← Visão Geral

Referência de Configuração do OmniCHF

A configuração do operador é lida da chave do ambiente da aplicação `:omnichf`, preenchida na inicialização pelo `config/runtime.exs`. Cada chave pode ser substituída por uma variável de ambiente do sistema operacional. O exemplo abaixo mostra a configuração completa de tempo de execução com os valores aplicados quando a variável de ambiente correspondente não está definida.

```
config :omnichf,
  # Ouvidor SBI (interface baseada em serviço)
  sbi_scheme: "http",
  sbi_addr:   "127.0.0.14",
  sbi_port:   7777,

  # Registro / heartbeat do NRF
  nrf_uri:    "http://127.0.0.1:7777",

  # Identidade do PLMN em serviço
  mcc: "999",
  mnc: "70",

  # Observabilidade
  prometheus_metrics_port: 9568,

  # Cadência do heartbeat do NRF
  heartbeat_interval: 10_000,

  # Integração do CGRateS para avaliação / saldo
  cgrates_enabled: true,
  cgrates_url:     "http://localhost:2080/jsonrpc",
  cgrates_tenant:  "cgrates.org",
  cgrates_timeout: 5000

# Registro estruturado em JSON
config :logger, :default_formatter,
  format: {OmniLogger.JsonFormatter, :format},
  metadata: :all
```

Parâmetros Principais

Parâmetro	Tipo	Necessário	Padrão
`sbi_scheme`	String	Não	`"http"`
`sbi_addr`	String	Não	`"127.0.0.14"`
`sbi_port`	Inteiro	Não	`7777`
`nrf_uri`	String	Não	`"http://127.0.0.1:777`
`mcc`	String	Não	`"999"`

Parâmetro	Tipo	Necessário	Padrão
`mnc`	String	Não	`"70"`
`prometheus_metrics_port`	Inteiro	Não	`9568`
`heartbeat_interval`	Inteiro (ms)	Não	`10000`

Parâmetros do CGRateS

Essas chaves controlam a integração com o motor de avaliação e saldo externo CGRateS. Veja Integração do CGRateS para detalhes comportamentais.

Parâmetro	Tipo	Necessário	Padrão
`cgrates_enabled`	Booleano	Não	`true` (`"true"`)
`cgrates_url`	String	Não	`"http://localhost:2080/jso`

Parâmetro	Tipo	Necessário	Padrão
`cgrates_tenant`	String	Não	`"cgrates.org"`
`cgrates_timeout`	Inteiro (ms)	Não	`5000`

> **Padrão quando desativado na hora da construção:** o padrão compilado (usado se o ambiente da aplicação nunca for preenchido por `runtime.exs`) é `cgrates_enabled: false`. Em uma implantação normal, `runtime.exs` o define, e lá o padrão é `true`. O perfil de teste força `cgrates_enabled: false`.

Chaves Avançadas / Opcionais

Essas chaves não são preenchidas pelo `runtime.exs` padrão, mas são lidas do ambiente `:omnichf` se estiverem presentes. Elas ajustam a resolução do grupo de avaliação, cobrança offline e geração de ID de cobrança. Elas não têm variável de ambiente dedicada e são definidas diretamente no bloco de configuração `:omnichf`.

Parâmetro	Tipo	Necessário	Padrão
`rating_groups`	Mapa	Não	`%{}`
`default_rating_group`	Inteiro	Não	`1`
`offline_charging_enabled`	Booleano	Não	`false`
`cdr_output_dir`	String	Não	`"/var/log/omnichf`
`node_id`	Inteiro	Não	`1`

Parâmetro	Tipo	Necessário	Padrão

Grupos de Avaliação

`rating_groups` mapeia um **DNN** (a chave do mapa, uma string) para o **grupo de avaliação** (o valor, um inteiro) aplicado às sessões desse DNN. O grupo de avaliação resolvido aparece no campo `ratingGroup` de `multipleUnitInformation` retornado ao consumidor e no campo `rating_group` do CDR. A resolução é:

1. Se o DNN da sessão for uma chave em `rating_groups`, use esse valor.
2. Caso contrário, use `default_rating_group`.

```
config :omnichf,
  rating_groups: %{
    "internet" => 10,
    "ims"      => 20,
    "iot"      => 30
  },
  default_rating_group: 1
```

A correspondência é feita na string DNN exata. Um DNN sem entrada (e sem `default_rating_group`) recai para o grupo de avaliação `1`. Um valor `rating_groups` malformado (não um mapa) é ignorado e o padrão é usado, portanto, uma configuração ruim nunca bloqueia a cobrança.

Registro

O registro estruturado em JSON é configurado via o `:logger` `:default_formatter`.

Parâmetro	Tipo	Padrão	Descrição
`format`	Tupla	`{OmniLogger.JsonFormatter, :format}`	Formatador de linha de log. Emite JSON estruturado, adequado para ingestão por um transportador de logs.
`metadata`	Átomo / Lista	`:all`	Metadados de log incluídos em cada linha. `:all` emite todos os metadados anexados (SUPI, DNN, ID da sessão PDU, ID de cobrança, procedimento, etc.).

Interfaces de Gerenciamento

O OmniCHF expõe superfícies de gerenciamento voltadas para o operador ao lado do SBI. Essas portas são relevantes para monitoramento e OAM, mas não fazem parte do serviço Nchf:

Interface	Porta	Esquema	Propósit
SBI (Nchf)	`sbi_port` (padrão `7777`)	HTTP	Tráfego de cobra inter-NF 5G.
Métricas do Prometheus	`prometheus_metrics_port` (padrão `9568`)	HTTP	Endpoint de cole métricas.
API de Gerenciamento	`8443`	HTTPS	Leituras de statu leituras de sessão/estatístic do CGRateS, bus CDR e ações de ((configuração, de de CDR, nível de aborto de sessão registro do NRF).
Painel de Controle	`7443`	HTTPS	UI web para recu configuração, lic logs.

Endpoints da API de Gerenciamento / OAM

A API de Gerenciamento na porta `8443` (HTTPS) expõe as seguintes superfícies para o operador:

Método	Caminho	Propósito
`GET`	`/api/status/api`	Ler status da API/serviço.
`GET`	`/api/status/license`	Ler status da licença.
`GET`	`/api/status/nrf`	Ler status de registro do NRF.
`GET`	`/api/status/nf`	Ler status do NF (nó).
`GET`	`/api/sessions`	Listar sessões de cobrança ativas.
`GET`	`/api/sessions/{id}`	Inspecionar uma única sessão de cobrança pelo `charging_data_ref`.
`GET`	`/api/statistics`	Ler estatísticas de cobrança (contagem de sessões ativas, volume total cobrado, acessibilidade do CGRateS).
`GET`	`/api/health/cgrates`	Ler saúde da conectividade do CGRateS.
`GET`	`/api/oam/config`	Ler configuração de tempo de execução.
`PATCH`	`/api/oam/config/{id}`	Atualizar uma chave de configuração de tempo de execução.
`POST`	`/api/oam/charging_session`	Abortar uma sessão de cobrança pelo `charging_data_ref`.
`GET`	`/api/oam/cdr`	Buscar CDRs concluídos. Parâmetros de consulta: `supi`,

Método	Caminho	Propósito
		`dnn`, `date_from`, `date_to` (todos opcionais).
`POST`	`/api/oam/cdr/flush`	Forçar o descarte do uso acumulado como um CDR para uma sessão (`{"charging_data_ref": "<ref>"}`) ou todas as sessões rastreadas (`{}` / `"all"`).
`POST`	`/api/oam/log_level`	Alterar o nível de log em tempo de execução.
`POST`	`/api/oam/nrf/reregister`	Acionar re-registro do NRF.

> **Visibilidade do armazenamento de sessões:** as superfícies de gerenciamento voltadas para sessões (`/api/sessions`, `/api/statistics`, `/api/oam/charging_session`, `/api/oam/cdr/flush`) apresentam cada sessão ativa. O caminho de cobrança SBI ativo mantém cada sessão em um processo **worker** por sessão, e cada sessão de worker é espelhada no **Context store** em memória na criação e atualização e removida na parada. As superfícies de gerenciamento consultam primeiro o registro de workers e recuam para o Context, portanto, uma sessão ativa está sempre visível e acionável. Veja Armazenamento de Sessões e Visibilidade de Gerenciamento para o que isso significa operacionalmente. A busca de CDR (`/api/oam/cdr`) lê os arquivos CDR offline e é independente de ambos os armazenamentos.

← Visão Geral

Referência de Métricas OmniCHF

As métricas são expostas na `prometheus_metrics_port` (padrão `9568`) em `/metrics`.

Métricas de Cobrança

Métrica: `omni_chf_charging_initial_count`
Tipo: Contador
Descrição: Solicitações de sessão de cobrança INICIAL (Criar) recebidas.
Rótulos: `supi`

Métrica: `omni_chf_charging_update_count`
Tipo: Contador
Descrição: Solicitações de sessão de cobrança ATUALIZAR recebidas.
Rótulos: `ref`

Métrica: `omni_chf_charging_release_count`
Tipo: Contador
Descrição: Solicitações de sessão de cobrança LIBERAR recebidas.
Rótulos: `ref`

Métrica: `omni_chf_charging_creates_total`
Tipo: Contador
Descrição: Resultados da operação de criação.
Rótulos: `result` (`success` / `failure`)

Métrica: `omni_chf_charging_updates_total`
Tipo: Contador
Descrição: Resultados da operação de atualização.
Rótulos: `result` (`success` / `failure`)

Métrica: `omni_chf_charging_releases_total`
Tipo: Contador
Descrição: Resultados da operação de liberação. Liberado via o SBI com um término CGRateS bem-sucedido (`success`), liberado via o SBI onde o `TerminateSession` do CGRateS falhou, mas o CDR ainda foi finalizado (`failure`, apresentado ao consumidor como um `500`), ou finalizado forçosamente através do endpoint de descarte de CDR OAM (`oam_flush`).
Rótulos: `result` (`success` / `failure` / `oam_flush`)

Métrica: `omni_chf_sessions_active_count`
Tipo: Gauge
Descrição: Sessões de cobrança ativas atualmente mantidas na memória. Emitido em cada criação e exclusão de armazenamento de contexto, que agora inclui cada sessão de trabalhador ao vivo espelhada, então o gauge rastreia a verdadeira contagem de sessões ativas.
Rótulos: nenhum

```
# Taxa de criação de sessão de cobrança
rate(omni_chf_charging_creates_total{result="success"}[5m])

# Razão de falha na criação
sum(rate(omni_chf_charging_creates_total{result="failure"}[5m]))
  / sum(rate(omni_chf_charging_creates_total[5m]))

# Sessões ativas atualmente
omni_chf_sessions_active_count
```

Métricas CGRateS

Métrica: `omni_chf_cgrates_requests_total`
Tipo: Contador
Descrição: Solicitações JSON-RPC do CGRateS.
Rótulos: `operation` (método SessionS), `result` (`success` / `failure`)

Métrica: `omni_chf_cgrates_request_duration_ms`
Tipo: Histograma
Descrição: Duração da solicitação JSON-RPC do CGRateS em milissegundos.

Rótulos: `operation`
Buckets: 5, 10, 25, 50, 100, 250, 500, 1000, 2500

Métrica: `omni_chf_cgrates_health`
Tipo: Gauge
Descrição: Saúde da conectividade do CGRateS (`1` = acessível, `0` = fora/desativado).
Rótulos: nenhum

```
# Latência p99 do CGRateS por operação
histogram_quantile(0.99,
  sum by (le, operation)
(rate(omni_chf_cgrates_request_duration_ms_bucket[5m])))

# Taxa de erro do CGRateS
sum(rate(omni_chf_cgrates_requests_total{result="failure"}[5m]))

# Acessibilidade do CGRateS (alerta quando 0)
omni_chf_cgrates_health
```

Métricas NRF

Métrica: `omni_chf_nrf_registration_status`
Tipo: Gauge
Descrição: Status de registro do NRF (`1` = registrado, `0` = não registrado).
Rótulos: `nf_type`

```
omni_chf_nrf_registration_status
```

Métricas BEAM VM

Os gauges de saúde padrão do tempo de execução também são exportados.

Métrica	Tipo	Descrição
`beam_memory_total`	Gauge	Memória total do BEAM (bytes)
`beam_memory_processes` / `beam_memory_processes_used`	Gauge	Memória de processo alocada / usada
`beam_memory_system`	Gauge	Memória do sistema (não-processo)
`beam_memory_atom` / `beam_memory_atom_used`	Gauge	Memória de átomo alocada / usada
`beam_memory_binary`	Gauge	Memória binária
`beam_memory_code`	Gauge	Memória de código
`beam_memory_ets`	Gauge	Memória da tabela ETS
`beam_processes_count`	Gauge	Contagem de processos Erlang
`beam_ports_count`	Gauge	Contagem de portas Erlang
`beam_atom_count`	Gauge	Contagem de átomos
`beam_vm_uptime`	Gauge	Tempo de atividade da VM (segundos)

← Visão Geral

Guia de Operações do OmniCHF

Índice

1. Visão Geral
2. Ciclo de Vida da Sessão de Cobrança
3. Referências de Função e Especificação 3GPP
4. Endpoints do SBI
5. Integração com CGRateS
6. Procedimentos Chave

Visão Geral

OmniCHF implementa a **Função de Cobrança (CHF)** do Core 5G. O CHF fornece cobrança online e offline convergente para sessões PDU 5G através do serviço **Nchf_ConvergedCharging** (3GPP TS 32.290 / TS 32.291 / TS 32.255).

Um consumidor de cobrança, tipicamente o **SMF**, cria uma sessão de cobrança para cada sessão PDU, relata o uso ao longo da vida útil dessa sessão e, finalmente, a libera. OmniCHF traduz cada solicitação de cobrança 5G em uma chamada JSON-RPC de **CGRateS** SessionS para autorização de crédito em tempo real (cobrança online / gerenciamento de cotas) e gera um **Registro de Dados de Cobrança (CDR)** na liberação da sessão (cobrança offline).

Cada sessão de cobrança é de propriedade de um **ChargingWorker** GenServer dedicado sob um `DynamicSupervisor` (arquitetura de processo por sessão). Todo o estado de uma sessão (SUPI, DNN, S-NSSAI, volume/duração acumulados, grupo de classificação, estado da sessão CGRateS, ID de cobrança) é mantido nesse processo de trabalho. A falha de uma sessão não

afeta as outras. O estado da sessão é mantido na memória e é reconstruído à medida que os consumidores criam novas sessões de cobrança após uma reinicialização.

Responsabilidades principais:

- Servir o serviço `nchf-convergedcharging/v3` no SBI.
- Alocar um `chargingDataRef` por sessão e gerenciar seu ciclo de vida INICIAL → ATUALIZAR → TERMINAR.
- Conceder unidades de serviço (GSU) em resposta a Unidades de Serviço Solicitadas (RSU), retornando volume/tempo concedido e **gatilhos** de gerenciamento de cotas ao consumidor.
- Conectar-se ao CGRateS para gerenciamento de classificação e saldo (cobrança online).
- Emitir CDRs para o log da aplicação e, opcionalmente, para arquivos de cobrança offline por dia.
- Registrar-se no NRF e manter um heartbeat.

Ciclo de Vida da Sessão de Cobrança

Cada sessão PDU mapeia para uma sessão de cobrança, rastreada por um `chargingDataRef` (UUID).

Estado	Gatilho	Ação
Criado	`POST /chargingdata`	ChargingWorker gerado, CGRateS `InitiateSession` chamado, cota inicial concedida
Atualizado	`POST /chargingdata/{ref}/update`	CGRateS `UpdateSession` chamado, uso acumulado, número de sequência incrementado, nova cota concedida
Liberado	`POST /chargingdata/{ref}/release`	CGRateS `TerminateSession` chamado, CDR construído e emitido, trabalhador parado

Armazenamento de Sessão e Visibilidade de Gerenciamento

OmniCHF mantém o estado da sessão ativa em **dois** armazenamentos em memória que são mantidos em sincronia, de modo que as superfícies de gerenciamento vejam e possam agir em cada sessão ativa:

Armazenamento	Suporte	Mantém	Papel
Registro de trabalhadores	`ChargingWorker` GenServer por sessão sob um `DynamicSupervisor`, indexado em um `Registry`	Estado completo por sessão (volume/duração acumulados, número de sequência, estado CGRateS)	Fonte da verdade para uma sessão ativa. Criar gera um trabalhador; Atualizar e Liberar resolvem o trabalhador primeiro
Armazenamento de contexto	Um único mapa `Agent` em memória indexado por `chargingDataRef`	Um espelho dos mesmos atributos da sessão	Modelo de leitura para observabilida e superfícies OAM

No SBI, **Criar** sempre gera um trabalhador, e **Atualizar** e **Liberar** resolvem esse trabalhador primeiro, recorrendo ao Armazenamento de Contexto apenas se nenhum trabalhador for encontrado (por exemplo, após uma falha do supervisor). Cada sessão de trabalhador ativa é espelhada no Armazenamento de Contexto na criação e em cada atualização, e é removida dele quando o trabalhador para (liberação normal, aborto ou falha). As superfícies de gerenciamento consultam o registro de trabalhadores primeiro e recorrem ao Contexto, de modo que ambos os armazenamentos apresentam uma visão consistente.

Operacionalmente, isso significa:

- Uma sessão criada normalmente é totalmente visível através das superfícies de gerenciamento: ela aparece em `GET /api/sessions` e nos totais de estatísticas, e pode ser abortada (`POST /api/oam/charging_session`) e CDR-liberada (`POST /api/oam/cdr/flush`) através dos endpoints OAM enquanto está ativa.

- O medidor de sessão ativa, estatísticas e listagens de sessão refletem o verdadeiro conjunto de sessões ativas, não apenas um subconjunto de fallback.
- Ambos os armazenamentos são mantidos na memória e são repopulados à medida que os consumidores criam novas sessões após uma reinicialização. Como as sessões são mantidas em um registro local, um `chargingDataRef` é de propriedade da instância que o criou.

O medidor `omni_chf_sessions_active_count` e os contadores `omni_chf_charging_creates_total` / `omni_chf_charging_releases_total` corroboram a contagem de sessões ativas relatadas por `GET /api/sessions`.

Referências de Função e Especificação 3GPP

Especificação	Relevância
TS 23.501	Arquitetura do sistema 5G: papel do CHF, conceito de cobrança convergente
TS 32.240	Arquitetura e princípios de cobrança
TS 32.290	Arquitetura de cobrança do sistema 5G e fluxos de mensagens
TS 32.291	Cobrança do sistema 5G: API do serviço **Nchf_ConvergedCharging** (HTTP/2 SBI), modelo de dados ChargingDataRequest/Response
TS 32.255	Cobrança do domínio de conectividade de dados 5G (cobrança de sessão PDU)
TS 32.251	Formato do ID de cobrança (`chargingId`), cláusula 5.2.1.6 do TS 32.251
TS 32.298	Descrições e codificação de parâmetros CDR
TS 29.500 / 29.501	Estrutura comum do SBI 5G e design da API
TS 29.510	Registro NF e heartbeat com o NRF

Referências de procedimentos Nchf_ConvergedCharging (TS 32.291):

Procedimento	Cláusula
Criação de Dados de Cobrança (INICIAL)	6.1.3.2.1
Atualização de Dados de Cobrança (ATUALIZAR)	6.1.3.2.2
Liberação de Dados de Cobrança (TERMINAR)	6.1.3.2.3
Notificação de Cobrança (REAUTORIZAÇÃO / ABORTAR_COBRANÇA)	6.1.6.2

Endpoints do SBI

Todos os endpoints Nchf são servidos sob a URL base `{sbi_scheme}://{sbi_addr}:{sbi_port}` e o caminho base `/nchf-convergedcharging/v3`.

Método	Caminho	Descrição
`POST`	`/chargingdata`	Criar uma sessão de cobrança (**INICIAL**). Aloca um `chargingDataRef`, inicia uma sessão CGRateS, concede cota inicial.
`POST`	`/chargingdata/{chargingDataRef}/update`	Atualizar uma sessão de cobrança (**ATUALIZAR**). Relata uso e solicita cota adicional.
`POST`	`/chargingdata/{chargingDataRef}/release`	Liberar uma sessão de cobrança (**TERMINAR**). Relata uso final, gera um CDR, termina a sessão CGRateS.
`POST`	`/notify`	Gatilho de callback de Notificação de Cobrança. Instruções para o OmniCHF enviar uma notificação de `REAUTORIZAÇÃO` ou

Método	Caminho	Descrição
		`ABORTAR_COBRANÇA` para o `notifyUri` armazenado do consumidor.

Na Liberação, um `500` (causa `SYSTEM_FAILURE`, título "Falha na Terminação de Cobrança") é retornado quando o `TerminateSession` do CGRateS falha. O CDR ainda é finalizado e escrito, mas a falha é exposta para que uma inconsistência no backend de cobrança (uma sessão CGRateS deixada aberta) não seja oculta por um falso `204`. Veja Liberação retorna 500.

Validação de Solicitações

Cada corpo de solicitação é validado para os seguintes Elementos de Informação obrigatórios antes do processamento. Um IE ausente retorna `400 Bad Request` com causa `MANDATORY_IE_MISSING`:

IE Obrigatório	Tipo
`nfConsumerIdentification`	objeto
`invocationTimeStamp`	string (ISO 8601)
`invocationSequenceNumber`	inteiro

ChargingDataRequest: Campos Chave Consumidos

Campo	Tipo	Usado em	Descriç
`subscriberIdentifier`	string	Todos	SUPI (por exemplo, `999700000000001`). U `Account` do CGRate `nfConsumerIdentif` se ausente.
`notifyUri`	string	Criar	URI de callback arm futura Notificação d (REAUTORIZAÇÃO / ABORTAR_COBRANÇ
`pDUSessionChargingInformation`	objeto	Todos	Detalhes da sessão `pduSessionInforma` ID da sessão PDU, ti tipo de RAT, `qoSInf` `networkSlicingInf`
`multipleUnitUsage`	array	Atualizar, Liberar	Contêineres de uso entradas de `usedUn` de cada elemento s para volume/duraçã acumulados. També o `ratingGroup` solic
`requestType`	string	Todos	`INITIAL_REQUEST`, `UPDATE_REQUEST` ou `TERMINATION_REQUE`

ChargingDataResponse: Campos Chave Retornados

Campo	Tipo	Descrição
`invocationSequenceNumber`	inteiro	`1` na Criação; incrementado em cada Atualização.
`invocationTimeStamp`	string	Timestamp ISO 8601 da resposta.
`multipleUnitInformation`	array	Unidades concedidas: uma entrada com `resultCode: "SUCCESS"`, `grantedUnit` (`totalVolume`, `time`) e o `ratingGroup` resolvido.
`triggers`	array	Gatilhos de gerenciamento de cotas instruindo o consumidor quando relatar a próxima vez: um `QUOTA_THRESHOLD` (IMMEDIATE_REPORT na metade do volume concedido) e um `TIME_LIMIT` (DEFERRED_REPORT em 3600 s).

Integração com CGRateS

CGRateS é o motor externo de classificação e saldo. O OmniCHF atua como um **cliente** SessionS, mapeando cada operação de cobrança Nchf em um método JSON-RPC SessionS:

Operação Nchf	Método CGRateS SessionS
Criar (INICIAL)	`SessionSv1.InitiateSession`
Atualizar (ATUALIZAR)	`SessionSv1.UpdateSession`
Liberar (TERMINAR)	`SessionSv1.TerminateSession`
Verificação de saúde	`SessionSv1.GetActiveSessions` (limite 1)

O valor `MaxUsage` que o CGRateS retorna é passado diretamente de volta ao consumidor como o `grantedUnit` (tanto `totalVolume` quanto `time`).

Comportamento Habilitado vs Desabilitado

Aspecto	`cgrates_enabled: true`	`cgrates_enabled: false` **(modo de bypass)**
Autorização de crédito	Real: CGRateS classifica a conta e retorna `MaxUsage`	Nenhum: uma concessão fixa de **86400** unidades é retornada para cada Criação e Atualização
Impacto no saldo	O saldo do assinante é debitado/reservado no CGRateS	Nenhum saldo é tocado
Terminar	`TerminateSession` finaliza a sessão CGRateS e CDR	No-op
Verificação de saúde	Prova ao vivo contra `cgrates_url`	Sempre relata `reachable: false`, `cgrates_url: "disabled"`
Assinatura de log	`Iniciando sessão CGRateS para ...`	`Integração CGRateS desativada, retornando autorização padrão` (aviso)
Caso de uso	Cobrança online em produção	Testes de laboratório / integração sem um motor de classificação

> No modo de bypass, o OmniCHF ainda aloca `chargingDataRef`s, rastreia sessões, incrementa números de sequência, emite gatilhos e gera CDRs; apenas o controle de crédito é simulado. Isso torna seguro exercer o fluxo completo SMF↔CHF sem uma implantação CGRateS.

A configuração da integração CGRateS (`cgrates_enabled`, `cgrates_url`, `cgrates_tenant`, `cgrates_timeout`) está documentada na Referência de

Configuração.

Mapeamento de Eventos 5G para CGRateS

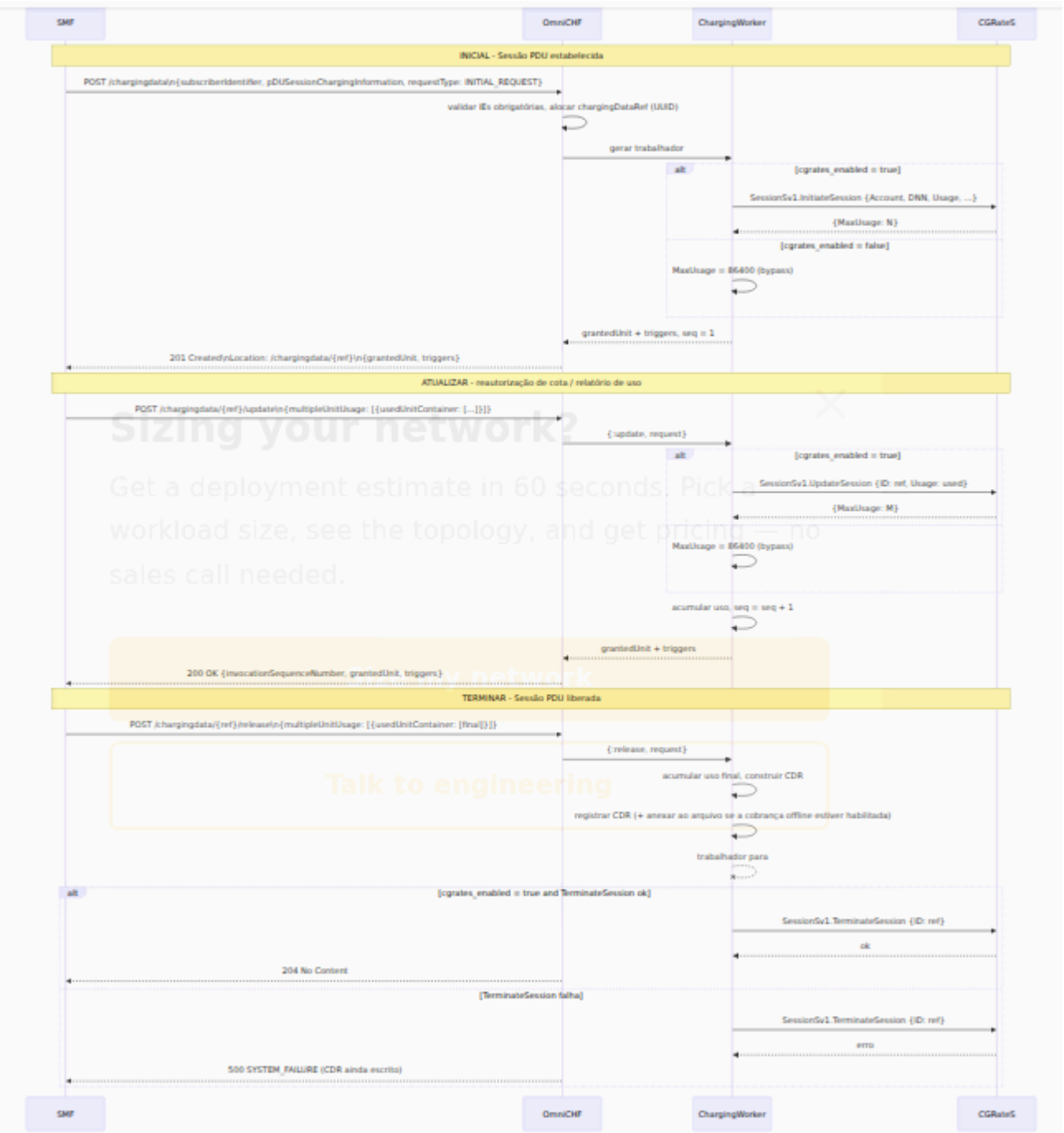
Cada chamada SessionS carrega um mapa `Event` construído a partir do ChargingDataRequest:

Campo de Evento CGRateS	Fonte	
`Account` / `Subject`	`subscriberIdentifier` (SUPI)	Recorre a `nfConsumerI` caso contrári
`Destination` / `DNN`	`pduSessionInformation.dnnId` (ou `.dnn`)	Nome da Rec
`ToR`	derivado de `pduType`	Sempre `*dat`
`RequestType`	`requestType`	Mapeado par
`Category`	estático	`"data"`.
`Usage`	`usedUnitContainer`	Primeiro não `totalVolume` `uplinkVolum` depois `time`.
`OriginID`	`chargingDataRef`	Único por ses
`OriginHost`	estático	`"OmniCHF"`.
`SUPI`	`subscriberIdentifier`	Campo de ex
`S-NSSAI_SST` / `S-NSSAI_SD`	`pduSessionInformation.sNSSAI`	Padrões: SST
`5QI`	`qoSInformation.5qI` (ou `fiveQI`)	Padrão `9`.
`RATType`	`pduSessionInformation.ratType`	Padrão `"NR"`
`PDUSessionID`	`pduSessionInformation.pduSessionID`	Padrão `0`.
`PDUSessionType`	`pduSessionInformation.pduType`	Padrão `"IPV4`

Campo de Evento CGRateS	Fonte	
`ChargingOperation`	operação	`Inicial` / `At`

Procedimentos Chave

Sessão de Cobrança Convergente (INICIAL → ATUALIZAR → TERMINAR)



> Se o `TerminateSession` do CGRateS falhar na liberação, o OmniCHF **ainda** finaliza e escreve o CDR, para que o registro offline não seja perdido. O consumidor então recebe `500` (causa `SYSTEM_FAILURE`) em vez de um falso `204`, porque a sessão do lado do CGRateS pode ser deixada aberta e o

> backend de cobrança pode precisar de reconciliação. Em uma terminação bem-sucedida, o consumidor recebe `204 No Content`.

Gerenciamento de Cota (RSU / GSU e Gatilhos)

Por TS 32.291, o consumidor envia Unidades de Serviço Solicitadas (RSU) e o OmniCHF retorna Unidades de Serviço Concedidas (GSU) mais **gatilhos** que dizem ao consumidor quando voltar:

O consumidor reautentica (envia uma ATUALIZAÇÃO) quando atinge o limite de volume ou o limite de tempo. O OmniCHF também pode conduzir proativamente a reautorização ou a desmontagem através do endpoint `/notify` (`REAUTORIZAÇÃO` / `ABORTAR_COBRANÇA`), que posta para o `notifyUri` fornecido na solicitação INICIAL.

Notificação de Cobrança (REAUTORIZAÇÃO / ABORTAR_COBRANÇA)

Por cláusula 6.1.6.2 do TS 32.291, o OmniCHF envia um `ChargingNotifyRequest` para o `notifyUri` armazenado do consumidor. O gatilho é o endpoint voltado para o operador `POST /nchf-convergedcharging/v3/notify`, cujo corpo carrega o `chargingDataRef` alvo e o `notificationType`. O OmniCHF consulta a sessão, resolve seu `notifyUri` e POSTa a notificação para o consumidor.

> A notificação é um esforço de boa fé em direção à superfície do operador: um `notifyUri` rejeitado ou inacessível é registrado, mas não altera o resultado do `/notify` uma vez que a sessão e o `notifyUri` são resolvidos. Uma sessão sem `notifyUri` armazenado resulta em um erro interno `no_notify_uri`.

Campos do CDR

Um CDR é construído na liberação, registrado em INFO (`CHF CDR: %{...}`), e, quando `offline_charging_enabled` é `true`, anexado como uma linha JSON a `cdr_YYYYMMDD.json`:

Campo	Fonte
`record_type`	Estático `"5G_PDU_SESSION"`
`supi`	Contexto da sessão
`dnn`	Contexto da sessão / `pduSessionInformation`
`snssai`	`{sst, sd}`
`qos_5qi`	`qoSInformation.5qI` (padrão `9`)
`rat_type`	`pduSessionInformation.ratType` (padrão `"NR"`)
`pdu_session_id` / `pdu_session_type`	Contexto da sessão / solicitação
`volume_uplink` / `volume_downlink` / `volume_total`	Acumulado de todos os `usedUnitContainer`s
`duration`	Tempo relatado, senão diferença de relógio
`start_time` / `end_time`	Timestamps de criação / liberação da sessão
`charging_data_ref`	UUID da sessão
`charging_id`	ID de cobrança de 32 bits do 3GPP (TS 32.251)
`recording_entity`	URL base do SBI do OmniCHF
`rating_group`	Resolvido a partir do mapeamento DNN

A saída do CDR offline (`offline_charging_enabled`, `cdr_output_dir`) e a composição do ID de cobrança (`node_id`) estão configuradas na Referência de Configuração.

Executando Cobrança Offline

Para capturar CDRs em arquivo para mediação/cobrança:

1. Defina `offline_charging_enabled: true` e aponte `cdr_output_dir` para um diretório gravável.
2. Os CDRs acumulam um objeto JSON por linha em um arquivo por dia nomeado `cdr_YYYYMMDD.json` (a data é a data de liberação UTC). Um novo arquivo é aberto a cada dia UTC; não há rotação ou retenção automática além da divisão do nome do arquivo diário, portanto, elimine arquivos antigos com sua própria manutenção.
3. Cada sessão liberada anexa exatamente um CDR. Sessões em andamento mantêm seu uso no trabalhador/contexto até a liberação; não são escritas em arquivo até então.

Se `offline_charging_enabled` for `false`, os CDRs ainda são construídos e registrados em INFO como `CHF CDR: %{...}`; apenas a gravação do arquivo é pulada. Para capturar CDRs sem saída de arquivo, envie essas linhas de log com seu pipeline de log. Veja CDRs ausentes de arquivos offline.

Localizando e Pesquisando CDRs

A API de Gerenciamento expõe `GET /api/oam/cdr` para pesquisar os arquivos CDR escritos. Ela lê cada arquivo `cdr_*.json` sob `cdr_output_dir` e filtra com base nos parâmetros de consulta fornecidos (todos opcionais, combinados com AND):

Parâmetro de consulta	Combina	Formato
`supi`	CDR `supi` exato	por exemplo, `imsi-999700000000001`
`dnn`	CDR `dnn` exato	por exemplo, `internet`
`date_from`	Data do arquivo CDR `>=` esta data	`YYYY-MM-DD` (hífens opcionais)
`date_to`	Data do arquivo CDR `<=` esta data	`YYYY-MM-DD` (hífens opcionais)

Os filtros de data são aplicados à **data do nome do arquivo**, não aos timestamps de registros individuais, portanto, o arquivo inteiro de um dia é incluído ou excluído como uma unidade. Sem parâmetros, o endpoint retorna cada CDR em disco. Como lê arquivos, `GET /api/oam/cdr` só retorna registros quando `offline_charging_enabled` estava ativado no momento da liberação.

Forçando a Liberação de um CDR (OAM)

Na operação normal, um CDR é produzido apenas quando uma sessão é liberada. `POST /api/oam/cdr/flush` permite que um operador emita um CDR antecipadamente para qualquer sessão ativa (veja Armazenamento de Sessão e Visibilidade de Gerenciamento), por exemplo, para encerrar sessões que o consumidor nunca liberou:

Corpo da solicitação	Efeito
`{"charging_data_ref": "<ref>"}`	Construir e escrever um CDR para essa sessão, em seguida, finalizá-lo. `404` se a referência não for encontrada.
`{}` ou `{"charging_data_ref": "all"}`	Liberar cada sessão atualmente rastreada.

A liberação constrói o CDR a partir do uso acumulado da sessão mais o último `ChargingDataRequest` e reutiliza o mesmo gravador de CDR como na liberação; não chama `TerminateSession` do CGRateS. A resposta relata `write_result` e `offline_charging_enabled` por sessão. Se a cobrança offline estiver desativada, o CDR é construído e a sessão finalizada, mas nada é escrito em arquivo (`write_result: "ok"` sem saída de arquivo). Cada liberação incrementa `omni_chf_charging_releases_total{result="oam_flush"}`.

← Visão Geral

Solução de Problemas do OmniCHF

400 Bad Request em qualquer solicitação de cobrança

Sintomas: O consumidor recebe `400` com a causa `MANDATORY_IE_MISSING`.

Possíveis causas:

- O corpo da solicitação está faltando `nfConsumerIdentification`, `invocationTimeStamp` ou `invocationSequenceNumber`.

Resolução:

1. Confirme que o consumidor inclui todos os três IEs obrigatórios em cada operação (Criar, Atualizar, Liberar).
2. Verifique o campo `detail` da resposta ProblemDetails; ele lista os nomes dos IEs ausentes.

404 ao Atualizar ou Liberar

Sintomas: O consumidor recebe `404` com a causa `CHARGING_DATA_NOT_FOUND`.

Possíveis causas:

- OmniCHF reiniciado entre Criar e a operação posterior; o estado da sessão é mantido na memória, então a sessão deve ser recriada após um reinício.
- O `chargingDataRef` no caminho está errado.
- A sessão já foi liberada (seu trabalhador parou).

Resolução:

1. Grep logs para `CHF Update: unknown ref=` / `CHF Release: unknown ref=`.
2. Confirme que o OmniCHF não reiniciou desde que a sessão foi criada.
3. Confirme que o consumidor está retornando o `chargingDataRef` do cabeçalho `Location` da criação.

500 ao Criar com CGRateS habilitado

Sintomas: O consumidor recebe `500` com a causa `CHARGING_FAILED`.

Possíveis causas:

- CGRateS `InitiateSession` falhou: motor inacessível, inquilino errado ou um erro do lado do CGRateS.

Resolução:

1. Verifique se `cgrates_url` aponta para um endpoint `/jsonrpc` do CGRateS acessível.
2. Verifique se `cgrates_tenant` corresponde ao inquilino do CGRateS.
3. Verifique o medidor `omni_chf_cgrates_health` (`1` = ativo).
4. Revise os logs para `CGRateS InitiateSession failed ...`; a razão é uma das `{:cgrates_error, msg}`, `{:http_error, status}`, ou `{:http_error, reason}`.
5. Confirme que o serviço `SessionSv1` está habilitado no CGRateS.

Liberação retorna 500

Sintomas: O consumidor recebe `500` com a causa `SYSTEM_FAILURE` e título "Falha na Terminação de Cobrança" em `POST /chargingdata/{ref}/release`.

Possíveis causas:

- O CGRateS `TerminateSession` falhou durante a Liberação (motor inacessível, inquilino errado ou um erro do lado do CGRateS).

Impacto:

- O CDR ainda é construído e gravado, então o registro offline não é perdido.
- A sessão do lado do CGRateS pode ficar aberta, então o backend de cobrança pode ser inconsistente e precisa de reconciliação.

Resolução:

1. Grep logs para `CGRateS TerminateSession failed` no `ref` afetado.
2. Confirme a acessibilidade do CGRateS com o medidor `omni_chf_cgrates_health` (`1` = ativo).
3. Reconcile qualquer sessão do CGRateS deixada aberta para esse `ref` contra o CDR gravado.
4. Correlacione com `omni_chf_charging_releases_total{result="failure"}` para ver a taxa de falhas de terminação.

Verificação de saúde do CGRateS falha enquanto o daemon está ativo

Sintomas: `omni_chf_cgrates_health` lê `0`, mas o CGRateS está em execução.

Possíveis causas:

- O CGRateS está lento: a verificação de saúde limita seu tempo limite a `min(cgrates_timeout, 3000)` ms.
- Incompatibilidade de esquema (HTTP vs HTTPS) em `cgrates_url`.

Resolução:

1. Confirme se o esquema e a porta de `cgrates_url` estão corretos.
2. Se o CGRateS estiver legitimamente lento, investigue a carga do CGRateS em vez de aumentar o limite.
3. Procure avisos `CGRateS unreachable at <url> ...`.

Alta latência do CGRateS

Sintomas: Operações de cobrança lentas; `omni_chf_cgrates_request_duration_ms` elevado.

Resolução:

1. Inspecione o histograma p95/p99 por `operação`.
2. Escale o CGRateS ou o caminho de rede entre OmniCHF e CGRateS.
3. Considere reduzir `cgrates_timeout` para que chamadas lentas falhem rapidamente em vez de bloquear a sessão.

CDRs ausentes dos arquivos offline

Sintomas: Linhas `CHF CDR:` aparecem no log, mas nenhum arquivo `cdr_YYYYMMDD.json` é gravado.

Possíveis causas:

- `offline_charging_enabled` é `false` (padrão): CDRs são apenas registrados.
- `cdr_output_dir` não é gravável.

Resolução:

1. Defina `offline_charging_enabled: true`.
2. Certifique-se de que `cdr_output_dir` exista e seja gravável; fique atento a erros `[CDR] Failed to write CDR to file`.
3. Para capturar CDRs sem saída de arquivo, envie as linhas de log `CHF CDR:` com seu pipeline de log.

Contagem de sessões ativas não diminui

Sintomas: `omni_chf_sessions_active_count` permanece elevado.

Possíveis causas:

- O consumidor nunca envia Liberação (as sessões só são limpas na liberação ou falha do trabalhador).
- As liberações estão retornando `404` (veja acima), então o consumidor as considera concluídas enquanto o OmniCHF pode não.

Resolução:

1. Correlacione a contagem ativa com as taxas de Criar vs Liberar.
2. Investigue o `404` na Liberação, que geralmente indica um reinício anterior.

Registro NRF não mantido

Sintomas: `omni_chf_nrf_registration_status` lê `0`.

Resolução:

1. Verifique se `nrf_uri` é acessível a partir do `sbi_addr` do OmniCHF.
2. Confirme se `mcc` / `mnc` correspondem à configuração PLMN do NRF.
3. Confirme se `sbi_addr` / `sbi_port` no perfil anunciado são acessíveis pelo NRF.
4. Revise os logs de inicialização em busca de erros de registro do NRF.