Mobile Network Code (MNC): Unknown (010)
          √ UE network capability
            Length: 2
            1... .. = EEA0: Supported
            .1... .. = 128-EEA1: Supported
            ..1... .. = 128-EEA2: Supported
            ...1... .. = 128-EEA3: Supported
            ....0... .. = EEA4: Not supported
            .... .0.. = EEA5: Not supported
            .... ..0. = EEA6: Not supported
            .... ...0 = EEA7: Not supported
            0... .. = EIA0: Not supported
            .1... .. = 128-EIA1: Supported
            ..1... .. = 128-EIA2: Supported
            ...1... .. = 128-EIA3: Supported
            ....0... .. = EIA4: Not supported
            .... .0.. = EIA5: Not supported
            .... ..0. = EIA6: Not supported
            .... ...0 = EIA7: Not supported
  
```

Figura 5.4: Initial UE message tra UE e MME.

```

Source      Destination      Protocol      Info
-----  -
10.250.2.55  10.250.2.49      DIAMETER     cmd=3GPP-Authentication-Information Request(318) flags=RP-- appl=3GPP S6a/S6d(16777251) h2h=5ebdf521 e2e=c68725b |
> Frame 126: 318 bytes on wire (2544 bits), 318 bytes captured (2544 bits)
> Ethernet II, Src: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9), Dst: fa:16:3e:27:58:4c (fa:16:3e:27:58:4c)
> Internet Protocol Version 4, Src: 10.250.2.55, Dst: 10.250.2.49
> Stream Control Transmission Protocol, Src Port: 3868 (3868), Dst Port: 45466 (45466)
√ Diameter Protocol
  Version: 0x01
  Length: 256
  √ Flags: 0xc0, Request, Proxyable
  Command Code: 318 3GPP-Authentication-Information
  ApplicationId: 3GPP S6a/S6d (16777251)
  Hop-by-Hop Identifier: 0x5ebdf521
  End-to-End Identifier: 0x0c68725b
  [Answer In: 127]
  √ AVP: Session-Id(263) l=44 f=-M- val=mme.localdomain;1604337862;3;app_s6a
  √ AVP: Auth-Session-State(277) l=12 f=-M- val=NO_STATE_MAINTAINED (1)
  √ AVP: Origin-Host(264) l=23 f=-M- val=mme.localdomain
  √ AVP: Origin-Realm(296) l=19 f=-M- val=localdomain
  √ AVP: Destination-Realm(283) l=19 f=-M- val=localdomain
  √ AVP: User-Name(1) l=23 f=-M- val=001010123456789
  √ AVP: Requested-EUTRAN-Authentication-Info(1408) l=44 f=VM- vnd=TGPP
  √ AVP: Visited-PLMN-Id(1407) l=15 f=VM- vnd=TGPP val=MCC 1 , MNC 01
  √ AVP: Vendor-Specific-Application-Id(260) l=32 f=-M-
    IMSI: 001010123456789
    √ [Association IMSI: 001010123456789]
      Mobile Country Code (MCC): Unknown (1)
      Mobile Network Code (MNC): Unknown1 (010)
  
```

Figura 5.5: Authentication Information Request inviata dalla MME all'HSS.

3. Ricevuta la richiesta dalla MME, l'HSS potrebbe avere degli Authentication Vectors, AVs, precomputati; in caso contrario dovrà computarli. L'HSS invia una **Authentication Information Answer** [5.6] alla MME che contiene un array ordinato di n AV (1 ... n). Se $n > 1$, gli EPS AV vengono ordinati in base al numero di sequenza. 3GPP nella specifica tecnica TS33.401 raccomanda $n=1$. Un AV si compone di 4 elementi:

- RAND. Un numero random che la MME utilizzerà come una sfida per l'autenticazione che l'UE dovrà sostenere.
- XRES. La risposta attesa per quanto riguarda la sfida. La risposta può essere calcolata solo dall'UE che possiede la chiave K.
- AUTN. Token di autenticazione, che può essere calcolato solo dalle entità che conoscono il valore di K (in questo caso l'HSS) e che include un numero di sequenza (SQN) per impedire attacchi di tipo replay.
- K_{ASME} . Chiave derivata da CK e IK e, in ultima analisi, dal valore dell'SN id e dallo xor tra un numero di sequenza (SQN) e un'Anonymity Key, AK, generata a sua volta a partire dai valori di K e RAND.

L'immagine [5.7] mostra come vengono generati i diversi campi di un AV; f1...f5 sono funzioni crittografiche. Maggiori dettagli in merito alla generazione degli AVs si possono trovare in [29].

```

Source      Destination  Protocol  Info
10.250.2.49 10.250.2.55 DIAMETER SACK cmd=3GPP-Authentication-Information Answer(318) flags=P- appl=3GPP S6a/S6d(16777251) h2h=5ebdf521 e2e=c68725b |
> Frame 127: 300 bytes on wire (3000 bits), 300 bytes captured (3000 bits)
> Ethernet II, Src: fa:16:3e:27:58:4c (fa:16:3e:27:58:4c), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9)
> Internet Protocol Version 4, Src: 10.250.2.49, Dst: 10.250.2.55
> Stream Control Transmission Protocol, Src Port: 45466 (45466), Dst Port: 3868 (3868)
v Diameter Protocol
  Version: 0x01
  Length: 308
  Flags: 0x40, Proxyable
  Command Code: 318 3GPP-Authentication-Information
  ApplicationId: 3GPP S6a/S6d (16777251)
  Hop-by-Hop Identifier: 0x5ebdf521
  End-to-End Identifier: 0x0c68725b
  [Request In: 126]
  [Response Time: 0.006743000 seconds]
  > AVP: Session-Id(263) l=44 f=M- val=mme.localdomain;1604337862;3;app_s6a
  > AVP: Authentication-Info(1413) l=144 f=VM- vnd=TGPP
  > AVP: Origin-Host(264) l=23 f=M- val=hss.localdomain
  > AVP: Origin-Realm(296) l=19 f=M- val=localdomain
  > AVP: Result-Code(268) l=12 f=M- val=DIAMETER_SUCCESS (2001)
  > AVP: Auth-Session-State(277) l=12 f=M- val=NO_STATE_MAINTAINED (1)
  > AVP: Vendor-Specific-Application-Id(260) l=32 f=M-
    v E-UTRAN-Vector: 000005a7c000001c000028af96c012bd6c16f5044302f5e6...
      > AVP: RAND(1447) l=28 f=VM- vnd=TGPP val=96c012bd6c16f5044302f5e6614afac5
      > AVP: XRES(1448) l=20 f=VM- vnd=TGPP val=f8674e1036600254
      > AVP: AUTN(1449) l=28 f=VM- vnd=TGPP val=ff087c23504b8000892d52dc8235fe2e
      > AVP: KASME(1450) l=44 f=VM- vnd=TGPP val=26bd2bd3381dfb2ee1f238bdb8880f5af19781cccd34e9e88...

```

Figura 5.6: Authentication Information Answer inviata dall'HSS alla MME.

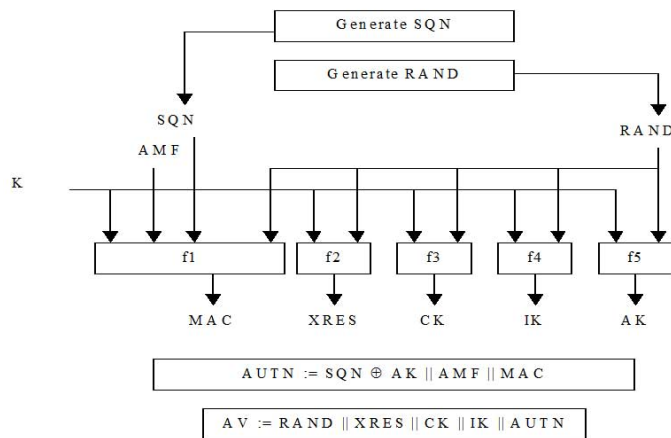


Diagram2: - Vector Derivation at Network side

Figura 5.7: Generazione dei campi di un AV da parte dell'HSS [1].

- la MME seleziona il primo AV non utilizzato nell'array inviatogli dall'HSS (nel nostro caso l'HSS ha inviato un solo AV) e invia all'USIM, tramite il Mobile Equipment, ME, gli elementi $RAND$ e $AUTN$ [5.8]. La MME include anche il KSI_{ASME} per la ME che verrà usato per identificare le chiavi K_{ASME} e quelle generate a partire da essa.

No.	Time	Source	Destination	Protocol	Length	Info
128	37.818167	10.250.2.55	10.250.2.246	S1AP/NAS-EPS	138	DownlinkNASTransport, Authentication request

```

> Frame 128: 138 bytes on wire (1104 bits), 138 bytes captured (1104 bits)
> Ethernet II, Src: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9), Dst: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e)
> Internet Protocol Version 4, Src: 10.250.2.55, Dst: 10.250.2.246
> Stream Control Transmission Protocol, Src Port: 36412 (36412), Dst Port: 52590 (52590)
v S1 Application Protocol
  v S1AP-PDU: initiatingMessage (0)
    v initiatingMessage
      procedureCode: id-downlinkNASTransport (11)
      criticality: ignore (1)
      v value
        v DownlinkNASTransport
          v protocolIEs: 3 items
            > Item 0: id-MME-UE-S1AP-ID
            > Item 1: id-eNB-UE-S1AP-ID
            > Item 2: id-NAS-PDU
              ... 0... = Type of security context flag (TSC): Native security context (for KSIasme)
              ... .000 = NAS key set identifier: (0) ASME
              v Authentication Parameter RAND - EPS challenge
                RAND value: 96c012bd6c16f5044302f5e6614afac5
              v Authentication Parameter AUTN (UMTS and EPS authentication challenge) - EPS challenge
                Length: 16
                v AUTN value: ff087c23504b8000892d52dc8235fe2e
                  SQN xor AK: ff087c23504b
                  AMF: 8000
                  MAC: 892d52dc8235fe2e
  
```

Figura 5.8: Authentication Information Request inviata dalla MME all'UE.

- Ricevuti $RAND$ e $AUTN$, l'USIM esegue la procedura in [5.9] per la computazione del risultato RES . In particolare, tramite K e $RAND$ computa l' AK . Estrapolando dall' $AUTN$ il campo $SQN \oplus AK$ [5.8] e mettendolo in xor con l' AK calcolato è in grado di ottenere il SQN . Tramite quest'ultimo, l'USIM è in grado di calcolare il MAC atteso,

$XMAC = f1_K(SQN || RAND || AMF)$ e verificare che questo sia uguale al MAC contenuto nell'AUTN. Un'ulteriore verifica che esegue l'USIM è che il SQN sia in un range corretto. Il metodo di verifica e di generazione del SQN non è standardizzato; siccome HSS e USIM sono sotto il controllo dell'operatore, questo può decidere le procedure da adottare. Nel caso in cui le verifiche vadano a buon fine, l'USIM computa la chiave crittografica, CK, la chiave per l'integrità IK e il risultato della sfida, RES e li invia alla ME. Questa tramite CK e IK è in grado di ottenere la chiave K_{ASME} e inoltra il messaggio di risposta contenente esclusivamente il RES alla MME [5.10].

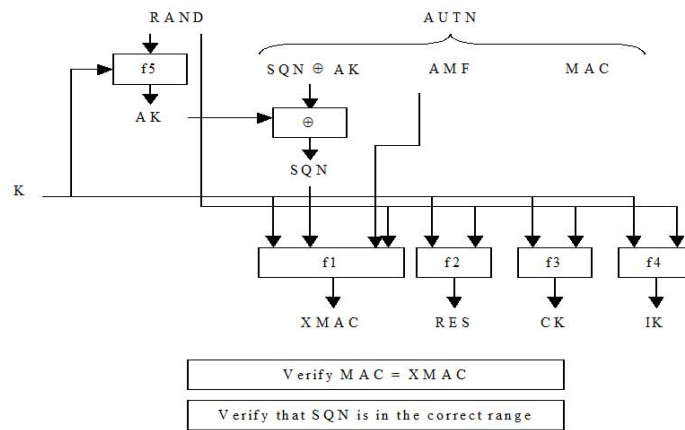


Diagram3: - Vector Derivation at USIM side

Figura 5.9: Derivazione dei campi di un AV da parte dell'USIM [1].

No.	Time	Source	Destination	Protocol	Length	Info
129	37.834645	10.250.2.246	10.250.2.55	S1AP/NAS-EPS	138	UplinkNASTransport, Authentication response

```

> Frame 129: 138 bytes on wire (1104 bits), 138 bytes captured (1104 bits)
> Ethernet II, Src: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9)
> Internet Protocol Version 4, Src: 10.250.2.246, Dst: 10.250.2.55
> Stream Control Transmission Protocol, Src Port: 52590 (52590), Dst Port: 36412 (36412)
v S1 Application Protocol
  v S1AP-PDU: initiatingMessage (0)
    v initiatingMessage
      procedureCode: id-uplinkNASTransport (13)
      criticality: ignore (1)
      v value
        v UplinkNASTransport
          v protocolIEs: 5 items
            > Item 0: id-MME-UE-S1AP-ID
            > Item 1: id-eNB-UE-S1AP-ID
            > Item 2: id-NAS-PDU
            > Item 3: id-EUTRAN-CGI
            > Item 4: id-TAI
          v NAS-PDU: 075308f8674e1036600254
            v Non-Access-Stratum (NAS)PDU
              0000 .... = Security header type: Plain NAS message, not security protected (0)
              .... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
              NAS EPS Mobility Management Message Type: Authentication response (0x53)
              v Authentication response parameter
                Length: 8
                RES: f8674e1036600254
  
```

Figura 5.10: Authentication Information Answer inviata dall'UE alla MME.

6. La MME controlla che il RES corrisponda al XRES dell'AV selezionato e in caso affermativo l'autenticazione ha avuto successo. A questo punto genera le chiavi di cifratura ed integrità NAS a partire dalla K_{ASME} dell'AV e seleziona gli algoritmi che dovrà usare la ME per generare le chiavi [5.11].

```

No.    Time           Source            Destination       Protocol           Length  Info
-----
130    37.834887      10.250.2.55      10.250.2.246     S1AP/NAS-EPS     126    DownlinkNASTransport, Security mode command
  > Frame 130: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits)
  > Ethernet II, Src: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9), Dst: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e)
  > Internet Protocol Version 4, Src: 10.250.2.55, Dst: 10.250.2.246
  > Stream Control Transmission Protocol, Src Port: 36412 (36412), Dst Port: 52590 (52590)
  > S1 Application Protocol
  > S1AP-PDU: initiatingMessage (0)
  >   initiatingMessage
  >     procedureCode: id-downlinkNASTransport (11)
  >     criticality: ignore (1)
  >     value
  >       DownlinkNASTransport
  >         protocolIEs: 3 items
  >         > Item 0: id-MME-UE-S1AP-ID
  >         > Item 1: id-eNB-UE-S1AP-ID
  >         > Item 2: id-NAS-PDU
  >           Item 2: id-NAS-PDU
  >             ProtocolIE-Field
  >               id: id-NAS-PDU (26)
  >               criticality: reject (0)
  >               value
  >                 NAS-PDU: 37cc21b6300075d02002f070c14f087a25ac699d102fb5
  >                 Non-Access-Stratum (NAS)PDU
  >                   0011 .... = Security header type: Integrity protected with new EPS security context (3)
  >                   ... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
  >                   Message authentication code: 0xcc21b630
  >                   Sequence number: 0
  >                   0000 .... = Security header type: Plain NAS message, not security protected (0)
  >                   ... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
  >                   NAS EPS Mobility Management Message Type: Security mode command (0x5d)
  >                   NAS security algorithms - Selected NAS security algorithms
  >                   0000 .... = Spare half octet: 0
  >                   ... 0... = Type of security context flag (TSC): Native security context (for KSIasme)
  >                   .... 0000 = NAS key set identifier: (0) ASME
  >                   UE security capability - Replayed UE security capabilities
  >                   IMEISV request
  >                   HashMME
  
```

Figura 5.11: Security Mode Command inviata dalla MME all'UE.

7. La ME genera le chiavi NAS utilizzando gli algoritmi specificati dalla MME ed esegue una verifica dell'integrità del *Security Mode Command* utilizzando la chiave opportuna. Infine, invia il messaggio di **Security Mode Complete** alla MME cifrandolo e proteggendone l'integrità 5.12.

```

No.    Time           Source            Destination       Protocol           Length  Info
-----
131    37.863599      10.250.2.246    10.250.2.55     S1AP/NAS-EPS     146    UplinkNASTransport, Security mode complete
  > Frame 131: 146 bytes on wire (1168 bits), 146 bytes captured (1168 bits)
  > Ethernet II, Src: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9)
  > Internet Protocol Version 4, Src: 10.250.2.246, Dst: 10.250.2.55
  > Stream Control Transmission Protocol, Src Port: 52590 (52590), Dst Port: 36412 (36412)
  > S1 Application Protocol
  > S1AP-PDU: initiatingMessage (0)
  >   initiatingMessage
  >     procedureCode: id-uplinkNASTransport (13)
  >     criticality: ignore (1)
  >     value
  >       UplinkNASTransport
  >         protocolIEs: 5 items
  >         > Item 0: id-MME-UE-S1AP-ID
  >         > Item 1: id-eNB-UE-S1AP-ID
  >         > Item 2: id-NAS-PDU
  >         > Item 3: id-EUTRAN-CGI
  >         > Item 4: id-TAI
  >           Item 2: id-NAS-PDU
  >             Non-Access-Stratum (NAS)PDU
  >               0100 .... = Security header type: Integrity protected and ciphered with new EPS security context (4)
  >               ... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
  >               Message authentication code: 0x4846e06d
  >               Sequence number: 0
  >               0000 .... = Security header type: Plain NAS message, not security protected (0)
  >               ... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
  >               NAS EPS Mobility Management Message Type: Security mode complete (0x5e)
  >               Mobile identity - IMEISV - IMEISV (3534900698733153)
  
```

Figura 5.12: Security Mode Complete inviata dall'UE alla MME.

8. Una volta completate le procedure per l'autenticazione e la configurazione della sicurezza NAS, la MME deve registrare l'abbonato nella rete e scoprire quali servizi può utilizzare quest'ultimo. A tal fine, la

MME notifica, tramite il protocollo diameter, all'HSS che l'abbonato è registrato nella rete e si trova nelle sue Tracking Area, quindi scarica le informazioni sull'abbonato dall'HSS 5.13.

```

Source      Destination      Protocol      Length  Info
10.250.2.55  10.250.2.49     DIAMETER      394    SACK cmd=3GPP-Update-Location Request(316) flags=RP-- appl=3GPP S6a/S6d(16777251) h2h=5ebdf522 e2e=c68725c |
10.250.2.49  10.250.2.55     DIAMETER      614    SACK cmd=3GPP-Update-Location Answer(316) flags=-P-- appl=3GPP S6a/S6d(16777251) h2h=5ebdf522 e2e=c68725c |
> Frame 132: 394 bytes on wire (3152 bits), 394 bytes captured (3152 bits)
> Ethernet II, Src: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9), Dst: fa:16:3e:27:58:4c (fa:16:3e:27:58:4c)
> Internet Protocol Version 4, Src: 10.250.2.55, Dst: 10.250.2.49
> Stream Control Transmission Protocol, Src Port: 3868 (3868), Dst Port: 45466 (45466)
▼ Diameter Protocol
  Version: 0x01
  Length: 316
  > Flags: 0xc0, Request, Proxyable
  Command Code: 316 3GPP-Update-Location
  ApplicationId: 3GPP S6a/S6d (16777251)
  Hop-by-Hop Identifier: 0x5ebdf522
  End-to-End Identifier: 0x0c68725c
  [Answer In: 133]
  > AVP: Session-Id(263) l=44 f=-M- val=mme.localdomain;1604337862;4;app_s6a
  > AVP: Auth-Session-State(277) l=12 f=-M- val=NO_STATE_MAINTAINED (1)
  > AVP: Origin-Host(264) l=23 f=-M- val=mme.localdomain
  > AVP: Origin-Realm(296) l=19 f=-M- val=localdomain
  > AVP: Destination-Realm(283) l=19 f=-M- val=localdomain
  > AVP: User-Name(1) l=23 f=-M- val=001010123456789
  > AVP: Terminal-Information(1401) l=56 f=VM- vnd=TGPP
  > AVP: RAT-Type(1032) l=16 f=V-- vnd=TGPP val=EUTRAN (1004)
  > AVP: ULR-Flags(1405) l=16 f=VM- vnd=TGPP val=2
  > AVP: Visited-PLMN-Id(1407) l=15 f=VM- vnd=TGPP val=MCC 1 , MNC 01
  > AVP: UE-SRVCC-Capability(1615) l=16 f=V-- vnd=TGPP val=UE-SRVCC-NOT-SUPPORTED (0)
  > AVP: Vendor-Specific-Application-Id(260) l=32 f=-M-

```

Figura 5.13: Registrazione abbonato e ottenimento delle sue informazioni tra MME e HSS.

Da ora in poi le diverse entità dell'EPC si scambieranno messaggi al fine di instaurare i tunnel: GTP-C e GTP-U e le sessioni PFCP affinché le entità di controllo possano supervisionare le entità del piano utente.

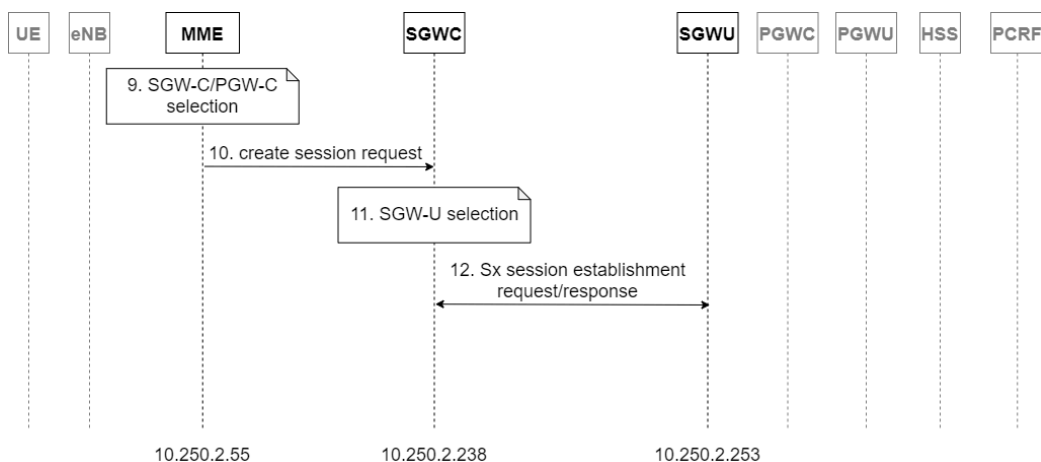


Figura 5.14: Flusso dati per la procedura di collegamento dell'UE senza contesto salvato. Parte 2.

9. La MME seleziona SGW-C e PGW-C come descritto in 3GPP TS 23.401 paragrafi 4.3.8.1 e 4.3.8.2.
10. La MME dopo aver selezionato SGW-C e PGW-C, invia al primo un messaggio di *Create Session Request*. Come abbiamo visto in precedenza, i tunnel GTP sono identificati su un nodo da un Tunnel End-Point Identifier. L'indirizzo IP del nodo, insieme al TEID ed eventualmente la porta, se non è quella di default per il protocollo GTP, costituiscono il Fully Qualified TEID, F-TEID. In questo caso, la MME contatta l'SGW-C sul TEID 0, che definiremo *butler*; questo indica che tra le entità non sussiste ancora un tunnel GTP, ma si vuole instaurarlo. Nella richiesta la MME indica all'SGW-C che vuole essere contattato sul TEID 0x1 e specifica a quest'ultimo l'indirizzo del PGW-C scelto e il TEID su cui contattarlo, ovvero il butler. Oltre alle informazioni per l'instaurazione del tunnel, la MME indica l'IMSI che identifica l'UE a cui saranno associati i pacchetti dati trasportati e l'EPS Bearer Identity, EBI. L'header del messaggio contiene un numero di sequenza che permetterà di associare una richiesta alla relativa risposta. Questo messaggio crea il downlink del tunnel GTP-C tra MME e SGW-C (interfaccia S11).

No.	Time	Source	Destination	Protocol	Length	Info
134	37.867711	10.250.2.55	10.250.2.238	GTPv2	245	Create Session Request

```

> Frame 134: 245 bytes on wire (1960 bits), 245 bytes captured (1960 bits)
> Ethernet II, Src: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9), Dst: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99)
> Internet Protocol Version 4, Src: 10.250.2.55, Dst: 10.250.2.238
> User Datagram Protocol, Src Port: 2123, Dst Port: 2123
v GPRS Tunneling Protocol V2
  > Flags: 0x48
  > Message Type: Create Session Request (32)
  > Message Length: 199
  > Tunnel Endpoint Identifier: 0x00000000 (0)
  > Sequence Number: 0x00000007 (7)
  > Spare: 0
  > International Mobile Subscriber Identity (IMSI) : 001010123456789
  > Mobile Equipment Identity (MEI) : 3534900698733153
  > User Location Info (ULI) : TAI ECGI
  > Serving Network : MCC 1 , MNC 01
  > RAT Type : EUTRAN (6)
  > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S11 MME GTP-C interface, TEID/GRE Key: 0x00000001, IPv4 10.250.2.55
  > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S5/S8 PGW GTP-C interface, TEID/GRE Key: 0x00000000, IPv4 10.250.2.30, IPv6 ::1
  > Access Point Name (APN) : internet
  > Selection Mode : MS or network provided APN, subscribed verified
  > PDN Type : IPv4
  > PDN Address Allocation (PAA) : IPv4 0.0.0.0
  > APN Restriction : No Existing Contexts or Restriction (0)
  > Aggregate Maximum Bit Rate (AMBR) :
  > Bearer Context : [Grouped IE] -> EPS Bearer ID (EBI) : 5
  > UE Time Zone :
  > Charging Characteristics :
  > [Response In: 150]

```

Figura 5.15: Create Session Request inviata dalla MME all'SGW-C.

11. L'SGW-C seleziona l'SGW-U, tramite appositi criteri, che farà da endpoint per i tunnel GTP-U sulle interfacce S1-U (da/verso eNB) e S5/8 (da/verso PGW-U).
12. Dopo aver selezionato l'SGW-U, l'SGW-C instaura una sessione PFCP con questo per poterlo controllare. Le sessioni PFCP funzionano in maniera simile ai tunnel GTP; al posto di TEID si parla di Session Endpoint Identifier, SEID. La differenza tra i due è che in una sessione PFCP le due entità usano lo stesso SEID scelto dal trasmittente, mentre in un tunnel GTP ogni entità ha il proprio TEID. In particolare, l'SGW-C invia all'SGW-U un messaggio di **Session Establishment Request** in cui specifica il SEID della sessione e le regole Packet Detection Rule, PDR, Forwarding Action Rule, FAR, Usage Reporting Rule, URR, QoS Enhancement Rule, QER, e Buffering Action Rule, BAR. Per quanto riguarda il messaggio in questione, le regole PDR servono per creare i TEID su cui l'SGW-U dovrà essere contattato dal PGW-U, 0x5 (Source Interface: Core), e dall'eNodeB, 0x6 (Source Interface: Access). Le regole FAR indicano che l'SGW-U dovrà instaurare i pacchetti verso PGW-U ed eNodeB, ma per il momento non sono presenti indicazioni per l'instradamento, visto che il PGW-U e l'eNodeB non hanno ancora creato i TEID per il tunnel. Il SGW-U confermerà al SGW-C l'instaurazione della sessione PFCP e la creazione delle regole specificate [5.16].

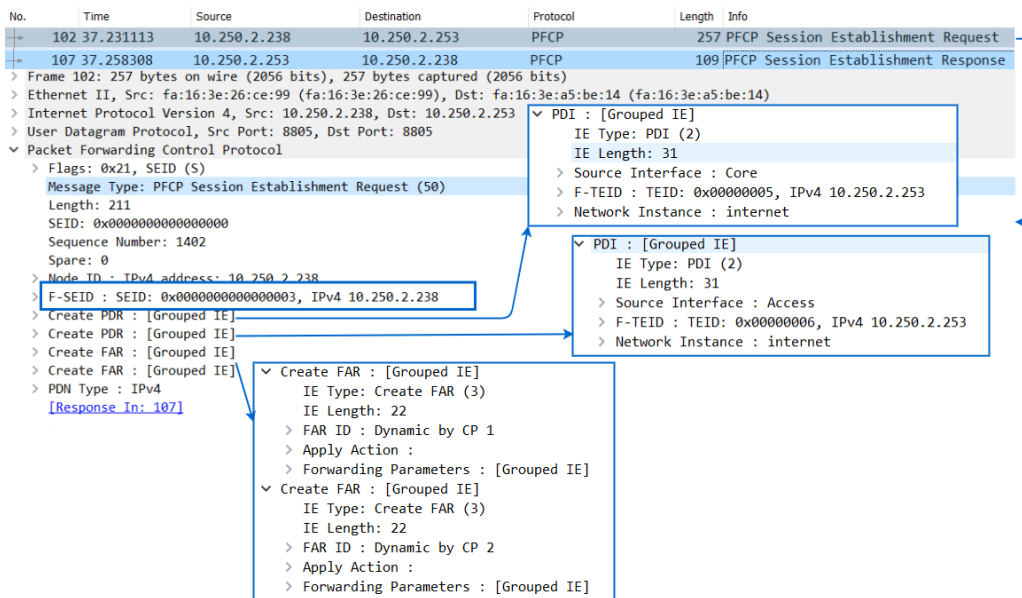


Figura 5.16: Session Establishment Request inviata dall'SGW-C all'SGW-U.

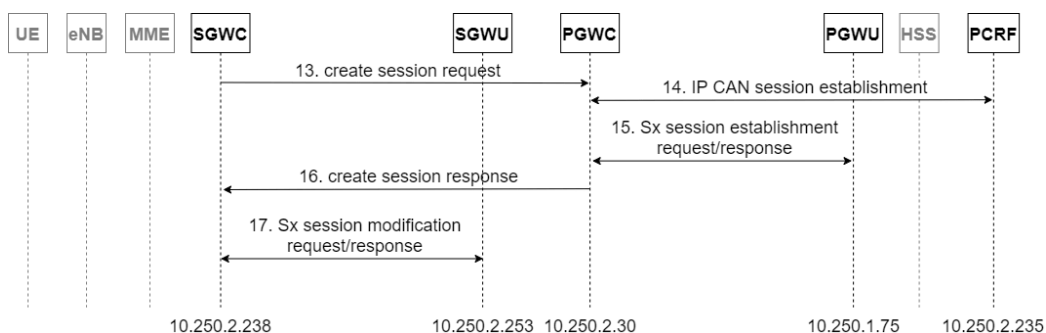


Figura 5.17: Flusso dati per la procedura di collegamento dell'UE senza contesto salvato. Parte 3.

13. Il SGW-C esegue una procedura simile a quella della MME al punto 10 al fine di instaurare un tunnel GTP con il PGW-C. In particolare, l'SGW-C invia al PGW-C il TEID che ha allocato per il tunnel, 0x3. Va ricordato, che la MME ha fornito all'SGW-C il F-TEID su cui contattare il PGW-C (IP 10.250.2.30, TEID butler). Oltre a ciò, l'SGW-C informa il PGW-C del TEID allocato sull'SGW-U per essere contattato dal PGW-U, 0x5. Questo messaggio crea il downlink del tunnel GTP-C tra SGW-C e PGW-C (interfaccia S5/8c) [5.18].


```

No.      Time      Source      Destination      Protocol      Length      Info
-----  -
108 37.258455  10.250.2.238  10.250.2.30     GTPv2        229         Create Session Request
> Frame 108: 229 bytes on wire (1832 bits), 229 bytes captured (1832 bits)
> Ethernet II, Src: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99), Dst: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d)
> Internet Protocol Version 4, Src: 10.250.2.238, Dst: 10.250.2.30
> User Datagram Protocol, Src Port: 2123, Dst Port: 2123
v GPRS Tunneling Protocol V2
  > Flags: 0x48
  > Message Type: Create Session Request (32)
  > Message Length: 183
  > Tunnel Endpoint Identifier: 0x00000000 (0)
  > Sequence Number: 0x00000005 (5)
  > Spare: 0
  > International Mobile Subscriber Identity (IMSI) : 001010123456789
  > Mobile Equipment Identity (MEI) : 3534900698733153
  > User Location Info (ULI) : TAI ECGI
  > Serving Network : MCC 1 , MNC 01
  > RAT Type : EUTRAN (6)
  > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S5/S8 SGW GTP-C interface, TEID/GRE Key: 0x00000003, IPv4 10.250.2.238
  > Access Point Name (APN) : internet
  > Selection Mode : MS or network provided APN, subscribed verified
  > PDN Type : IPv4
  > PDN Address Allocation (PAA) : IPv4 0.0.0.0
  > APN Restriction : No Existing Contexts or Restriction (0)
  > Aggregate Maximum Bit Rate (AMBR) :
  > Bearer Context : [Grouped IE]
  > UE Time Zone :
  > Charging Characteristics :
  > [Response In: 116]
  > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S5/S8 SGW GTP-U interface, TEID/GRE Key: 0x00000005, IPv4 10.250.2.253

```

Figura 5.18: Create Session Request inviata dall' SGW-C al PGW-C.

- Se è abilitato il Policy and Charging Control, PCC, dinamico e non si tratta di una procedura di handoff, ma di registrazione iniziale, come nel caso in esame, il PDN GW tramite protocollo diameter ottiene dal PCRF le regole PCC associate all'UE [5.19].

```

10.250.2.30 10.250.2.235 DIAMETER cmd=Credit-Control Request(272) flags=RP-- appl=3GPP Gx(16777238) h2h=50ad8150 e2e=e037ef03 |
10.250.2.235 10.250.2.30 DIAMETER cmd=Credit-Control Answer(272) flags=-P-- appl=3GPP Gx(16777238) h2h=50ad8150 e2e=e037ef03 |
> Frame 100: 570 bytes on wire (4560 bits), 570 bytes captured (4560 bits)
> Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:77:33:a7 (fa:16:3e:77:33:a7)
> Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.2.235
> Transmission Control Protocol, Src Port: 3868, Dst Port: 33728, Seq: 89, Ack: 77, Len: 504
v Diameter Protocol
  Version: 0x01
  Length: 504
  > Flags: 0xc0, Request, Proxyable
  > Command Code: 272 Credit-Control
  > ApplicationId: 3GPP Gx (16777238)
  > Hop-by-Hop Identifier: 0x50ad8150
  > End-to-End Identifier: 0xe037ef03
  > [Answer In: 102]
  > AVP: Session-Id(263) l=43 f=-M- val=smf.localdomain;1604394499;3;
  > AVP: Origin-Host(264) l=23 f=-M- val=smf.localdomain
  > AVP: Origin-Realm(296) l=19 f=-M- val=localdomain
  > AVP: Destination-Realm(283) l=19 f=-M- val=localdomain
  > AVP: Auth-Application-Id(258) l=12 f=-M- val=3GPP Gx (16777238)
  > AVP: CC-Request-Type(416) l=12 f=-M- val=INITIAL_REQUEST (1)
  > AVP: CC-Request-Number(415) l=12 f=-M- val=0
  > AVP: Subscription-Id(443) l=44 f=-M- val=
  > AVP: Supported-Features(628) l=44 f=V-- vnd=TGPP
  > AVP: Network-Request-Support(1024) l=16 f=VM- vnd=TGPP val=NETWORK_REQUEST SUPPORTED (1)
  > AVP: Framed-IP-Address(8) l=12 f=-M- val=10.45.0.4
  > AVP: IP-CAN-Type(1027) l=16 f=VM- vnd=TGPP val=3GPP-EPS (5)
  > AVP: RAT-Type(1032) l=16 f=V-- vnd=TGPP val=EUTRAN (1004)
  > AVP: QoS-Information(1016) l=44 f=VM- vnd=TGPP
  > AVP: Default-EPS-Bearer-QoS(1049) l=88 f=V-- vnd=TGPP
  > AVP: 3GPP-User-Location-Info(22) l=25 f=VM- vnd=TGPP val=MCC 1 , MNC 01 , ECGI 0x19b01
  > AVP: 3GPP-MS-TimeZone(23) l=14 f=VM- vnd=TGPP val=Timezone: GMT + 0 hours 0 minutes No adjustment
  > AVP: Called-Station-Id(30) l=16 f=-M- val=internet
  > AVP Code: 22 3GPP-User-Location-Info
  > AVP Flags: 0xc0, Vendor-Specific: Set, Mandatory: Set
  > 1... .. = Vendor-Specific: Set
  > .1.. .. = Mandatory: Set
  > ..0. .... = Protected: Not set
  > ...0 .... = Reserved: Not set
  > ....0... = Reserved: Not set
  > .... .0.. = Reserved: Not set
  > .... ..0. = Reserved: Not set
  > .... ...0 = Reserved: Not set
  > AVP Length: 25
  > AVP Vendor Id: 3GPP (10415)
  > 3GPP-User-Location-Info: 8200f110000700f11000019b01
  > Geographic Location Type: TAI and ECGI (130)
  > Tracking Area Identity (TAI)
  > E-UTRAN Cell Global Identifier (ECGI)
  > Padding: 000000

```

Figura 5.19: IP CAN Session Establishment tra PGW-C e PCRF.

15. Il PGW-C esegue una procedura simile a quella dell'SGW-C al punto 12 instaurando una sessione PFCP con il PGW-U. In questo caso, tramite le regole PDR alloca il TEID su cui il PGW-U dovrà essere contattato dall'SGW-U (Source Interface: Access), mentre per l'interfaccia core verso la data network non viene specificato TEID, perchè il tunnel GTP si interromperà e si utilizzerà l'instradamento IP. In questo caso, la regola FAR verso l'SGW-U ne specifica il F-TEID (ricevuto nel messaggio al punto 13). Questo messaggio crea il downlink del tunnel GTP-U tra SGW-U e PGW-U (interfaccia S5/8u). Il PGW-U confermerà al PGW-C l'instaurazione della sessione PFCP e la creazione delle regole specificate [5.20].
16. Il PGW-C risponde alla richiesta dall'SGW-C al punto 13 con una **Create Session Response** indicando l'IP allocato per l'UE, il F-TEID su cui vuole essere contattato dall'SGW-C e il F-TEID allocato al PGW-U affinché possa essere contattato dall'SGW-U. Questo messaggio crea l'uplink del tunnel GTP-C tra SGW-C e PGW-C (interfaccia S5/8c) [5.21].

No.	Time	Source	Destination	Protocol	Length	Info
107	35.867506	10.250.2.30	10.250.1.75	PFCP	353	PFCP Session Establishment Request
109	35.875818	10.250.1.75	10.250.2.30	PFCP	122	PFCP Session Establishment Response


```

> Frame 107: 353 bytes on wire (2824 bits), 353 bytes captured (2824 bits)
> Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:98:49:95 (fa:16:3e:98:49:95)
> Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.1.75
> User Datagram Protocol, Src Port: 8805, Dst Port: 8805
< Packet Forwarding Control Protocol
  < Flags: 0x21, SEID (S)
    Message Type: PFCP Session Establishment Request (50)
    Length: 307
    SEID: 0x0000000000000000
    Sequence Number: 1371
    Spare: 0
  < Node ID : IPv6 address: ::1
  < F-SEID : SEID: 0x0000000000000003, IPv4 10.250.2.30, IPv6 ::1
  < Create PDR : [Grouped IE]
  < Create PDR : [Grouped IE]
  < Create FAR : [Grouped IE]
  < Create FAR : [Grouped IE]
  < Create QER : [Grouped IE]
  < PDN Type : IPv4
  [Response In: 109]
  < Create FAR : [Grouped IE]
    IE Type: Create FAR (3)
    IE Length: 36
    > FAR ID : Dynamic by CP 1
    > Apply Action :
    < Forwarding Parameters : [Grouped IE]
      IE Type: Forwarding Parameters (4)
      IE Length: 19
      > Destination Interface : Access
      < Outer Header Creation :
        IE Type: Outer Header Creation (84)
        IE Length: 10
        Outer Header Creation Description: GTP-U/UDP/IPv4 (256)
        TEID: 0x00000005
        IPv4 Address: 10.250.2.253
      < Create FAR : [Grouped IE]
        IE Type: Create FAR (3)
        IE Length: 22
        > FAR ID : Dynamic by CP 2
        > Apply Action :
        < Forwarding Parameters : [Grouped IE]
          IE Type: Forwarding Parameters (4)
          IE Length: 5
          > Destination Interface : Core
    < PDI : [Grouped IE]
      IE Type: PDI (2)
      IE Length: 31
      > Source Interface : Access
      > F-TEID : TEID: 0x00000003, IPv4 10.250.1.75
      > Network Instance : internet
  < UE IP Address :
    IE Type: UE IP Address (93)
    IE Length: 5
    > Flags: 0x06, S/D, V4 (IPv4)
    IPv4 address: 10.45.0.4
  < PDI : [Grouped IE]
    IE Type: PDI (2)
    IE Length: 27
    > Source Interface : Core
    > Network Instance : internet
  < Precedence : 4294967295
  < PDR ID : 1
  IE Type: Create PDR (1)
  IE Length: 61
  
```

Figura 5.20: Session Establishment Request inviata dal PGW-C al PGW-U.

No.	Time	Source	Destination	Protocol	Length	Info
116	37.309250	10.250.2.30	10.250.2.238	GTPv2	131	Create Session Response
> Frame 116: 131 bytes on wire (1048 bits), 131 bytes captured (1048 bits) > Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99) > Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.2.238 > User Datagram Protocol, Src Port: 2123, Dst Port: 2123 > GPRS Tunneling Protocol V2 > Flags: 0x48 > Message Type: Create Session Response (33) > Message Length: 85 > Tunnel Endpoint Identifier: 0x00000003 (3) > Sequence Number: 0x00000005 (5) > Spare: 0 > Cause : Request accepted (16) > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S5/S8 PGW GTP-C interface, TEID/GRE Key: 0x00000003, IPv4 10.250.2.30, IPv6 ::: > PDN Address Allocation (PAA) : IPv4 10.45.0.4 → UE ip address > APN Restriction : No Existing Contexts or Restriction (0) > Bearer Context : [Grouped IE] [Response To: 108] [Response Time: 0.050795000 seconds] > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S5/S8 PGW GTP-U interface, TEID/GRE Key: 0x00000003, IPv4 10.250.1.75						

Figura 5.21: Create Session Response inviata dal PGW-C all'SGW-C.

17. A questo punto, il SGW-C conosce il F-TEID su cui è in ascolto il PGW-U; per questo motivo, invia sulla sessione PFCP una richiesta di modifica in cui specifica che la regola FAR, avente un certo ID, deve instradare i pacchetti GTP verso il F-TEID del PGW-U. Questo messaggio crea l'uplink del tunnel GTP-U tra SGW-U e PGW-U (interfaccia S5/8u) [5.22].

No.	Time	Source	Destination	Protocol	Length	Info
117	37.309384	10.250.2.238	10.250.2.253	PFCP	88	PFCP Session Modification Request
118	37.310091	10.250.2.253	10.250.2.238	PFCP	63	PFCP Session Modification Response
> Frame 117: 88 bytes on wire (704 bits), 88 bytes captured (704 bits) > Ethernet II, Src: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99), Dst: fa:16:3e:a5:be:14 (fa:16:3e:a5:be:14) > Internet Protocol Version 4, Src: 10.250.2.238, Dst: 10.250.2.253 > User Datagram Protocol, Src Port: 8805, Dst Port: 8805 > Packet Forwarding Control Protocol > Flags: 0x21, SEID (5) > Message Type: PFCP Session Modification Request (52) > Length: 42 > SEID: 0x0000000000000003 > Sequence Number: 1403 > Spare: 0 > Update FAR : [Grouped IE] → [Response In: 118] > FAR ID : Dynamic by CP 2 > Update Forwarding Parameters : [Grouped IE] > IE Type: Update Forwarding Parameters (11) > IE Length: 14 > Outer Header Creation : > IE Type: Outer Header Creation (84) > IE Length: 10 > Outer Header Creation Description: GTP-U/UDP/IPv4 (256) > TEID: 0x00000003 > IPv4 Address: 10.250.1.75						

Figura 5.22: PFCP Session Modification tra SGW-C e SGW-U.

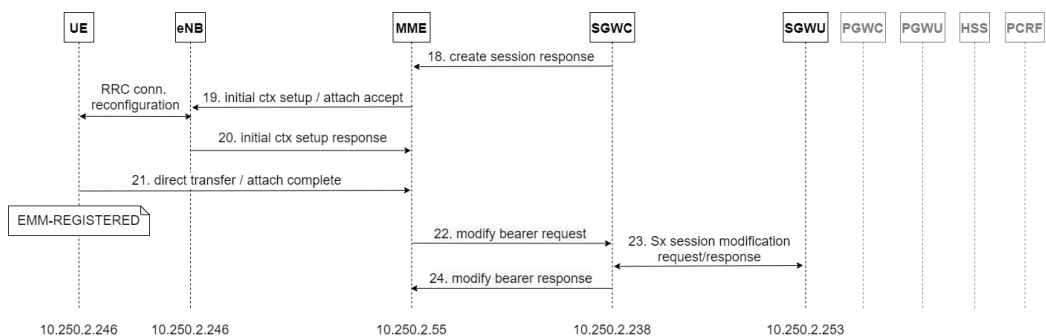


Figura 5.23: Flusso dati per la procedura di collegamento dell'UE senza contesto salvato. Parte 4.

18. L'SGW-C risponde alla richiesta della MME al punto 10 con una **Create Session Response** indicando il suo F-TEID e quello del PGW-C; in questo modo è possibile creare l'uplink del tunnel GTP-C tra MME e SGW-C e si conclude la procedura di instaurazione del tunnel di controllo. Oltre a ciò, l'SGW-C specifica i F-TEID allocati dall'SGW-U e dal PGW-U affinché sia possibile completare l'instaurazione del tunnel utente con l'eNodeB [5.24].

No.	Time	Source	Destination	Protocol	Length	Info
150	37.984692	10.250.2.238	10.250.2.55	GTPv2	157	Create Session Response

```

> Frame 150: 157 bytes on wire (1256 bits), 157 bytes captured (1256 bits)
> Ethernet II, Src: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9)
> Internet Protocol Version 4, Src: 10.250.2.238, Dst: 10.250.2.55
> User Datagram Protocol, Src Port: 2123, Dst Port: 2123
v GPRS Tunneling Protocol V2
  > Flags: 0x48
    Message Type: Create Session Response (33)
    Message Length: 111
    Tunnel Endpoint Identifier: 0x00000001 (1)
    Sequence Number: 0x00000007 (7)
    Spare: 0
  > Cause : Request accepted (16)
  > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S11/S4 SGW GTP-C interface, TEID/GRE Key: 0x00000003, IPv4 10.250.2.238
  > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S5/S8 PGW GTP-C interface, TEID/GRE Key: 0x00000003, IPv4 10.250.2.30, IPv6 ::1
  > PDN Address Allocation (PAA) : IPv4 10.45.0.4
  > APN Restriction : No Existing Contexts or Restriction (0)
  > Bearer Context : [Grouped IE]
  [Response To: 134]
  [Response Time: 0.116981000 seconds]
  > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S1-U SGW GTP-U interface, TEID/GRE Key: 0x00000006, IPv4 10.250.2.253
  > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S5/S8 PGW GTP-U interface, TEID/GRE Key: 0x00000003, IPv4 10.250.1.75
  
```

Figura 5.24: Create Session Response inviata dall'SGW-C alla MME.

19. La MME invia all'eNodeB un messaggio di Attach Accept contenuto in un messaggio di controllo S1AP di tipo **Initial Context Setup**. In tale messaggio la MME specifica all'eNodeB l'indirizzo IP e il TEID allocato dall'SGW-U al fine di instaurare il tunnel GTP-U. Questo messaggio crea l'uplink del tunnel GTP-U tra eNodeB e SGW-U (interfaccia S1-U) [5.25]

Source	Destination	Protocol	Info
10.250.2.55	10.250.2.246	S1AP/NAS-EPS	InitialContextSetupRequest, Attach accept, Activate default EPS bearer context request
<ul style="list-style-type: none"> > Frame 151: 262 bytes on wire (2096 bits), 262 bytes captured (2096 bits) > Ethernet II, Src: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9), Dst: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e) > Internet Protocol Version 4, Src: 10.250.2.55, Dst: 10.250.2.246 > Stream Control Transmission Protocol, Src Port: 36412 (36412), Dst Port: 52590 (52590) 			
<ul style="list-style-type: none"> ▼ S1 Application Protocol <ul style="list-style-type: none"> ▼ S1AP-PDU: initiatingMessage (0) <ul style="list-style-type: none"> ▼ initiatingMessage <ul style="list-style-type: none"> procedureCode: id-InitialContextSetup (9) criticality: reject (0) ▼ value <ul style="list-style-type: none"> ▼ InitialContextSetupRequest <ul style="list-style-type: none"> ▼ protocolIEs: 7 items <ul style="list-style-type: none"> > Item 0: id-MME-UE-S1AP-ID > Item 1: id-eNB-UE-S1AP-ID > Item 2: id-uEAggregateMaximumBitrate > Item 3: id-E-RABToBeSetupListCtxtSUrReq > Item 4: id-UESecurityCapabilities > Item 5: id-SecurityKey > Item 6: id-Masked-IMEISV 			
<div style="border: 1px solid black; padding: 5px;"> <ul style="list-style-type: none"> ▼ transportLayerAddress: 0afa02fd [bit length 32, 0000 1010 1111 1010 0000 0010 1111 1101 decimal value 184156925] transportLayerAddress(IPv4): 10.250.2.253 gTP-TEID: 00000006 </div>			

Figura 5.25: Attach Accept e Initial Context Setup inviato dalla MME all'eNodeB.

20. L'eNodeB risponde alla MME con un messaggio di **Initial Context Setup Response** in cui include il suo indirizzo e il TEID per il tunnel GTP-U [5.26].

No.	Time	Source	Destination	Protocol	Length	Info
152	38.018983	10.250.2.246	10.250.2.55	S1AP	118	InitialContextSetupResponse
<ul style="list-style-type: none"> > Frame 152: 118 bytes on wire (944 bits), 118 bytes captured (944 bits) > Ethernet II, Src: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9) > Internet Protocol Version 4, Src: 10.250.2.246, Dst: 10.250.2.55 > Stream Control Transmission Protocol, Src Port: 52590 (52590), Dst Port: 36412 (36412) 						
<ul style="list-style-type: none"> ▼ S1 Application Protocol <ul style="list-style-type: none"> ▼ S1AP-PDU: successfulOutcome (1) <ul style="list-style-type: none"> ▼ successfulOutcome <ul style="list-style-type: none"> procedureCode: id-InitialContextSetup (9) criticality: reject (0) ▼ value <ul style="list-style-type: none"> ▼ InitialContextSetupResponse <ul style="list-style-type: none"> ▼ protocolIEs: 3 items <ul style="list-style-type: none"> > Item 0: id-MME-UE-S1AP-ID > Item 1: id-eNB-UE-S1AP-ID > Item 2: id-E-RABSetupListCtxtSUrRes 						
<div style="border: 1px solid black; padding: 5px;"> <ul style="list-style-type: none"> ▼ transportLayerAddress: 0afa02f6 [bit length 32, 0000 1010 1111 1010 0000 0010 1111 0110 decimal value 184156918] transportLayerAddress(IPv4): 10.250.2.246 gTP-TEID: 00000001 </div>						

Figura 5.26: Initial Context Setup Response inviato dall'eNodeB alla MME.

21. L'UE invia un messaggio di trasferimento diretto all'eNodeB/MME che incapsula un messaggio di Attach Complete. Quest'ultimo contiene l'EBI, il NAS-MAC e il numero di sequenza del messaggio NAS [5.27].

No.	Time	Source	Destination	Protocol	Length	Info
158	38.223409	10.250.2.246	10.250.2.55	S1AP/NAS-EPS	122	UplinkNASTransport, Attach complete, Activate default EPS bearer context accept
159	38.223872	10.250.2.55	10.250.2.246	S1AP/NAS-EPS	138	DownlinkNASTransport, EPM information


```

> Frame 158: 122 bytes on wire (976 bits), 122 bytes captured (976 bits)
> Ethernet II, Src: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9)
> Internet Protocol Version 4, Src: 10.250.2.246, Dst: 10.250.2.55
> Stream Control Transmission Protocol, Src Port: 52590 (52590), Dst Port: 36412 (36412)
< S1 Application Protocol
  < S1AP-PDU: 27c3854bd001074300035200c2
    < Non-Access-Stratum (NAS)PDU
      < 0010 .... = Security header type: Integrity protected and ciphered (2)
      < .... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
      < Message authentication code: 0xc3854bd0
      < Sequence number: 1
      < 0000 .... = Security header type: Plain NAS message, not security protected (0)
      < .... 0111 = Protocol discriminator: EPS mobility management messages (0x7)
      < NAS EPS Mobility Management Message Type: Attach complete (0x43)
      < ESM message container
        < Length: 3
        < ESM message container contents: 5200c2
          < 0101 .... = EPS bearer identity: EPS bearer identity value 5 (5)
          < .... 0010 = Protocol discriminator: EPS session management messages (0x2)
          < Procedure transaction identity: 0
          < NAS EPS session management messages: Activate default EPS bearer context accept (0xc2)
  
```

Figura 5.27: Direct Transfer inviato dall'UE all'eNodeB/MME.

22. La MME, a questo punto, conosce il F-TEID dell'eNode-B per l'instaurazione del tunnel GTP-U e invia questa informazione all'SGW-C in un messaggio di **Modify Bearer Request**, sfruttando il tunnel di controllo instaurato precedentemente [5.28].

No.	Time	Source	Destination	Protocol	Length	Info
160	38.223979	10.250.2.55	10.250.2.238	GTPv2	76	Modify Bearer Request


```

> Frame 160: 76 bytes on wire (608 bits), 76 bytes captured (608 bits)
> Ethernet II, Src: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9), Dst: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99)
> Internet Protocol Version 4, Src: 10.250.2.55, Dst: 10.250.2.238
> User Datagram Protocol, Src Port: 2123, Dst Port: 2123
< GPRS Tunneling Protocol V2
  < Flags: 0x48
  < Message Type: Modify Bearer Request (34)
  < Message Length: 30
  < Tunnel Endpoint Identifier: 0x00000003 (3)
  < Sequence Number: 0x00000008 (8)
  < Spare: 0
  < Bearer Context : [Grouped IE]
    < [Response In: 161]
  < EPS Bearer ID (EBI) : 5
  < Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S1-U eNodeB GTP-U interface, TEID/GRE Key: 0x00000001, IPv4 10.250.2.246
  
```

Figura 5.28: Modify Bearer Request inviato dalla MME all'SGW-C.

23. L'SGW-C, tramite la sessione PFCP, aggiorna la regola FAR, avente un certo id, sull'SGW-U, fornendogli il F-TEID su cui l'eNodeB è in ascolto per il tunnel GTP-U. Questo messaggio crea il downlink del tunnel GTP-U tra eNodeB e SGW-U (interfaccia S1-U) completando l'instaurazione del tunnel GTP-U. L'SGW-U conferma all'SGW-C la modifica delle regole [5.29].

No.	Time	Source	Destination	Protocol	Length	Info
124	37.550481	10.250.2.238	10.250.2.253	PFCP	88	PFCP Session Modification Request
125	37.551167	10.250.2.253	10.250.2.238	PFCP	63	PFCP Session Modification Response

```

> Frame 124: 88 bytes on wire (704 bits), 88 bytes captured (704 bits)
> Ethernet II, Src: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99), Dst: fa:16:3e:a5:be:14 (fa:16:3e:a5:be:14)
> Internet Protocol Version 4, Src: 10.250.2.238, Dst: 10.250.2.253
> User Datagram Protocol, Src Port: 8805, Dst Port: 8805
< Packet Forwarding Control Protocol
  < Flags: 0x21, SEID (S)
    Message Type: PFCP Session Modification Request (52)
    Length: 42
    SEID: 0x0000000000000003
    Sequence Number: 1404
    Spare: 0
  < Update FAR : [Grouped IE]
    IE Type: Update FAR (10)
    IE Length: 26
    > FAR ID : Dynamic by CP 1
    > Update Forwarding Parameters : [Grouped IE]
      < Outer Header Creation :
        IE Type: Outer Header Creation (84)
        IE Length: 10
        Outer Header Creation Description: GTP-U/UDP/IPv4 (256)
        TEID: 0x00000001
        IPv4 Address: 10.250.2.246
  
```

Figura 5.29: PFCP Session Modification tra SGW-C e SGW-U.

24. L'SGW-C conferma alla MME la modifica del bearer con un messaggio di **Modify Bearer Response**, terminando la procedura di instaurazione dei tunnel GTP [5.30].

No.	Time	Source	Destination	Protocol	Length	Info
161	38.225732	10.250.2.238	10.250.2.55	GTPv2	88	Modify Bearer Response

```

> Frame 161: 88 bytes on wire (704 bits), 88 bytes captured (704 bits)
> Ethernet II, Src: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9)
> Internet Protocol Version 4, Src: 10.250.2.238, Dst: 10.250.2.55
> User Datagram Protocol, Src Port: 2123, Dst Port: 2123
< GPRS Tunneling Protocol V2
  < Flags: 0x48
    Message Type: Modify Bearer Response (35)
    Message Length: 42
    Tunnel Endpoint Identifier: 0x00000001 (1)
    Sequence Number: 0x00000008 (8)
    Spare: 0
  < Cause : Request accepted (16)
  < Bearer Context : [Grouped IE]
    IE Type: Bearer Context (93)
    IE Length: 24
    0000 .... = CR flag: 0
    .... 0000 = Instance: 0
    > EPS Bearer ID (EBI) : 5
    > Fully Qualified Tunnel Endpoint Identifier (F-TEID) : S1-U eNodeB GTP-U interface, TEID/GRE Key: 0x00000001, IPv4 10.250.2.246
    > Cause : Request accepted (16)
  [Response To: 160]
  [Response Time: 0.001753000 seconds]
  
```

Figura 5.30: Modify Bearer Response inviato dall'SGW-C alla MME.

La procedura di instaurazione dei tunnel GTP, a seguito del CUPS, prevede diversi passaggi. Per comprendere meglio il suo funzionamento, in [5.31] sono riportati i numeri di porta e i TEID usati per lo scambio dei messaggi tra le diverse entità e parte del contenuto di questi. Ogni messaggio è contrassegnato da un numero che lo identifica nei diagrammi di sequenza sopra stanti.

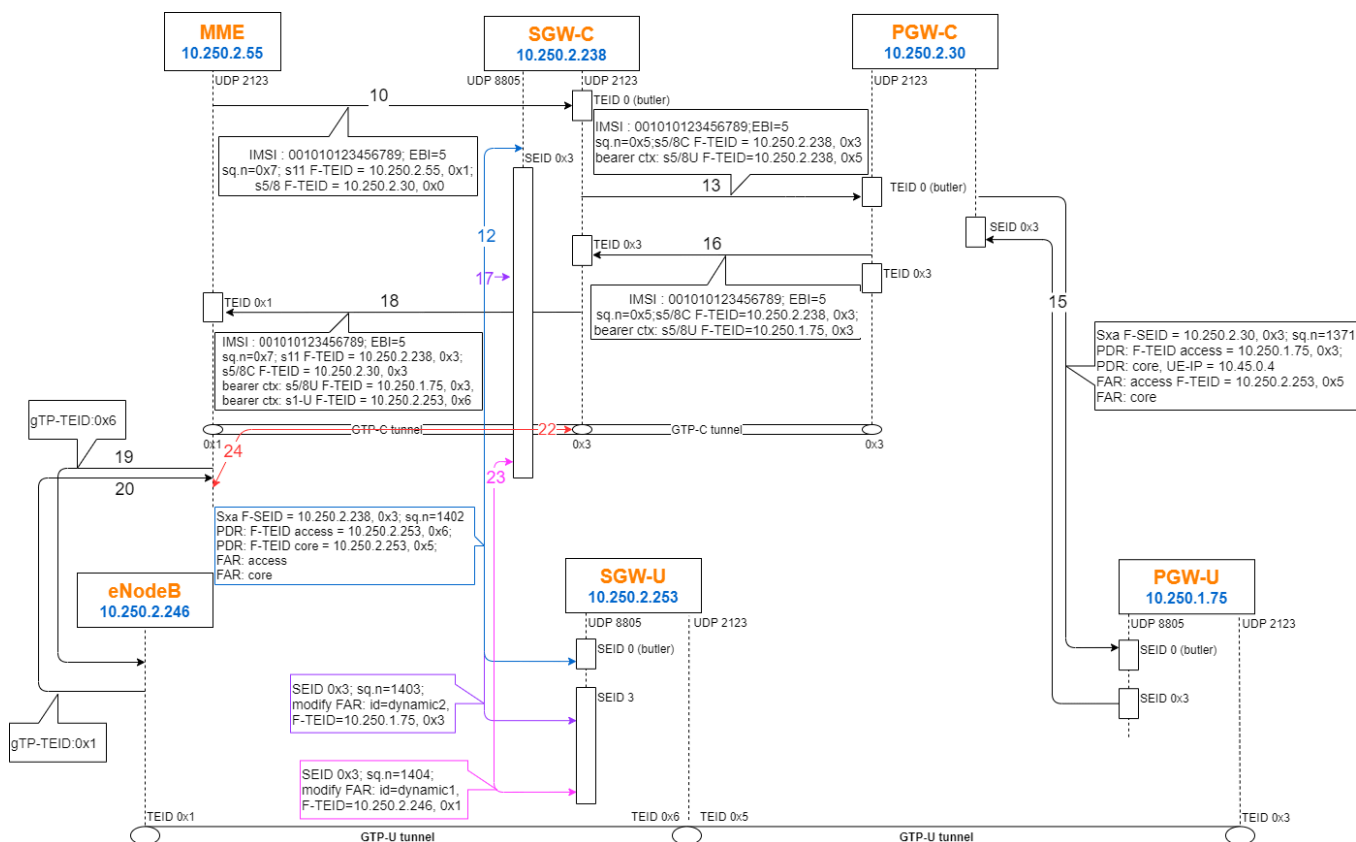


Figura 5.31: Setup dei tunnel GTP

5.1.3 Procedura di collegamento UE con contesto salvato nell'EPC

Il caso in questione avviene quando l'UE contatta una MME a cui si è collegato in precedenza e quindi esiste, nella rete, un contesto associato all'UE. In particolare, nell'**InitialUEMessage** l'UE include il vecchio GUTI, anziché l'IMSI; oltre a ciò, include un'indicazione sul fatto che il GUTI sia nativo e non derivato a partire da P-TMSI¹ e RAI² [5.32]. Questo permette di saltare gli step di autenticazione e iniziare la procedura di instaurazione del default bearer. In riferimento all'immagine [5.3] i passi dal 2-8 saranno saltati nella procedura in questione.

Il resto della procedura prosegue come descritto nella sezione [5.1.2].

¹Identità temporanea, univoca all'interno di una determinata Routing Area, rilasciata a un dispositivo mobile abilitato al GPRS per tracciare la sua posizione all'interno di una rete GSM o UMTS.

²Routing Area Identification

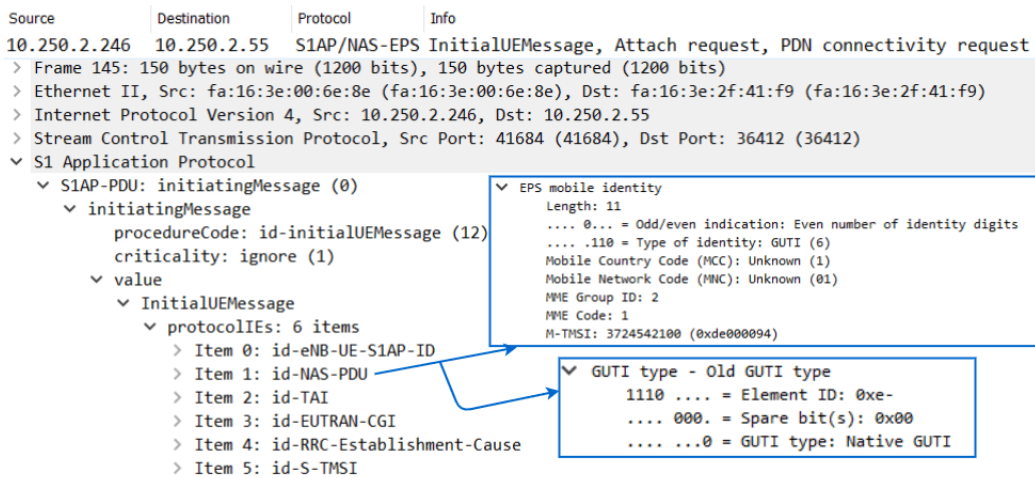


Figura 5.32: Initial UE Message inviato dall'UE alla MME usando il GUTI.

5.1.4 Procedura di abbattimento del collegamento tra UE ed EPC

In questa sezione analizzeremo la procedura di abbattimento del collegamento tra UE ed EPC a fronte dello spegnimento del primo. Tale procedura segue quella di instaurazione del collegamento descritta in [5.1.2].

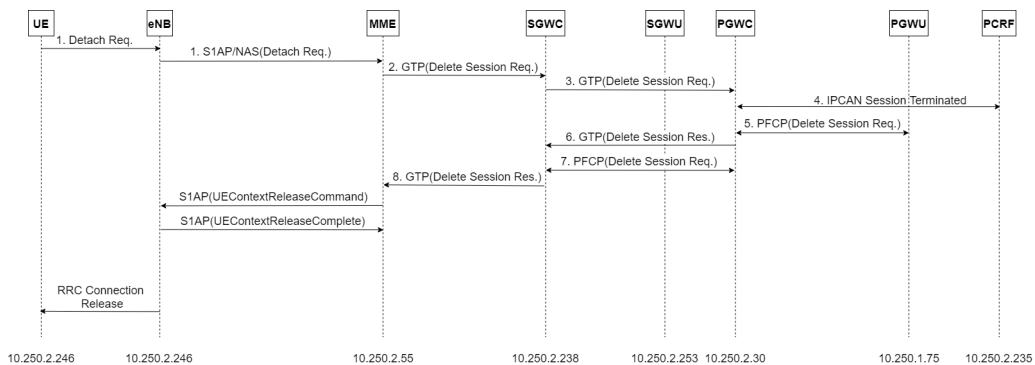


Figura 5.33: Flusso dati per l'abbattimento del collegamento tra UE ed EPC.

1. L'UE invia un messaggio NAS all'eNodeB di tipo **Detach Request** contenente il GUTI e un campo Switch Off che indica se la richiesta di abbattimento è dovuta o meno a uno spegnimento (nel caso in esame avviene a fronte di uno spegnimento). L'eNodeB inoltra questo messaggio alla MME incapsulandolo in un messaggio S1AP contenente il TAI e l'ECGI della cella utilizzata dall'UE [5.34].

Source	Destination	Protocol	Info
10.250.2.246	10.250.2.55	S1AP/NAS-EPS	UplinkNASTransport, Detach request (EPS detach / switch-off)

```

> Frame 192: 130 bytes on wire (1040 bits), 130 bytes captured (1040 bits)
> Ethernet II, Src: fa:16:3e:00:6e:8e (fa:16:3e:00:6e:8e), Dst: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9)
> Internet Protocol Version 4, Src: 10.250.2.246, Dst: 10.250.2.55
> Stream Control Transmission Protocol, Src Port: 41684 (41684), Dst Port: 36412 (36412)
√ S1 Application Protocol
  √ S1AP-PDU: initiatingMessage (0) S1AP Message
    √ initiatingMessage
      procedureCode: id-uplinkNASTransport (13)
      criticality: ignore (1)
      √ value
        √ UplinkNASTransport
          √ protocolIEs: 5 items
            > Item 0: id-MME-UE-S1AP-ID
            > Item 1: id-eNB-UE-S1AP-ID
            > Item 2: id-NAS-PDU
            > Item 3: id-EUTRAN-CGI
            > Item 4: id-TAI
  
```

```

.... 1... = Switch off: Switch off (1)
.... .001 = Detach Type: EPS detach (1)
EPS mobile identity UE Message
  
```

Figura 5.34: Detach Request inviata dall'UE alla MME

2. La MME invia all'SGW-C un messaggio GTP di tipo **Delete Session Request** per eliminare i bearer EPS delle connessioni PDN, associate all'UE in questione, attive con il PGW. Il collegamento è costituito da un bearer con ID, EBI, pari a 0x5 che a sua volta si compone di diversi tunnel GTP come mostrato in [5.31]. Il messaggio inviato dalla MME all'SGW-C contiene l'EBI, informazioni sulla posizione dell'utente e diverse indicazioni. Tra queste ultime, ve n'è una: Idle mode Signalling Reduction Supported Indication, ISRSI, che se attiva indica che il TEID allocato dall'SGW-C per il tunnel GTP-C con la MME non deve essere rilasciato fino a che non gli viene inviato un messaggio di conferma. Nel caso in esame questa opzione è disabilitata [5.35].
3. L'SGW-C invia una richiesta di **Delete Session Request** al PGW-C contenente le stesse informazioni di quella al punto precedente. Per semplicità, non verranno mostrati la struttura e il contenuto del pacchetto in quanto quest'ultimo differisce esclusivamente per gli indirizzi e per il numero di sequenza. Il PGW-C può rilasciare il TEID allocato per il tunnel GTP-C con l'SGW-C.

Source	Destination	Protocol	Info
10.250.2.55	10.250.2.238	GTPv2	Delete Session Request
<ul style="list-style-type: none"> > Frame 181: 84 bytes on wire (672 bits), 84 bytes captured (672 bits) > Ethernet II, Src: fa:16:3e:2f:41:f9 (fa:16:3e:2f:41:f9), Dst: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99) > Internet Protocol Version 4, Src: 10.250.2.55, Dst: 10.250.2.238 > User Datagram Protocol, Src Port: 2123, Dst Port: 2123 ▼ GPRS Tunneling Protocol V2 <ul style="list-style-type: none"> > Flags: 0x48 <ul style="list-style-type: none"> Message Type: Delete Session Request (36) Message Length: 38 Tunnel Endpoint Identifier: 0x00000003 (3) Sequence Number: 0x00000009 (9) Spare: 0 > EPS Bearer ID (EBI) : 5 > User Location Info (ULI) : TAI ECGI > Indication : <ul style="list-style-type: none"> [Response In: 182] 			
<div style="border: 1px solid blue; padding: 2px;"> 0.. = ISRSI (Idle mode Signalling Reduction Supported Indication): False </div>			

Figura 5.35: Delete Session Request inviata dalla MME all'SGW-C

4. Il PGW-C invia al PCRF una richiesta per l'abbattimento della sessione IP-CAN come definito nella specifica tecnica 3GPP 23.203. Il PCRF confermerà l'abbattimento della sessione [5.36].
5. Il PGW-C invia un messaggio di **PFCP Session Deletion Request** al PGW-U sulla sessione PFCP per richiederne l'abbattimento. Il PGW-U rilascerà il SEID allocato e invierà una risposta di conferma, **PFCP Session Deletion Response**. A questo punto, anche il PGW-C rilascerà il SEID allocato.

Source	Destination	Protocol	Info
10.250.2.30	10.250.2.235	DIAMETER	cmd=Credit-Control Request(272) flags=RP-- appl=3GPP Gx(16777238) h2h=50ad8151 e2e=e037ef04
10.250.2.235	10.250.2.30	DIAMETER	cmd=Credit-Control Answer(272) flags=P-- appl=3GPP Gx(16777238) h2h=50ad8151 e2e=e037ef04

```

> Frame 136: 290 bytes on wire (2320 bits), 290 bytes captured (2320 bits)
> Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:77:33:a7 (fa:16:3e:77:33:a7)
> Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.2.235
> Transmission Control Protocol, Src Port: 3868, Dst Port: 33728, Seq: 593, Ack: 409, Len: 224
v Diameter Protocol
  Version: 0x01
  Length: 224
  Flags: 0xc0, Request, Proxyable
  Command Code: 272 Credit-Control
  ApplicationId: 3GPP Gx (16777238)
  Hop-by-Hop Identifier: 0x50ad8151
  End-to-End Identifier: 0xe037ef04
  [Answer In: 137]
  > AVP: Session-Id(263) l=43 f=-M- val=smf.localdomain;1604394499;3;app_gx
  > AVP: Origin-Host(264) l=23 f=-M- val=smf.localdomain
  > AVP: Origin-Realm(296) l=19 f=-M- val=localdomain
  > AVP: Destination-Realm(283) l=19 f=-M- val=localdomain
  > AVP: Auth-Application-Id(258) l=12 f=-M- val=3GPP Gx (16777238)
  > AVP: CC-Request-Type(416) l=12 f=-M- val=TERMINATION_REQUEST (3)
  > AVP: CC-Request-Number(415) l=12 f=-M- val=1
  > AVP: Subscription-Id(443) l=44 f=-M-
  > AVP: Called-Station-Id(30) l=16 f=-M- val=internet

```

Figura 5.36: IP-CAN Session Termination Procedure tra PGW-C e PCRF

Source	Destination	Protocol	Info
10.250.2.30	10.250.1.75	PFCP	PFCP Session Deletion Request
10.250.1.75	10.250.2.30	PFCP	PFCP Session Deletion Response

```

> Frame 139: 58 bytes on wire (464 bits), 58 bytes captured (464 bits)
> Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:98:49:95 (fa:16:3e:98:49:95)
> Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.1.75
> User Datagram Protocol, Src Port: 8805, Dst Port: 8805
v Packet Forwarding Control Protocol
  Flags: 0x21, SEID (S)
  Message Type: PFCP Session Deletion Request (54)
  Length: 12
  SEID: 0x0000000000000003
  Sequence Number: 1373
  Spare: 0
  [Response In: 140]

```

Figura 5.37: PFCP Session Deletion Request/Response tra PGW-C e PGW-U

- Il PGW-C invia una risposta, **Delete Session Response**, all'SGW-C. A questo punto, anche quest'ultimo può rilasciare il TEID associato al tunnel GTP-C con il PGW-C [5.38].

Source	Destination	Protocol	Info
10.250.2.30	10.250.2.238	GTPv2	Delete Session Response
<ul style="list-style-type: none"> > Frame 141: 60 bytes on wire (480 bits), 60 bytes captured (480 bits) > Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:26:ce:99 (fa:16:3e:26:ce:99) > Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.2.238 > User Datagram Protocol, Src Port: 2123, Dst Port: 2123 ▼ GPRS Tunneling Protocol V2 <ul style="list-style-type: none"> > Flags: 0x48 <ul style="list-style-type: none"> Message Type: Delete Session Response (37) Message Length: 14 Tunnel Endpoint Identifier: 0x00000003 (3) Sequence Number: 0x00000006 (6) Spare: 0 > Cause : Request accepted (16) <ul style="list-style-type: none"> [Response To: 135] [Response Time: 0.006219000 seconds] 			

Figura 5.38: Delete Session Response inviata dal PGW-C all'SGW-C

7. L'SGW-C esegue una procedura, analoga a quella eseguita dal PGW-C al punto 5, per l'abbattimento della sessione PFCP con l'SGW-U. Per semplicità, non verranno mostrati la struttura e il contenuto del pacchetto in quanto differisce esclusivamente per gli indirizzi e per il numero di sequenza.
8. L'SGW-C invia una risposta, **Delete Session Request**, alla MME. A questo punto, anche quest'ultimo può rilasciare il TEID associato al tunnel GTP-C con l'SGW-C. Per lo stesso motivo del punto precedente la struttura e il contenuto del pacchetto non verranno mostrati.

I messaggi successivi raffigurati in [5.33] sono dedicati all'abbattimento del collegamento tra eNodeB e MME e a quello dell'interfaccia aerea.

5.1.5 Procedura di collegamento del gNodeB all'AMF

Il gNodeB stabilisce un collegamento con l'AMF inviandogli un messaggio di **NGSetupRequest**. In questo messaggio specifica il suo id globale, che si compone del suo ID più quello della PLMN a cui appartiene, le tracking area supportate e il paging cycle di default [5.39].

Source	Destination	Protocol	Info
10.250.2.165	10.250.2.85	NGAP	NGSetupRequest
<ul style="list-style-type: none"> > Frame 49: 110 bytes on wire (880 bits), 110 bytes captured (880 bits) > Ethernet II, Src: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69) > Internet Protocol Version 4, Src: 10.250.2.165, Dst: 10.250.2.85 > Stream Control Transmission Protocol, Src Port: 57282 (57282), Dst Port: 38412 (38412) √ NG Application Protocol <ul style="list-style-type: none"> √ NGAP-PDU: initiatingMessage (0) <ul style="list-style-type: none"> √ initiatingMessage <ul style="list-style-type: none"> procedureCode: id-NGSetup (21) criticality: reject (0) √ value <ul style="list-style-type: none"> √ NGSetupRequest <ul style="list-style-type: none"> √ protocolIEs: 3 items <ul style="list-style-type: none"> > Item 0: id-GlobalRANNodeID > Item 1: id-SupportedTAList > Item 2: id-DefaultPagingDRX 			

Figura 5.39: NGSetupRequest inviata dal gNodeB all'AMF

L'AMF risponde con un messaggio di **NGSetupResponse** specificando il suo nome, che il gNodeB potrebbe usare per identificarla, la lista delle PLMN supportate, che si compone dell'ID della PLMN più gli slice, che possono essere utilizzati, di questa, la lista dei GUAMI serviti e la sua capacità relativa [5.40].

Source	Destination	Protocol	Info
10.250.2.85	10.250.2.165	NGAP	NGSetupResponse
<ul style="list-style-type: none"> > Frame 51: 118 bytes on wire (944 bits), 118 bytes captured (944 bits) > Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1) > Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.2.165 > Stream Control Transmission Protocol, Src Port: 38412 (38412), Dst Port: 57282 (57282) √ NG Application Protocol <ul style="list-style-type: none"> √ NGAP-PDU: successfulOutcome (1) <ul style="list-style-type: none"> √ successfulOutcome <ul style="list-style-type: none"> procedureCode: id-NGSetup (21) criticality: reject (0) √ value <ul style="list-style-type: none"> √ NGSetupResponse <ul style="list-style-type: none"> √ protocolIEs: 4 items <ul style="list-style-type: none"> > Item 0: id-AMFName > Item 1: id-ServedGUAMList > Item 2: id-RelativeAMFCapacity > Item 3: id-PLMNSupportList 			

Figura 5.40: NGSetupResponse inviata dall'AMF al gNodeB

5.1.6 Procedura di registrazione dell'UE al 5GC

Prima di entrare nel vivo delle procedure 5G è necessario chiarire un piccolo aspetto. A differenza di quanto avviene nell'EPC, i collegamenti tra i blocchi funzionali del core non sono specificati nei file di configurazione; eccezion

fatta per il reference point N4 tra UPF e SMF. Ciascun blocco conosce solo l'indirizzo e la porta del NRF e colloquia con questo per ottenere le posizioni degli altri. Per semplicità, le richieste di discovery inviate al NRF e le relative risposte non vengono mostrate, ma un esempio è raffigurato in [5.41].

Source	Destination	Protocol	Info
10.250.2.85	10.250.1.251	HTTP	GET /nnrf-disc/v1/nf-instances?requester-nf-type=AMF&target-nf-type=SMF HTTP/1.1
10.250.1.251	10.250.2.85	HTTP	HTTP/1.1 200 OK (application/json)

```

> Frame 91: 640 bytes on wire (5120 bits), 640 bytes captured (5120 bits)
> Ethernet II, Src: fa:16:3e:1d:c2:67 (fa:16:3e:1d:c2:67), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69)
> Internet Protocol Version 4, Src: 10.250.1.251, Dst: 10.250.2.85
> Transmission Control Protocol, Src Port: 7777, Dst Port: 60824, Seq: 427, Ack: 1037, Len: 574
> [2 Reassembled TCP Segments (736 bytes): #89(162), #91(574)]
> Hypertext Transfer Protocol
< JavaScript Object Notation: application/json
  Object
    Member Key: validityPeriod
    Member Key: nfInstances
      Array
        Object
          Member Key: nfInstanceId
          Member Key: nfType
          Member Key: nfStatus
          Member Key: ipv4Addresses
          Member Key: nfServices
        Key: nfInstances
  
```

Figura 5.41: Richiesta da parte dell'AMF al NRF dell'indirizzo e della porta del SMF e relativa risposta

Anche in questo caso, per semplicità, suddivideremo il flusso dei messaggi in due fasi. La trattazione farà riferimento alla procedura di registrazione in [12].

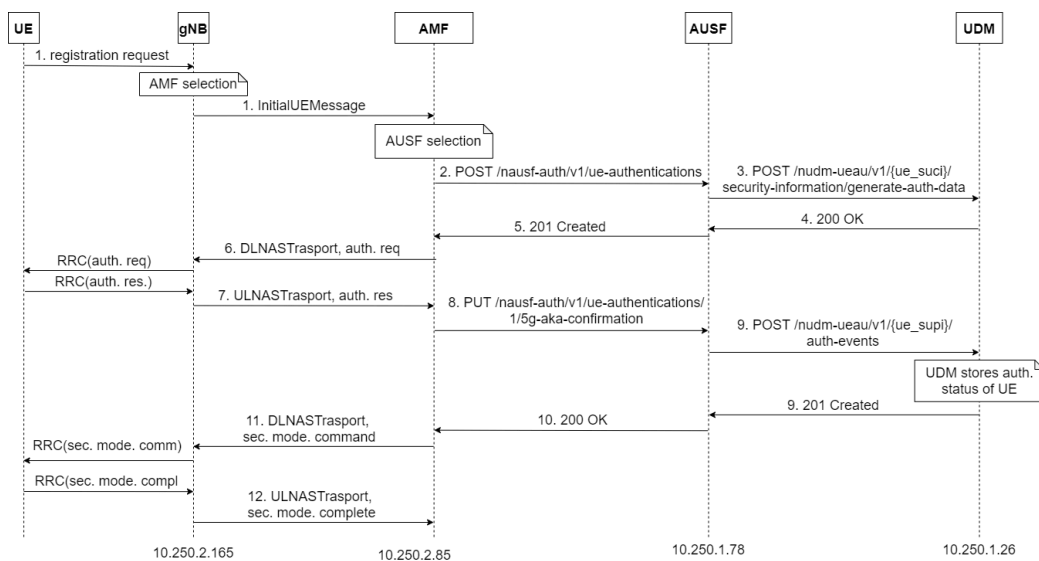


Figura 5.42: Flusso dati per la procedura di registrazione dell'UE al 5GC. Parte 1.

1. In maniera simile a quanto avviene nell'EPC, l'UE invia un messaggio di tipo **Registration Request** alla gNodeB che contiene diverse informazioni, tra cui:

- registration type - initial registration (caso in esame), mobility registration, periodic registration o emergency registration;
- user identity - SUPI³, SUCI⁴, PEI⁵ o 5G-GUTI mappato a partire dal 4G-GUTI nel caso di interoperabilità tra 4G e 5G;
- security capability - algoritmi per la cifratura e per garantire l'integrità supportati dall'UE;
- requested NSSAI - NSSAI su cui l'UE vuole essere collocato.

Tramite il SUCI o l'esistenza di una connessione N2 per l'UE, la gNodeB seleziona l'AMF. Nel caso queste informazioni non siano disponibili, viene selezionata l'AMF predefinita in base all'NSSAI se SUPI o PEI sono inclusi nel messaggio. Dopo aver selezionato l'AMF, il gNodeB invia a questo un messaggio NGAP, **InitialUEMessage** che incapsula

³Subscription Permanent Identifier è un identificatore 5G univoco e globale assegnato a ciascun abbonato. Il valore SUPI è condiviso tra la USIM e la funzione UDM/UDR del 5GC.

⁴Subscription Concealed Identifier è un identificatore che nasconde il SUPI al fine di garantire la privacy dell'abbonato.

⁵Identificatore tramite cui la rete riconosce un UE.

la richiesta di registrazione dell'UE. Questo messaggio include l'identità della cella a cui l'UE vuole collegarsi [5.43].

2. A questo punto ha inizio l'AKA 5G che ha il compito di generare una chiave di ancoraggio, K_{SEAF} che verrà fornita dall'AUSF della rete domestica alla Security Anchor Function, SEAF, che nel caso in esame è situata nell'AMF. K_{SEAF} viene derivata da una chiave intermedia K_{AUSF} . Le chiavi per più di un contesto di sicurezza possono essere derivate da K_{SEAF} senza la necessità di una nuova procedura AKA [13]. Tramite SUPI o SUCI, l'AMF seleziona un AUSF come descritto in TS 23.501, clausola 6.3.4. L'AMF invoca il servizio **Nausf_UEAuthentication** tramite HTTP POST (**/nausf-auth/v1/ue-authentications**) includendo il SUPI o SUCI dell'UE (nel caso in analisi si utilizza il SUCI) e l'SN id [5.44].
3. l'AUSF, per prima cosa, verifica che il SEAF richiedente abbia il diritto di utilizzare l'SN id, in caso contrario invierà una risposta contenente l'indicazione *rete di servizio non autorizzata*. L'AUSF memorizza temporaneamente l'SN id ricevuto. Nel caso la verifica vada a buon fine, esegue un'HTTP POST specificando come parametro del percorso il SUCI dell'UE: **/nudm-ueau/v1/suci-0-901-70-0000-0-0-0000000003/security-information/generate-auth-data**. Il corpo della richiesta contiene l'SN id e l'id dell'AUSF richiedente [5.45].

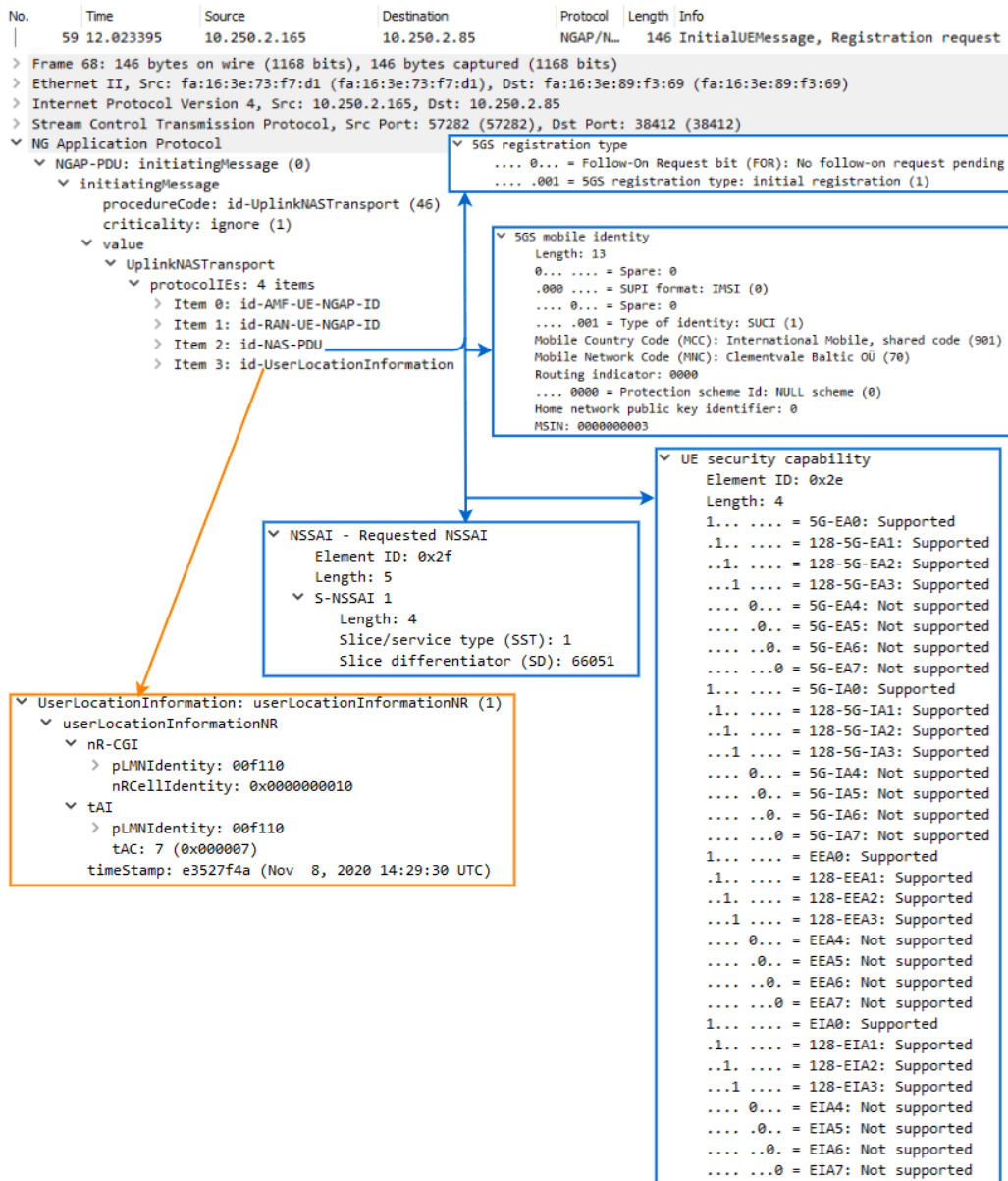


Figura 5.43: InitialUEMessage tra l'UE e l'AMF.

Source	Destination	Protocol	Length	Info
10.250.2.85	10.250.1.78	HTTP	366	POST /nausf-auth/v1/ue-authentications HTTP/1.1 (application/json)
> Frame 60: 366 bytes on wire (2928 bits), 366 bytes captured (2928 bits)				
> Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:08:34:66 (fa:16:3e:08:34:66)				
> Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.1.78				
> Transmission Control Protocol, Src Port: 39934, Dst Port: 7777, Seq: 1, Ack: 1, Len: 300				
> Hypertext Transfer Protocol				
JavaScript Object Notation: application/json				
Object				
Member Key: supiOrSuci				
String value: suci-0-901-70-0000-0-0-0000000003				
Key: supiOrSuci				
Member Key: servingNetworkName				
String value: 5G:mnc001.mcc001.3gppnetwork.org				
Key: servingNetworkName				

Figura 5.44: AuthenticationRequest tra SEAF e l'AUSF.

Source	Destination	Protocol	Info
10.250.1.78	10.250.1.26	HTTP	POST /nudm-ueau/v1/suci-0-901-70-0000-0-0-0000000003/security-information/generate-auth-data HTTP/1.1 (application/json)
> Frame 19: 419 bytes on wire (3352 bits), 419 bytes captured (3352 bits)			
> Ethernet II, Src: fa:16:3e:08:34:66 (fa:16:3e:08:34:66), Dst: fa:16:3e:80:9f:ea (fa:16:3e:80:9f:ea)			
> Internet Protocol Version 4, Src: 10.250.1.78, Dst: 10.250.1.26			
> Transmission Control Protocol, Src Port: 38862, Dst Port: 7777, Seq: 1, Ack: 1, Len: 353			
> Hypertext Transfer Protocol			
JavaScript Object Notation: application/json			
Object			
Member Key: servingNetworkName			
String value: 5G:mnc001.mcc001.3gppnetwork.org			
Key: servingNetworkName			
Member Key: ausfInstanceId			
String value: 12c7f61c-1cfa-41eb-a30d-1590f6e8b9e6			
Key: ausfInstanceId			

Figura 5.45: UEAuthenticationRequest tra l'AUSF e l'UDM.

4. l'UDM invia una risposta HTTP 200 OK all'AUSF che contiene un array ordinato di n AV (1 ... n). Se $n > 1$, gli AV vengono ordinati in base al numero di sequenza [5.46]. 3GPP raccomanda $n=1$. I campi di un AV sono analoghi a quelli del 4G, l'unica differenza è che la chiave K_{ASME} è sostituita dalla K_{AUSF} ; ricapitolando, i campi di un AV sono:

- RAND;
- XRES*;
- AUTN;
- K_{AUSF} .

Anche in questo caso, per la procedura di generazione dei diversi campi è possibile fare riferimento all'immagine [5.7].

No.	Time	Source	Destination	Protocol	Length	Info
22	7.671499	10.250.1.26	10.250.1.78	HTTP/1.1	367	HTTP/1.1 200 OK , JavaScript Object Notation (application/json)
<pre> > Frame 22: 367 bytes on wire (2936 bits), 367 bytes captured (2936 bits) > Ethernet II, Src: fa:16:3e:80:9f:ea (fa:16:3e:80:9f:ea), Dst: fa:16:3e:80:34:66 (fa:16:3e:80:34:66) > Internet Protocol Version 4, Src: 10.250.1.26, Dst: 10.250.1.78 > Transmission Control Protocol, Src Port: 7777, Dst Port: 38862, Seq: 134, Ack: 354, Len: 301 > [2 Reassembled TCP Segments (434 bytes): #20(133), #22(301)] > Hypertext Transfer Protocol JavaScript Object Notation: application/json Object Member Key: authType Member Key: authenticationVector </pre>						
<pre> Object Member Key: avType String value: 5G_HE_AKA Key: avType Member Key: rand String value: 81b987f130a86abe9043a6cfc35b49cc Key: rand SQN xor AK Member Key: autn String value: 28da684c7cd1800005c261bdd7f76537 Key: autn AMF MAC Member Key: xresStar String value: 81fe647a3faf34a8f0cb39fe72533eae Key: xresStar Member Key: kausf String value: feb6454e0bc874d08bf427c8e201bc0e3878ea30f8501dabb3d5b2bacb3ae44 Key: kausf Key: authenticationVector </pre>						

Figura 5.46: UEAuthenticationResponse tra l'UDM e l'AUSF.

- L'AUSF computa un hash dell'XRES*, HXRES*, memorizza la K_{AUSF} e invia alla SEAF una risposta HTTP 201 Created contenente la sfida, RAND, l'hash del risultato atteso, HXRES e il token AUTN [5.47].

No.	Time	Source	Destination	Protocol	Length	Info
25	7.671959	10.250.1.78	10.250.2.85	HTTP	381	HTTP/1.1 201 Created (application/3gpphal+json)
<pre> > Frame 25: 381 bytes on wire (3048 bits), 381 bytes captured (3048 bits) > Ethernet II, Src: fa:16:3e:80:34:66 (fa:16:3e:80:34:66), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69) > Internet Protocol Version 4, Src: 10.250.1.78, Dst: 10.250.2.85 > Transmission Control Protocol, Src Port: 7777, Dst Port: 39934, Seq: 217, Ack: 301, Len: 315 > [2 Reassembled TCP Segments (531 bytes): #24(216), #25(315)] > Hypertext Transfer Protocol Media Type Media type: application/3gpphal+json (315 bytes) </pre>						
<pre> { "authType": "5G_AKA", "5gAuthData": { "rand": "81b987f130a86abe9043a6cfc35b49cc", "xresStar": "61b37f77990695a9737eb8d7180d7de6", "autn": "28da684c7cd1800005c261bdd7f76537" }, "_links": { "5g-aka": { "href": "http://10.250.1.78:7777/nausf-auth/v1/ue-authentications/1/5g-aka-confirmation" } } } </pre>						

Figura 5.47: AuthenticationResponse tra l'AUSF e SEAF.

- Il SEAF, memorizzerà HXRES* e inoltrerà in modo trasparente il messaggio di richiesta ricevuto verso l'UE incapsulandolo in una richiesta di autenticazione NAS. L'ME riceverà il messaggio NAS e inoltrerà all'USIM RAND e AUTN. Il SEAF inserirà nella richiesta NAS due parametri aggiuntivi: ngKSI e ABBA [5.48]. Il primo, verrà utilizzato dall'UE e dall'AMF per identificare il contesto di sicurezza nativo parziale che verrà creato se l'autenticazione ha successo. Il secondo,

Anti-Bidding down Between Architectures, fornisce protezione contro la riduzione delle funzionalità di sicurezza da una versione superiore a una inferiore.

```

No.      Time           Source            Destination      Protocol Length Info
...      ...           ...              ...              ...      ...    ...
67      12.034182    10.250.2.85      10.250.2.165    NGAP/N... 146 DownlinkNASTransport, Authentication request
> Frame 77: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits)
> Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1)
> Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.2.165
> Stream Control Transmission Protocol, Src Port: 38412 (38412), Dst Port: 57282 (57282)
v NG Application Protocol
  v NGAP-PDU: initiatingMessage (0)
    v initiatingMessage
      procedureCode: id-DownlinkNASTransport (4)
      criticality: ignore (1)
      v value
        v DownlinkNASTransport
          v protocolIEs: 3 items
            > Item 0: id-AMF-UE-NGAP-ID
            > Item 1: id-RAN-UE-NGAP-ID
            > Item 2: id-NAS-PDU
              v NAS key set identifier - ngKSI
                ... 0... = Type of security context flag (TSC): Native security context (for KSIAMF)
                ... .000 = NAS key set identifier: 0
              v ABBA
                Length: 2
                ABBA Contents: 0x0000
              v Authentication Parameter RAND - 5G authentication challenge
                Element ID: 0x21
                RAND value: 81b987f130a86abe9043a6cfc35b49cc
              v Authentication Parameter AUTN (UMTS and EPS authentication challenge) - 5G authentication challenge
                Element ID: 0x20
                Length: 16
                v AUTN value: 28da684c7cd1800005c261bdd7f76537
                  SQN xor AK: 28da684c7cd1
                  AMF: 8000
                  MAC: 05c261bdd7f76537
  
```

Figura 5.48: DownlinkNASTransport, AuthenticationRequest tra SEAF e l'UE.

7. A questo punto, l'USIM esegue la procedura di verifica vista per il 4G [5.9] per la computazione del risultato RES*. In particolare, tramite K e RAND computa l'AK. Estrapolando dall'AUTN il campo SQN xor AK e mettendolo in xor con l'AK calcolato è in grado di ottenere l'SQN. Tramite quest'ultimo, l'USIM è in grado di calcolare il MAC atteso, $XMAC = f_{1K}(SQN||RAND||AMF)$ e verificare che questo sia uguale al MAC contenuto nell'AUTN. Un'ulteriore verifica che esegue l'USIM è che l'SQN sia in un range corretto. Nel caso in cui le verifiche vadano a buon fine, l'USIM computa la chiave crittografica, CK, la chiave per l'integrità, IK, e il risultato della sfida, RES*, e li invia all'ME. Questo inoltra il messaggio di risposta contenente esclusivamente il RES* al SEAF [5.49].

```

No.      Time           Source           Destination      Protocol Length Info
...
68 12.225105 10.250.2.165    10.250.2.85     NGAP/NL... 146 UplinkNASTransport, Authentication response
> Frame 68: 146 bytes on wire (1168 bits), 146 bytes captured (1168 bits)
> Ethernet II, Src: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69)
> Internet Protocol Version 4, Src: 10.250.2.165, Dst: 10.250.2.85
> Stream Control Transmission Protocol, Src Port: 57282 (57282), Dst Port: 38412 (38412)
< NG Application Protocol
  < NGAP-PDU: initiatingMessage (0)
    < initiatingMessage
      procedureCode: id-UplinkNASTransport (46)
      criticality: ignore (1)
      < value
        < UplinkNASTransport
          < protocolIEs: 4 items
            > Item 0: id-AMF-UE-NGAP-ID
            > Item 1: id-RAN-UE-NGAP-ID
            > Item 2: id-NAS-PDU
            > Item 3: id-UserLocationInformation
          < Authentication response parameter
            Element ID: 0x2d
            Length: 16
            RES: 81fe647a3faf34a8f0cb39fe72533eae

```

Figura 5.49: UplinklinkNASTransport, AuthenticationResponse tra l'UE e SEAF.

- Il SEAF computerà HRES*, come descritto in Annex A.5 in [13], a partire da RES* e lo confronterà con HXRES*; se questi coincidono, la procedura di autenticazione, per la rete di servizio, è andata a buon fine. Dopo la verifica, effettuerà una HTTP PUT (**/nausf-auth/v1/ue-authentications/1/5g-aka-confirmation**) verso l'AUSF inoltrando nel corpo della richiesta il RES* ottenuto dall'UE [5.50].

```

No.      Time           Source           Destination      Protocol Length Info
...
69 12.225676 10.250.2.85     10.250.1.78     HTTP           315 PUT /nausf-auth/v1/ue-authentications/1/5g-aka-confirmation HTTP/1.1 (application/json)
> Frame 69: 315 bytes on wire (2520 bits), 315 bytes captured (2520 bits)
> Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:08:34:66 (fa:16:3e:08:34:66)
> Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.1.78
> Transmission Control Protocol, Src Port: 39934, Dst Port: 7777, Seq: 301, Ack: 532, Len: 249
< Hypertext Transfer Protocol
  < JavaScript Object Notation: application/json
    < Object
      < Member Key: resStar
        String value: 81fe647a3faf34a8f0cb39fe72533eae
        Key: resStar

```

Figura 5.50: AuthenticationResponse tra SEAF e l'AUSF.

- L'AUSF validerà il risultato; in caso di esito negativo informerà il SEAF tramite un errore, in caso di esito positivo considererà l'autenticazione valida dal punto di vista della home network e procederà come descritto nei punti seguenti. L'AUSF informerà l'UDM tramite HTTP POST (**/nudm-ueau/v1/imsi-90170000000003/auth-events**) del risultato dell'autenticazione [5.51]. L'UDM memorizzerà lo stato dell'autenticazione dell'UE e risponderà con un HTTP 201 Created.

```

No.    Time          Source           Destination      Protocol    Length  Info
---    -
29 7.864987      10.250.1.78     10.250.1.26     HTTP        466    POST /nudm-ueau/v1/imsi-90170000000003/auth-events HTTP/1.1 (application/json)
> Frame 29: 466 bytes on wire (3728 bits), 466 bytes captured (3728 bits)
> Ethernet II, Src: fa:16:3e:08:34:66 (fa:16:3e:08:34:66), Dst: fa:16:3e:80:9f:ea (fa:16:3e:80:9f:ea)
> Internet Protocol Version 4, Src: 10.250.1.78, Dst: 10.250.1.26
> Transmission Control Protocol, Src Port: 38862, Dst Port: 7777, Seq: 354, Ack: 435, Len: 400
> Hypertext Transfer Protocol
▼ JavaScript Object Notation: application/json
  ▼ Object
    > Member Key: nfInstanceId
      ▼ Member Key: success
        True value
        Key: success
      > Member Key: timeStamp
      ▼ Member Key: authType
        String value: 5G_AKA
        Key: authType
    > Member Key: servingNetworkName

```

Figura 5.51: AuthenticationResponse tra l'AUSF e l'UDM.

10. L'AUSF deriva a partire dalla K_{AUSF} la K_{SEAF} come descritto nella clausola A.6 in [13] e invia al SEAF una risposta HTTP 200 OK contenente nel corpo: l'informazione di esito con successo della procedura di autenticazione, il SUPI e la K_{SEAF} . K_{SEAF} diventa la chiave di ancoraggio per generare le chiavi della gerarchia raffigurata in [5.53] al fine di garantire l'integrità e la cifratura dei messaggi [5.52].

```

No.    Time          Source           Destination      Protocol    Length  Info
---    -
75 12.231879     10.250.1.78     10.250.2.85     HTTP        220    HTTP/1.1 200 OK (application/json)
> Frame 75: 220 bytes on wire (1760 bits), 220 bytes captured (1760 bits)
> Ethernet II, Src: fa:16:3e:08:34:66 (fa:16:3e:08:34:66), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69)
> Internet Protocol Version 4, Src: 10.250.1.78, Dst: 10.250.2.85
> Transmission Control Protocol, Src Port: 7777, Dst Port: 39934, Seq: 665, Ack: 550, Len: 154
> [2 Reassembled TCP Segments (287 bytes): #73(133), #75(154)]
> Hypertext Transfer Protocol
▼ JavaScript Object Notation: application/json
  ▼ Object
    ▼ Member Key: authResult
      String value: AUTHENTICATION_SUCCESS
      Key: authResult
    ▼ Member Key: supi
      String value: imsi-90170000000003
      Key: supi
    ▼ Member Key: kseaf
      String value: 22ac7ccc1aa2a398935f4078d636d80eb341f1a7c905f77a08a43dd96114647b
      Key: kseaf

```

Figura 5.52: AuthenticationResponse tra l'AUSF e SEAF.

11. In maniera simile al 4G, l'AMF invia all'UE un messaggio **NAS Security Mode Command** contenente: le capacità di sicurezza dell'UE specificate nel messaggio iniziale (punto 1), gli algoritmi NAS selezionati e il NAS key set identifier, ngKSI. L'AMF può richiedere, come in questo caso, all'UE di specificare l'IMEISV (PEI)⁶. L'header del

⁶l'International Mobile Station Equipment Identity Software Version è un codice che identifica univocamente un dispositivo mobile e la versione del software di questo

messaggio NAS riporta la dicitura *Integrity protected with new 5GS security context* ad indicare che l'integrità del messaggio è stata protetta usando l'apposita chiave NAS [5.54].

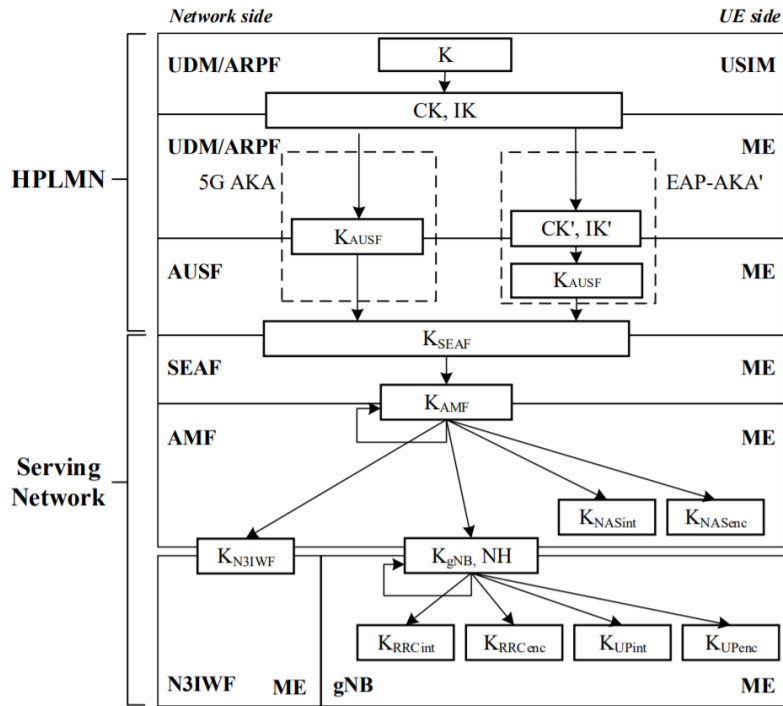


Figura 5.53: Gerarchia generazione chiavi in un sistema 5G [13].

12. A questo punto l'UE genera le chiavi NAS, verifica l'integrità del messaggio ricevuto e invia un messaggio **NAS Security Mode Complete** cifrato la cui integrità è protetta. Il corpo del messaggio conterrà informazioni simili a quelle dell'InitialUEMessage e in aggiunta l'IMEISV [5.55].

No.	Time	Source	Destination	Protocol	Length	Info
77	12.232264	10.250.2.85	10.250.2.165	NGAP/NAS-5GS	126	DownlinkNASTransport, Security mode command

```

> Frame 77: 126 bytes on wire (1008 bits), 126 bytes captured (1008 bits)
> Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1)
> Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.2.165
> Stream Control Transmission Protocol, Src Port: 38412 (38412), Dst Port: 57282 (57282)
v NG Application Protocol
  v NGAP-PDU: initiatingMessage (0)
    v initiatingMessage
      procedureCode: id-DownlinkNASTransport (4)
      criticality: ignore (1)
      v value
        v DownlinkNASTransport
          v protocolIEs: 3 items
            > Item 0: id-AMF-UE-NGAP-ID
            > Item 1: id-RAN-UE-NGAP-ID
            > Item 2: id-NAS-PDU
          v Security protected NAS 5GS message
            Extended protocol discriminator: 5G mobility management messages (126)
            0000 .... = Spare Half Octet: 0
            .... 0011 = Security header type: Integrity protected with new 5GS security context (3)
            Message authentication code: 0xd817d0ec
            Sequence number: 0
          v Plain NAS 5GS Message
            Extended protocol discriminator: 5G mobility management messages (126)
            0000 .... = Spare Half Octet: 0
            .... 0000 = Security header type: Plain NAS message, not security protected (0)
            Message type: Security mode command (0x5d)
            v NAS security algorithms
              0000 .... = Spare Half Octet: 0
            v NAS key set identifier - ngKSI
            v UE security capability - Replayed UE security capabilities
            v IMEISV request
            v Additional 5G security information
  
```

Figura 5.54: SecurityModeCommand tra l'AMF e l'UE.

No.	Time	Source	Destination	Protocol	Length	Info
78	12.321857	10.250.2.165	10.250.2.85	NGAP/NAS-5GS/NAS...	190	UplinkNASTransport, Security mode complete, Registration request

```

> Frame 78: 190 bytes on wire (1520 bits), 190 bytes captured (1520 bits)
> Ethernet II, Src: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69)
> Internet Protocol Version 4, Src: 10.250.2.165, Dst: 10.250.2.85
> Stream Control Transmission Protocol, Src Port: 57282 (57282), Dst Port: 38412 (38412)
v NG Application Protocol
  v NGAP-PDU: initiatingMessage (0)
    v initiatingMessage
      procedureCode: id-UplinkNASTransport (46)
      criticality: ignore (1)
      v value
        v UplinkNASTransport
          v protocolIEs: 4 items
            > Item 0: id-AMF-UE-NGAP-ID
            > Item 1: id-RAN-UE-NGAP-ID
            > Item 2: id-NAS-PDU
            > Item 3: id-UserLocationInformation
          v Security protected NAS 5GS message
            Extended protocol discriminator: 5G mobility management messages (126)
            0000 .... = Spare Half Octet: 0
            .... 0100 = Security header type: Integrity protected and ciphered with new 5GS security context (4)
            Message authentication code: 0x5cbbcbba
            Sequence number: 0
          v 5GS mobile identity
            Element ID: 0x77
            Length: 8
            .... 1... = Odd/even indication: Odd number of identity digits
            .... .101 = Type of identity: IMEISV (5)
            IMEISV: 356938035643803
  
```

Figura 5.55: SecurityModeComplete tra l'UE e l'AMF.

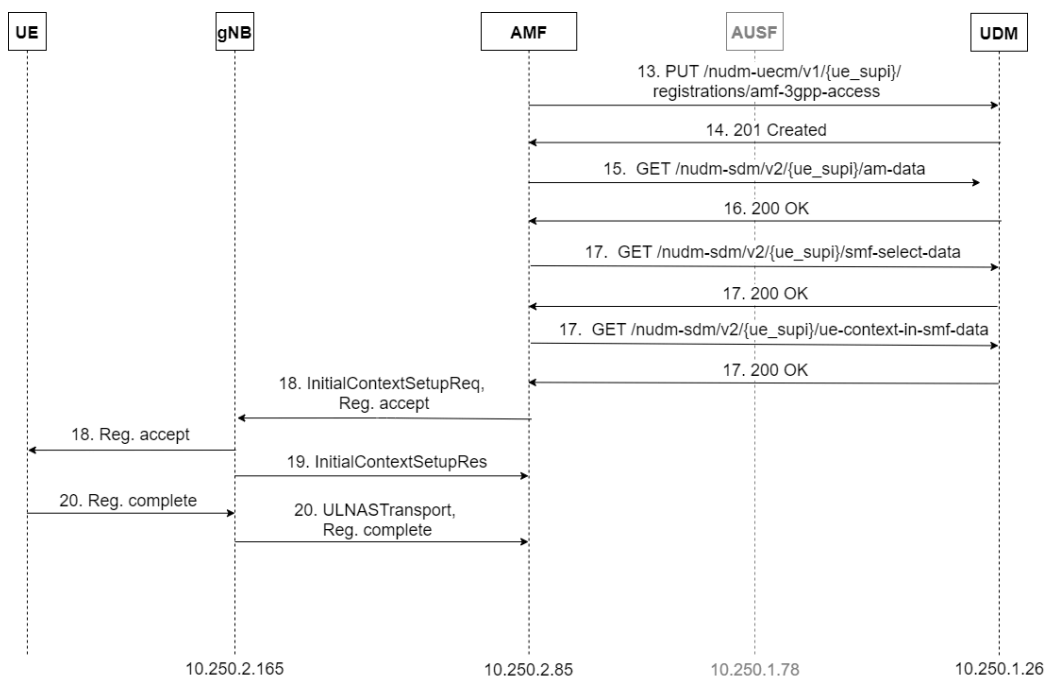


Figura 5.56: Flusso dati per la procedura di registrazione dell'UE al 5GC. Parte 2.

13. L'AMF si registra presso l'UDM tramite una richiesta HTTP PUT specificando come parametro del percorso il SUPI (imsi-901700000000003) dell'UE: **/nudm-uecm/v1/imsi-901700000000003/registrations/amf-3gpp-access**). Il corpo della richiesta contiene l'id dell'istanza AMF e il GUAMI, al fine di identificare univocamente l'AMF. Il messaggio contiene anche un campo che specifica l'URI del vecchio AMF affinché l'UDM possa invocare l'API di de-registrazione; questo avviene nel caso in cui l'UE cambi l'AMF a cui è connesso [5.57].

Source	Destination	Protocol	Length	Info
10.250.2.85	10.250.1.26	HTTP	535	PUT /nudm-uecm/v1/imsi-90170000000003/registrations/amf-3gpp-access HTTP/1.1
> Frame 79: 535 bytes on wire (4280 bits), 535 bytes captured (4280 bits)				
> Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:80:9f:ea (fa:16:3e:80:9f:ea)				
> Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.1.26				
> Transmission Control Protocol, Src Port: 48004, Dst Port: 7777, Seq: 1, Ack: 1, Len: 469				
> Hypertext Transfer Protocol				
JavaScript Object Notation: application/json				String value: a92a8484-21ce-41eb-a6ae-ad8636e86056 Key: amfInstanceId
Object				
> Member Key: amfInstanceId				
> Member Key: deregCallbackUri				
> Member Key: guami				
> Member Key: ratType				

Member Key	Value
plmnId	Object
mcc	String value: 001 Key: mcc
mnc	String value: 01 Key: mnc
plmnId	Key: plmnId
amfId	String value: 020040 Key: amfId
guami	Key: guami

Figura 5.57: RegistrationRequest tra l'AMF e l'UDM.

- L'UDM aggiornerà la sottoscrizione dell'AMF nel caso sia presente e risponderà con un HTTP 200 OK o 204 No Content. Nel caso la risorsa non sia presente, l'UDM memorizza i dati ricevuti dall'AMF e risponde con un HTTP 201 Created; il corpo della risposta conterrà le informazioni ricevute ed eventualmente informazioni aggiuntive che potrebbero essere di interesse per la NF consumatrice (l'AMF in questo caso) [5.58]. Nel caso la sottoscrizione non possa essere autorizzata, la risposta presenterà un codice HTTP 403 Forbidden.

No.	Time	Source	Destination	Protocol	Length	Info
82	12.325270	10.250.1.26	10.250.2.85	HTTP	326	HTTP/1.1 201 Created (application/json)
> Frame 82: 326 bytes on wire (2608 bits), 326 bytes captured (2608 bits)						
> Ethernet II, Src: fa:16:3e:80:9f:ea (fa:16:3e:80:9f:ea), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69)						
> Internet Protocol Version 4, Src: 10.250.1.26, Dst: 10.250.2.85						
> Transmission Control Protocol, Src Port: 7777, Dst Port: 48004, Seq: 238, Ack: 470, Len: 260						
> [2 Reassembled TCP Segments (497 bytes): #80(237), #82(260)]						
> Hypertext Transfer Protocol						
JavaScript Object Notation: application/json						
Object						
> Member Key: amfInstanceId						
> Member Key: deregCallbackUri						
> Member Key: guami						
> Member Key: ratType						

Figura 5.58: RegistrationResponse tra l'UDM e l'AMF.

- Dopo la registrazione, l'AMF richiede all'UDM i dati legati all'accesso e alla mobilità dell'abbonamento dell'UE. Per fare ciò esegue una richiesta HTTP GET specificando come parametro del percorso il SUPI [5.59].

Source	Destination	Protocol	Length	Info
10.250.2.85	10.250.1.26	HTTP	199	GET /nudm-sdm/v2/imsi-901700000000003/am-data HTTP/1.1

Figura 5.59: AccessAndMobilityDataRequest tra l'AMF e l'UDM.

16. In caso di successo, UDM risponde con 200 OK con il corpo del messaggio contenente i dati rilevanti per l'AMF [5.60](#); nel caso non ci sia una sottoscrizione valida, l'UDM risponderà con un 404 Not Found.

No.	Time	Source	Destination	Protocol	Length	Info
87	12.328222	10.250.1.26	10.250.2.85	HTTP	152	HTTP/1.1 200 OK (application/json)
<ul style="list-style-type: none"> > Frame 87: 152 bytes on wire (1216 bits), 152 bytes captured (1216 bits) > Ethernet II, Src: fa:16:3e:80:9f:ea (fa:16:3e:80:9f:ea), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69) > Internet Protocol Version 4, Src: 10.250.1.26, Dst: 10.250.2.85 > Transmission Control Protocol, Src Port: 7777, Dst Port: 48004, Seq: 630, Ack: 603, Len: 86 > [2 Reassembled TCP Segments (218 bytes): #86(132), #87(86)] > Hypertext Transfer Protocol 						
<ul style="list-style-type: none"> ▼ JavaScript Object Notation: application/json <ul style="list-style-type: none"> ▼ Object <ul style="list-style-type: none"> ▼ Member Key: subscribedUeAmbr <ul style="list-style-type: none"> ▼ Object <ul style="list-style-type: none"> ▼ Member Key: uplink <ul style="list-style-type: none"> String value: 1024000 Kbps Key: uplink ▼ Member Key: downlink <ul style="list-style-type: none"> String value: 1024000 Kbps Key: downlink Key: subscribedUeAmbr 						

Figura 5.60: AccessAndMobilityDataResponse tra l'UDM e l'AMF.

17. L'AMF invia due ulteriori HTTP GET agli URI `/nudm-sdm/v2/imsi-901700000000003/smf-select-data` e `/nudm-sdm/v2/imsi-901700000000003/ue-context-in-smf-data`. Le risposte hanno codice 200 OK ad indicare che tutto è andato a buon fine, ma non ci sono dati rilevanti per quanto richiesto, motivo per cui il corpo è vuoto [\[5.61\]](#).

Source	Destination	Protocol	Info
10.250.2.85	10.250.1.26	HTTP	GET /nudm-sdm/v2/imsi-901700000000003/smf-select-data HTTP/1.1
10.250.1.26	10.250.2.85	HTTP	HTTP/1.1 200 OK
10.250.2.85	10.250.1.26	HTTP	GET /nudm-sdm/v2/imsi-901700000000003/ue-context-in-smf-data HTTP/1.1
10.250.1.26	10.250.2.85	HTTP	HTTP/1.1 200 OK

Figura 5.61: Ottenimento da parte dell'AMF dei dati: SMF Selection Subscription Data e UE Context In SMF Data.

18. L'AMF invia all'UE un messaggio di **InitialContextSetup** con lo scopo di stabilire il contesto UE complessivo nel gNodeB, incluso il contesto della PDU Session, la chiave di sicurezza, l'elenco di limitazioni di mobilità, la capacità radio UE, le capacità di sicurezza UE, eccetera. Tale messaggio incapsula in una segnalazione NAS l'indicazione di registrazione accettata [\[5.62\]](#).

No.	Time	Source	Destination	Protocol	Length	Info
94	12.331964	10.250.2.85	10.250.2.165	NGAP/NAS-5GS	242	InitialContextSetupRequest, Registration accept
<ul style="list-style-type: none"> > Frame 94: 242 bytes on wire (1936 bits), 242 bytes captured (1936 bits) > Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1) > Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.2.165 > Stream Control Transmission Protocol, Src Port: 38412 (38412), Dst Port: 57282 (57282) ▼ NG Application Protocol <ul style="list-style-type: none"> ▼ NGAP-PDU: initiatingMessage (0) <ul style="list-style-type: none"> ▼ initiatingMessage <ul style="list-style-type: none"> procedureCode: id-InitialContextSetup (14) criticality: reject (0) ▼ value <ul style="list-style-type: none"> ▼ InitialContextSetupRequest <ul style="list-style-type: none"> ▼ protocolIEs: 9 items <ul style="list-style-type: none"> > Item 0: id-AMF-UE-NGAP-ID > Item 1: id-RAN-UE-NGAP-ID > Item 2: id-UEAggregateMaximumBitRate > Item 3: id-GUAMI > Item 4: id-AllowedNSSAI > Item 5: id-UESecurityCapabilities > Item 6: id-SecurityKey > Item 7: id-MaskedIMEISV > Item 8: id-NAS-PDU 						

Figura 5.62: InitialContextSetupRequest, Registration Accept tra l'AMF e l'UE.

19. Il gNodeB segnala la creazione del contesto con un messaggio di **InitialContextSetupResponse** [5.63].

No.	Time	Source	Destination	Protocol	Length	Info
96	12.418700	10.250.2.165	10.250.2.85	NGAP	98	InitialContextSetupResponse
<ul style="list-style-type: none"> > Frame 96: 98 bytes on wire (784 bits), 98 bytes captured (784 bits) > Ethernet II, Src: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69) > Internet Protocol Version 4, Src: 10.250.2.165, Dst: 10.250.2.85 > Stream Control Transmission Protocol, Src Port: 57282 (57282), Dst Port: 38412 (38412) ▼ NG Application Protocol <ul style="list-style-type: none"> ▼ NGAP-PDU: successfulOutcome (1) <ul style="list-style-type: none"> ▼ successfulOutcome <ul style="list-style-type: none"> procedureCode: id-InitialContextSetup (14) criticality: reject (0) ▼ value <ul style="list-style-type: none"> ▼ InitialContextSetupResponse <ul style="list-style-type: none"> ▼ protocolIEs: 2 items <ul style="list-style-type: none"> > Item 0: id-AMF-UE-NGAP-ID > Item 1: id-RAN-UE-NGAP-ID 						

Figura 5.63: InitialContextSetupResponse tra gNodeB e l'AMF.

20. L'UE segnala all'AMF il completamento della registrazione tramite un messaggio NAS **Registration Complete** [5.64].

No.	Time	Source	Destination	Protocol	Length	Info
99	12.620403	10.250.2.165	10.250.2.85	NGAP/NAS-5GS	122	UplinkNASTransport, Registration complete
> Frame 99: 122 bytes on wire (976 bits), 122 bytes captured (976 bits)						
> Ethernet II, Src: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69)						
> Internet Protocol Version 4, Src: 10.250.2.165, Dst: 10.250.2.85						
> Stream Control Transmission Protocol, Src Port: 57282 (57282), Dst Port: 38412 (38412)						
<ul style="list-style-type: none"> ▼ NG Application Protocol <ul style="list-style-type: none"> ▼ NGAP-PDU: initiatingMessage (0) <ul style="list-style-type: none"> ▼ initiatingMessage <ul style="list-style-type: none"> procedureCode: id-UplinkNASTransport (46) criticality: ignore (1) ▼ value <ul style="list-style-type: none"> ▼ UplinkNASTransport <ul style="list-style-type: none"> ▼ protocolIEs: 4 items <ul style="list-style-type: none"> > Item 0: id-AMF-UE-NGAP-ID > Item 1: id-RAN-UE-NGAP-ID > Item 2: id-NAS-PDU > Item 3: id-UserLocationInformation 						

Figura 5.64: RegistrationComplete tra l'UE e l'AMF.

5.1.7 Procedura di creazione di una PDU Session

La procedura in questione assume che l'UE sia già registrato all'AMF; quindi, quanto descritto in [5.1.6], deve essere stato portato a termine con successo.

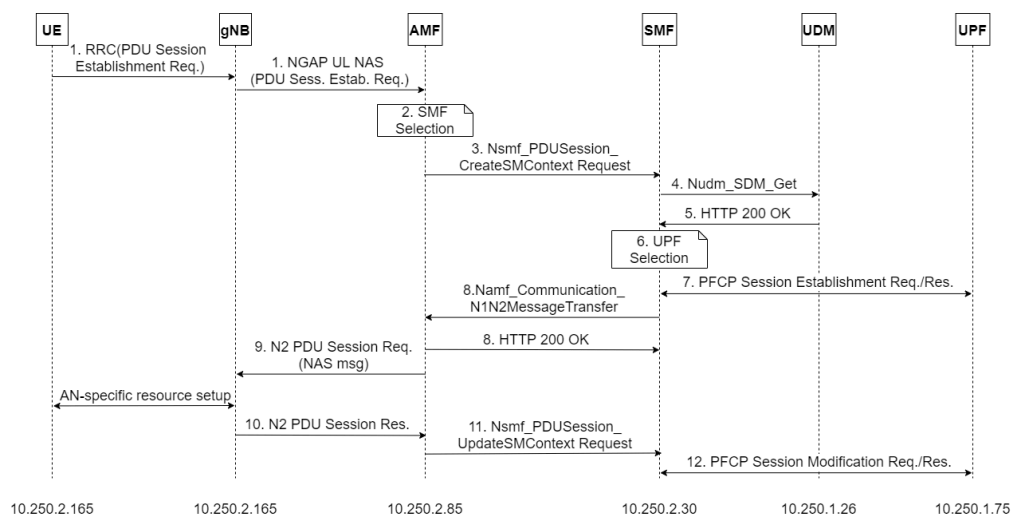


Figura 5.65: Flusso dati per la procedura di creazione di una PDU Session.

1. L'UE invia un messaggio NAS **ULNASTransport** che al suo interno contiene: S-NSSAIs, DNN richiesti (Internet nel caso in esame), PDU Session ID, Request type, N1 SM container. Il campo **Request type** può assumere i valori:

- **Initial request**, nel caso in cui si voglia instaurare una nuova PDU Session;
- **Existing PDU Session** nel caso si tratti di un passaggio da un accesso 3GPP a uno non 3GPP o in caso di handoff;
- **Emergency request**, nel caso in cui si voglia instaurare una nuova PDU Session per un servizio di emergenza;
- **”Existing Emergency PDU Session**, per lo stesso motivo sopra riportato.

Il campo N1 SM container incapsula una richiesta di **PDU Session Establishment Request** che a sua volta include:

- PDU Session ID
- tipo richiesto della PDU Session (IPv4 nel caso in esame)
- modalità di Session and Service Continuity (SSC) richiesta (SSC1 nel caso in esame)
- velocità dati massima per cui è garantita la protezione dell'integrità da parte dell'UE.

Per soddisfare i vari requisiti di continuità delle diverse applicazioni e/o servizi, l'architettura del 5GS fornisce tre modalità SSC. Una volta che una modalità SSC è associata a una sessione PDU, non cambia durante il ciclo di vita di quest'ultima. Le applicazioni possono scegliere la modalità SSC da utilizzare. Tale modalità determina il modo in cui l'UPF della sessione PDU viene allocata e gestita durante la durata della sessione. La modalità SSC1 è quella tradizionale e fornisce una connettività di rete stabile all'UE grazie all'ancoraggio dell'IPv4 o IPv6 da parte dell'UPF che viene mantenuto per tutta la durata della PDU Session a prescindere dalla mobilità dell'UE. Nella modalità SSC2 la rete potrebbe interrompere la connettività e rilasciare una PDU Session mappata prima di crearne una nuova; in questo caso anche l'IP dell'UE potrebbe essere rilasciato. Se viene stabilita una nuova PDU Session, è possibile scegliere una nuova UPF di ancoraggio. In SSC3 la rete garantisce che l'UE non perda la connettività effettuando una nuova connessione prima di interrompere quella esistente; in questo modo è garantita la continuità del servizio. Il messaggio NAS è incapsulato dal gNodeB in un messaggio N2 che include informazioni sulla posizione dell'utente e sul tipo di accesso. In [5.66] è raffigurato quanto riportato.

2. L'AMF determina che il messaggio corrisponde a una richiesta per una nuova PDU Session in base al tipo di richiesta (campo Request Type

al punto 1). Se il messaggio NAS contiene un solo S-NSSAI dovrà essere usato quello, se non contiene S-NSSAI, l'AMF seleziona l'NSSAI di default; se contiene l'S-NSSAI, ma non contiene un DNN, l'AMF seleziona il DNN predefinito per quell'NSSAI, se l'UE ha l'abbonamento a quel DNN, altrimenti seleziona un DNN locale. Infine, se il messaggio contiene più S-NSSAI la scelta su quale usare prevede di consultare i dati dell'abbonamento dell'UE. L'AMF seleziona un SMF come descritto nella clausola 6.3.2 in [15]. Se il tipo di richiesta è *Initial request* o è stato effettuato un cambio di AMF a fronte di una procedura di handoff, il nuovo AMF memorizza i seguenti parametri della richiesta: S-NSSAI, DNN, ID della PDU Session, l'ID della SMF e il tipo di accesso della sessione PDU.

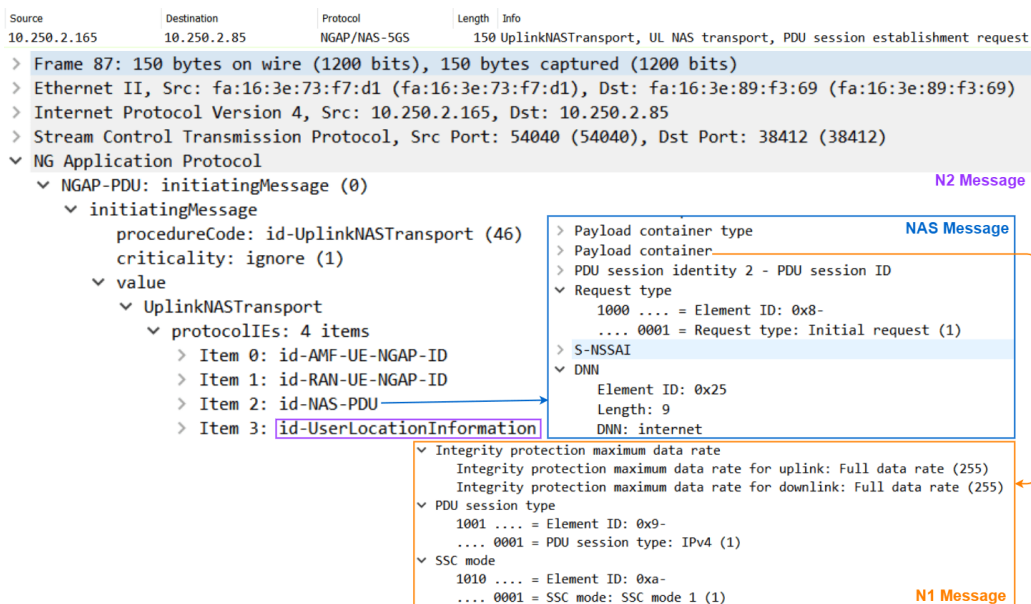


Figura 5.66: PDU Session Establishment Request inviata dall'UE all'AMF.

3. Quando l'AMF non è associato ad un SMF per l'ID della PDU Session fornito dall'UE, ad esempio quando la richiesta è di tipo iniziale come nel caso in esame, invoca il servizio **Nsmf_PDUSession_CreateSMContext Request** tramite HTTP POST. Nel caso sussista già un'associazione, l'AMF invocherà il servizio **Nsmf_PDUSession_UpdateSMContext Request**. Il corpo della POST incapsula la richiesta di **PDU Session Establishment Request** (messaggio N1) descritta al punto 1 e contiene informazioni come:

- SUPI. Può non essere fornito nel caso di richiesta di emergenza;

- PEI
- DNN richiesto e selezionato. Nel caso in esame coincidono;
- S-NSSAI
- ID della PDU Session
- ID dell'istanza AMF (campo servingNFId)
- GUAMI
- posizione dell'UE in termini di tai (PLMN ID + TAC) e New Radio Cell Global Identity (PLMN ID + NR Cell ID)
- l'URI contenente le informazioni in merito alla sessione.

```

Source          Destination    Protocol      Length  Info
10.250.2.85     10.250.2.30  HTTP/NAS-5GS 1353    POST /nsmf-pdusession/v1/sm-contexts HTTP/1.1 (application/json), PDU session establishment request
> Frame 96: 1353 bytes on wire (10824 bits), 1353 bytes captured (10824 bits)
> Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d)
> Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.2.30
> Transmission Control Protocol, Src Port: 34752, Dst Port: 7777, Seq: 1, Ack: 1, Len: 1287
> Hypertext Transfer Protocol
v MIME Multipart Media Encapsulation, Type: multipart/related, Boundary: "--nLhf1pP8oHcCvf0WkK1JJA=="
  [Type: multipart/related]
  First boundary: "--nLhf1pP8oHcCvf0WkK1JJA=="\r\n
  > Encapsulated multipart part: (application/json)
  Boundary: \r\n--nLhf1pP8oHcCvf0WkK1JJA=="\r\n
  > Encapsulated multipart part: (application/vnd.3gpp.5gnas)
  Last boundary: \r\n--nLhf1pP8oHcCvf0WkK1JJA===--\r\n
  Extended protocol discriminator: 5G session management messages (46)
  PDU session identity: PDU session identity value 1 (1)
  Procedure transaction identity: 1
  Message type: PDU session establishment request (0xc1)
  > Integrity protection maximum data rate
  > PDU session type
  > SSC mode
  > Member Key: supi
  > Member Key: pei
  > Member Key: pduSessionId
  > Member Key: dnn
  > Member Key: sNssai
  > Member Key: servingNFId
  > Member Key: guami
  > Member Key: servingNetwork
  > Member Key: n1SmMsg
  > Member Key: anType
  > Member Key: ueLocation
  > Member Key: ueTimeZone
  > Member Key: smContextStatusUri

```

Figura 5.67: Richiesta Nsmf_PDUSESSION_CreateSMContext inviata dall'AMF alla SMF

4. Se i dati dell'abbonamento relativi alla gestione della sessione per SUPI, DNN e S-NSSAI della HPLMN non sono disponibili, SMF li recupera dall'UDM tramite **Nudm_SDM_Get** [5.68]. L'SDN può sottoscrivere all'UDM, tramite **Nudm_SDM_Subscribe**, per venire notificato nel caso in cui i dati dell'abbonamento relativi alla gestione della sessione cambino. L'UDM reperisce queste informazioni interrogando l'UDR. La risposta inviata dall'UDM alla SMF, HTTP 200 OK, contiene: tipo della PDU Session di default e tipi concessi, modalità SSC di default e modalità concesse, informazioni in merito al profilo di QoS dell'abbonamento, Session Aggregate Maximum Bit Rate, Session-AMBR [5.69]. Quest'ultimo parametro limita il bit rate ottenibile aggregando tutti i flussi QoS Non-GBR⁷ di una PDU Session.

⁷Non-Guaranteed Bit Rate. Per cui il bit rate non è garantito

Source	Destination	Protocol	Length	Info
10.250.2.30	10.250.1.26	HTTP	257	GET /nudm-sdm/v2/imsi-901700000000003/sm-data?single-nssai=%7B%0A%09%22sst%22%3A%091%0A%7D&dnn=internet


```

{
  "sst": 1
}

```

Figura 5.68: Richiesta Nudm_SDM_Get inviata dalla SMF all'UDM

Source	Destination	Protocol	Length	Info
10.250.1.26	10.250.2.30	HTTP	661	HTTP/1.1 200 OK (application/json)


```

> Frame 75: 661 bytes on wire (5288 bits), 661 bytes captured (5288 bits)
> Ethernet II, Src: fa:16:3e:80:9f:ea (fa:16:3e:80:9f:ea), Dst: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d)
> Internet Protocol Version 4, Src: 10.250.1.26, Dst: 10.250.2.30
> Transmission Control Protocol, Src Port: 7777, Dst Port: 51792, Seq: 134, Ack: 192, Len: 595
> [2 Reassembled TCP Segments (728 bytes): #73(133), #75(595)]
> Hypertext Transfer Protocol
  JavaScript Object Notation: application/json
  Object
    Member Key: singleNssai
      Object
        Member Key: sst
          Number value: 1
          Key: sst
          Key: singleNssai
    Member Key: dnnConfigurations
      Object
        Member Key: internet
          Key: dnnConfigurations
    Member Key: pduSessionTypes
      Object
        Member Key: defaultSessionType
        Member Key: allowedSessionTypes
          Key: pduSessionTypes
        Member Key: sscModes
          Object
            Member Key: defaultSscMode
            Member Key: allowedSscModes
          Key: sscModes
        Member Key: 5gQosProfile
        Member Key: sessionAmbr

```

Figura 5.69: Risposta HTTP 200 OK inviata dall'UDM alla SMF

5. Può essere una risposta di tipo **Nsmf_PDUSession_CreateSMContext Response** o **Nsmf_PDUSession_UpdateSMContext Response** a seconda del tipo di richiesta al punto 3. Nel caso in esame, si tratta del primo tipo e a fronte della creazione del contesto, la SMF risponde all'AMF fornendo l'URI del contesto creato [5.70].

Source	Destination	Protocol	Length	Info
10.250.2.30	10.250.2.85	HTTP	69	HTTP/1.1 201 Created (application/json)


```

> Frame 108: 69 bytes on wire (552 bits), 69 bytes captured (552 bits)
> Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69)
> Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.2.85
> Transmission Control Protocol, Src Port: 7777, Dst Port: 34752, Seq: 205, Ack: 1288, Len: 3
> [2 Reassembled TCP Segments (207 bytes): #106(204), #108(3)]
> Hypertext Transfer Protocol
  JavaScript Object Notation: application/json

```


[Request URI: http://10.250.2.30:7777/nsmf-pdusession/v1/sm-contexts/5/modify]

Figura 5.70: Risposta Nsmf_PDUSession_CreateSMContext Response, HTTP 201 Created, inviata dalla SMF all'AMF

6. Se il tipo di richiesta al punto 3 è "Initial request", la SMF seleziona una modalità SSC per la PDU Session come descritto nella clausola 5.6.9.3 in [15]. Oltre a ciò, seleziona uno o più UPF in base alle esigenze come descritto nella clausola 6.3.3 della stessa specifica tecnica. Se il tipo della PDU Session è IPv4 o IPv6 o IPv4v6, SMF alloca un indirizzo/prefisso IP questa (a meno che non sia configurato diversamente) come descritto nella clausola 5.8.2 in [15].
7. In maniera simile a quanto avviene per il 4G, la SMF mediante un messaggio di tipo **PFCP Session Establishment Request** richiede all'UDM l'instaurazione di una sessione PFCP, sull'interfaccia N4, al fine di controllo. Per l'instaurazione della sessione, la SMF contatta l'UDM sul SEID 0x0 (buttlar) e specifica il SEID da utilizzare di qui in avanti, 0x5. Tramite le regole PDR, la SMF specifica all'UDM il TEID per il tunnel GTP-U sull'interfaccia N3 verso la rete di accesso; per la rete core, invece, specificherà le informazioni utili per l'instradamento IP. La richiesta contiene anche le regole FAR utili per istruire l'UDM in merito al forwarding dei pacchetti. In questo caso, la regola per instradare i pacchetti provenienti dal core e diretti verso l'UE viene creata, ma la SMF non è in grado di specificare il TEID che il gNodeB ha associato al tunnel GTP in quanto l'allocazione non è ancora avvenuta. L'UDM specifica tramite il messaggio **PFCP Session Establishment Response** la creazione delle regole e l'instaurazione della sessione PFCP [5.71].
8. L'SMF invoca, tramite HTTP POST, il servizio **Namf_Communications_N1N2MessageTransfer** offerto dall'AMF includendo informazioni N2 SM che l'AMF dovrà inoltrare alla RAN e un container N1 SM che l'AMF dovrà inoltrare all'UE [5.72]. Le informazioni N2 SM includono:
 - informazioni in merito al TEID allocato dall'UDM per il tunnel GTP-U;
 - uno o più profili QoS
 - ID della PDU Session che può essere usato dalla rete di accesso, AN, nelle segnalazioni inviate all'UE per indicare a questo l'associazione tra le risorse allocate nell'AN e la PDU Session.

Il container N1 SM incapsula l'accettazione dell'instaurazione della PDU Session e contiene informazioni come:

- tipo della PDU Session (nel caso in esame IPv4)
- indirizzo allocato all'UE (nel caso in esame 10.45.0.6)

- Session-AMBR consentito
- S-NSSAI allocato
- regole QoS.

L'AMF risponde con un messaggio di conferma che nel caso di successo sarà un HTTP 200 OK.

Source	Destination	Protocol	Length	Info
10.250.2.30	10.250.1.75	PFCP	349	PFCP Session Establishment Request
10.250.1.75	10.250.2.30	PFCP	122	PFCP Session Establishment Response

```

> Frame 81: 349 bytes on wire (2792 bits), 349 bytes captured (2792 bits)
> Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:98:49:95 (fa:16:3e:98:49:95)
> Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.1.75
> User Datagram Protocol, Src Port: 8805, Dst Port: 8805
< Packet Forwarding Control Protocol
  > Flags: 0x21, SEID (S)
  > Message Type: PFCP Session Establishment Request (50)
  > Length: 303
  > SEID: 0x0000000000000000
  > Sequence Number: 45257
  > Spare: 0
  > Node ID : IPv6 address: ::1
  > F-SEID : SEID: 0x0000000000000005, IPv4 10.250.2.30, IPv6 ::1
  > Create PDR : [Grouped IE]
  > Create PDR : [Grouped IE]
  > Create FAR : [Grouped IE]
  > Create FAR : [Grouped IE]
  > Create QER : [Grouped IE]
  > PDN Type : IPv4
    < Create FAR : [Grouped IE]
      > IE Type: Create FAR (3)
      > IE Length: 22
      > FAR ID : Dynamic by CP 1
      > Apply Action :
      > Forwarding Parameters : [Grouped IE]
    < Create FAR : [Grouped IE]
      > IE Type: Create FAR (3)
      > IE Length: 22
      > FAR ID : Dynamic by CP 2
      > Apply Action :
      < Forwarding Parameters : [Grouped IE]
        > IE Type: Forwarding Parameters (4)
        > IE Length: 5
        > Destination Interface : Core
    < PDI : [Grouped IE]
      > IE Type: PDI (2)
      > IE Length: 27
      > Source Interface : Core
      > Network Instance : internet
      < UE IP Address :
        > IE Type: UE IP Address (93)
        > IE Length: 5
        > Flags: 0x06, S/D, V4 (IPv4)
        > IPv4 address: 10.45.0.6
      > FAR ID : Dynamic by CP 1
    < PDI : [Grouped IE]
      > IE Type: PDI (2)
      > IE Length: 36
      > Source Interface : Access
      > F-TEID : TEID: 0x00000005, IPv4 10.250.1.75
      > Network Instance : internet
      > QFI :
      > Outer Header Removal : GTP-U/UDP/IPv4
      > FAR ID : Dynamic by CP 2
  
```

Figura 5.71: Instaurazione sessione PFCP tra SMF e UDM

```

Source      Destination  Protocol Length  Info
10.250.2.30 10.250.2.85 HTTP_1051 POST /namf-comm/v1/ue-contexts/imsi-901700000000003/n1-n2-messages HTTP/1.1 (application/json), PDU session establishment accept
> Frame 91: 1051 bytes on wire (8408 bits), 1051 bytes captured (8408 bits)
> Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69)
> Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.2.85
> Transmission Control Protocol, Src Port: 51644, Dst Port: 7777, Seq: 1, Ack: 1, Len: 985
> Hypertext Transfer Protocol
> MIME Multipart Media Encapsulation, Type: multipart/related, Boundary: "--e81dLS9EeYsoRJ2JmYmUWw=="
  [Type: multipart/related]
  First boundary: "--e81dLS9EeYsoRJ2JmYmUWw==\r\n"
  > Encapsulated multipart part: (application/json)
  Boundary: \r\n--e81dLS9EeYsoRJ2JmYmUWw==\r\n
  > Encapsulated multipart part: (application/vnd.3gpp.5gnas)
  Boundary: \r\n--e81dLS9EeYsoRJ2JmYmUWw==\r\n
  > Encapsulated multipart part: (application/vnd.3gpp.ngap)
  Last boundary: \r\n--e81dLS9EeYsoRJ2JmYmUWw==--\r\n

  > Object
    > Member Key: n1MessageContainer
    > Member Key: n2InfoContainer
      > Object
        > Member Key: n2InformationClass
        > Member Key: smInfo
          > Object
            > Member Key: pduSessionId
            > Member Key: n2InfoContent
              Key: smInfo
              Key: n2InfoContainer
            > Member Key: pduSessionId

  > NG Application Protocol
    > PDUSessionResourceSetupRequestTransfer
      > protocolIEs: 4 items
        > Item 0: id-PDUSessionAggregateMaximumBitRate
        > Item 1: id-UL-NGU-UP-TNLInformation
          > ProtocolIE-Field
            id: id-UL-NGU-UP-TNLInformation (139)
            criticality: reject (0)
            > value
              > UPTransportLayerInformation: gTPTunnel (0)
                > gTPTunnel
                  > transportLayerAddress: 0afa014b
                    > TransportLayerAddress (IPv4): 10.250.1.75
                    > gTP-TEID: 00000005
              > Item 2: id-PDUSessionType
              > Item 3: id-QoSFlowSetupRequestList
                N2 SM

  > Plain NAS 5GS Message
    Extended protocol discriminator: 5G session management messages (46)
    PDU session identity: PDU session identity value 1 (1)
    Procedure transaction identity: 1
    Message type: PDU session establishment accept (0xc2)
    0001 .... = Selected SSC mode: SSC mode 1 (1)
    > PDU session type - Selected PDU session type
    > QoS rules - Authorized QoS rules
    > Session-AMBR
    > PDU address
    > S-NSSAI
    > QoS flow descriptions - Authorized
    > DNN
    N1 SM

```

Figura 5.72: Namf_Communication_N1N2MessageTransfer invocato dalla SMF

9. L'AMF invia, tramite NAS, una richiesta di **PDUSessionResourceSetupRequest** indirizzata all'UE, contenente l'ID della PDU Session e l'accettazione dell'instaurazione di questa. Inoltre, invia le informazioni N2 SM ricevute dalla SMF e indirizzate alla (R)AN [5.73].
10. La (R)AN risponde all'AMF con un messaggio **PDUSessionResourceSetupResponse** che include al suo interno il TEID allocato dall'AN per il GTP-U tunnel sull'interfaccia N3 [5.74].
11. L'AMF inoltra le informazioni ricevute dalla (R)AN alla SMF in modo tale che questo possa aggiornare le regole di inoltro nell'UPF; per fare ciò, effettua una POST all'indirizzo del contesto SM [5.75]. L'SMF risponderà all'AMF, con un HTTP 204 No Content, per confermare la ricezione delle informazioni.
12. L'SMF, mediante un messaggio di tipo **PFCP Session Modification Request**, aggiorna la regola FAR per l'instradamento dei pacchetti verso la RAN. L'UPF conferma la modifica mediante un messaggio di **PFCP Session Modification Response** [5.76]. A questo punto la PDU Session è stata creata con successo e l'UE può inviare/ricevere dati dalla Data Network (nel caso in esame Internet).

```

Source      Destination  Protocol  Info
10.250.2.85 10.250.2.165 NGAP/NAS-5GS PDU SessionResourceSetupRequest, DL NAS transport, PDU session establishment accept
> Frame 116: 226 bytes on wire (1808 bits), 226 bytes captured (1808 bits)
> Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1)
> Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.2.165
> Stream Control Transmission Protocol, Src Port: 38412 (38412), Dst Port: 54040 (54040)
v NG Application Protocol
  v NGAP-PDU: initiatingMessage (0)
    v initiatingMessage
      procedureCode: id-PDU SessionResourceSetup (29)
      criticality: reject (0)
      v value
        v PDU SessionResourceSetupRequest
          v protocolIEs: 3 items
            > Item 0: id-AMF-UE-NGAP-ID
            > Item 1: id-RAN-UE-NGAP-ID
            > Item 2: id-PDU SessionResourceSetupListSUReq
              v value
                v PDU SessionResourceSetupListSUReq: 1 item
                  v Item 0
                    v PDU SessionResourceSetupItemSUReq
                      PDU Session Establishment Accept
                      pDU SessionID: 1
                      > pDU SessionNAS-PDU: 7e02140a5ada037e00680100352e0101c211000901000631...
                      > s-NSSAI
                      > pDU SessionResourceSetupRequestTransfer: 0000040082000a0c3e800000303e800000008b000a01f00a...

```

Figura 5.73: Richiesta PDU SessionResourceSetupRequest inviata dall'AMF alla (R)AN

```

Source      Destination  Protocol  Info
10.250.2.165 10.250.2.85 NGAP      PDU SessionResourceSetupResponse
> Frame 121: 118 bytes on wire (944 bits), 118 bytes captured (944 bits)
> Ethernet II, Src: fa:16:3e:73:f7:d1 (fa:16:3e:73:f7:d1), Dst: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69)
> Internet Protocol Version 4, Src: 10.250.2.165, Dst: 10.250.2.85
> Stream Control Transmission Protocol, Src Port: 54040 (54040), Dst Port: 38412 (38412)
v NG Application Protocol
  v NGAP-PDU: successfulOutcome (1)
    v successfulOutcome
      procedureCode: id-PDU SessionResourceSetup (29)
      criticality: reject (0)
      v value
        v PDU SessionResourceSetupResponse
          v protocolIEs: 3 items
            > Item 0: id-AMF-UE-NGAP-ID
            > Item 1: id-RAN-UE-NGAP-ID
            > Item 2: id-PDU SessionResourceSetupListSURes
              v gTPTunnel
                v transportLayerAddress: 7f000101
                  TransportLayerAddress (IPv4): 10.250.2.165
                  gTP-TEID: 00000005

```

Figura 5.74: Richiesta PDU SessionResourceSetupResponse inviata dalla (R)AN all'AMF

Source	Destination	Protocol	Info
10.250.2.85	10.250.2.30	HTTP/NGAP	POST /nsmf-pdusession/v1/sm-contexts/5/modify HTTP/1.1 (application/json) ;

```

> Frame 122: 583 bytes on wire (4664 bits), 583 bytes captured (4664 bits)
> Ethernet II, Src: fa:16:3e:89:f3:69 (fa:16:3e:89:f3:69), Dst: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d)
> Internet Protocol Version 4, Src: 10.250.2.85, Dst: 10.250.2.30
> Transmission Control Protocol, Src Port: 34752, Dst Port: 7777, Seq: 1288, Ack: 208, Len: 517
> Hypertext Transfer Protocol
  MIME Multipart Media Encapsulation, Type: multipart/related, Boundary: "--UiH6kchX101b8A4XCwPpRg=="
    [Type: multipart/related]
    First boundary: "--UiH6kchX101b8A4XCwPpRg==\r\n"
    > Encapsulated multipart part: (application/json)
      Boundary: \r\n--UiH6kchX101b8A4XCwPpRg==\r\n
    > Encapsulated multipart part: (application/vnd.3gpp.ngap)
      Last boundary: \r\n--UiH6kchX101b8A4XCwPpRg==--\r\n
      gTPTunnel
        transportLayerAddress: 7f000101
        TransportLayerAddress (IPv4): 10.250.2.165
        gTP-TEID: 00000005
  
```

Figura 5.75: Nsmf_PDUSession_UpdateSMContext Request inviata dall'AMF alla SMF

Source	Destination	Protocol	Length	Info
10.250.2.30	10.250.1.75	PFCP	88	PFCP Session Modification Request
10.250.1.75	10.250.2.30	PFCP	63	PFCP Session Modification Response

```

> Frame 99: 88 bytes on wire (704 bits), 88 bytes captured (704 bits)
> Ethernet II, Src: fa:16:3e:1c:41:8d (fa:16:3e:1c:41:8d), Dst: fa:16:3e:98:49:95 (fa:16:3e:98:49:95)
> Internet Protocol Version 4, Src: 10.250.2.30, Dst: 10.250.1.75
> User Datagram Protocol, Src Port: 8805, Dst Port: 8805
  Packet Forwarding Control Protocol
    Flags: 0x21, SEID (S)
    Message Type: PFCP Session Modification Request (52)
    Length: 42
    SEID: 0x0000000000000005
    Sequence Number: 45258
    Spare: 0
    > Update FAR : [Grouped IE]
      [Response In: 100]
      FAR ID : Dynamic by CP 1
      Update Forwarding Parameters : [Grouped IE]
        IE Type: Update Forwarding Parameters (11)
        IE Length: 14
        Outer Header Creation :
          IE Type: Outer Header Creation (84)
          IE Length: 10
          Outer Header Creation Description: GTP-U/UDP/IPv4 (256)
          TEID: 0x00000005
          IPv4 Address: 10.250.2.165
  
```

Figura 5.76: Modifica della sessione PFCP tra SMF e UDM

5.2 Creazione Network Service

La V.M su cui sono in esecuzione OSM e Juju è dotata di 4 VCPU e 8 GB di RAM. La creazione delle VNF (nello specifico, delle sue VDU) e la configurazione di Day 0 sono estremamente veloci, nell'ordine del paio di minuti. Per la configurazione di Day 1 invece, le risorse risultano scarse e spesso la macchina va in crash non completando l'operazione. Ciò è comprensibile, visto che per ogni VNF è necessario eseguire un container per il proxy charm che si occuperà della sua configurazione. Attualmente, l'unica modalità di validazione possibile per la configurazione di Day 1 prevede

di mandare in esecuzione un NS composto dalla sola VNF di cui si vuole verificare la configurazione. Per ovviare a questo problema, sarà necessario aumentare le risorse della V.M che ospita OSM oppure valutare l'adozione dei native charm.

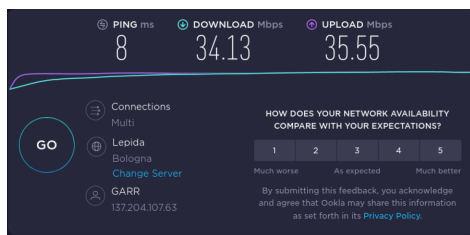
5.3 Limitazione AMBR per abbonamento 5G

Uno dei parametri che può essere settato nel database degli abbonati è l'UE Aggregate Maximum Bit-Rate, ovvero il massimo bit-rate a cui può inviare e ricevere i pacchetti un utente abbonato. Allo stato di sviluppo attuale, UERANSIM riceve dal core di rete 5G questa informazione, ma non la utilizza per limitare il traffico dell'UE. Nel dettaglio, la gNodeB si dovrebbe occupare di limitare il traffico di ciascuna sessione PDU in base al $\min(\sum APNs_AMBR, UE_AMBR)$. Al fine di testare questa funzionalità, è possibile implementare in UERANSIM un semplice algoritmo di token bucketing in grado di limitare il rateo di invio e ricezione dei pacchetti sulla/dalla socket UDP associata alla sessione PDU.

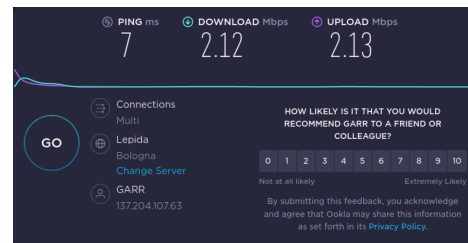
Il principio su cui si basa l'algoritmo è semplice: è disponibile un secchio (bucket) che contiene dei token ognuno dei quali garantisce la possibilità di inviare o ricevere un byte o un pacchetto di dimensione predefinita (nel caso in esame utilizzeremo i byte). I token vengono generati e inseriti nel secchio con una certa frequenza; se questo è pieno, i token generati verranno scartati. Se un processo vuole inviare N byte deve consumare N token e nel caso non siano presenti deve attendere la loro generazione.

Prima di poter implementare l'algoritmo, è necessario comprendere come funziona la gestione di una sessione PDU da parte della gNodeB. Il task che si occupa di ciò, implementa un event loop per gestire i dati. In particolare, ad ogni ciclo preleva un messaggio da una coda che può essere un messaggio di downlink o uplink. Nel primo caso, si tratta di un datagramma UDP da cui è possibile ottenere il messaggio GTP e successivamente il pacchetto IP; questo verrà passato al livello inferiore. Nel secondo caso, si tratta di un pacchetto IP che deve essere incapsulato in un messaggio GTP che a sua volta verrà inviato come datagramma UDP.

Siccome i messaggi vengono gestiti tramite event loop è possibile implementare una classe che descrive il bucket senza dover preoccuparsi degli accessi concorrenti. Alla costruzione del bucket è necessario specificare la capacità e il tempo impiegato per riempirsi. La classe implementa un metodo `tryConsume` che prende in input il numero di token da consumare; questo metodo verrà invocato ogni volta che deve essere inviato o ricevuto un pacchetto passandogli la dimensione di quest'ultimo. La prima azione svolta dal



(a) Speed test senza limitazione



(b) Speed test con AMBR limitato a 2Mb/s sia in upload che download

Figura 5.77: Comparazione risultati speed test

metodo è computare il numero in più di token nel secchio a partire dall'ultima volta che è stato invocato; in questo modo non è necessario implementare il bucket come entità proattiva che periodicamente incrementa il suo numero di token. Il codice che implementa il bucket è contenuto in [A.4.1].

Implementata la classe, è necessario creare due oggetti TokenBuket da utilizzare per l'upload e il download. Tutte le volte che si deve inviare/ricevere una pacchetto PDU dal tunnel GTP è necessario richiedere, dal bucket apposito, un numero di token pari alla grandezza del pacchetto IP che questo incapsula per poter completare la procedura. Purtroppo, l'utilizzo di un bucket così semplice e la gestione dei dati ricevuti e inviati mediante event-loop causa dei problemi qualora si vogliono inviare e ricevere dati simultaneamente. Infatti, bloccare l'event loop per limitare una delle due direzioni (upload o download) causa anche il rallentamento dell'altra. Ad ogni modo, l'implementazione in oggetto serve esclusivamente per verificare se è possibile limitare upload e download in base al valore AMBR impostato nel database degli abbonati e per fare ciò ci si avvarrà di uno speed test. Perciò è possibile ritenere l'implementazione sopra descritta sufficiente per eseguire tale validazione.

La macchina virtuale su cui è in esecuzione UERANSIM possiede 2 VCPU e 4 GB di RAM ed è dotata di una connettività nell'ordine del Gigabit. Stabilendo una sessione PDU e reindirizzando il traffico di Firefox è possibile navigare a 35 Mb/s sia in upload che in download. Ovviamente, l'hardware su cui esegue il simulatore fa da collo di bottiglia e non permette di navigare a velocità superiori. Inserendo nel database degli abbonati un AMBR per l'UE è possibile limitarlo a velocità desiderate. Nelle figure sottostanti è mostrato il risultato di uno speed test senza limitazione e con limitazione a 2Mb/s [5.77].

Conclusioni e sviluppi futuri

La tesi si è concentrata sullo studio, la messa in esecuzione e la configurazione automatizzata dei core di rete 4G e 5G. Dopo una prima panoramica introduttiva in merito agli aspetti teorici rilevanti, sono stati presentati gli strumenti software utili per la parte pratica.

Quest'ultima ha permesso di capire come NFV, CUPS e Cloud possano garantire flessibilità e scalabilità dell'architettura minimizzando i costi d'investimento e gestione. In più è stato possibile studiare in maniera dettagliata il traffico scambiato nelle diverse procedure e comprendere come vengono gestiti aspetti fondamentali quali l'instaurazione dei tunnel GTP e delle sessioni PFCP. Il controllo diretto dell'infrastruttura Cloud ha permesso di notare un aspetto che spesso viene ignorato a causa della dematerializzazione delle risorse apportata dallo stesso: l'aumento di scalabilità e flessibilità è sicuramente notevole, ma deve comunque fare i conti con la potenza dell'hardware su cui si appoggia l'intera infrastruttura. Inoltre, la presenza massiva di software implica l'aumentare del numero di possibili errori che possono rendere i servizi di rete non disponibili. Ovviamente, queste considerazioni sono frutto di sperimentazioni fatte con software open-source e auto gestito; in uno scenario reale, l'utilizzo di soluzioni standard offerte da aziende leader diminuisce problemi di questa natura. Tuttavia, è ragionevole pensare che non li estingua.

Gli sviluppi futuri sono innumerevoli. Lo scenario messo in campo ha permesso di acquisire familiarità con i diversi strumenti e comprenderne i limiti. Costituisce una base di partenza che verrà migliorata per uno studio più accurato delle possibilità offerte dal 5G. In particolare, potrebbe essere interessante approfondire: la messa in esecuzione dei gateway che si occupano del piano dati sulla parte edge del Cloud, la creazione di slice di rete in grado di soddisfare requisiti differenti, valutare l'adozione di native charms anziché proxy charms in modo da distribuire la potenza computazionale richiesta su più V.M e introdurre la parte di configurazione di day 2 per il monitoraggio del core.

Appendice A

Implementazioni

A.1 Descrittori

A.1.1 radio_access_vnfd.yaml

```
vnfd:vnfd-catalog:
  vnfd:
  - id: radio_access_vnfd
    name: radio_access_vnfd
    short-name: radio_access_vnfd
    description: Generated by OSM package generator
    vendor: OSM
    version: '1.0'

    mgmt-interface:
      cp: vnf-infrastructure-mgmt

    vdu:
  - id: mme_amf_vdu
    name: mme_amf_vdu
    description: mme_amf_vdu
    count: 1

    vm-flavor:
      vcpu-count: 1
      memory-mb: 2048
      storage-gb: 20

    image: 'Open5GS'
    cloud-init-file: cloud-config
    supplemental-boot-data:
      boot-data-drive: true

    interface:
  - name: eth0
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-infrastructure-mgmt
  - name: eth1
    type: EXTERNAL
```

```

    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-infrastructure-external
  - name: eth2
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-pc-internal

connection-point:
- name: vnf-infrastructure-mgmt
  id: vnf-infrastructure-mgmt
  type: VPORT
- name: vnf-infrastructure-external
  id: vnf-infrastructure-external
  type: VPORT
  port-security-enabled: false
- name: vnf-pc-internal
  id: vnf-pc-internal
  type: VPORT

vnf-configuration:
  initial-config-primitive:
  - seq: '1'
    name: config
    parameter:
    - name: ssh-hostname
      value: <rw_mgmt_ip>
    - name: ssh-username
      value: ubuntu
    - name: ssh-password
      value: ubuntu
  - seq: '2'
    name: configure-mme-yaml
    parameter:
    - name: slap-ip
      data-type: STRING
      value: '10.250.2.68'
    - name: gtpc-ip
      data-type: STRING
      value: '192.168.3.2'
    - name: sgwc-ip
      data-type: STRING
      value: '192.168.3.3'
    - name: smf-ip
      data-type: STRING
      value: '192.168.3.3'
    - name: sgwc-port
      data-type: STRING
      value: '2124'
  - seq: '3'
    name: configure-mme-conf
    parameter:
    - name: fr-diam-listen-ip
      data-type: STRING
      value: '192.168.3.2'
    - name: fr-diam-connect-ip
      data-type: STRING
      value: '192.168.3.4'
  - seq: '4'
    name: configure-amf
    parameter:

```

```

-     name: sbi-ip
      data-type: STRING
      value: '192.168.3.2'
-     name: ngap-ip
      data-type: STRING
      value: '192.168.3.2'
-     name: nrf-ip
      data-type: STRING
      value: '192.168.3.5'
-   seq: '5'
      name: start-services
juju:
  charm: mmeamfcharm

```

A.1.2 user_gateway_vnfd.yaml

```

vnfd:vnfd-catalog:
  vnfd:
  - id: user_gateway_vnfd
    name: user_gateway_vnfd
    short-name: user_gateway_vnfd
    description: Generated by OSM package generator
    vendor: OSM
    version: '1.0'

  mgmt-interface:
    cp: vnf-infrastructure-mgmt

  vdu:
  - id: sgwu_upf_pgwu_vdu
    name: sgwu_upf_pgwu_vdu
    description: sgwu_upf_pgwu_vdu
    count: 1

  vm-flavor:
    vcpu-count: 1
    memory-mb: 2048
    storage-gb: 20

  image: 'Open5GS'
  cloud-init-file: cloud-config
  supplemental-boot-data:
    boot-data-drive: true

  interface:
  - name: eth0
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-infrastructure-mgmt
  - name: eth1
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-infrastructure-external
  - name: eth2
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-pc-internal
  - name: eth3

```

```

        type: EXTERNAL
        virtual-interface:
            type: PARAVIRT
        external-connection-point-ref: vnf-pc-datanetwork

connection-point:
-   name: vnf-infrastructure-mgmt
    id: vnf-infrastructure-mgmt
    type: VPORT
-   name: vnf-infrastructure-external
    id: vnf-infrastructure-external
    type: VPORT
-   name: vnf-pc-internal
    id: vnf-pc-internal
    type: VPORT
-   name: vnf-pc-datanetwork
    id: vnf-pc-datanetwork
    type: VPORT

vnf-configuration:
  initial-config-primitive:
  -   seq: '1'
      name: config
      parameter:
      -   name: ssh-hostname
          value: <rw_mgmt_ip>
      -   name: ssh-username
          value: ubuntu
      -   name: ssh-password
          value: ubuntu
  -   seq: '2'
      name: configure-sgwu
      parameter:
      -   name: gtpu-ip
          data-type: STRING
          value: '192.168.3.6'
      -   name: gtpu-port
          data-type: STRING
          value: '2153'
      -   name: pfcp-ip
          data-type: STRING
          value: '192.168.3.6'
      -   name: pfcp-port
          data-type: STRING
          value: '8806'
      -   name: sgwc-ip
          data-type: STRING
          value: '192.168.3.3'
      -   name: sgwc-port
          data-type: STRING
          value: '8806'
  -   seq: '3'
      name: configure-upf-pgwu-yaml
      parameter:
      -   name: gtpu-ip
          data-type: STRING
          value: '192.168.3.6'
      -   name: pfcp-ip
          data-type: STRING
          value: '192.168.3.6'
      -   name: smf-ip
          data-type: STRING

```

```

        value: '192.168.3.3'
-   seq: '4'
    name: add-nat-rule
-   seq: '5'
    name: start-services
juju:
  charm: sguupfpgwucharm

```

A.1.3 control_gateway_vnfd.yaml

```

vnfd:vnfd-catalog:
  vnfd:
  - id: control_gateway_vnfd
    name: control_gateway_vnfd
    short-name: control_gateway_vnfd
    description: Generated by OSM package generator
    vendor: OSM
    version: '1.0'

  mgmt-interface:
    cp: vnf-infrastructure-mgmt

  vdu:
  - id: sgwc_smf_pgwc_vdu
    name: sgwc_smf_pgwc_vdu
    description: sgwc_smf_pgwc_vdu
    count: 1

  vm-flavor:
    vcpu-count: 1
    memory-mb: 2048
    storage-gb: 20

  image: 'Open5GS'
  cloud-init-file: cloud-config
  supplemental-boot-data:
    boot-data-drive: true

  interface:
  - name: eth0
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-infrastructure-mgmt
  - name: eth1
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-pc-internal

  connection-point:
  - name: vnf-infrastructure-mgmt
    id: vnf-infrastructure-mgmt
    type: VPORT
  - name: vnf-pc-internal
    id: vnf-pc-internal
    type: VPORT

  vnf-configuration:
    initial-config-primitive:
    - seq: '1'

```

```

name: config
parameter:
- name: ssh-hostname
  value: <rw_mgmt_ip>
- name: ssh-username
  value: ubuntu
- name: ssh-password
  value: ubuntu
- seq: '2'
  name: configure-sgwc
  parameter:
- name: gtpc-ip
  data-type: STRING
  value: '192.168.3.3'
- name: gtpc-port
  data-type: STRING
  value: '2124'
- name: pfcip-ip
  data-type: STRING
  value: '192.168.3.3'
- name: pfcip-port
  data-type: STRING
  value: '8806'
- name: sgwu-ip
  data-type: STRING
  value: '192.168.3.6'
- name: sgwu-port
  data-type: STRING
  value: '8806'
- seq: '3'
  name: configure-smf-pgwc-conf
  parameter:
- name: fr-diam-listen-ip
  data-type: STRING
  value: '192.168.3.3'
- name: fr-diam-connect-ip
  data-type: STRING
  value: '192.168.3.5'
- seq: '4'
  name: configure-smf-pgwc-yaml
  parameter:
- name: sbi-ip
  data-type: STRING
  value: '192.168.3.3'
- name: gtpc-ip
  data-type: STRING
  value: '192.168.3.3'
- name: pfcip-ip
  data-type: STRING
  value: '192.168.3.3'
- name: nrf-ip
  data-type: STRING
  value: '192.168.3.5'
- name: upf-ip
  data-type: STRING
  value: '192.168.3.6'
- seq: '5'
  name: start-services
juju:
  charm: sgwcmfpgwccharm

```


A.1.4 auth_functional_block_vnfd.yaml

```
vnfd:vnfd-catalog:
  vnfd:
  - id: auth_functional_block_vnfd
    name: auth_functional_block_vnfd
    short-name: auth_functional_block_vnfd
    description: Generated by OSM package generator
    vendor: OSM
    version: '1.0'

  mgmt-interface:
    cp: vnf-infrastructure-mgmt

  vdu:
  - id: hss_udm_uds_auaf_vdu
    name: hss_udm_uds_auaf_vdu
    description: hss_udm_uds_auaf_vdu
    count: 1

  vm-flavor:
    vcpu-count: 1
    memory-mb: 2048
    storage-gb: 20

  image: 'Open5GS'
  cloud-init-file: cloud-config
  supplemental-boot-data:
    boot-data-drive: true

  interface:
  - name: eth0
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-infrastructure-mgmt
  - name: eth1
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-pc-internal

  connection-point:
  - name: vnf-infrastructure-mgmt
    id: vnf-infrastructure-mgmt
    type: VPORT
  - name: vnf-pc-internal
    id: vnf-pc-internal
    type: VPORT

  vnf-configuration:
  initial-config-primitive:
  - seq: '1'
    name: config
    parameter:
  - name: ssh-hostname
    value: <rw_mgmt_ip>
  - name: ssh-username
    value: ubuntu
  - name: ssh-password
    value: ubuntu
```

```

- seq: '2'
  name: configure-ausf
  parameter:
  - name: sbi-ip
    data-type: STRING
    value: '192.168.3.4'
  - name: sbi-port
    data-type: STRING
    value: '7779'
  - name: nrf-ip
    data-type: STRING
    value: '192.168.3.5'
- seq: '3'
  name: configure-udr
  parameter:
  - name: sbi-ip
    data-type: STRING
    value: '192.168.3.4'
  - name: sbi-port
    data-type: STRING
    value: '7778'
  - name: nrf-ip
    data-type: STRING
    value: '192.168.3.5'
  - name: mongodb-ip
    data-type: STRING
    value: '192.168.3.4'
- seq: '4'
  name: configure-udm
  parameter:
  - name: sbi-ip
    data-type: STRING
    value: '192.168.3.4'
  - name: nrf-ip
    data-type: STRING
    value: '192.168.3.5'
- seq: '5'
  name: configure-hss-conf
  parameter:
  - name: fr-diam-listen-ip
    data-type: STRING
    value: '192.168.3.4'
  - name: fr-diam-connect-ip
    data-type: STRING
    value: '192.168.3.2'
- seq: '6'
  name: register-user
  parameter:
  - name: imsi
    data-type: STRING
    value: '001010123456789'
  - name: opc
    data-type: STRING
    value: '63BFA50EE6523365FF14C1F45F88737D'
  - name: k
    data-type: STRING
    value: '00112233445566778899AABBCCDDEEFF'
  - name: type
    data-type: STRING
    value: 'opc'
  - name: pgwc-ip
    data-type: STRING

```

```

        value: '10.102.2.159'
-   seq: '7'
    name: register-user
    parameter:
-     name: imsi
      data-type: STRING
      value: '901700000000003'
-     name: opc
      data-type: STRING
      value: 'E8ED289DEBA952E4283B54E88E6183CA'
-     name: k
      data-type: STRING
      value: '465B5CE8B199B49FAA5F0A2EE238A6BC'
-     name: type
      data-type: STRING
      value: 'op'
-     name: pgwc-ip
      data-type: STRING
      value: '10.102.2.159'
-   seq: '8'
    name: start-services
juju:
  charm: authblockscharm

```

A.1.5 misc_functional_block_vnfd.yaml

```

vnfd:vnfd-catalog:
  vnfd:
-   id: misc_functional_block_vnfd
    name: misc_functional_block_vnfd
    short-name: misc_functional_block_vnfd
    description: Generated by OSM package generator
    vendor: OSM
    version: '1.0'

  mgmt-interface:
    cp: vnf-infrastructure-mgmt

  vdu:
-   id: pcrf_nrf_vdu
    name: pcrf_nrf_vdu
    description: pcrf_nrf_vdu
    count: 1

  vm-flavor:
    vcpu-count: 1
    memory-mb: 2048
    storage-gb: 20

  image: 'Open5GS'
  cloud-init-file: cloud-config
  supplemental-boot-data:
    boot-data-drive: true

  interface:
-   name: eth0
    type: EXTERNAL
    virtual-interface:
      type: PARAVIRT
    external-connection-point-ref: vnf-infrastructure-mgmt
-   name: eth1

```

```

        type: EXTERNAL
        virtual-interface:
            type: PARAVIRT
        external-connection-point-ref: vnf-pc-internal

connection-point:
-   name: vnf-infrastructure-mgmt
    id: vnf-infrastructure-mgmt
    type: VPORT
-   name: vnf-pc-internal
    id: vnf-pc-internal
    type: VPORT

vnf-configuration:
    initial-config-primitive:
    -   seq: '1'
        name: config
        parameter:
        -   name: ssh-hostname
            value: <rw_mgmt_ip>
        -   name: ssh-username
            value: ubuntu
        -   name: ssh-password
            value: ubuntu
    -   seq: '2'
        name: configure-nrf
        parameter:
        -   name: nrf-ip
            data-type: STRING
            value: '192.168.3.5'
    -   seq: '3'
        name: configure-pcrf-conf
        parameter:
        -   name: fr-diam-listen-ip
            data-type: STRING
            value: '192.168.3.5'
        -   name: fr-diam-connect-ip
            data-type: STRING
            value: '192.168.3.3'
    -   seq: '4'
        name: configure-pcrf-yaml
        parameter:
        -   name: mongodb-ip
            data-type: STRING
            value: '192.168.3.4'
    -   seq: '5'
        name: start-services
    juju:
        charm: nrfpcrfcharm

```

A.1.6 Open5GS_nsd.yaml

```

nsd:nsd-catalog:
  nsd:
  -   id: Open5GS_nsd
      name: Open5GS_nsd
      short-name: Open5GS_nsd
      description: Generated by OSM package generator
      vendor: OSM
      version: '1.0'

```

```

constituent-vnfd:
- member-vnf-index: 1
  vnfd-id-ref: radio_access_vnfd
- member-vnf-index: 2
  vnfd-id-ref: control_gateway_vnfd
- member-vnf-index: 3
  vnfd-id-ref: user_gateway_vnfd
- member-vnf-index: 4
  vnfd-id-ref: auth_functional_block_vnfd
- member-vnf-index: 5
  vnfd-id-ref: misc_functional_block_vnfd

ip-profiles:
- ip-profile-params:
  dhcp-params:
    enabled: 'true'
  ip-version: ipv4
  subnet-address: 192.168.3.0/24
  name: ip-internal

vld:
- id: mgmt_net
  name: mgmt_net
  short-name: mgmt_net
  type: ELAN
  mgmt-network: 'true'
  vim-network-name: mgmtNet
  vnfd-connection-point-ref:
  - member-vnf-index-ref: 1
    vnfd-id-ref: radio_access_vnfd
    vnfd-connection-point-ref: vnf-infrastructure-mgmt
  - member-vnf-index-ref: 2
    vnfd-id-ref: control_gateway_vnfd
    vnfd-connection-point-ref: vnf-infrastructure-mgmt
  - member-vnf-index-ref: 3
    vnfd-id-ref: user_gateway_vnfd
    vnfd-connection-point-ref: vnf-infrastructure-mgmt
  - member-vnf-index-ref: 4
    vnfd-id-ref: auth_functional_block_vnfd
    vnfd-connection-point-ref: vnf-infrastructure-mgmt
  - member-vnf-index-ref: 5
    vnfd-id-ref: misc_functional_block_vnfd
    vnfd-connection-point-ref: vnf-infrastructure-mgmt
- id: external_net
  name: external_net
  short-name: external_net
  type: ELAN
  vim-network-name: external
  vnfd-connection-point-ref:
  - member-vnf-index-ref: 1
    vnfd-id-ref: radio_access_vnfd
    vnfd-connection-point-ref: vnf-infrastructure-external
    ip-address: 10.250.2.68
  - member-vnf-index-ref: 3
    vnfd-id-ref: user_gateway_vnfd
    vnfd-connection-point-ref: vnf-infrastructure-external
    ip-address: 10.250.2.164
- id: transport_net
  name: transport_net
  short-name: transport_net
  type: ELAN

```

```

vim-network-name: transportNet
vnfd-connection-point-ref:
- member-vnf-index-ref: 3
  vnfd-id-ref: user_gateway_vnfd
  vnfd-connection-point-ref: vnf-pc-datanetwork
  ip-address: 10.102.2.159
- id: internal_net
  name: internal_net
  ip-profile-ref: ip-internal
  short-name: internal_net
  type: ELAN
  vnfd-connection-point-ref:
- member-vnf-index-ref: 1
  vnfd-id-ref: radio_access_vnfd
  vnfd-connection-point-ref: vnf-pc-internal
  ip-address: 192.168.3.2
- member-vnf-index-ref: 2
  vnfd-id-ref: control_gateway_vnfd
  vnfd-connection-point-ref: vnf-pc-internal
  ip-address: 192.168.3.3
- member-vnf-index-ref: 3
  vnfd-id-ref: user_gateway_vnfd
  vnfd-connection-point-ref: vnf-pc-internal
  ip-address: 192.168.3.6
- member-vnf-index-ref: 4
  vnfd-id-ref: auth_functional_block_vnfd
  vnfd-connection-point-ref: vnf-pc-internal
  ip-address: 192.168.3.4
- member-vnf-index-ref: 5
  vnfd-id-ref: misc_functional_block_vnfd
  vnfd-connection-point-ref: vnf-pc-internal
  ip-address: 192.168.3.5

```

A.2 Script my-open5gs-dbctl

```

# TO EXECUTE:
# my-open5gs-dbctl add_with_pgw_and_op_or_opc {imsi} {k}
# {op_or_opc_val} {op_or_opc_type} {pgw_ip}
if [ "$1" = "add_with_pgw_and_op_or_opc" ]; then
  if [ "$#" -eq 6 ]; then
    imsi=$2
    ki=$3
    pgw_ip=$6
    op_and_opc_values=( [0]=null [1]=null )
    [[ "$5" = "opc" ]] && op_and_opc_values[1]='\'$4\'
    || op_and_opc_values[0]='\'$4\'

    mongo --eval "db.subscribers.update({'imsi' : \"\$imsi\"},
    { \$setOnInsert: { 'imsi' : \"\$imsi\", 'pdn' :
    [ { 'apn' : \"internet\", '_id' : new ObjectId(),
    'pcc_rule' : [ ], 'ambr' : {
    'downlink' : NumberLong(1024000),
    'uplink' : NumberLong(1024000) },
    'qos' : { 'qci' : NumberInt(9),
    'arp' : { 'priority_level' : NumberInt(8),

```

```

        \ "pre_emption_vulnerability\ " : NumberInt(1),
        \ "pre_emption_capability\ " : NumberInt(0) } } ,
        \ "type\ " : NumberInt(0),
        \ "pgw\ " : { \ "addr\ " : \ "$pgw_ip\ " } } ],
        \ "ambr\ " : { \ "downlink\ " : NumberLong(1024000),
        \ "uplink\ " : NumberLong(1024000) },
        \ "subscribed_rau_tau_timer\ " : NumberInt(12),
        \ "network_access_mode\ " : NumberInt(2),
        \ "subscriber_status\ " : NumberInt(0),
        \ "access_restriction_data\ " : NumberInt(32),
        \ "security\ " : { \ "k\ " : \ "$ki\ " , \ "amf\ " : \ "8000\ " ,
        \ "op\ " : ${op_and_opc_values[0]} ,
        \ "opc\ " : ${op_and_opc_values[1]} } } ,
        \ "__v\ " : 0 } } , upsert=true);" open5gs
    exit 0
fi

echo "my-open5gs-dbctl: incorrect number of args, format is
\ "my-open5gs-dbctl add_with_pgw_and_op_or_opc key opc_val
type(op or opc) pgw_ip\ ""
exit 1
fi

```

A.3 Configurazione day 1

A.3.1 Codice python per file corrispondenti alle azioni

```

#!/usr/bin/env python3
import sys
sys.path.append('lib')

from charms.reactive import main
from charms.reactive import set_state
from charmhelpers.core.hookenv import action_fail, action_name

set_state('actions.{}'.format(action_name()))
try:
    main()
except Exception as e:
    action_fail(repr(e))

```

A.3.2 Struttura sgwuupfpgwcharm.py

```

from charms.reactive import when, when_not, set_flag
from charmhelpers.core.hookenv import (
    function_get,
    function_fail,
    function_set,
    config,
    status_set,
)
from charms.reactive import (

```

```

        remove_state as remove_flag,
        set_state as set_flag,
        when, when_not
    )
import charms.sshproxy

@when_not("sgwuupfpgwucharm.installed")
def install_sgwuupfpgwucharm():
    set_flag("sgwuupfpgwucharm.installed")

@when("actions.configure-upf-pgwu")
def configure_upfpgwu():
    #upf and pgwu configuration code
    remove_flag("actions.configure-upf-pgwu")

@when("actions.configure-sgwu")
def configure_sgwu():
    #sgwu configuration code
    remove_flag("actions.configure-sgwu")

@when("actions.add-nat-rule")
def add_nat_rule():
    #add nat rule code
    remove_flag("actions.add-nat-rule")

@when("actions.start-services")
def start_services():
    #start services code
    remove_flag("actions.start-services")

```

A.3.3 Cambio indirizzi nel file hss.conf

```

@when("actions.configure-hss-conf")
def configure_hssconf():
    filePath = "/etc/freeDiameter/hss.conf"
    remove_comment(filePath)
    run_cmd(
        changeDiameterAddress.format(
            originalIp="127.0.0.8",
            ipAddress=function_get("fr-diam-listen-ip"),
            filePath=filePath,
        )
    )
    run_cmd(
        changeDiameterAddress.format(
            originalIp="127.0.0.2",
            ipAddress=function_get("fr-diam-connect-ip"),
            filePath=filePath,
        )
    )
    remove_flag("actions.configure-hss-conf")

@when("actions.register-user")
def register_user():
    charms.sshproxy._run(
        'my-open5gs-dbctl add_with_pgw_op_or_opc "{imsi}" "{k}" "{opcval}"
↪ "{type}" "{pgwip}"'.format(
            imsi = function_get("imsi"),
            k = function_get("k"),
            opcval = function_get("opc"),

```



```

        type = function_get("type"),
        pgwip = function_get("pgwc-ip")
    )
    remove_flag("actions.register-user")

```

A.4 Limitazione AMBR

A.4.1 TokenBucket.java

```

public class TokenBucket {

    private final long capacity;
    private final double refillTokensPerOneMillis;

    private double availableTokens;
    private long lastRefillTimestamp;

    public TokenBucket(long capacity, long refillPeriodMillis) {
        this.capacity = capacity;
        this.refillTokensPerOneMillis = (double) capacity / (double) refillPeriodMillis;
        this.availableTokens = capacity;
        this.lastRefillTimestamp = System.currentTimeMillis();
    }

    public boolean tryConsume(int numberTokens) {
        refill();
        if (availableTokens < numberTokens) {
            return false;
        } else {
            availableTokens -= numberTokens;
            return true;
        }
    }

    private void refill() {
        long currentTimeMillis = System.currentTimeMillis();
        if (currentTimeMillis > lastRefillTimestamp) {
            long millisSinceLastRefill = currentTimeMillis - lastRefillTimestamp;
            double refill = millisSinceLastRefill * refillTokensPerOneMillis;
            this.availableTokens = Math.min(capacity, availableTokens + refill);
            this.lastRefillTimestamp = currentTimeMillis;
        }
    }
}

```

Ringraziamenti

Il presente elaborato segna la fine di un importante percorso di studi. Inizialmente, la vita universitaria mi ha destabilizzato: classi numerose piene di persone nuove e un ambiente formale in cui sei “solo” una matricola. Inoltre, ero molto pragmatico e non comprendevo le motivazioni per cui dovessi studiare argomenti già conosciuti.

Con il tempo mi sono ricreduto, ho scoperto che gli argomenti che pensavo di conoscere, in realtà non li conoscevo abbastanza e che forse non li conoscerò mai in maniera esaustiva. L’Università, oltre alla base di conoscenza teorica, mi ha regalato il desiderio di continuo miglioramento e un metodo con cui portare a termine i diversi compiti professionali. Non è stato un percorso facile, gli anni di laurea magistrale mi hanno messo alla prova. Il raggiungimento di questo traguardo non è frutto esclusivamente del mio impegno; l’affetto di molte persone mi ha permesso di arrivare fino a qui. Per questo, vorrei dedicare un po’ di spazio per ringraziarle.

In primo luogo, ringrazio Marina e Paolino che hanno sempre svolto con eccellenza uno dei ruoli più difficili, quello dei genitori. I loro sacrifici mi hanno permesso di condurre una vita agiata e concentrarmi al meglio sugli studi. Non hanno mai avuto la pretesa di controllare chi dovessi diventare, hanno lasciato che la vita mi guidasse pronti a sostenermi nei momenti più difficili. Spero che questo traguardo li renda fieri di me, quanto io lo sono di loro e ripaghi parte dei loro sforzi.

Il secondo ringraziamento va ai miei fratelli, Davide e Deborah. Essendo molto più piccolo, sono stati i miei secondi genitori. Mi hanno sempre incoraggiato a raggiungere i miei obiettivi esprimendo ammirazione per i risultati ottenuti. Inoltre, mi hanno regalato quattro splendide nipoti che continuamente mi ricordano l’importanza della curiosità e mi riempiono il cuore di gioia.

Un sentito ringraziamento va alla mia fidanzata, Martina. Progettare insieme il nostro futuro mi spinge a dare il massimo in ogni situazione per raggiungere i nostri sogni. In questi mesi mi ha sempre sostenuto rendendo meno pesanti gli sforzi necessari per arrivare fino a qui.

Grazie a tutti gli amici, quelli di una vita e quelli conosciuti grazie a questo percorso. Il tempo condiviso con loro mi ha permesso di “ricaricare le pile”.

Infine, voglio ringraziare con tutto il cuore i professori incontrati lungo il percorso che amano il proprio mestiere e hanno come primo obiettivo professionale l’istruzione dei ragazzi. Quello che sono diventato è anche merito loro; come disse Rita Levi-Montalcini: “la scelta di un giovane dipende dalla sua inclinazione, ma anche dalla fortuna di incontrare un grande maestro”. Conserverò con cura gli insegnamenti di ciascuno. Tra questi, un ringraziamento particolare va a Franco Callegati, Chiara Grasselli, Chiara Contoli e Davide Borsatti che hanno dedicato parte del loro tempo per curare questo progetto e hanno condiviso con me il loro sapere.

Mi piacerebbe ringraziare tante altre persone, ma i limiti di spazio non me lo permettono. Ci tengo a precisare che questa sezione non è una formalità, ma è scritta con il cuore in quanto ritengo che l’amore e l’affetto siano i sentimenti che ci permettono di fare grandi cose nel migliore dei modi.

Bibliografia

Libri

- [21] Chandramouli Devaki, Liebhart Rainer e Pirskanen Juho. *5G for the connected world*. John Wiley & Sons, 2019.
- [29] Dan Forsberg et al. *LTE Security*. John Wiley & Sons, 2013.
- [35] Keith W. Ross James F. Kurose. *Computer Networking - A Top-down Approach*. 7th, global. Pearson, 2017. ISBN: 1-292-15359-8.
- [40] Magnus Olsson e Catherine Mulligan (Eds.) *EPC and 4G Packet Networks. Driving the Mobile Broadband Revolution*. 2^a ed. Academic Press, 2013. ISBN: 978-0-12-394595-2.
- [49] Martin Sauter. *From GSM to LTE-Advanced: An Introduction to Mobile Networks and Mobile Broadband*. Revised Second Edition. Wiley, 2014. ISBN: 1118861957,9781118861950.

Articoli

- [16] Marco Ajmone Marsan et al., cur. *The 5G Italy Book 2019: a Multi-perspective View of 5G*. Consorzio Nazionale Interuniversitario per le Telecomunicazioni, 2019. ISBN: 9788832170030. URL: https://www.5gitaly.eu/2019/wp-content/uploads/2019/12/libro5G_2019_online.pdf.
- [17] NGMN Alliance. *NGMN 5G White Paper*. Rapp. tecn. NGMN, mar. 2015.
- [18] Bego Blanco et al. “Technology pillars in the architecture of future 5G mobile networks: NFV, MEC and SDN”. In: *Computer Standards & Interfaces* 54 (nov. 2017), pp. 216–228. DOI: [10.1016/j.csi.2016.12.007](https://doi.org/10.1016/j.csi.2016.12.007). URL: <https://doi.org/10.1016/j.csi.2016.12.007>.

- [19] Wolfgang Braun e Michael Menth. “Software-Defined Networking Using OpenFlow: Protocols, Applications and Architectural Design Choices”. In: *Future Internet* 6.2 (mag. 2014), pp. 302–336. DOI: [10.3390/fi6020302](https://doi.org/10.3390/fi6020302). URL: <https://doi.org/10.3390/fi6020302>.
- [21] Chandramouli Devaki, Liebhart Rainer e Pirskanen Juho. *5G for the connected world*. John Wiley & Sons, 2019.
- [30] GSMA. “From Vertical Industry Requirements to Network Slice Characteristics”. en. In: *GSMA* (ago. 2018). URL: <https://www.gsma.com/futurenetworks/wp-content/uploads/2018/09/5G-Network-Slicing-Report-From-Vertical-Industry-Requirements-to-Network-Slice-Characteristics.pdf>.
- [31] GSMA. “Global System for Mobile Communications Alliance (GSMA): An introduction to 5G network slicing”. en. In: *GSMA* (nov. 2017). URL: <https://www.gsma.com/futurenetworks/wp-content/uploads/2017/11/GSMA-An-Introduction-to-Network-Slicing.pdf>.
- [34] Adrian Hoban et al. “OSM Release FOUR Technical Overview”. In: *ETSI* (mag. 2018), p. 17.
- [37] Frank J. Knaesel, Pedro Neves e Susana Sargento. “IEEE 802.21 MIH-enabled Evolved Packet System Architecture”. In: *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*. Springer Berlin Heidelberg, 2012, pp. 61–75. DOI: [10.1007/978-3-642-30422-4_5](https://doi.org/10.1007/978-3-642-30422-4_5). URL: https://doi.org/10.1007/978-3-642-30422-4_5.
- [38] NFV White Paper. *Network Functions Virtualisation: An Introduction, Benefits, Enablers, Challenges & Call for Action. Issue 1*. Rapp. tecn. ETSI, ott. 2012.
- [39] Nishtha e M. Sood. “Software defined network — Architectures”. In: *2014 International Conference on Parallel, Distributed and Grid Computing*. 2014, pp. 451–456.
- [48] Andy Reid et al. “OSM SCOPE, FUNCTIONALITY, OPERATION AND INTEGRATION GUIDELINES”. In: *ETSI* (feb. 2019), p. 43.
- [54] Jiao Zhang et al. “Enabling Efficient Service Function Chaining by Integrating NFV and SDN: Architecture, Challenges and Opportunities”. In: *IEEE Network* 32.6 (nov. 2018), pp. 152–159. DOI: [10.1109/mnet.2018.1700467](https://doi.org/10.1109/mnet.2018.1700467). URL: <https://doi.org/10.1109/mnet.2018.1700467>.
- [55] S. Zhang. “An Overview of Network Slicing for 5G”. In: *IEEE Wireless Communications* 26.3 (2019), pp. 111–117.

Report tecnici

- [1] 3GPP. *3G security; Security architecture*. Technical Specification (TS) 33.102. Version 16.0.0. 3rd Generation Partnership Project (3GPP), lug. 2020. URL: <http://www.3gpp.org/DynaReport/33102.htm>.
- [2] 3GPP. *3GPP Evolved Packet System (EPS); Evolved General Packet Radio Service (GPRS) Tunnelling Protocol for Control plane (GTPv2-C); Stage 3*. Technical Specification (TS) 29.274. Version 16.5.0. 3rd Generation Partnership Project (3GPP), set. 2020. URL: <http://www.3gpp.org/DynaReport/29274.htm>.
- [3] 3GPP. *5G System; Technical Realization of Service Based Architecture; Stage 3*. Technical Specification (TS) 29.500. Version 17.0.0. 3rd Generation Partnership Project (3GPP), set. 2020. URL: <http://www.3gpp.org/DynaReport/29500.htm>.
- [4] 3GPP. *Architecture enhancements for control and user plane separation of EPC nodes*. Technical Specification (TS) 23.214. Version 16.2.0. 3rd Generation Partnership Project (3GPP), set. 2020. URL: <http://www.3gpp.org/DynaReport/23214.htm>.
- [5] 3GPP. *Circuit Switched (CS) fallback in Evolved Packet System (EPS); Stage 2*. Technical Specification (TS) 23.272. Version 16.0.0. 3rd Generation Partnership Project (3GPP), lug. 2020. URL: <http://www.3gpp.org/DynaReport/23272.htm>.
- [6] 3GPP. *General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access*. Technical Specification (TS) 23.401. Version 16.8.0. 3rd Generation Partnership Project (3GPP), set. 2020. URL: <http://www.3gpp.org/DynaReport/23401.htm>.
- [7] 3GPP. *General Packet Radio Service (GPRS); GPRS Tunnelling Protocol (GTP) across the Gn and Gp interface*. Technical Specification (TS) 29.060. Version 16.0.0. 3rd Generation Partnership Project (3GPP), mar. 2020. URL: <http://www.3gpp.org/DynaReport/29060.htm>.
- [8] 3GPP. *General Packet Radio System (GPRS) Tunnelling Protocol User Plane (GTPv1-U)*. Technical Specification (TS) 29.281. Version 16.1.0. 3rd Generation Partnership Project (3GPP), set. 2020. URL: <http://www.3gpp.org/DynaReport/29281.htm>.

- [9] 3GPP. *Interface between the Control Plane and the User Plane nodes*. Technical Specification (TS) 29.244. Version 16.5.0. 3rd Generation Partnership Project (3GPP), set. 2020. URL: <http://www.3gpp.org/DynaReport/29244.htm>.
- [10] 3GPP. *Network architecture*. Technical Specification (TS) 23.002. Version 16.0.0. 3rd Generation Partnership Project (3GPP), lug. 2020. URL: <http://www.3gpp.org/DynaReport/23002.htm>.
- [11] 3GPP. *Policy and charging control architecture*. Technical Specification (TS) 23.203. Version 16.2.0. 3rd Generation Partnership Project (3GPP), dic. 2019. URL: <http://www.3gpp.org/DynaReport/23203.htm>.
- [12] 3GPP. *Procedures for the 5G System (5GS)*. Technical Specification (TS) 23.502. Version 16.6.0. 3rd Generation Partnership Project (3GPP), set. 2020. URL: <http://www.3gpp.org/DynaReport/23502.htm>.
- [13] 3GPP. *Security architecture and procedures for 5G System*. Technical Specification (TS) 33.501. Version 16.4.0. 3rd Generation Partnership Project (3GPP), set. 2020. URL: <http://www.3gpp.org/DynaReport/33501.htm>.
- [14] 3GPP. *Study on Control Plane (CP) and User Plane (UP) separation of Evolved Packet Core (EPC) nodes*. Technical Report (TR) 23.714. Version 14.0.0. 3rd Generation Partnership Project (3GPP), giu. 2016. URL: <http://www.3gpp.org/DynaReport/23714.htm>.
- [15] 3GPP. *System architecture for the 5G System (5GS)*. Technical Specification (TS) 23.501. Version 16.6.0. 3rd Generation Partnership Project (3GPP), set. 2020. URL: <http://www.3gpp.org/DynaReport/23501.htm>.
- [20] P. Calhoun et al. *Diameter Base Protocol*. Rapp. tecn. RFC, set. 2003. DOI: [10.17487/rfc3588](https://doi.org/10.17487/rfc3588). URL: <https://doi.org/10.17487/rfc3588>.
- [24] ETSI. *MEC in 5G networks*. Rapp. tecn. ETSI, giu. 2018.
- [25] ETSI GS MEC 003. *Multi-access Edge Computing (MEC); Framework and Reference Architecture*. Rapp. tecn. ETSI, gen. 2019.
- [26] ETSI GS NFV 002. *Network Functions Virtualization (NFV); Architectural Framework v1.2.1*. Rapp. tecn. ETSI, dic. 2014. URL: https://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.02.01_60/gs_nfv002v010201p.pdf.

- [27] ETSI GS NFV-EVE 005. *Network Functions Virtualisation (NFV); Ecosystem; Report on SDN Usage in NFV Architectural Framework*. Rapp. tecn. ETSI, dic. 2015.
- [28] ETSI GS NFV-MAN 001. *Network Functions Virtualization (NFV); Management and Orchestration*. Rapp. tecn. ETSI, dic. 2014. URL: https://www.etsi.org/deliver/etsi_gs/NFV-MAN/001_099/001/01.01.01_60/gs_nfv-man001v010101p.pdf.
- [33] Joel M. Halpern e Carlos Pignataro. *Service Function Chaining (SFC) Architecture*. Internet-Draft draft-ietf-sfc-architecture-00. Internet Engineering Task Force, 2014. 26 pp. URL: <https://datatracker.ietf.org/doc/html/draft-ietf-sfc-architecture-00>.

Siti

- [22] *Documentazione Juju*. <https://juju.is/docs/concepts-and-terms>.
- [23] *Driver basato su ZeroMQ per srsLTE*. https://docs.srslte.com/en/latest/app_notes/source/zeromq/source/index.html#zeromq-appnote.
- [32] *Guida OSM per onboarding*. <http://osm-download.etsi.org/ftp/Documentation/vnf-onboarding-guidelines/06-walkthrough.html>.
- [41] *Open5GS*. <https://open5gs.org/open5gs/>.
- [42] *Openstack overview*. <https://www.openstack.org/software/>.
- [43] *Ottava edizione hackfest - Native charms*. <http://osm-download.etsi.org/ftp/osm-6.0-six/8th-hackfest/presentations/8th%20OSM%20Hackfest%20-%20Session%207.3%20-%20New%20Native%20Charms.pdf>.
- [44] *Pagina Github del progetto srsGUI*. <https://github.com/srslte/srsgui>.
- [45] *Pagina Github del progetto srsLTE*. <https://github.com/srsLTE/srsLTE>.
- [46] *Pagina Github del progetto UERANSIM*. <https://github.com/aligungr/UERANSIM>.
- [47] *Pagina Github del progetto yq*. <https://github.com/mikefarah/yq>.

- [50] *Strumento OSM per generatore descrittori*. https://osm.etsi.org/wikipub/index.php/Creating_your_own_VNF_package.
- [51] *Wiki configurazione TUN interface in UERANSIM*. <https://github.com/aligungr/UERANSIM/wiki/Configuring-the-TUN-interface>.
- [52] *Wiki installazione UERANSIM*. <https://github.com/aligungr/UERANSIM/wiki/Installation-and-Usage>.
- [53] *Wiki utilizzo TUN interface in UERANSIM*. <https://github.com/aligungr/UERANSIM/wiki/Using-the-TUN-interface>.