

安裝

安裝說明書

安裝

OmniMessage SMPP 安裝說明書 SMPP 安裝 OmniMessage 安裝說明書
OmniMessage Core 安裝 - 安裝說明書

1. 安裝 OmniMessage SMPP PDU
2. 安裝 REST API 安裝 OmniMessage 安裝
3. 安裝 OmniMessage 安裝說明書
4. 安裝 SMPP PDU 安裝
5. 安裝 OmniMessage 安裝

OmniMessage 安裝DiameterMAPIMS安裝說明書 - 安裝說明書
OmniMessage Core

安裝

```
/opt/omnimessage-smpp/config/runtime.exs
```

安裝說明書

```
sudo systemctl restart omnimessage-smpp
```

安裝

安裝 Elixir 安裝說明書

```
import Config

# 配置
config :omnimessage_smpp,
  setting_name: value

# SMPP 配置
config :omnimessage_smpp, :binds, [
  %{
    name: "bind_name",
    # ... 配置
  }
]
```

配置

API_BASE_URL

OmniMessage Core 配置 URL

```
config :omnimessage_smpp,
  api_base_url: "https://omnimessage-core.example.com:8443"
```

配置	配置	配置	配置
api_base_url	URL (URL)		-

OmniMessage Core 配置 URL 配置 REST API 配置 OmniMessage 配置

- 配置 SMPP 配置 OmniMessage 配置
- 配置 SMPP 配置
- 配置 OmniMessage
- 配置

配置“配置”配置 OmniMessage 配置

- ✓ 配置
- ✓ 配置
- ✓ 配置
- ✓ 配置
- ✓ 配置
- ✓ 配置
- ✓ 配置

配置 SMPP ↔ OmniMessage 配置

配置

```
# IP 上 HTTPS
api_base_url: "https://192.168.1.100:8443"

# 配置 HTTPS
api_base_url: "https://omnimessage-core.company.com:8443"

# HTTP 配置
api_base_url: "http://192.168.1.100:8080"
```

配置

- 配置 OmniMessage Core
- 配置 HTTPS 配置 `verify_ssl_peer`
- 配置 HTTPS 配置

OAM/配置 API 配置 (`api_ex`)

配置 OAM/配置 API 配置

配置 `api_ex` 配置 OAM/配置 API (`/status`, `/smpp/client_peers`, `/smpp/server_peers`, `/smpp/binds`, `/smpp/summary`)配置 8443 (`config/config.exs`)配置 `api_base_url` 配置 OmniMessage Core 配置

配置

項目	項目	項目	項目
api_base_url	API → Core (API)	...:8443	API/メッセージ
api_ex port	API (API)	8443	API OAM API

OmniMessage Core 8443 API SMPP Core 8443 :eaddrinuse /api/messages 404

config/runtime.exs /etc/omnimessage_smpp/runtime.exs api_ex OAM API Core api_base_url Core 8443

```
# Core 8443 API OAM API 8444
config :api_ex,
  api:
    Application.get_env(:api_ex, :api, %{})
    |> Map.put(:port, 8444)
    |> Map.put(:listen_ip, "0.0.0.0")
```

8443Core

SMPP_POLL_INTERVAL

```
config :omnimessage_smpp,
  smpp_poll_interval: 100
```

項目	項目	項目	項目
smpp_poll_interval	項目	項目	100

時間

- 高負荷 (>100 TPS) 100-500ms
- 中負荷 (10-100 TPS) 500-1000ms
- 低負荷 (<10 TPS) 1000-2000ms

環境変数 `SMPP_POLL_INTERVAL`

VERIFY_SSL_PEER

SSL 検証

```
config :omnimessage_smpp,  
  verify_ssl_peer: false
```

項目	デフォルト	型	説明
<code>verify_ssl_peer</code>	false	Boolean	SSL 接続時のピア検証を有効にするかどうか

API 呼び出し時に SSL 接続を使用するかどうか

値

- `true` SSL 接続を有効にする
- `false` SSL 接続を無効にする

環境変数 `VERIFY_SSL_PEER`

SMSC_NAME

説明

```
config :omnimessage_smpp,  
  smsc_name: "smpp_gateway"
```


配置示例

```
config :omnimessage_smpp, :binds, [
  %{
    # 名称
    name: "vodafone_uk",

    # 模式
    mode: :client,

    # SMPP 绑定
    bind_type: :transceiver,

    # SMPP 主机
    host: "smpp.vodafone.co.uk",
    port: 2775,

    # 绑定源 IP/端口
    # 绑定源 IP/端口
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5000,

    # 系统 ID
    system_id: "your_username",
    password: "your_password",

    # SMPP 系统类型
    system_type: "",
    addr_ton: 0,
    addr_npi: 0,
    address_range: "",

    # TPS 限制
    tps_limit: 100,

    # 队列
    # 队列
    queue: "vodafone_uk",

    # 队列检查频率
    queue_check_frequency: 1000,

    # 队列检查频率 0 禁用
```

```
    enquire_link_interval: 60,  
  
    # 配置缓存  
    cache_enabled: false,  
    cache_max_size: 10000,  
    cache_retry_interval: 60  
  }  
]
```

配置名称

name

配置名称

名称	值	说明
名称	值	"vodafone_uk"

配置名称 SMPP 名称

- 配置名称
- 配置名称
- 配置名称

配置名称

- carrier_region "vodafone_uk" "att_us"
- purpose_number "marketing_1" "alerts_primary"

mode

配置名称

名称	值	说明
名称	值	:client

ESME SMSC

:client

bind_type

SMPP

SMPP	bind_type	bind_parameters
SMPP	bind_type	:transmitter, :receiver, :transceiver

- :transmitter - submit_sm
- :receiver - deliver_sm
- :transceiver - transceiver

:transceiver

host

SMPP IP

SMPP	IP	host
SMPP	IP	"smpp.carrier.com" "10.5.1.100"

SMPP

```
host: "smpp.vodafone.co.uk"
host: "10.20.30.40"
host: "smpp-primary.carrier.net"
```

port

SMPP □□□□□

00	00	00	00
00	0	2775	1-65535

□□□SMPP □□□ TCP □□□

□□□□□2775

111

```
port: 2775 # []
port: 3000 # []
```

bind_source_ip

□□□□□□□□ IP

□□	□□	□□	□□
□□□	□	(□□□□□□)	"10.20.30.40"

TCP/IP

```

:binds Web UI / Mnesia

```

bind_source_port

□ □ □ □ □ □ □ □ □ □

□□	□□	□□	□□
□□	□	(□□)	1-65535

SO_REUSEADDR TCP/IP TIME_WAIT :eaddrinuse

bind_source_ip Web UI

system_id

□□□□□□□

□□	□□	□□
□□□	□	"company_user"

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

□□□□□□□□ - □□□□□□□□

password

□□□□□□

00	00	00
000	0	"secret_password"

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □

□□□□

- 
- 
- 

tps_limit

□□□□□□

項目	単位	値	範囲
tps_limit	個	100	1-10000

tps_limit の設定

設定

- tps_limit を 70-80%
- tps_limit / 100
- tps_limit を 100

設定

```
tps_limit: 10    # 1000  
tps_limit: 50    # 5000  
tps_limit: 100   # 10000  
tps_limit: 1000  # 100000
```

設定

```
tps_limit = 100 TPS  
tps_limit = 70-80  
tps_limit 20-30 TPS
```

queue

queue

項目	単位	値	範囲
queue	個	name	"sinch.smp"

queue の設定
name: sinch.smp
queue: sinch.smp

設定

```
# 配置示例
%{name: "sinch_1", queue: "sinch.smpp", ...}
%{name: "sinch_2", queue: "sinch.smpp", ...}
```

queue_check_frequency

配置示例

名称	单位	默认值	范围
queue_check_frequency	秒	1000	100-10000

配置说明

说明

- 高流量 (>100 TPS) 500-1000ms
- 中流量 (10-100 TPS) 1000-2000ms
- 低流量 (<10 TPS) 2000-5000ms

配置

- 配置项 = 配置项名称 API 名称
- 配置项 = 配置项名称 API 名称

enqueue_link_interval

SMPP 配置示例

名称	单位	默认值	范围
enqueue_link_interval	秒	60	0-3600

配置说明 SMPP enqueue_link PDU 配置项 enqueue_link_resp 配置

配置

項目	項目	項目
名前	値	true

この設定は、メッセージキャッシュを有効にするかどうかを指定します。

cache_enabled

デフォルト値

項目	項目	項目
名前	値	false

この設定は、API のメッセージキャッシュの有効性を指定します。
[MESSAGE_CACHE.md](#)

cache_max_size

デフォルト値

項目	項目	項目	項目
名前	値	10000	1-1000000

この設定は、メッセージキャッシュの最大サイズを指定します。FIFO方式で、cache_enabled が true の場合

cache_retry_interval

デフォルト値

項目	項目	項目
名前	値	60

この設定は、メッセージキャッシュの再接続間隔を指定します。0は60s、1は120s、2は240s。??
 この設定は、cache_enabled が true の場合

Web UI

SMPP

SMPP is a protocol for sending and receiving SMS messages between a mobile network and a service provider. It is used by many mobile network operators and service providers to send and receive SMS messages. The protocol is based on the SMPP (Short Message Peer-to-Peer) protocol, which is a standard for sending and receiving SMS messages between a mobile network and a service provider. The protocol is used by many mobile network operators and service providers to send and receive SMS messages. The protocol is based on the SMPP (Short Message Peer-to-Peer) protocol, which is a standard for sending and receiving SMS messages between a mobile network and a service provider. The protocol is used by many mobile network operators and service providers to send and receive SMS messages.

配置服务器

```
config :omnimessage_smpp, :server_binds, [
  %{
    # 合作伙伴名称
    name: "partner_acme",

    # 系统ID
    system_id: "acme_corp",
    password: "acme_secret",

    # 允许的绑定类型
    allowed_bind_types: [:transmitter, :receiver, :transceiver],

    # IP 白名单
    ip_whitelist: ["192.168.1.0/24", "10.50.1.100"],

    # 源地址白名单 = 空列表
    source_address_whitelist: [],

    # TPS 限制
    tps_limit: 50,

    # 队列检查频率
    queue_check_frequency: 1000,

    # 链路轮询间隔
    enquire_link_interval: 60,

    # 缓存配置
    cache_enabled: false,
    cache_max_size: 10000,
    cache_retry_interval: 60
  }
]
```

合作伙伴名称

name

合作伙伴名称

配置方法

項目	値	説明
接続先	IP	-

接続先を指定する

例

```
allowed_bind_types: [:transceiver] # 両方
allowed_bind_types: [:transmitter, :receiver] # TX だけ RX
allowed_bind_types: [:transmitter, :receiver, :transceiver] # 両方
```

接続先を指定する

ip_whitelist

接続先を指定する IP

項目	値	説明	接続先
接続先	IP	[]	IP だけ CIDR

接続先 - 接続先 IP

例

- IP "192.168.1.100" /32
- CIDR "192.168.1.0/24" "10.0.0.0/8"
- ["192.168.1.0/24", "10.50.1.100"]

例

```
# 空白 IP Whitelist
ip_whitelist: []

# 单个 IP
ip_whitelist: ["203.0.113.50"]

# 多个 IP
ip_whitelist: ["203.0.113.50", "203.0.113.51"]

# CIDR
ip_whitelist: ["192.168.1.0/24"]

# 混合
ip_whitelist: ["192.168.1.0/24", "10.50.1.100", "10.60.0.0/16"]
```

说明

- `/32` - 指定 IP 地址
- `/24` - 256 个地址 192.168.1.0-255
- `/16` - 65,536 个地址 10.50.0.0-255.255
- `/8` - 16,777,216 个地址 10.0.0.0-255.255.255

source_address_whitelist

说明

源地址	端口	协议	目标地址
192.168.1.0/24	80	HTTP	10.0.0.0/8

源地址、端口、协议和目标地址 ID 都是必填项

注意

```
# 空列表
source_address_whitelist: []

# 正则表达式列表
source_address_whitelist: ["^MyBrand$", "^AlertService$"]

# 正则表达式
source_address_whitelist: ["^614", "^\\+61"]

# 正则表达式列表
source_address_whitelist: ["^61[45]\\d{8}$"]

# 正则表达式
source_address_whitelist: ["^MyBrand$", "^614", "^\\+61400000001$"]
```

ESME_RINVSRCADR 参数在 [SOURCE_ADDRESS_WHITELIST.md](#) 中定义。

tps_limit

配置项

tps_limit 参数 - 限制 outbound deliver_sm 消息的速率。

queue_check_frequency

配置项

queue_check_frequency 参数 - 检查队列的频率。

enquire_link_interval

SMPP 配置项

enquire_link_interval 参数 - 发送 enquire_link PDU 消息的间隔。

enabled

配置项

cache_enabled 是否启用缓存

cache_enabled

是否启用缓存

cache_enabled 是否启用缓存 [MESSAGE_CACHE.md](#)

cache_max_size

缓存最大大小

cache_max_size 缓存最大大小

cache_retry_interval

缓存重试间隔

cache_retry_interval 缓存重试间隔

Web UI

Web UI

Web UI

配置

```
config :omnimessage_smpp, :listen, %{  
  host: "0.0.0.0",  
  port: 2775,  
  max_connections: 100  
}
```

主机

host

主机 IP

主机	IP	主机	IP
主机	IP	"0.0.0.0"	"0.0.0.0" 和 "127.0.0.1"

主机 IP

IP

- "0.0.0.0" - 所有 IP
- "127.0.0.1" - 本地主机
- "192.168.1.10" - 主机 IP

端口

端口 TCP

端口	端口	端口	端口
端口	端口	2775	1-65535

端口 SMPP

端口 2775

max_connections

???

??	??	??	??
??	?	100	1-10000

????????????

??

- ?????
 - ?????
 - ???10-100 ??
-

□□□□□□

□□ **1**□□□□□□□□

```
import Config

config :omnimessage_smpp,
  api_base_url: "https://smcsc.company.com:8443",
  verify_ssl_peer: true,
  smsc_name: "smpp_prod"

config :omnimessage_smpp, :binds, [
  %{
    name: "att_primary",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.att.com",
    port: 2775,
    system_id: "company_user",
    password: "secure_pass_123",
    tps_limit: 100,
    queue_check_frequency: 1000
  }
]
```

2

```
import Config

config :omnimessage_smpp,
  api_base_url: "https://smcsc.companys.com:8443"

config :omnimessage_smpp, :binds, [
  #
  %{
    name: "att_us",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.att.com",
    port: 2775,
    system_id: "att_username",
    password: "att_password",
    tps_limit: 100,
    queue_check_frequency: 1000
  },

  #
  %{
    name: "vodafone_uk",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.vodafone.co.uk",
    port: 2775,
    system_id: "voda_username",
    password: "voda_password",
    tps_limit: 50,
    queue_check_frequency: 1000
  }
]
```

3

```
import Config

config :omnimessage_smpp,
  api_base_url: "https://smcsc.companay.com:8443"

#
config :omnimessage_smpp, :binds, [
  %{
    name: "upstream_carrier",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.carrier.com",
    port: 2775,
    system_id: "my_username",
    password: "my_password",
    tps_limit: 100,
    queue_check_frequency: 1000
  }
]

#
config :omnimessage_smpp, :server_binds, [
  %{
    name: "partner_alpha",
    system_id: "alpha_corp",
    password: "alpha_secret",
    allowed_bind_types: [:transmitter, :receiver, :transceiver],
    ip_whitelist: ["203.0.113.0/24"],
    tps_limit: 50,
    queue_check_frequency: 1000
  },
  %{
    name: "partner_beta",
    system_id: "beta_inc",
    password: "beta_password",
    allowed_bind_types: [:transceiver],
    ip_whitelist: ["198.51.100.50"],
    tps_limit: 25,
    queue_check_frequency: 2000
  }
]
```

```
# 配置
config :omnimessage_smpp, :listen, %{
  host: "0.0.0.0",
  port: 2775,
  max_connections: 100
}
```

第4章 配置

在配置文件中，我们使用 `sinch.smpp` 来配置 IP 地址和端口。在配置文件中，我们使用 `sinch.smpp` 来配置 IP 地址和端口。

```

import Config

config :omnimessage_smpp,
  api_base_url: "https://smcsc.com:8443"

config :omnimessage_smpp, :binds, [
  %{
    name: "sinch_1",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.sinch.com",
    port: 2775,
    system_id: "sinch_user",
    password: "sinch_pass",
    tps_limit: 100,
    queue_check_frequency: 1000,

    # 消息队列
    queue: "sinch.smpp",

    # 本地IP/本地端口
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5001
  },
  %{
    name: "sinch_2",
    mode: :client,
    bind_type: :transceiver,
    host: "smpp.sinch.com",
    port: 2775,
    system_id: "sinch_user",
    password: "sinch_pass",
    tps_limit: 100,
    queue_check_frequency: 1000,

    # 与 sinch_1 共享消息队列
    queue: "sinch.smpp",

    # 本地 IP
    bind_source_ip: "10.20.30.40",
    bind_source_port: 5002
  }
]

```

```
}  
]
```

```
##### sinch.smpd ##### dest_smsc: "sinch.smpd"  
#####  
IP ##### SO_REUSEADDR #####  
  
##### queue ##### bind_source_ip ##### bind_source_port #####
```

####

#####

####

```
# ## Elixir ##  
/opt/omnimessage-smpd/bin/omnimessage-smpd eval "File.read!  
( 'config/runtime.exs' )"
```

#####

####

```
# #####  
sudo -u omnimessage-smpd /opt/omnimessage-smpd/bin/omnimessage-  
smpd console
```

Ctrl+C

####

systemd ##### shell

環境変数	設定名	設定値
API_BASE_URL	api_base_url	"https://10.17
SMSC_NAME	smsc_name	"smpp_gateway"
SMPP_POLL_INTERVAL	smpp_poll_interval	100
VERIFY_SSL_PEER	verify_ssl_peer	false
CACHE_FLUSH_INTERVAL	cache_flush_interval	10000
CACHE_MAX_RETRY_ATTEMPTS	cache_max_retry_attempts	10
CACHE_BACKOFF_MULTIPLIER	cache_backoff_multiplier	2
MNESIA_STORAGE_TYPE	mnesia_storage_type	disc_copies

systemd の設定

```
sudo systemctl edit omnimessage-smpp
```

```
[Service]
Environment="API_BASE_URL=https://omnimessage-
core.company.com:8443"
Environment="SMSC_NAME=smpp_prod_01"
Environment="VERIFY_SSL_PEER=true"
```

インストール

1. パッケージのインストール

```
sudo chmod 600 /opt/omnimessage-smpp/config/runtime.exs
sudo chown omnimessage-smpp:omnimessage-smpp /opt/omnimessage-smpp/config/runtime.exs
```

2. 設定ファイルの編集

- 12 行目
- 13 行目
- 14 行目

3. IP の設定

- 15 行目 `ip_whitelist`
- 16 行目 `[]`

4. SSL の設定

- 17 行目 `verify_ssl_peer: true`

5. ログの設定

- 18 行目
- 19 行目

ドキュメント

- [MONITORING.md](#) モニタリング
- [USAGE.md](#) 使用方法
- [TROUBLESHOOTING.md](#) トラブルシューティング
- [README.md](#) 概要



□□□□

A

API (□□□□□□□) □□□□□□□□□□□□□□□□

□□□□ Web UI □□□□□□□□□□□□□□□□□□□□

B

□□ SMPP □□□□□□□□□□□□□□□□

□□ □□□□□□ SMPP □□□□□□□□□□□□□□□□

□□□□ SMPP □□□□□□

- □□□□□□□□□□
- □□□□□□□□□□
- □□□□□□□□□□

□□□□ SMPP □□□□□□□□□□□□□□□□ IP □□□□

C

CIDR (□□□□□□) □□ IP □□□□□□□□□□□□ **192.168.1.0/24** □□ 256 □ IP □□□□

□□□□□ □□ SMPP □□□□□□□□□□ **ESME** □□□□□ **SMSC**□□□□□□□□□□ SMPP □□□□□□□□□□□□□□□□□□□□

□□□□ SMPP □□□□□□□□□□

- □□□□□□□□□□
- □□□□□□□□

- □ □ □ □ □ □ □ □ □ □ □ □

□ □ □ □

D

SMPP GSM-7 UCS-2

Deliver_SM

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

□□□□ (DLR) □□□□□□□□□□□□□□□□

[illegible]

dest smsc 000000000000 SMPP 00000000000000000000000000000000 0000 000000

[illegible]

E

[illegible]

ESM □ **SMPP** □□□□□□□□□□□□□□□□

ESME (ESME) は SMPP 経由で SMSC へメッセージを送信する際に、メッセージの送信先を指定する SMSC の ESME 番号を指定する必要がある。ESME 番号は

[illegible]

F

[illegible]

透過 SMPP 與 OmniMessage Core 連接後 60 秒內發送訊息失敗時，會自動重試 90 次。OmniMessage 會自動將訊息放入 `dest_smsc` 或 `queue` 中，等待重新發送。

G

透過 `SMPP` 發送訊息。

透過 `SMPP` 發送訊息時，會自動將訊息放入 `queue` 中，等待重新發送。

Grafana 是一個基於 Prometheus 的監控工具。

GSM-7 是一種 SMS 編碼，每個字元佔 160 位元。

H

HTTP/HTTPS 是一種 Web 協議，用於傳輸數據。

I

IP 是一個地址，用於識別網路設備。

ISDN (Integrated Services Digital Network) 是一種數位通訊技術。

J

(空行)

K

透過 `KPI` 發送訊息。

KPI (Key Performance Indicator) 是一個用於衡量效能的指標。

L

```
[[{"Prometheus": "bind_name=vodafone_uk"}]]
```

Phoenix Web UI

M

□□□□ □□□□□□□□□□□□□□□□□□□□

○○ ○○○○○○○○○○ Prometheus ○○○○○

MO ()

MT ()

MSISDN (XXXXXXXXXX) XXXXXXXXXXXX

N

NPI () SMPP ISDN

O

□□ □□□□□□□□□□

□□ □□□□□□□□□□

P

PDU () SMPP submit_sm deliver_sm

Prometheus

Q

□ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □

□ □ □ □ □ □

□□□□ □□□□□□□□□□ SMPP □□□□□

R

TPS

SMPP deliver_sm

□□□□ □□□□□□ SMPP □□

[illegible]

```

0000 0000 OmniMessage Core 00000000 dest_smsc 000000000000000000000000
0000 0000 queue 00000000 queue 00000000000000000000000000000000

```

S

[illegible]

ESMEs

SMSC

□□ □□□□□□□□ SMPP □□□

`sinch.smp` `queue`

000 0000000000000000 SMS 00/00000000 5-6 00000000 deliver_sm 0000 TON
00 000000000000000000000000

000 00 bind_source_ip 0 bind_source_port 0000000000000000 IP 000/0 TCP
 000000000000 IP 0000000000000000 CONFIGURATION.md

source_smsc **deliver_sm**

SMPP (Short Message Peer-to-Peer) SMS 送信

SMSC (送信元) → SMPP 送信元 → ESME(送信元) → SMS 送信元(送信元) →
→ 送信元 → 送信元 → ESME 送信元 → SMSC

SSL/TLS □□□□□□□□□□□□

Submit_SM SMPP PDU

Submit_SM_Resp □ submit_sm □ SMPP □□□□□□□□□□

□□ **ID** □□ SMPP □□□□□□□□

T

□ □ □ □ □ □ □ □ □ □ □ □ □ □

TON () SMPP

TPS ()

□□□ □□□□□□□□□□□□ SMPP □□□□□□□□□□

SMPP submit_sm

[illegible]

U

UCS-2 □□ SMS □ 16 □ Unicode □□□□□□□□□□ 70 □□□□

[illegible]

V

○○○ ○○○○○○○○○○○○○○○○○○○

W

Web ○○○ ○○○○○○○○○○○○○○○○○○○

○○○ ○ IP ○○○○○○○○○○○○○

X

(○○○)

Y

(○○○)

Z

(○○○)

□□□□□□□□

□□□	□□
API	□□□□□□□□
CIDR	□□□□□□
DLR	□□□□
ESME	□□□□□□□□
GSM	□□□□□□□□
HTTP	□□□□□□□□
HTTPS	□□□□□□□□□□
IP	□□□□□□
ISDN	□□□□□□□□
KPI	□□□□□□□□
MO	□□□□□
MSISDN	□□□□□□□□□□□□
MT	□□□□□
NPI	□□□□□□□□
PDU	□□□□□□□□
SMPP	□□□□□□□□
SMSC	□□□□□□□□

項目	説明
SMS	短メッセージサービス
SSL	セキュア・ソケット・レイヤー
TLS	トランスポート・レイヤー・セキュリティ
TON	トランスポート・ネットワーク
TPS	トランスポート・プロトコル
UCS	ユニーク・コンテンツ・システム
UI	ユーザー・インターフェース
URL	ユニーク・リソース・Locator

目次

- **README.md** - 概要
 - **CONFIGURATION.md** - 設定
 - **USAGE.md** - 使い方
 - **MONITORING.md** - 監視
 - **TROUBLESHOOTING.md** - トラブルシューティング
-

SMPP ☐☐☐☐

11

SMPP 0000000000000000 SMPP 000000 API 0000000000000000 Mnesia 00000000
000000000000 API0000000000000000

11

- **API** - **API** **SMPP**
- **Mnesia** - **:disc_copies**
- **Queue** - **Queue**
- **Queue** - **Queue** **SMPP**
- **Queue** - **Queue** **FIFO**
- **Queue** - **Queue**
- **Queue** - **Queue** **LiveView**
- **Prometheus** - **Prometheus**

11

111

□□□□□

SMPP Client → submit_sm → Gateway Server

↓

Cache in Mnesia (immediate response)

↓

CacheFlushWorker (background)

↓

Backend API

□□□□

```
SMPP Client → submit_sm → Gateway Server
                        ↓
                    Backend API (direct)
```

□□

1. **MessageCache** □□

- □□□□□□
- □□□□
- □□ LiveView □□□□□□□□

2. **CacheFlushWorker**

- □□□□□□□□□□ GenServer
- □□□□□□□□□□
- □□□□□□
- □□□□□□□□□□

3. **Mnesia** □ (:smpp_message_cache)

- □□ :disc_copies □□□□□
- □ bind_name□next_retry_at □ status □□
- □□□□□□□□□□

□□

□□□□

□□ config/runtime.exs:

```

config :omnimessage_smpp,
  # 配置缓存
  cache_flush_interval: 10_000,

  # 配置重试
  cache_max_retry_attempts: 10,

  # 配置退避
  cache_backoff_multiplier: 2,

  # Mnesia 存储 (:disc_copies 或 :ram_copies)
  mnesia_storage_type: :disc_copies

```

配置 SMPP

配置 SMPP 连接参数

```

config :omnimessage_smpp, :binds, [
  %{
    name: "my_smpp_bind",
    mode: :server,
    system_id: "username",
    password: "password",

    # 配置
    cache_enabled: true,           # 是否启用缓存
    cache_max_size: 10_000,       # 缓存最大大小
    cache_retry_interval: 60     # 缓存重试间隔
  }
]

```

环境变量

```

# 配置
CACHE_FLUSH_INTERVAL=10000           # 缓存刷新间隔
CACHE_MAX_RETRY_ATTEMPTS=10          # 缓存最大重试次数
CACHE_BACKOFF_MULTIPLIER=2           # 缓存退避系数
MNESIA_STORAGE_TYPE=disc_copies      # Mnesia 存储

```

メッセージ

キュー

メッセージキューの仕組み

メッセージ: 60 件

メッセージ: 2

キュー 0: 60s (1 件)

キュー 1: 120s (2 件)

キュー 2: 240s (4 件)

キュー 3: 480s (8 件)

キュー 4: 960s (16 件)

キュー 5: 1920s (32 件)

...

キュー 9: 30,720s (8.5 日)

メッセージ

約 10 分間隔でメッセージを送信する `failed_permanent` キュー

- メッセージを送信する
- メッセージを送信する
- メッセージを送信する“メッセージ”を送信
- メッセージを送信する

メッセージ

```
:pending → :delivering → SUCCESS (メッセージ)
               → FAILURE → :pending (メッセージ)
                           → :failed_permanent (メッセージ)
```



LiveView 简介

通过 `http://your-server:4000/smpp` → “接收”数据

接收数据

- 接收 - 接收数据
- 接收 - 接收数据
- 接收 - 接收数据

接收数据

- 接收
- 接收
- 接收 / 接收
- 接收
- 接收

Prometheus 简介

```
# 接收数据
smpp_cache_size{bind_name="my_bind",mode="server"} 42

# 接收数据
smpp_cache_delivered_total{bind_name="my_bind"} 1234

# 接收数据
smpp_cache_retry_total{bind_name="my_bind"} 56

# 接收数据
smpp_cache_permanent_failures_total{bind_name="my_bind"} 2

# 接收数据
smpp_cache_overflow_total{bind_name="my_bind"} 0
```

□□□□

```
INFO Message -123456789 cached for my_smpp_bind
INFO Successfully delivered cached message -123456789, API ID:
99999
WARN Failed to deliver message -123456789 (retry 3/10), next
retry at 2026-02-01 12:34:56Z
ERROR Message -123456789 exceeded max retries (10), marking as
failed_permanent
WARN Cache overflow for my_smpp_bind: deleted oldest message
```

□□

□□/□□□□

□□ **LiveView UI**

1. □□□ `http://your-server:4000/smpp`
2. □□“□□□□□”□“□□□□□”□□□
3. □□□□
4. □□“□□□□□”□□□
5. □□□□

□□ **IEx** □□□

```
# 测试缓存
SmsC.SMPPConfig.update_server_peer("my_bind", "username",
    "password",
    cache_enabled: true,
    cache_max_size: 10_000,
    cache_retry_interval: 60
)

# 测试不缓存
SmsC.SMPPConfig.update_server_peer("my_bind", "username",
    "password",
    cache_enabled: false
)
```

测试消息缓存

```
# 测试消息缓存
SmsC.SMPP.MessageCache.get_cache_summary()
# => %{total_cached: 42, pending: 40, failed: 2}

# 测试按绑定获取消息缓存
SmsC.SMPP.MessageCache.get_cache_by_bind()
# => [
#   %{bind_name: "bind1", total: 30, pending: 28, failed: 2,
#     max_size: 10_000},
#   %{bind_name: "bind2", total: 12, pending: 12, failed: 0,
#     max_size: 10_000}
# ]

# 测试统计缓存消息数量
SmsC.SMPP.MessageCache.count_cached_messages("my_bind")
# => 42
```

测试发送

测试发送消息

```
# 删除失败消息
{:atomic, failed_messages} = :mnesia.transaction(fn ->
  :mnesia.match_object({:smpp_message_cache, :_, "my_bind", :_,
    :_, :_, :_, :_, :_, :failed_permanent})
end)

# 遍历
Enum.each(failed_messages, fn {_, {bind_name, msg_id}, _, _, _, _,
  _, _, _, _} ->
  SmsC.SMPP.MessageCache.delete_cache_record(bind_name, msg_id)
end)
```

删除失败消息

```
# 将 failed_permanent 消息改为 pending
SmsC.SMPP.MessageCache.update_cache_record("my_bind", -123456, %{
  status: :pending,
  retry_count: 0,
  next_retry_at: DateTime.utc_now(),
  last_error: nil
})
```

配置

配置 / 配置

配置: `cache_overflow_total` 配置

配置: 配置

配置:

1. 配置 `cache_max_size`
2. 配置 API 配置 API 配置
3. 配置
4. 配置

□□□□□□

□□: □□□□ :pending □□

□□□□:

1. API □□□

- □□ API □□□
- □□□□ API □□
- □□□□□□

2. next_retry_at □□□

- □□□□□□ next_retry_at ⚠️⚠️□□
- □□□□□□□□

3. □□□□□□□□

```
# □□□□□□□□□□  
Supervisor.which_children(SmsC.SMPP.Supervisor)
```

4. □□□□□

- □□□□□□□□ cache_enabled: true

□□□□□□

□□: □□□□□□□□ retry_count □

□□:

```
# 消息队列实现
{:atomic, messages} = :mnesia.transaction(fn ->
  :mnesia.match_object({:smpp_message_cache, :_, "my_bind", :_,
    :_, :_, :_, :_, :_, :_})
end)

high_retry = Enum.filter(messages, fn {_ , _ , _ , _ , _ , _ ,
  retry_count, _ , _ , _} ->
  retry_count >= 5
end)

# 记录最后错误
Enum.each(high_retry, fn {_ , _ , _ , msg_id, _ , _ , retry_count, _ ,
  last_error, _} ->
  IO.puts("Message #{msg_id}: #{retry_count} retries, error: #
    {inspect(last_error)}")
end)
```

Mnesia 简介

简介: 消息队列

简介 **Mnesia** 简介:

```
ls -lh Mnesia.*
du -sh Mnesia.*
```

简介:

1. 消息队列实现
2. 消息队列 `cache_max_size`
3. 消息队列实现 FIFO 队列

メモリ

メモリ

- メモリ容量 500-1000 MB程度
- 10,000 項目 $\approx$ 5-10 MB 程度
- `:disc_copies` 項目数

CPU

- プロセッサ 10 個程度
- メモリ 100 MB程度
- プロセッサ 10 個 API 項目

I/O

- `:disc_copies` 項目数
- プロセッサ数>1000 項目/項目
 - `:ram_copies` 項目数
 - プロセッサ
 - プロセッサ

メモリ

項目	cache_max_size	cache_flush_interval
項目 (<100 項目/項目)	10,000	10,000ms
項目 (100-500 項目/項目)	50,000	5,000ms
項目 (>500 項目/項目)	100,000	3,000ms

□□□□

□□□□

1. Mnesia □□□□□□ □ `:disc_copies` □
2. □□□□□□□□
3. □□□□□□□□□□□□□□

□□□□□

□□□□□□□□□□□□

1. □□□□□□□□□□□□□□□□
2. □□□□ `cache_enabled: false` □ `cache_max_size: 10,000` □
`cache_retry_interval: 60`
3. □□□□□
4. □□□□□□□□□□□□

API □□□□

1. □□□□□□□□□□□□□□
2. □ API □□□□□□□□□□□□□□
3. □□□□□□□□□□FIFO□
4. □□□□□□□□□□ API □□

□□□□

1. □□□□□□ - □□□□□□□□□□□□
2. □□□□ - □□□□ `cache_permanent_failures_total` □ `cache_overflow_total` □□□□
3. □□□□□□ - □□□□□□□□□□□□□□□□ `cache_max_size`
4. □□□□□□ - □□□□□□□□□□□□□□□□□□□
5. □□□□□□ - □□ API □□□□□□□□□□
6. □□□□□□ - □□ API □□□□□□□□□□□□
7. □□□□□□ - □□□□□□□□ `mnesia_storage_type: disc_copies`



- Four small orange squares arranged horizontally in a row.
- Five small orange squares arranged horizontally in a row.
- Four small orange squares arranged horizontally in a row.

Getting Started

SMPP Configuration

Prerequisites

SMPP requires Prometheus to be installed on the server.

OmniMessage Core provides an **OmniMessage** API endpoint for monitoring.

1. **SMPP** - Setup
2. **OmniMessage API** - Configuration

Configuration

URL: `http://your-server:4000/metrics`

Target: Prometheus

Host: `localhost`

Example

```
curl http://localhost:4000/metrics
```

Labels

Labels: `smpp_`

□ □ □ □ □

omnmessage_smpp_license_status

□□: Gauge

□□ : □□□□□□

2

- 1 = □□□□□
- 0 = □□/□□□□□

□□: □

11

```
omnimessage_smpp_license_status 1
```

□□:

- 0000000000000000
- 0000000000000000SMPP000000
- Web UI000000000000

```
0000: omnimessage_smpp
```

□□:

- `license_status == 0`
- SMPP
-
- UI

0000

```
- alert: SMPP_License_Invalid
  expr: omnimessage_smpp_license_status == 0
  for: 1m
  labels:
    severity: critical
  annotations:
    summary: "SMPP许可证过期"
    description: "许可证过期 - 请及时续费"
```

配置项

smpp_connection_status

类型: Gauge

单位: SMPP许可证数量

值:

- 1 = 许可证已过期
- 0 = 许可证有效

配置:

- bind_name - 许可证名称 "vodafone_uk"
- mode - 模式 "client" 或 "server"
- host - 许可证服务器地址
- port - 许可证服务器端口
- bind_type - SMPP许可证类型
- system_id - 许可证ID

示例:

```
smpp_connection_status{bind_name="vodafone_uk",mode="client",host="sn
1
```

备注:

- 0000000000000000
- 0000000000000000
- 000000

000000

smpp_messages_sent_total

00: Counter

00: 00SMPP0000000000

00: 00

00: 0connection_status00

00:

```
smpp_messages_sent_total{bind_name="vodafone_uk",mode="client",...}
150234
```

00:

- 0000000000/00
- 0000/0000
- 000000000000

smpp_messages_received_total

00: Counter

00: 00SMPP0000000000

00: 00

00: 0connection_status00

00:

```
smpp_messages_received_total{bind_name="partner_acme",mode="server",...}
45123
```

📌:

- 📶📶📶📶
- 📶📶📶📶MO📶📶
- 📶📶📶📶📶📶

📶📶📶

smpp_delivery_failures_total

📌: Counter

📌: 📶📶📶📶📶📶

📌: 📶

📌: 📶connection_status📶

📌:

```
smpp_delivery_failures_total{bind_name="vodafone_uk",mode="client",...}
234
```

📌:

- 📶📶📶📶
- 📶📶📶📶📶📶
- 📶📶📶📶

📶📶📶:

```
success_rate = (messages_sent - delivery_failures) / messages_sent
* 100
```

概要

smpp_bind_success_total

単位: Counter

単位: 整数

単位: 整数

単位:

```
smpp_bind_success_total{bind_name="vodafone_uk",...} 45
```

単位:

- 整数
- 整数

smpp_bind_failures_total

単位: Counter

単位: 整数

単位: 整数

単位:

```
smpp_bind_failures_total{bind_name="vodafone_uk",...} 3
```

単位:

- 整数
- 整数
- 整数

概要

smpp_connection_attempts_total

Counter

Counter

Counter

Counter

```
smpp_connection_attempts_total{bind_name="vodafone_uk",...} 48
```

Counter

- Counter
- Counter

smpp_disconnection_total

Counter

Counter

Counter

Counter

```
smpp_disconnection_total{bind_name="vodafone_uk",...} 3
```

Counter

- Counter
- Counter
- Counter

Counter

smpp_enquire_link_sent_total

Counter

Counter

Counter

📄: 📄connection_status📄

📄:

```
smpp_enquire_link_sent_total{bind_name="vodafone_uk",mode="client",...
1440
```

📄:

- 📄📄📄📄
- 📄📄📄📄📄📄📄📄📄📄

smpp_enquire_link_received_total

📄: Counter

📄: 📄📄📄📄📄📄📄📄📄PDU📄

📄: PDU

📄: 📄connection_status📄

📄:

```
smpp_enquire_link_received_total{bind_name="vodafone_uk",mode="client
1438
```

📄:

- 📄📄📄📄📄📄📄 >> 📄📄
- 📄📄📄📄📄📄📄📄📄📄

📄📄📄📄📄📄

smpp_uptime_seconds

📄: Gauge

📄: 📄SMPP📄📄📄📄📄📄📄📄

📄: 📄

11

```
smpp_uptime_seconds{bind_name="vodafone_uk",...} 86400
```

11

- 
- 
- 

□ □ □ □ □ □

MESSAGE_CACHE.md

smpp_cache_size

□□: Gauge

□□ : □□□□□□□□□□□□□□□□

□□ : □□

11

- `bind_name` - 字符串
- `mode` - 字符串“client”或“server”

11

```
smpp_cache_size{bind_name="partner_acme",mode="server"} 42
```

11

- 00000000
- 000 cache_max_size 000000

smpp_cache_delivered_total

Counter

API

```
smpp_cache_delivered_total{bind_name="partner_acme"} 1234
```

smpp_cache_retry_total

Counter

```
smpp_cache_retry_total{bind_name="partner_acme"} 56
```

smpp_cache_permanent_failures_total

Counter

```
smpp_cache_permanent_failures_total{bind_name="partner_acme"} 2
```

- > 0

smpp_cache_overflow_total

Counter

00:

```
smpp_cache_overflow_total{bind_name="partner_acme"} 0
```

00:

- 0000000000000000API00000000

OmniMessage API0000

00000000SMPP000000OmniMessage API00000000000000

00OmniMessage0000000000

- omnimessage_api_requests_total - 000000API0000
- omnimessage_api_request_duration_seconds - API0000
- omnimessage_queue_depth - OmniMessage0000000000

0000000000000000

0000000000API0000

- "api.*connection refused" - 000000OmniMessage
- "api.*timeout" - OmniMessage0000
- "api.*http 503" - OmniMessage000000
- "api.*parse error" - 000000

Prometheus00

00000000

0000/etc/prometheus/prometheus.yml0

```

scrape_configs:
  - job_name: 'omnimessage-smpp'
    scrape_interval: 15s
    static_configs:
      - targets: ['your-server:4000']
        labels:
          environment: 'production'
          service: 'omnimessage-smpp'

```

□□□□

```

scrape_configs:
  - job_name: 'omnimessage-smpp-instances'
    scrape_interval: 15s
    static_configs:
      - targets:
          - 'smpp-gw-1:4000'
          - 'smpp-gw-2:4000'
          - 'smpp-gw-3:4000'
        labels:
          environment: 'production'

```

□□□□

□□□□□□□□□□

```

scrape_configs:
  - job_name: 'omnimessage-smpp-instances'
    file_sd_configs:
      - files:
          - '/etc/prometheus/targets/smpp-*.json'

```

□□/etc/prometheus/targets/smpp-production.json□

```
[
  {
    "targets": ["smpp-gw-1:4000", "smpp-gw-2:4000"],
    "labels": {
      "environment": "production",
      "datacenter": "us-east"
    }
  }
]
```

Grafana

:

```
smpp_connection_status{job="omnimessage-smpp"}
```

: Stat

:

- : < 1
- : == 1

:

```
rate(smpp_messages_sent_total{job="omnimessage-smpp"}[5m])
```

: Graph

: /

: {{bind_name}}

报警规则

规则:

```
100 * (1 - (
    rate(smpp_delivery_failures_total{job="omnimessage-smpp"}[5m])
    /
    rate(smpp_messages_sent_total{job="omnimessage-smpp"}[5m])
))
```

类型: Gauge

范围: 0-100

阈值:

- 阈值: < 95%
- 阈值: 95-98%
- 阈值: > 98%

报警规则

规则:

```
smpp_uptime_seconds{job="omnimessage-smpp"} / 3600
```

类型: Stat

范围: 0

报警规则

Prometheus报警规则

规则文件: `/etc/prometheus/rules/smpp-alerts.yml`

```

groups:
- name: smpp_gateway
  interval: 30s
  rules:
    # 连接
    - alert: SMPPConnectionDown
      expr: smpp_connection_status == 0
      for: 2m
      labels:
        severity: critical
      annotations:
        summary: "SMPP连接{{ $labels.bind_name }}断开"
        description: "连接{{ $labels.bind_name }}已断开2分钟"

    # 失败率
    - alert: SMPPHighFailureRate
      expr: |
        (
          rate(smpp_delivery_failures_total[5m])
          /
          rate(smpp_messages_sent_total[5m])
        ) > 0.05
      for: 5m
      labels:
        severity: warning
      annotations:
        summary: "连接{{ $labels.bind_name }}失败率过高"
        description: "连接{{ $labels.bind_name }}失败率{{ $value | humanizePercentage }}%{{ $labels.bind_name }}"

    # 绑定失败
    - alert: SMPPBindFailures
      expr: increase(smpp_bind_failures_total[10m]) > 3
      labels:
        severity: warning
      annotations:
        summary: "连接{{ $labels.bind_name }}绑定失败"
        description: "连接{{ $labels.bind_name }}在10分钟内发生了{{ $value }}次绑定失败"

    # 无流量
    - alert: SMPPNoTraffic
      expr: rate(smpp_messages_sent_total[10m]) == 0

```

```

    for: 30m
    labels:
      severity: warning
    annotations:
      summary: "⚠️ {{ $labels.bind_name }} 连接异常"
      description: "{{ $labels.bind_name }} 连接异常持续 30 分钟"

# 连接异常
- alert: SMPPFrequentDisconnections
  expr: increase(smpp_disconnection_total[1h]) > 5
  labels:
    severity: warning
  annotations:
    summary: "⚠️ {{ $labels.bind_name }} 连接异常"
    description: "{{ $labels.bind_name }} 连接异常持续 {{ $value }} 分钟"

# OmniMessage API 连接异常
- alert: OmniMessageAPIUnreachable
  expr: |
    count(count_over_time({job="omnmessage-smpp"} |=
"api.*connection refused"[5m])) > 0
  for: 1m
  labels:
    severity: critical
  annotations:
    summary: "OmniMessage API 连接异常"
    description: "SMPP 连接异常 OmniMessage API 连接异常 API_BASE_URL 连接异常"

# OmniMessage API 超时
- alert: OmniMessageAPITimeout
  expr: |
    count(count_over_time({job="omnmessage-smpp"} |=
"api.*timeout"[5m])) > 5
  for: 2m
  labels:
    severity: warning
  annotations:
    summary: "OmniMessage API 超时"
    description: "SMPP 连接异常 OmniMessage API 连接异常"

# 连接异常 API 连接异常
- alert: NoMessageFlow

```

```
    expr: rate(smpp_messages_sent_total[10m]) == 0 and
rate(smpp_messages_received_total[10m]) == 0
    for: 30m
    labels:
      severity: warning
    annotations:
      summary: "SMPP - 0 OmniMessage"
      description: "SMPP 30초 동안 SMPP 메시지 전송/수신 0 OmniMessage API 호출됨"
```

prometheus.yml

```
rule_files:
  - '/etc/prometheus/rules/smpp-alerts.yml'
```

Web

Web UI Prometheus

URL: https://your-server:8087

SMPP →

:

-
-
-
-
-

:

- 
- 
- 

□ □ □ □ □ □

- [illegible]

□ □ □ □

--	--	--	--

□□□□:

```
# 確認
sudo journalctl -u omnimessage-smpp -f

# 100行
sudo journalctl -u omnimessage-smpp -n 100

# 1時間前以降
sudo journalctl -u omnimessage-smpp --since "1 hour ago"

# エラー
sudo journalctl -u omnimessage-smpp -p err
```

Web UI

Web UI

:

-
-
-
-
-

- 目標値: 000000000000000000000000
- 単位: 0000000000000000
- 範囲値: 00/0000000000000000
- 単位/範囲: 0000000000000000
- 単位: 000000000000

00000000**KPI**0

00000

単位: 000000000000

```
avg_over_time(smpp_connection_status[24h]) * 100
```

単位: > 99.9%

000000

単位: 00000000

```
rate(smpp_messages_sent_total[5m])
```

単位: 000000

000000

単位: 0000000000

```
100 * (1 - rate(smpp_delivery_failures_total[5m]) /  
rate(smpp_messages_sent_total[5m]))
```

単位: > 98%

📌 📌 📌 📌

📌: 📌 📌 📌 📌

```
rate(smpp_bind_success_total[1h]) * 3600
```

📌: < 10 📌 📌 📌 📌 📌 📌 📌

📌 📌 📌 📌 📌

1. 📌 📌

- 📌 📌 📌 Prometheus 📌
- 📌 PagerDuty/OpsGenie 📌 24/7 📌
- 📌 📌 📌

2. 📌 📌 📌

- 📌 📌 📌 Grafana 📌
- 📌 📌 📌 📌 📌 📌
- 📌 📌 📌 📌

3. 📌 📌

- 📌 📌 📌
- 📌 📌 📌 📌
- 📌 📌 📌

4. 📌 📌

- 📌 📌 📌 📌
- 📌 📌 TPS 📌
- 📌 📌 📌 / 📌

5. 配置

- 配置API
 - 配置参数
 - 配置
-

配置参数

配置

配置: `smpp_connection_status`

- 0 = 配置参数
- 配置 = 配置

配置

配置: `smpp_delivery_failures_total`

- 配置 = 配置参数
- 配置参数 = 配置

配置

配置: `smpp_messages_sent_total`

- 配置 = 配置TPS
- 配置API

配置

配置: `smpp_bind_failures_total`

- 配置 = 配置参数
- 配置参数system_id

□□□□

- **CONFIGURATION.md** - □□□□□
 - **USAGE.md** - □□□□
 - **TROUBLESHOOTING.md** - □□□□
 - **README.md** - □□□□□□□
-

Regex	Match	Result
<code>^61[45]\d{8}\$</code>	614/615 11	61400000001, 61512345678
<code>^(MyBrand AlertSvc)\$</code>		MyBrand, AlertSvc
<code>(?i)^mybrand\$</code>		MyBrand, mybrand, MYBRAND
<code>^(?!Test).+</code>	Test	MyBrand, TestBrand Test

Regex

Regex is a powerful tool for matching patterns in text.

Example: `^MyBrand$, ^614, ^\+61400000001$`

Input	Regex	Match
MyBrand	<code>^MyBrand\$</code>	Match
61412345678	<code>^614</code>	Match
+614000000001	<code>^\+614000000001\$</code>	Match
OtherBrand		Match
615000000001		Match

Regex

Example: `submit_sm`

項目	値
PDU	submit_sm_resp
送信先	0x00000000A
送信元	ESME_RINVSRCADR送信元アドレス
送信ID	

送信元アドレスがホワイトリストに登録されていないため拒否されました

```
SMPP Server: Rejected submit_sm from partner_acme - source_addr 'UnauthorisedBrand' not in whitelist
```

送信元アドレスがホワイトリストに登録されていないため拒否されました

??

Web UI

- 1. 送信元 SMPP > Server Peers
- 2. 送信元アドレス Edit 送信元 Add New Server Peer
- 3. 送信元 Source Address Whitelist 送信元 IP アドレス
- 4. 送信元アドレスを登録する

```
^MyBrand$,^614,^\+61400000001$
```

- 5. 送信元 Save

送信元アドレスを登録する submit_sm PDU

?? 送信元

送信元 source_address_whitelist 送信元 runtime.exs 送信元

```
config :omnimessage_smpp, :server_binds, [
  %{
    name: "partner_acme",
    system_id: "acme_corp",
    password: "secure_password",
    allowed_bind_types: [:transmitter, :receiver, :transceiver],
    ip_whitelist: ["203.0.113.0/24"],
    source_address_whitelist: ["^MyBrand$", "^614",
    "^\\+614000000001$"],
    tps_limit: 50,
    queue_check_frequency: 1000
  }
]
```

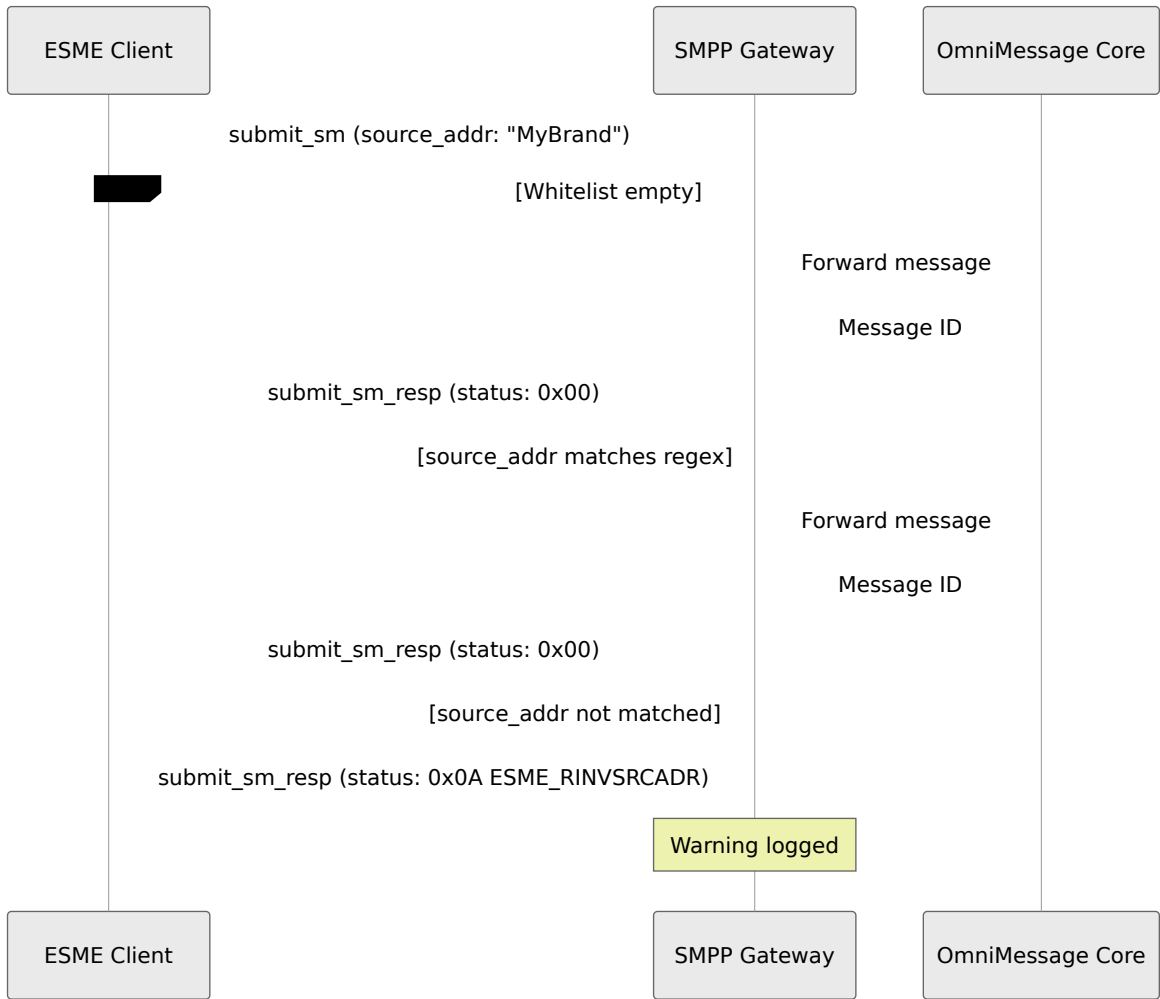
Table

Field	Type	Default	Options	Description
source_address_whitelist	Array	["^MyBrand\$", "^614", "^\+614000000001\$"]	["^MyBrand\$", "^614", "^\+614000000001\$"]	Regexes to match source addresses

Table

Table with 10 columns and 1 row

□□□□



□□

□□□□□□□□

□□□□□□ID□AcmeCorp□□□□

^AcmeCorp\$

□□□□□□□□□□

□□□□□□□□□□□□614□□□□

^614

□□□□□□□□

□□□□□614□615□□□□□□11□□□□□

^61[45]\d{8}\$

□□□□□□□□

□□□□□□□□□□□□

^AcmeCorp\$,^614,^\+61290000001\$

□□□□□□□□□□

□□□□□□□□□□□□

(?i)^acmecorp\$

□□□□□□□□

□□□□□□□□□□□□□□□□

□□□□

□□□□□□□□□□**ESME_RINVSRCADR**

□□□□□□□□ submit_sm_resp □□□□□□ 0x0A □

□□□□□

- 000000000000000000000000
- 000000000000000000000000
- 000000^/\$0000000000000000
- 000000000000000000000000 (?i) 0000000000000000

000000

1. 00Web UI0000000000000000
2. 000000000000000000000000
3. 00Elixir0000000000000000 Regex.match?(~r/^pattern\$/, "address")
4. 0000000000000000
5. 0000000000000000000000 +000000 \+

00000000

000000000000000000000000

000000

- 0000000000000000
- 0000000000000000000000 614 00 X614Y
- ESME0000000000000000
- 0000000000000000000000

000000

1. 00Web UI0000000000000000
2. 0000000000000000000000 ^\$
3. 00ESME0000000000000000000000
4. 0000000000000000000000

000000

- 00000 - 0000000000000000
- 00000 - 00SMPP

- 0000 - 00000000

API 実装

インストール

インストール / 実行

OmniMessage Core 8443 :eaddrinuse

確認

- systemctl show <unit> -p NRestarts
-

```
Running ApiEx.Endpoint with Bandit ... at https failed, port
8443 already in use
... failed to start child: ApiEx.Supervisor ... ** (EXIT)
:eaddrinuse
```

- curl -sk https://<host>:8443/api/messages 404

SMPP OmniMessage Core 8443 api_ex API

OAM/API Core Core 8443

/etc/omnmessage_smpp/runtime.exs api_ex

api_base_url Core API

```
# /etc/omnmessage_smpp/runtime.exs
config :api_ex,
  api:
    Application.get_env(:api_ex, :api, %{})
    |> Map.put(:port, 8444)
    |> Map.put(:listen_ip, "0.0.0.0")
```

インストール

```
sudo systemctl restart omnimessage-smpp
sudo systemctl restart omnimessage # Core 8443
```

Core 8443

```
ss -lntp | grep -E ':8443|:8444'
curl -sk -o /dev/null -w '%{http_code}\n'
https://127.0.0.1:8443/api/messages # 404
```

CONFIGURATION.md → OAM/API

OmniMessage

SMPP OmniMessage Core OmniMessage

OmniMessage

-
-
- API
- "HTTP 503"

1. OmniMessage

```
#
curl -k -v https://omnimessage-
core.example.com:8443/api/system/health

#
ssh gateway-server 'curl -k https://omnimessage-
core.example.com:8443/api/system/health'
```

2. 確認 API URL

```
# 確認
grep -Al 'api_base_url' /opt/omnimessage-smpp/config/runtime.exs

# 確認接続
ping omnimessage-core.example.com
nc -zv omnimessage-core.example.com 8443
```

3. 確認 API ログ

```
# API ログを確認
sudo journalctl -u omnimessage-smpp -f | grep -i
'api\|omnimessage\|connect'

# エラーログを確認
sudo journalctl -u omnimessage-smpp -n 200 | grep -i error
```

確認

OmniMessage 確認

- OmniMessage 確認
- 確認
- 確認 `SMPP_POLL_INTERVAL`
- OmniMessage 確認

OmniMessage 確認

- 確認 HTTPS
- DNS 確認 `nslookup omnimessage-core.example.com`
- 確認 `traceroute omnimessage-core.example.com`
- SSL 確認 HTTPS

API URL 確認

- `/opt/omnimessage-smpp/config/runtime.exs`
- `api_base_url` 確認 HTTPS

3. 確認 `sudo systemctl restart omnimessage-smpp`

確認

確認

確認

- 確認“確認”確認
- 確認確認
- 確認確認

確認確認

1. 確認

確認

```
# 確認 DNS 確認
nslookup smpp.carrier.com

# 確認
ping -c 3 smpp.carrier.com

# 確認
telnet smpp.carrier.com 2775
# 確認
nc -zv smpp.carrier.com 2775
```

確認

- 確認 DNS 確認確認 IP 確認確認
- 確認 ping 確認確認確認確認
- 確認確認確認確認確認確認

2. 確認

📌

- 📄 “📄” “📄”
- Web UI → SMPP → 📄 system_id 📄

📌

- 📄
- 📄
- 📄

3. IP 📄

📌

- 📄
- 📄 IP

📌

- 📄 IP

```
curl ifconfig.me
```

- 📄 IP 📄
- 📄 IP 📄 IP

4. 📄

📌

```
# 📄
sudo iptables -L -n | grep 2775

# 📄 UFW📄Ubuntu/Debian📄
sudo ufw status | grep 2775

# 📄 firewalld📄RHEL/CentOS📄
sudo firewall-cmd --list-ports | grep 2775
```

5

```
# Ubuntu/Debian
sudo ufw allow out 2775/tcp

# RHEL/CentOS
sudo firewall-cmd --permanent --add-port=2775/tcp
sudo firewall-cmd --reload
```

5. □□□□□/□□□□□

```
bind_source_ip 0/0 bind_source_port 000000
CONFIGURATION.md
```

111

- `bind_source_ip`
- `bind_source_port`

□□□□□

- `netstat -tln | grep bind_source_ip` 查看 IP 地址是否被监听
- `ss -ltn | grep bind_source_ip` 查看 IP 地址是否被监听
- `ss -ltn | grep bind_source_ip` 查看 IP 地址是否被监听

□ □ □ □ □ □

111

- 00000000
- `smpp_disconnection_total` 0000
- 00000000

□ □ □ □ □ □ □ □ □ □

1. 网络测试

网络

```
# 测试
ping -c 100 smpp.carrier.com | grep loss

# 测试丢包
netstat -s | grep -i error
```

网络

- 网络延迟/抖动
- 网络带宽/ISP
- 网络拥塞/丢包

2. 网络配置

网络

- 配置“enquire_link timeout”
- 配置网络参数

网络

- 配置 30 s
- 配置网络参数
- 配置网络参数

3. 配置 TPS 参数

网络

- 配置网络参数
- 配置网络参数

网络

- 配置 `tps_limit` 参数

- TPS 70-80%
-

4.

-
-

-
-

-
- `smpp_messages_sent_total`
-

1. `dest_smsc`

- Web UI → → `dest_smsc`
- SMPP →

- `dest_smsc`

- 00000000000000000000 `dest_smsc`
- 00 `dest_smsc` 0 NULL000000000000

2. 000000000000

0000

- Web UI → 00 → 00 `deliver_after` 00
- 00000000000000000000

0000

- 00000000000000000000 `deliver_after`
- 00000000000000000000

000000

- 00000000
- 000000000000000000000000

3. TPS 00000

0000

- 00000000
- 000000000000

000000

- 00000000 `tps_limit`
- 000000000000000000000000
- 00 **CONFIGURATION.md**

4. 000000000000

0000

- 000000
- 00000000

5/5

```
# 確認
sudo systemctl restart omnimessage-smpp

# 確認
sudo journalctl -u omnimessage-smpp -f
```

5. □□□□□□□□

OmniMessage Core dest_smsc

- `queue` `queue`
-

111

- `Successfully registered SMPP peer frontend '<name>'` `<name>` `dest_smsc`
- `queue`

□□□□□

- 配置 `dest_smsc` 和 `queue` 为 `{hostname}_{peer}` 格式
- 配置 `queue` 为 `queue` 格式
- 配置 `queue` 为 `queue` 格式

□ □ □ □ □

111

- `smpp_delivery_failures_total` □□
- □□□□“submit_sm_resp”□□□□□□
- □□□□□□□□

□□□□□□□□□□

1. □□□□□□□□

□□□

- □□□□□□□□□□
- □□□□□□□□□□

□□□□□□□□

- 0x0000000B - □□□□□
- 0x00000001 - □□□□□□
- 0x00000003 - □□□□

□□□□□

- □□□□□□  E.164□
- □□□□□□□□□□□□
- □□□□□□□□

2. □□□□□□□□

□□□

- □□□□
- □□□□
- □□

□□□□□

- GSM-7□□□ 160 □□□
- UCS-2□□□ 70 □□□
- □□□□□□□□□□
- □□□□□□□□

3. □□□□□□

□□□

- 環境変数
- 設定

確認

- 環境変数
- 設定
- 環境変数/設定

4. 確認

確認

- 環境変数 `expires` 設定
- 設定

確認

- 環境変数
- 環境変数/設定

Web UI 確認

環境変数 Web 確認

確認

- 環境変数 `https://your-server:8087`
- 設定

環境変数/設定

1. 確認

確認

```
sudo systemctl status omnimessage-smpp
```

□□□□□

```
# □□□□□□□□□□
sudo systemctl start omnimessage-smpp

# □□□□□□□□□□
sudo journalctl -u omnimessage-smpp -n 50
```

2. □□□□□□□ 8087

□□□

```
sudo ufw status | grep 8087
# □
sudo firewall-cmd --list-ports | grep 8087
```

□❓❓□□□

```
# Ubuntu/Debian
sudo ufw allow 8087/tcp

# RHEL/CentOS
sudo firewall-cmd --permanent --add-port=8087/tcp
sudo firewall-cmd --reload
```

3. SSL □□□□

□□□

- □□□□□□□□□□
- □□□□□□□□

□□□□□

- □□□□□□□□□□□□□□□□

- 証明書 SSL 用
- 証明書インストール

```
ls -l /opt/omnimessage-smpp/priv/cert/
```

4. URL 用

用

- 証明書 HTTPS 用 HTTP
- 証明書 IP/用
- 証明書 8087

Web UI 用

用

- 証明書
- 証明書
- 証明書

用

1. 証明書

- Ctrl+F5 用
- 証明書 Cookie

2. 証明書

- F12
- 証明書 JavaScript 用
- 証明書

3. 証明書

- Chrome Firefox Edge 用
- 証明書

4. 確認

```
sudo journalctl -u omnimessage-smpp -f
```

確認

Prometheus 確認

確認

- `curl http://localhost:4000/metrics` 確認
- Prometheus 確認
- 確認

確認

1. 確認

確認

```
sudo systemctl status omnimessage-smpp
```

確認

```
sudo systemctl start omnimessage-smpp
```

2. 確認

確認

```
# 本地测试
curl http://localhost:4000/metrics

# 远程测试
curl http://your-server-ip:4000/metrics
```

验证步骤

- 使用浏览器或命令行工具访问 `http://localhost:4000/metrics`
- 看到 Prometheus 返回的文本内容 4000

3. 部署 Prometheus

部署

- 访问 `/metrics` 返回 `/prometheus` 和 `/stats`
- 访问 4000 端口 8087

部署 Prometheus

部署

- 部署 Prometheus
- 部署 Grafana
- 部署 Alertmanager

部署 Prometheus

1. 部署 Prometheus

- 部署 Prometheus
- 部署 Grafana
- 使用 Prometheus 的 `increase()` 和 `rate()`

2. 部署 Grafana

- 部署 Grafana

- 消息队列
- 消息中间件

3. 消息队列

- 消息队列
- 消息中间件
- 消息 Prometheus 监控



CPU 监控

监控

```
top -p $(pgrep -f omnimessage-smpp)
```

监控

- 监控
- 监控
- 监控

监控

- 监控
- 监控 TPS
- 监控 CPU 使用率

监控

监控

```
ps aux | grep omnimessage-smpp
```

環境構築

- 仮想マシン構築
- ネットワーク構築

基本操作

- 仮想マシン起動
- ネットワーク接続
- 仮想マシン管理ツールのインストール

インストール

準備

- ディレクトリ作成
- ファイルコピー
- パッケージインストール

実行

1. TPS 実行 - 実行コマンド
2. `queue_check_frequency` - 設定値
3. API 実行 - 実行コマンド
4. ログ出力確認

確認

- 仮想マシン稼働確認 TPS
 - `queue_check_frequency` 設定確認
 - API 実行確認
 - ログ出力確認
-

環境構築

インストール

前提

- 64bit OS
- 64bit CPU

インストール

```
# Elixir をインストール
/opt/omnimessage-smpp/bin/omnimessage-smpp eval "File.read!
('config/runtime.exs')"
```

確認

- Elixir のバージョンを確認
- " vs ' の違いを確認
- Elixir のインストールを確認
- Elixir のインストールを確認 `import Config`

確認

- Elixir のバージョンを確認
- Elixir のインストールを確認
- Elixir のインストールを確認

開発環境

前提

- Elixir のバージョンを確認
- Elixir のインストールを確認

確認

```
# 重启 omnimessage-smpp
sudo systemctl restart omnimessage-smpp

# 查看状态
sudo systemctl status omnimessage-smpp

# 查看日志
sudo journalctl -u omnimessage-smpp -n 50
```

检查

检查磁盘

检查

1. 检查系统资源

```
# 磁盘
df -h

# 内存
free -h

# CPU 使用率
uptime
```

2. 检查服务状态

```
sudo systemctl status omnimessage-smpp
```

3. 查看日志

```
sudo journalctl -u omnimessage-smpp -n 200
```

4. 確認

```
sudo systemctl restart omnimessage-smpp
```

5. 確認

- 確認
- SSL 確認
- 確認
- 確認

6. 確認

```
# 確認
sudo cp /opt/omnimessage-smpp/config/runtime.exs.backup \
/opt/omnimessage-smpp/config/runtime.exs

# 確認
sudo systemctl restart omnimessage-smpp
```

7. 確認

確認

確認

確認

1. 確認 `cat /opt/omnimessage-smpp/VERSION`

2. 確認

```
sudo journalctl -u omnimessage-smpp -n 200 > /tmp/smpp-logs.txt
```

3. 確認

```
sudo cp /opt/omnimessage-smpp/config/runtime.exs  
/tmp/config.exs  
# [] /tmp/config.exs []
```

4. []

```
curl http://localhost:4000/metrics > /tmp/metrics.txt
```

5. []

```
uname -a > /tmp/system-info.txt  
free -h >> /tmp/system-info.txt  
df -h >> /tmp/system-info.txt
```

[]

- [] support@omnitouch.com
 - [] +61 XXXX XXXX (24/7)
 - [] []
-

[]

- **USAGE.md** - []
 - **CONFIGURATION.md** - []
 - **MONITORING.md** - []
 - **README.md** - []
-



XXXXXX

XXXXXX OmniMessage Core

XXXX OmniMessage SMPP XXXXXXXX OmniMessage Core XXXXXXXXXXXXXXXXXXXX
OmniMessage XXX - XXXXXXXXXXXXXXXX

XX OmniMessage XXXXX

- XX XXXXXXXX
- XX XXXXXXXXXXXX
- XX XXXXXXXX
- XX XXXXXXXXXXXX

XX **OmniMessage** XXXXX

```
# XX API XX
curl -k https://omnimessage-
core.example.com:8443/api/system/health

# XXXXXXXX API URL
grep api_base_url /opt/omnimessage-smpp/config/runtime.exs
```



XXXXXX

XXXXXXXXXXXXXXXXXX

1. XX **Web** XXX

- URL: `https://your-server:8087`

- 配置数据库连接

2. 配置

- 配置SMPP → 配置
- 配置“”
- 配置

3. 配置

- 配置
- 配置
- 配置

4. 配置

- 配置
- 配置
- 配置

5. 配置 Prometheus

- `curl http://localhost:4000/metrics`
- 配置 Grafana
- 配置

配置

配置参数

- 配置 > 2 个配置
- 配置 > 5%
- 配置
- 配置

配置 **MONITORING.md** 配置

配置

配置 OmniMessage Core 的 SMPP 配置

- **dest_smsc** — 配置 OmniMessage 配置 **dest_smsc:**
"vodafone_uk" 配置 **vodafone_uk** 配置 SMPP **submit_sm** 配置
- **source_smsc** — 配置 OmniMessage 配置 **source_smsc:**
"partner_acme" 配置 SMPP **deliver_sm** 配置 **partner_acme** 配置

配置 **submit_sm** PDU 配置 ESME 配置 **deliver_sm** PDU
配置 ESME 配置 SMSC 配置

配置

配置 OmniMessage Core 的 SMPP 配置

- 配置 **smsc_name** 配置 "smpp_gateway" 配置 **SMSC_NAME** 配置
- 配置 60 配置
- 配置 90 配置
- 配置 **dest_smsc** 配置
 - 配置 **queue** 配置 **queue** 配置

- 設定ファイルの作成

インストール時に OmniMessage Core のインストールが完了します。

インストールが完了したら、以下の手順に従って設定を行います。

1. ファイル“frontend_register”を
2. `smc_name` を OmniMessage の名前
3. として OmniMessage Core の `api_base_url` を

SMPP ? ? ?

SMPP のインストール

SMPP のインストールは、以下の手順で行います。

1. Web UI のインストール

- インストールディレクトリの作成
- SMPP → インストール / インストールディレクトリ
- インストールディレクトリの作成
- インストールディレクトリ Mnesia のインストール
- インストールディレクトリのインストール

2. インストール

- インストールディレクトリの作成
- `/opt/omnmessage-smpp/config/runtime.exs`
- Elixir のインストール
- インストールディレクトリのインストール
- インストールディレクトリ
- インストールディレクトリのインストール

Web UI のインストールが完了したら、

CONFIGURATION.md のインストール

環境構築

環境構築には **SMSC** と **ESME** が必要

環境構築

- SMPP プロトコル/IP
- ポート番号 2775
- ユーザー ID
- パスワード
- 送信機名
- TPS 値

環境構築

Web UI

環境構築

環境構築

1. 環境構築

- Web UI `https://your-server:8087`
- SMPP → 送信機

2. 送信機

- “送信機”
- 送信機
 - `vodafone_uk`
 - `smpp.vodafone.co.uk`
 - `2775`
 - **ID** `your_username`
 - `your_password`
 - 送信機名 `Transceiver`
 - **TPS** 値 `100`
 - 送信機名 `1000`

- 00“00”

3. 00000000

- 0000000000
- 0000SMPP → 0000
- 0000 10-30 00000“000”0000
- 000000000000000000

4. 0000000

- 0000000000
- 00000000 dest_smsc 00000000
- 000000000000
- 00000000

00 **B**00000000

000000000000000000

000

1. 00000000

```
sudo nano /opt/omnimessage-smpp/config/runtime.exs
```

2. 設定ファイルの編集

```
config :omnimessage_smpp, :binds, [  
  # 設定...  
  
  # 設定  
  %{  
    name: "vodafone_uk",  
    mode: :client,  
    bind_type: :transceiver,  
    host: "smpp.vodafone.co.uk",  
    port: 2775,  
    system_id: "your_username",  
    password: "your_password",  
    tps_limit: 100,  
    queue_check_frequency: 1000,  
  
    # 送信元IP/ポートを指定する  
    bind_source_ip: "10.20.30.40",  
    bind_source_port: 5000,  
  
    # 送信キューを指定する  
    queue: "vodafone_uk"  
  }  
]
```

設定ファイル `CONFIGURATION.md` の `bind_source_ip` と `bind_source_port` を `queue` で

3. サービスの再起動

```
# サービスを nano で Ctrl+X, Y, Enter  
  
# サービスを再起動  
sudo systemctl restart omnimessage-smpp
```

4. 確認

- サービスのステータスを確認
- ログを確認

- 設置“SMS”項目
- 設置發送號碼

5. 設定

- 設定項目 
- 設定項目 `dest_smsc` 設定項目
- 設定項目
- 設定項目

設定項目

設定項目 **ESME** 設定項目 **SMSC**

設定

1. 設定

- 設定項目 ID `partner_name`
- 設定項目
- 設定項目

2. 設定項目

- 設定項目 IP 設定
- 設定項目 TPS 設定
- 設定項目

設定項目

設定 **A** **Web UI**

設定項目

設定

1. 設定項目

- 設定 Web UI `https://your-server:8087`

- 配置SMPP → 配置

2. 配置

- “”
- 配置
 - 配置partner_acme
 - 配置ID acme_corp
 - 配置secure_password_123
 - 配置
 - IP 配置203.0.113.0/24
 - TPS 配置50
 - 配置1000
- “”

3. 配置

- 配置
- 配置

4. 配置

- 配置IP
- 配置2775
- 配置ID acme_corp

- `secure_password_123`
- `secure_password_123`

5. `secure_password_123`

- `secure_password_123` → `secure_password_123`
- `secure_password_123`
- `secure_password_123`
- `secure_password_123` IP `secure_password_123`

`secure_password_123` **B** `secure_password_123`

`secure_password_123` `secure_password_123`

`secure_password_123`

1. `secure_password_123`

```
sudo nano /opt/omnimessage-smpp/config/runtime.exs
```

2. `secure_password_123`

```
# 配置 server_binds 配置
config :omnimessage_smpp, :server_binds, [
  # 配置...

  # 配置
  %{
    name: "partner_acme",
    system_id: "acme_corp",
    password: "secure_password_123",
    allowed_bind_types: [:transmitter, :receiver,
:transceiver],
    ip_whitelist: ["203.0.113.0/24"],
    tps_limit: 50,
    queue_check_frequency: 1000
  }
]

# 配置监听
config :omnimessage_smpp, :listen, %{
  host: "0.0.0.0",
  port: 2775,
  max_connections: 100
}
```

3. 配置监听

```
sudo systemctl restart omnimessage-smpp
```

4. 配置配置

- 配置 IP 配置
- 配置 2775
- 配置 ID 配置 acme_corp
- 配置 secure_password_123
- 配置配置

5. 配置配置

- 配置 SMPP → 配置

- 設定する
- 設定を確認する
- 設定 IP 設定を確認する

設定を確認する

設定を確認するTPS 設定を確認IP 設定を確認

設定を確認する

設定 **A** 設定 **Web UI** 設定

設定を確認する

設定

1. 設定を確認する

- 設定 Web UI <https://your-server:8087>
- 設定を確認するSMPP → 設定を確認する
- 設定を確認するSMPP → 設定を確認する

2. 設定を確認する

- 設定を確認する
- 設定“設定”設定
- 設定を確認する
 - 設定を確認するTPS 設定を確認IP 設定を確認/設定
- 設定“設定”

3. 設定を確認する

- 設定を確認する
- 設定を確認する
- 設定  SMPP → 設定を確認する

4. 設定を確認する

- 設定を確認する

- 設定ファイルの作成
- サービスの起動

1. **インストール**

インストールの準備

1.1

1. インストール

```
sudo nano /opt/omnimessage-smpp/config/runtime.exs
```

2. 設定

- `:binds` と `:server_binds` の設定
- 接続先
- 接続先

```
%{
  name: "vodafone_uk",
  # ... 接続先
  tps_limit: 150, # 100 以下
  password: "new_password" # 接続先
}
```

3. サービスの再起動

```
sudo systemctl restart omnimessage-smpp
```

4. 確認

- サービスの起動状況を確認
- 接続先を確認
- 接続先を確認
- 接続先を確認

環境構築

環境構築 SMPP 側

準備

1. 環境構築

- 環境構築/インストール
- 環境構築

2. Web UI 環境構築

- 環境構築SMPP → 環境構築
- 環境構築
- Web “環境構築”
- 環境構築❓❓❓

3. 環境構築

- 環境構築SMPP → 環境構築/環境構築
- 環境構築
- Web “環境構築”
- 環境構築

4. 環境構築

- 環境構築 - 環境構築
- 環境構築環境構築

環境構築環境構築

環境構築環境構築環境構築環境構築

環境構築 **enabled** 環境構築環境構築環境構築環境構築環境構築環境構築環境構築環境構築環境構築環境構築

Web UI

1. 環境構築SMPP → 環境構築環境構築環境構築

2. 配置参数
3. 配置“链路”
4. 配置“链路”参数
5. 配置“链路”

配置参数

配置参数

- 配置参数
- 配置参数
- 配置参数

配置参数

配置参数

配置参数

- 配置参数 30
- 配置参数 500
- 配置参数

配置参数

配置参数 SMPP enquire_link PDU 配置参数

- 配置参数 60 enquire_link_interval 配置参数
- 配置参数 enquire_link_interval: 0 配置参数
- 配置参数 enquire_link 配置参数

配置 Prometheus 配置 smpp_enquire_link_sent_total

配置 smpp_enquire_link_received_total 配置参数

TPS 限制

限制 TPS 的方法有 `tps_limit`

- 限制 1 秒内请求数
- 限制 TPS 的峰值
- 限制 TPS 的均值 100 毫秒内请求数
- 限制 TPS 的方差

限制 TPS 的方法有

1. `tps_limit` 限制 TPS 的峰值
 2. `queue_check_frequency` 限制 TPS 的均值
 3. 限制 TPS 的方差
-

限制 TPS

限制 TPS

限制 TPS

限制

1. 限制 TPS
 - 限制 TPS 的峰值
 - 限制 TPS 的均值

2. 發送簡訊

- 發送
- 接收
 - 發送
 - 接收
 - 簡訊 SMSC (dest_smsc)
 - 發送
 - 接收

3. 接收簡訊

- 接收簡訊
- 接收簡訊 SMSC 接收

發送簡訊

接收簡訊

發送

1. 接收簡訊

- 接收簡訊 SMPP → 接收
- 接收簡訊接收
- 接收簡訊 接收

2. 消息的发送

- 消息的发送
- 消息的接收
- 消息的 `dest_smsc` 消息的接收
- 消息的 `deliver_after` 消息的接收

3. 消息的接收

- 消息 = 消息
- 消息的接收
- 消息的接收

4. 消息的接收

- 消息的接收
- 消息的接收
- 消息的接收

消息的接收

消息的接收

消息的接收“消息”消息

消息

1. 消息的接收

```
ping -c 3 carrier-smpp-server.com
telnet carrier-smpp-server.com 2775
```

2. 消息的接收

- 消息的接收
- 消息的接收

- 0000000000000000

3. 000000

- 0000SMPP → 000/0000000
- 0000 ID 00000000
- 00000000000000

4. 00000000

- 0000SMPP → 0000
- 00000000
- 00“0000”00
- 00 10-30 0
- 0000000000“0000”

5. 0000000000

- 00000000
- 0000000000000000
- 00000000
- 000 [TROUBLESHOOTING.md](#)

0000000000

00000000000000

📌

- 📄/📁
- IP 📄
- 📄/📁

📌

1. 📄

- 📄SMPP → 📄
- 📄 ID 📄
- 📄

2. 📄 IP 📄

- 📄 IP 📄
- 📄 IP 📄

3. 📄

- 📄
- 📄
- 📄

4. 📄

- 📄 Web UI 📄
- 📄“📄”📄

📄

📄 Prometheus 📄

📄

```
curl http://localhost:4000/metrics | grep smpp_connection_status
```

🔍 🔍 🔍 🔍

```
smpp_connection_status{bind_name="vodafone_uk",...} 1  
smpp_connection_status{bind_name="att_us",...} 1
```

🔍 🔍 🔍 🔍 1 🔍 🔍 🔍 🔍

🔍 🔍 🔍 🔍

🔍 🔍 🔍 🔍 🔍 🔍

1. 🔍 Web UI → SMPP → 🔍 🔍
2. 🔍 🔍 🔍 🔍 🔍 🔍
3. 🔍 🔍 🔍 🔍 🔍 🔍
4. 🔍 🔍 🔍 🔍 🔍 🔍 🔍 🔍 🔍 🔍
5. 🔍 🔍 [TROUBLESHOOTING.md](#)

🔍 🔍 🔍 🔍 🔍

1. 🔍 🔍 🔍 🔍 🔍 🔍 🔍 🔍
2. 🔍 🔍 🔍 🔍 🔍 🔍
3. 🔍 🔍 🔍 🔍 🔍 🔍 🔍 🔍 🔍 🔍
4. 🔍 🔍 🔍 🔍 🔍 🔍

🔍 🔍 🔍 🔍 🔍

1. 🔍 🔍 🔍 🔍 🔍 🔍 🔍 🔍
 2. 🔍 🔍 `dest_smsc` 🔍 🔍 🔍 🔍
 3. 🔍 🔍 TPS 🔍 🔍 🔍 🔍 🔍 🔍
 4. 🔍 🔍 `queue_check_frequency` 🔍 🔍
-

目錄

目錄

目錄

1. 目錄

- 目錄
- 目錄
- 目錄

2. 目錄

- 目錄
- 目錄
- 目錄

3. 目錄

- 目錄 SMPP 目錄
- 目錄
- 目錄 IP 目錄

4. 目錄

- 目錄
- 目錄 TPS 目錄
- 目錄

目錄

目錄

- 目錄
- 目錄
- 目錄

000

```
# 000000
sudo systemctl status omnimessage-smpp

# 0000
sudo systemctl restart omnimessage-smpp

# 0000
sudo systemctl status omnimessage-smpp

# 0000
sudo journalctl -u omnimessage-smpp -n 50
```

00 **Web UI** 000

1. 0000000000 30-60 000000
2. 0000SMPP → 0000
3. 00000000001-2 000
4. 0000000000

00000

00000000000000

```
# 0000
sudo cp /opt/omnimessage-smpp/config/runtime.exs \
  /opt/omnimessage-smpp/config/runtime.exs.backup.$(date +%Y%m%d)

# 0000
sudo tar -czf /tmp/smpp-certs-$(date +%Y%m%d).tar.gz \
  /opt/omnimessage-smpp/priv/cert/
```

00000000

```
# 確認
sudo cp /opt/omnimessage-smpp/config/runtime.exs.backup.YYYMMDD \
/opt/omnimessage-smpp/config/runtime.exs

# 再起動
sudo systemctl restart omnimessage-smpp
```

確認

確認インストール

確認

1. サービスの有無

```
sudo systemctl status omnimessage-smpp
```

2. サービスの起動

```
sudo systemctl start omnimessage-smpp
```

3. サービスのログ

```
sudo journalctl -u omnimessage-smpp -n 100
```

4. システムの状態

- サービスの有無
- SSL のインストール
- ディスクの使用状況 `df -h`
- メモリ使用状況 `free -h`

5. サービスの再起動

TPS 介紹

TPS

1. 什麼是 TPS

- 什麼是 SMPP → 通訊
- 什麼是 TPS
- 什麼是“TPS”

2. 什麼是 TPS

- 什麼是 TPS
- 什麼是 TPS
- 什麼是 TPS

3. 什麼是 TPS

- 什麼是 TPS
- 什麼是 TPS
- 什麼是 TPS

4. 什麼是 TPS

- 什麼是 TPS
- 什麼是 TPS
- 什麼是 TPS

TPS 介紹

TPS 介紹

TPS

1. 什麼是 TPS

- 什麼是 SMPP → 通訊
- 什麼是 TPS
- 什麼是 TPS TPS

2. 認證與授權

- 認證與授權
- 認證與授權

3. 認證與授權

- 認證與授權
- 認證與授權

4. 認證與授權

- 認證與授權
- 認證與授權

認證與授權

認證與授權

- ☐ 認證與 SMPP 認證與授權
- ☐ 認證與授權
- ☐ 認證與授權
- ☐ 認證 Prometheus/Grafana 認證
- ☐ 認證 > 98%

認證與授權

- ☐ 認證與授權
- ☐ 認證與授權
- ☐ 認證與授權
- ☐ 認證與授權
- ☐ 認證與授權

📁 目录

- ☐ 简介
 - ☐ 快速入门
 - ☐ 部署指南
 - ☐ 配置 TPS 参数
 - ☐ 常见问题
-

📄 文件

- **CONFIGURATION.md** - 配置文件
 - **SOURCE_ADDRESS_WHITELIST.md** - 源地址白名单
 - **MONITORING.md** - Prometheus 监控
 - **TROUBLESHOOTING.md** - 故障排除
 - **README.md** - 本文件
-

