



□□□□□

## 1. □□□□□□□□□□ /24□

□□□ □□□□□□□□□□□□

□□□

- OmniMME□□□□□□□□□□
- OmniSGW□□□□□□□□
- OmniPGW-C□PDN □□□□□□□□
- OmniUPF/PGW-U□□□□□□□□ / PDN □□□□□□□□

□□□ 10.179.1.0/24

```
mme:  
  hosts:  
    omni-site-mme01:  
      ansible_host: 10.179.1.15  
      gateway: 10.179.1.1  
      host_vm_network: "vmbr1"
```

---

## 2. □□□□□□□□□□ /24□

□□□ Diameter □□□□□□□□□□□□□□

□□□

- OmniHSS□□□□□□□□□□
- OmniCharge OCS□□□□□□□□□□
- OminiHSS PCRF□□□□□□□□□□□□
- OmniDRA DRA□Diameter □□□□□□
- DNS □□□
- TAP3/CDR □□□
- □□/□□□

- SIP
- 
- RAN
- Omnitouch CBC -
- APT -

10.179.2.0/24

```
hss:
  hosts:
    omni-site-hss01:
      ansible_host: 10.179.2.140
      gateway: 10.179.2.1
      host_vm_network: "vmbr2"
```

### 3. IMS /24

IMS SIP

- OmniCSCF S-CSCF
- OmniCSCF I-CSCF
- OmniTAS /
- OmniMessage SMPP IMS
- OmniSS7 STP SS7
- OmniSS7 HLR - 2G/3G
- OmniSS7 IP-SM-GW MAP SMS
- OmniSS7 CAMEL

10.179.3.0/24

```
scscf:
  hosts:
    omni-site-scscf01:
      ansible_host: 10.179.3.45
      gateway: 10.179.3.1
      host_vm_network: "vmbr3"
```

---

## 4. UE 10.179.4.0/24

UE 10.179.4.0/24 IMS & DNS

UE

- OmniCSCF P-CSCF
- XCAP
- DNS
- DNS

UE 10.179.4.0/24

```
pcscf:
  hosts:
    omni-site-pcscf01:
      ansible_host: 10.179.4.165
      gateway: 10.179.4.1
      host_vm_network: "vmbr4"
```

---

UE

OmniCore 10.179.4.0/24

UE 10.179.4.0/24

UE NIC 10.179.4.0/24



```

# vbr12 VLAN
applicationserver:
  hosts:
    ons-lab08sbc01:
      ansible_host: 10.178.2.213
      gateway: 10.178.2.1
      host_vm_network: "ovsbr1"
      vlanid: "402"

dra:
  hosts:
    ons-lab08dra01:
      ansible_host: 10.178.2.211
      gateway: 10.178.2.1
      host_vm_network: "ovsbr1"
      vlanid: "402"

dns:
  hosts:
    ons-lab08dns01:
      ansible_host: 10.178.2.178
      gateway: 10.178.2.1
      host_vm_network: "ovsbr1"
      vlanid: "402"

```

•••••

- vbr12 VLAN
- vbr12 VLAN
- vbr12 VLAN

•• **VLAN** •••

```

VLAN 10: 10.x.1.0/24 (•••••)
VLAN 20: 10.x.2.0/24 (••)
VLAN 30: 10.x.3.0/24 (IMS ••)
VLAN 40: 10.x.4.0/24 (UE ••)

```

# OmniCore IP

OmniCore IP

- **DRA** -
- **SGW/PGW** - GTP
- **ePDG** - WiFi UE IPsec
- **SMSC** - SMS SMPP
- **P-CSCF** - UE SIP

# IP

IP `ansible_host` IP

IP **SGW/PGW**

```

sgw:
  hosts:
    #   SGW
    opt-site-sgw01:
      ansible_host: 10.4.1.25
      gateway: 10.4.1.1
      host_vm_network: "v400-omni-packet-core"

    #   IP   SGW
    opt-site-roaming-sgw01:
      ansible_host: 203.0.113.10
      gateway: 203.0.113.9
      netmask: 255.255.255.248      # /29
      host_vm_network: "498-public-servers"
      in_pool: False
      cdrs_enabled: True

smf: # PGWs
  hosts:
    #   IP   PGW
    opt-site-roaming-pgw01:
      ansible_host: 203.0.113.20
      gateway: 203.0.113.17
      netmask: 255.255.255.240      # /28
      host_vm_network: "497-public-services-LTE"
      in_pool: False
      ip_pools:
        - '100.64.24.0/22'

```

## IP DRA

```

dra:
  hosts:
    opt-site-dra01:
      ansible_host: 198.51.100.50
      gateway: 198.51.100.49
      netmask: 255.255.255.240      # /28
      host_vm_network: "497-public-services-LTE"

```

## IP ePDG

```
epdg:
  hosts:
    opt-site-epdg01:
      ansible_host: 198.51.100.51
      gateway: 198.51.100.49
      netmask: 255.255.255.240      # /28
      host_vm_network: "497-public-services-LTE"
```

## IP

IP

- SGW GTP
- SGW
- PGW-C SGW

OmniCore - IP

---

# Omnitouch Ansible

Omnitouch Ansible  
4G/5G Ansible

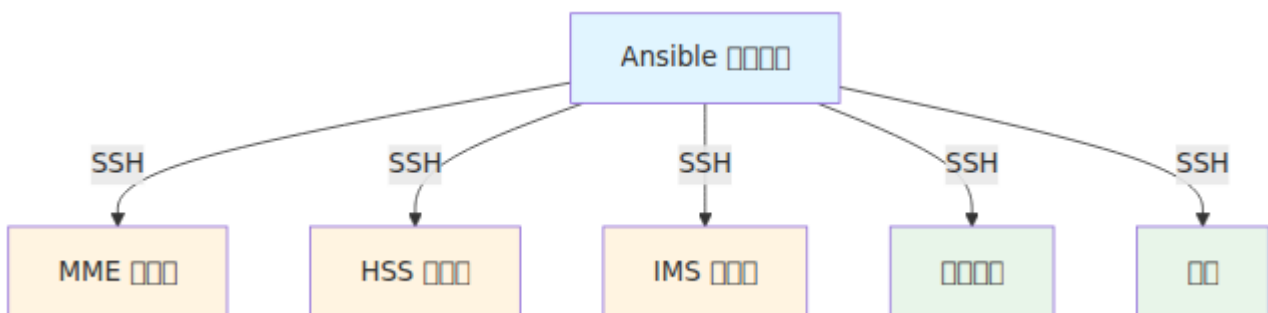
## Ansible

Ansible

- 
- 
- 
- 

Ansible - Ansible

## Omnitouch Ansible

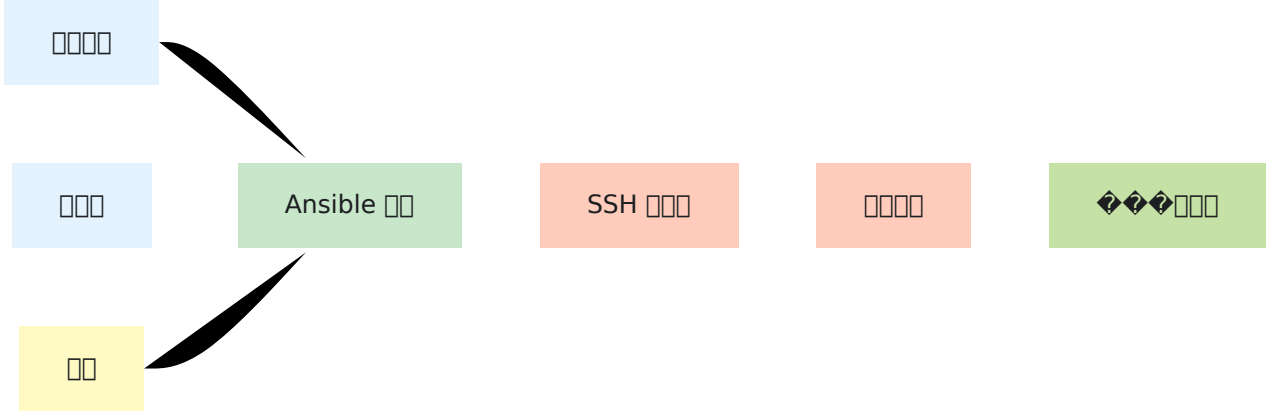


1.



□□□□□□□□

□□□□



□□□□

## 1. □□□□□□

□□□□□□□□□□□□□□□□

□□□□□□□ □□□□□□□□□□□□ IP □□□□□□□□□□□□□□□□ IP □□□□□□□□□□□□

**Proxmox** □□□ □□□ Proxmox □□□□□□□□ Proxmox VM/LXC □□ □□□□□□ VM/□□□□□□

□□□□□□□□□ □ □□□□

```
mme:
  hosts:
    customer-mme01:
      ansible_host: 10.10.1.15
      mme_code: 1
```

## 2. □□□□□□

□ group\_vars □□□□□□□□□□□□

```
plmn_id:
  mcc: '001'
  mnc: '01'
customer_name_short: customer
```

#0000 - 00000000000000000000

### 3. 0000

00000

```
ansible-playbook -i hosts/customer/host_files/production.yml
services/epc.yml
```

### 4. 00000

Ansible 00

- 00/000000000000 Proxmox/VMware 000
- 0000
- 0 APT 00000000
- 000000000
- 00000000000
- 0000
- 0000

00000000000

## OmniCore 4G/5G 00000000

- **OmniHSS** - 00000000
- **OmniSGW** - 0000000000
- **OmniPGW** - 0000000000
- **OmniUPF** - 0000000

- **OmniDRA** - Diameter □□□□
- **OmniTWAG** - □□ WLAN □□□□

□□□ <https://docs.omnitouch.com.au/docs/repos/OmniCore>

## OmniCall□□□□□□□□

- **OmniCall CSCF** - □□□□□□□□P-CSCF□I-CSCF□S-CSCF□
- **OmniTAS** - IMS □□□□□□VoLTE/VoNR □□□
- **OmniMessage** - □□□□□SMS-C□
- **OmniMessage SMPP** - SMPP □□□□
- **OmniSS7** - SS7 □□□□□STP□HLR□CAMEL□
- **VisualVoicemail** - □□□□□□

□□□ <https://docs.omnitouch.com.au/docs/repos/OmniCall>

## OmniCharge/OmniCRM

- **CRM** □□ - □□□□□□□□□□□□□□

□□□ <https://docs.omnitouch.com.au/docs/repos/OmniCharge>

## □□□□

- **DNS** - □□ DNS □□
- □□□□□□ - □□□□□□
- □□ - Prometheus□Grafana

□□□□□□□□□□

## □□□

□□□□□□□□□□□□

# 如何安装 APT

如何安装 Omnitouch 在 Debian 上 .deb 包

- 如何 CI/CD 部署
- 如何安装
- 如何更新

## APT 简介

简介

1. 什么是 APT - 包管理工具
2. 如何安装 - 安装 Omnitouch 包

如何安装 APT 包

## 如何安装

如何安装 Omnitouch 包

- 如何安装
- 如何更新/卸载
- 如何配置

如何安装

## 如何安装

如何安装

如何安装 Ansible 包

- 如何安装
- 如何配置
- 如何运行

- 環境

環境

環境構築

環境

環境構築 Git 環境

- 環境
- 環境
- 環境

環境構築

環境構築 `group_vars` 環境構築

環境構築

環境構築

環境

環境構築

環境構築 Ansible 環境構築 Python 環境構築

## 1. Python 環境構築

Ansible 環境構築 Python 環境構築

```
python3 -m venv .venv
```

## 2. 環境構築



- □□□□ - □□□□□□□□□□□□□□□□

# APT 配置

## 概要

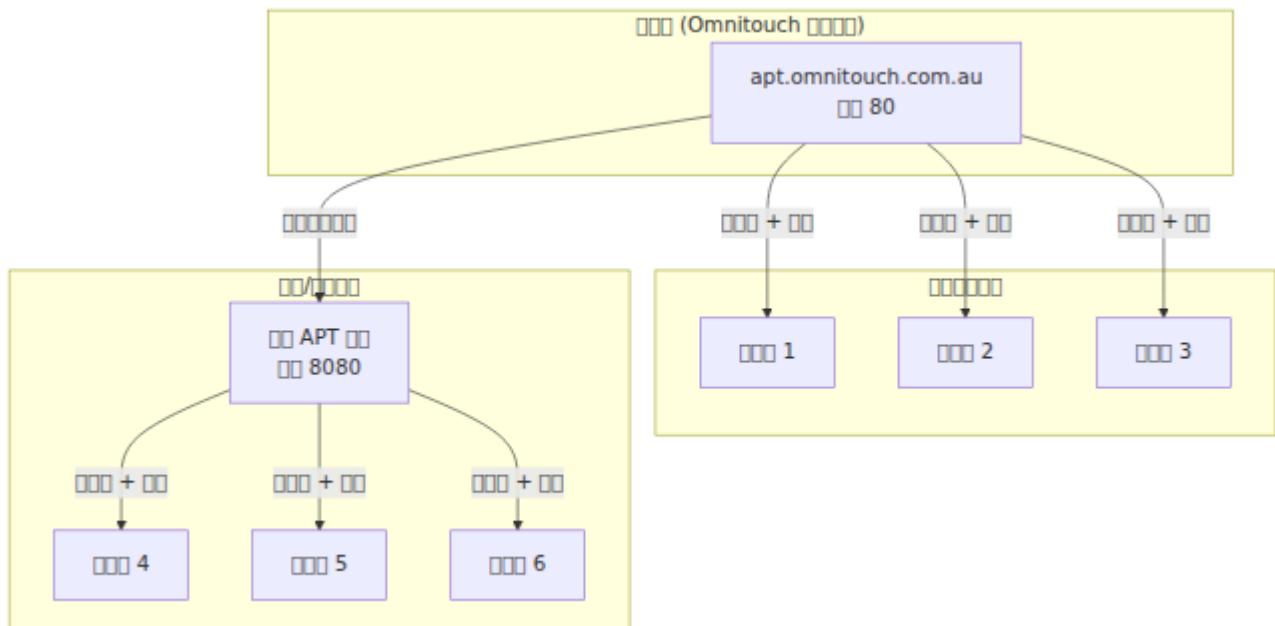
Omnitouch APT 環境構築

1. **APT** 環境 — `apt install` Debian 環境
2. 環境 — Prometheus 環境

環境構築

1. 環境 — `apt.omnitouch.com.au` 環境
2. 環境 — Omnitouch 環境/環境

## 環境



## 環境構築

APT 環境構築

パッケージ	ソース	パス
Omnitouch パッケージ	パッケージ .deb omnihss omnimme	/dists/<distro>/
Ubuntu パッケージ	Ubuntu パッケージ	/<distro>/pool/main/
GitHub パッケージ	Prometheus Grafana Homer	/releases/<org>/<repo>/
パッケージ	Web CGrateS_UI speedtest	/repos/
パッケージ	Galera FRR InfluxDB KeyDB	/releases/<vendor>/

## インストール

パッケージをインストールする

```

# apt_repo
# (APT リポジトリ)

remote_apt_*
# (パッケージ)

```

```

# /etc/apt/sources.list

remote_apt_*
# /releases/*

```

## 表 1

名前	説明	値
apt_repo	APT リポジトリのパス	/etc/apt/sources.list /etc/apt/sources.list.d/*.list
remote_apt_*	リモート APT リポジトリの URL	/releases/ Node Exporter Zabbix Nagios

## 表 2

名前	APT (apt_repo)	リモート (remote_apt_*)
use_apt_cache: true	apt_repo.apt_server	apt_repo.apt_server
use_apt_cache: false	apt_repo.*	remote_apt_*

use\_apt\_cache: false を指定すると、リモート APT リポジトリを使用します。

## 1. 設定

OmniTouch APT をインストールします。

### 設定

IP アドレスを IP アドレスで OmniTouch APT をインストールする場合は、OmniTouch をインストールします。

- IP アドレスを IP アドレスで HTTP をインストールする
- **FQDN** を APT リポジトリ

OmniTouch IP

IPv4	144.79.167.0/24
IPv4	160.22.43.0/24
IPv6	2001:df3:dec0::/48
ASN	AS152894

OmniTouch

APT	80	TCP	
APT	53	TCP/UDP	apt.omnitouch.com.au DNS
	123	UDP	NTP
	53	TCP/UDP	DNS

HTTP (TCP/80) NTP (UDP/123) DNS (TCP+UDP/53) OmniTouch IP

□□

```
all:
  vars:
    use_apt_cache: false

# APT □□□□□□
# □□ /etc/apt/sources.list □□ apt install □□
apt_repo:
  apt_server: "apt.omnitouch.com.au"
  apt_repo_username: "your-username"
  apt_repo_password: "your-password"

# □□□□□□
# □□□ /releases/ □□□□□□
remote_apt_server: "apt.omnitouch.com.au"
remote_apt_port: 80
remote_apt_protocol: "http"
remote_apt_user: "your-username"
remote_apt_password: "your-password"
```

□□

### APT □□□□ (apt\_repo)

□□	□□	□□	□□	□□
apt_repo.apt_server	□□□	□	-	APT □□□□□□□□ IP □□
apt_repo.apt_repo_username	□□□	□	-	APT □□ HTTP □□□□□□□□□□
apt_repo.apt_repo_password	□□□	□	-	APT □□ HTTP □□□□□□□□

### □□□□□ (remote\_apt\_\*)

名前	型	必須	デフォルト	説明
<code>remote_apt_server</code>	文字列	否	-	リモートリポジトリの IP
<code>remote_apt_port</code>	整数	否	80	リモートリポジトリのポート
<code>remote_apt_protocol</code>	文字列	否	http	リモートリポジトリのプロトコル (http または https)
<code>remote_apt_user</code>	文字列	否	-	リモートリポジトリの HTTP ユーザー名
<code>remote_apt_password</code>	文字列	否	-	リモートリポジトリの HTTP パスワード

名前

名前	型	必須	デフォルト	説明
<code>use_apt_cache</code>	ブール値	否	-	リポジトリをキャッシュするかどうか (false)

## URL の指定

**APT** の `/etc/apt/sources.list`

```
deb [trusted=yes] http://{apt_repo_username}:
{apt_repo_password}@{apt_server}/ noble main
```

Ansible の `get_url` タスク

```
http://{remote_apt_user}:
{remote_apt_password}@{remote_apt_server}:
{remote_apt_port}/releases/prometheus/node_exporter/node_exporter-
1.8.1.linux-amd64.tar.gz
```





項目	単位	初期値	コメント	説明
apt_repo.apt_server	文字列			APTリポジトリのサーバーIPアドレスを指定します。指定しない場合は、apt_cache_serversで指定されたサーバーを使用します。
apt_repo.apt_repo_username	文字列		-	APTリポジトリのユーザー名を指定します。
apt_repo.apt_repo_password	文字列		-	APTリポジトリのパスワードを指定します。

リモートリポジトリ (remote\_apt\_\*)

OmniTouchのインストールに必要です。

項目	単位	初期値	コメント	説明
remote_apt_server	文字列		-	OmniTouch APTのサーバーを指定します。
remote_apt_port	数値	80		OmniTouch APTのポート番号を指定します。
remote_apt_protocol	文字列	http		使用するプロトコルを指定します。
remote_apt_user	文字列		-	OmniTouchのユーザー名を指定します。
remote_apt_password	文字列		-	OmniTouchのパスワードを指定します。

注

名前	型	デフォルト	説明
<code>use_apt_cache</code>	boolean	true	apt_cache_servers が true の場合 true
<code>apt_cache_port</code>	integer	8080	

## URL 指定

APT ソースリスト `/etc/apt/sources.list`

```
deb [trusted=yes] http://192.168.1.100:8080/noble noble main
```

Ansible `get_url` タスク

```
http://192.168.1.100:8080/releases/prometheus/node_exporter/node_exporter-1.8.1.linux-amd64.tar.gz
```

APT ソースリストに `[trusted=yes]` を追加

## 手順

1. LXC コンテナに 50 GB を確保
2. コンテナを起動

```
ansible-playbook -i hosts/customer/production.yml
services/apt_cache.yml
```

3. `http://192.168.1.100:8080/` にアクセス

# 目次

インストール wget オmnitouch APT リポジトリ

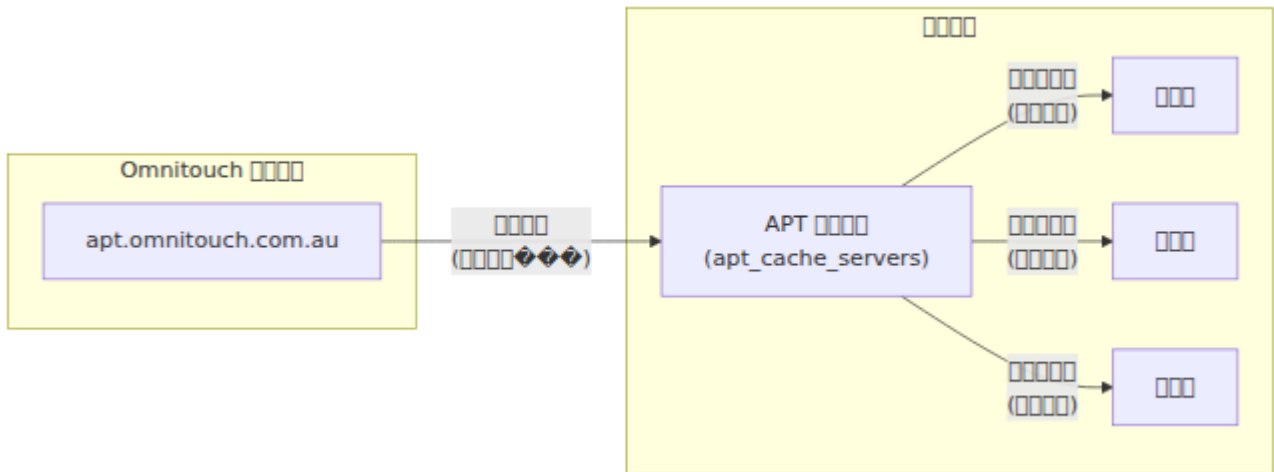


## 目次

パス	説明
<code>/dists/&lt;distro&gt;/</code>	APT リポジトリ
<code>/pool/main/</code>	Omnitouch .deb
<code>/&lt;distro&gt;/pool/main/</code>	Ubuntu
<code>/releases/</code>	GitHub Prometheus Grafana Zabbix
<code>/repos/</code>	Erlang Elixir CGrateS_UI

インストール

## Network Diagram



Use the following command to recursively fetch the APT repository from Omnitouch:

```
wget --recursive http://apt.omnitouch.com.au
```

## Configuration

Configure the `apt_cache_servers` group:

1. Set `use_apt_cache: true`
2. Set `ansible_host` to the IP of the `apt_repo.apt_server`

## Ansible Playbook

```
apt_cache_servers:
  hosts:
    apt-cache-01:
      ansible_host: 192.168.1.100
      gateway: 192.168.1.1
  vars:
    # Omnitouch repository
    remote_apt_server: "apt.omnitouch.com.au"
    remote_apt_user: "your-username"
    remote_apt_password: "your-password"
```

Run the playbook:

- `use_apt_cache: true`
- `apt_repo.apt_server: "192.168.1.100"`
- `http://192.168.1.100:8080/`
- `http://your-username:your-password@apt.omnitech.com.au/`

apt

apt

```
all:
  vars:
    use_apt_cache: false #
  apt_repo:
    apt_server: "apt.omnitech.com.au"
    apt_repo_username: "user"
    apt_repo_password: "pass"
  remote_apt_server: "apt.omnitech.com.au"
  remote_apt_user: "user"
  remote_apt_password: "pass"
```

apt

## 1 APT

APT

```
all:
  vars:
    use_apt_cache: false

    # APT 設定 - 設定
    apt_repo:
      apt_server: "apt.omnitech.com.au"
      apt_repo_username: "user"
      apt_repo_password: "pass"

    # 設定 - 設定
    remote_apt_server: "apt.omnitech.com.au"
    remote_apt_port: 80
    remote_apt_protocol: "http"
    remote_apt_user: "user"
    remote_apt_password: "pass"
```

```
deb [trusted=yes] http://user:pass@apt.omnitech.com.au/
noble main
```

## 2 hosts 設定 APT 設定

Ansible 設定

```
apt_cache_servers:
  hosts:
    cache-server:
      ansible_host: 192.168.1.100
      gateway: 192.168.1.1
  vars:
    # 設定
    remote_apt_server: "apt.omnitech.com.au"
    remote_apt_port: 80
    remote_apt_protocol: "http"
    remote_apt_user: "user"
    remote_apt_password: "pass"

# all: vars: 設定
# 設定 apt_cache_servers 設定
```



```
# APT
apt_cache_servers:
  hosts:
    customer-apt-cache:
      ansible_host: 10.179.1.114
      gateway: 10.179.1.1
      host_vm_network: "vibr0"
      num_cpus: 4
      memory_mb: 16384
      proxmoxLxcDiskSizeGb: 120
  vars:
    #
    remote_apt_server: "apt.omnitouch.com.au"
    remote_apt_port: 80
    remote_apt_protocol: "http"
    remote_apt_user: "customer-username"
    remote_apt_password: "customer-secure-token"

#
hss:
  hosts:
    customer-hss01:
      ansible_host: 10.179.2.140
      gateway: 10.179.2.1

mme:
  hosts:
    customer-mme01:
      ansible_host: 10.179.1.15
      gateway: 10.179.1.1

dns:
  hosts:
    customer-dns01:
      ansible_host: 10.179.2.177
      gateway: 10.179.2.1

#
all:
  vars:
    #
```

```
# - use_apt_cache: true apt_cache_servers
# - apt_repo.apt_server: "10.179.1.114"
```

10.179.1.114

### 1. 10.179.1.114

- vars: remote\_apt\_\*
- http://customer-username:customer-secure-token@apt.omnitouch.com.au:80/
- nginx 8080

### 2. (customer-hss01 customer-mme01 customer-dns01)

- apt\_cache\_servers
- use\_apt\_cache: true
- apt\_repo.apt\_server: "10.179.1.114"
- deb [trusted=yes] http://10.179.1.114:8080/noble noble main
- 

10.179.1.114

10.179.1.114

```
ansible-playbook -i hosts/customer/production.yml
services/apt_cache.yml
```

10.179.1.114 Omnitouch APT

- Omnitouch
- Ubuntu
- GitHub
- 

```
wget --timestamping
```

OmniTouch APT (apt.omnitouch.com.au) APT services/apt.yml services/apt\_cache.yml

APT

## APT 401

APT

```
http://10.179.1.115:80/noble/dists/noble/main/binary-  
amd64/Packages 401
```

APT

- apt\_repo all: vars: apt\_cache\_servers: vars:
- 
- apt\_repo\_username apt\_repo\_password
- IP OmniTouch APT
- 

APT

- apt\_repo apt\_cache\_servers: vars: all: vars:
- 8080 80
- /etc/apt/sources.list.d/omnitouch.list
  - deb [trusted=yes] http://10.179.1.114:8080/noble noble main
  - deb [trusted=yes] http://user:pass@10.179.1.115:80/noble noble main
- 
- IP OmniTouch

# Node Exporter Zabbix

Ansible `/releases/`

- `remote_apt_*`
- `remote_apt_user` `remote_apt_password`
- `use_apt_cache: false` `remote_apt_server`

1. `remote_apt_*`
2. Omnitouch
3. `remote_apt_server`

- 
- `remote_apt_*`
- Omnitouch

1. 80 `apt.omnitouch.com.au`
2. `remote_apt_*`
- 3.

- —
- —

- **VMware** — **ESXi**
- **Proxmox VE** — **OpenStack LXC**





名前	名前	名前
mme	mme	omnimme, open5gs
sgw	sgw	omnisgwc, open5gs
pgw	pgw	omnipgwc, open5gs
upf	upf	omniupf, frf
hss	hss	omnihss, pyhss, mysql
dra	dra	omnidra, freediam
pcscf, scscf, icscf	cscf	kamailio, rtpengine
applicationserver	applicationserver	freeswitch, omnitas
ocs	ocs	cgrates, ocs, pcef, keydb
cgrates	cgrates	cgrates, mysql
omnimessage	omnimessage	omnimessage, kamailio
dns	dns	named, bind
omnicrm	omnicrm	omnicrm, nginx, mysql
monitoring	monitoring	prometheus, grafana, loki, nginx

名前

名前

項目	説明	値
job	ジョブ名	customer-mme01
hostname	ジョブが実行されるホスト名	customer-mme01
component	監視対象のコンポーネント	mme, cscf, ocs
unit	systemd のユニット	omnimme.service
level	ログレベル	info, warning, error
service	Syslog サービス	omnimme

##

###

監視対象のコンポーネント

1. 監視対象 `monitoring` を
2. 監視対象のコンポーネント

監視対象のコンポーネント

###

```
monitoring:
  hosts:
    customer-monitoring01:
      ansible_host: 10.10.2.200
      gateway: 10.10.2.1
```

監視対象 `monitoring` の監視

- Loki 配置
- Alloy 配置

## 配置

Loki 配置

配置项	值	说明
保留周期	7 天	保留 7 天的日志数据
存储空间	50 GB	保留 50 GB 的日志数据

配置示例

配置示例

```
all:
  vars:
    loki_retention_period: "168h" # 7 天
```

## Alloy 配置

Loki 配置 Alloy 配置

配置项	值	说明
WAL 大小	500 MB	写入 ahead log 的大小
WAL 保留时间	1 天	保留 1 天的 WAL 数据

配置示例

# Grafana □□□

□□□□□□□□□□□□□□□□□□□□

□□□□□□

□□□	□□	□□
CSCF □□	□□ → CSCF □□	P-CSCF, S-CSCF, I-CSCF (Kamailio) □□
MME □□	□□ → MME □□	MME □□/□□□□□□□□
SGW □□	□□ → SGW □□	SGW □□□□□□□□
PGW □□	□□ → PGW □□	PGW PDN □□□□□□
HSS □□	□□ → HSS □□	HSS Diameter □□□□□□□□
OmniMessage □□	□□ → OmniMessage □□	□□□□□ SMPP □□
OCS/CGrateS ◆◆◆◆□	□□ → OCS/CGrateS □ □	□□□□□ JSONRPC □□
CGrateS RPC □□	□□ → CGrateS RPC □ □	□□□□□ RPC □□/□□□□□□

□□□□□□

□□□□□□□□

- □□□ — □□□□□□□□□□□□□□□□
- □□□□□ — □□□□□□□□□□□□□□□□□□
- □□□□□ — □□□□□□□□□□P-CSCF, S-CSCF, I-CSCF□
- □□□□□ — □□□□□□□□□□□□

- 10 分钟 — 10 分钟

## 10 分钟

1. 访问 Grafana 地址 `http://<monitoring-host>:3000`
2. 登录 → 仪表盘 → 仪表盘
3. 创建仪表盘
4. 配置仪表盘

## 10 分钟

### LogQL 查询

Loki 使用 LogQL 查询日志

```
{label="value"} |= "search string"
```

## 10 分钟

查询日志

```
{hostname="customer-mme01"}
```

查询 MME 日志

```
{component="mme"}
```

查询 CSCFs 日志

```
{component="cscf"} |~ "(?i)error"
```

查询 systemd 日志

```
{unit="kamailio.service"}
```

□□□□□□

```
{component="hss", hostname="customer-hss01"} |= "ULR" |=  
"DIAMETER_SUCCESS"
```

□□□□□□□□

□□□□□□□□□□

```
sum by (component) (rate({job=~".+"} [5m]))
```

**CSCF** □□□□□

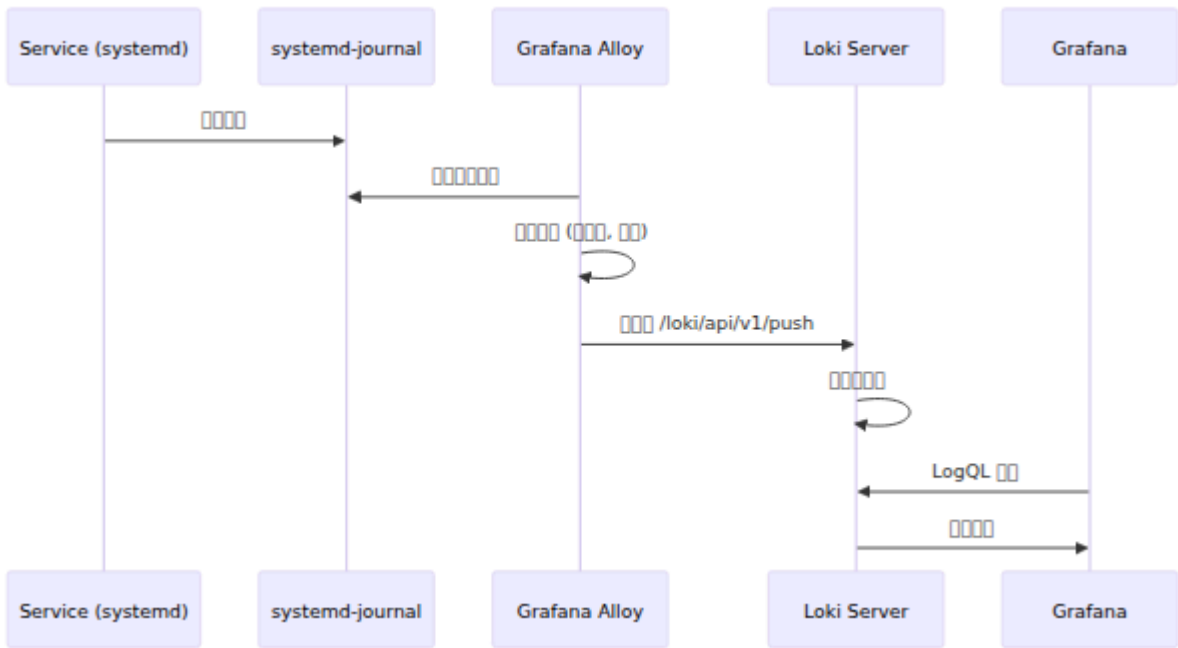
```
sum(rate({component="cscf"} |~ "(?i)error" [5m]))
```

---

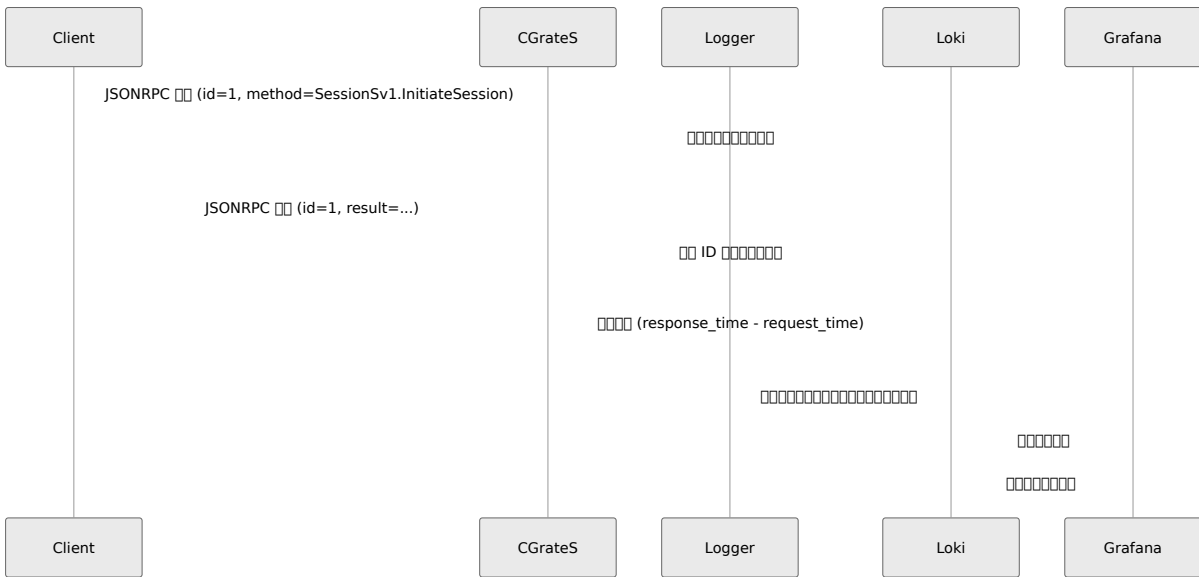
## **CGrateS JSONRPC** □□□□

□□ OCS/CGrateS □□□JSONRPC □□□□□□□□□□□□/□□□□□□□□□□□□□□□□□□□□□□□□

□□



□□/□□□□



□□□□

□□ CGrateS □□□□□□□□□□

Port	Service	Protocol	Notes
2012	rpc_json	TCP	JSONRPC over TCP
2080	http	HTTP	HTTP API endpoints: /metrics, /health

Configuration

- Prometheus scrape (GET /metrics)
- Health check (GET /health)
- Gzip compression

Log

JSONRPC log: /var/log/cgrates/jsonrpc.log

```
2026-03-01T10:30:45.123456 port=2012 service=rpc_json request_id=1
method=SessionSv1.InitiateSession latency_ms=2.45 status=ok error=
src=10.10.1.50:45678 dst=10.10.2.100:2012 request=
{"id":1,"method":"SessionSv1.InitiateSession",...} response=
{"id":1,"result":{...}}
```

□□□□

□□	□□	□□
timestamp	□□□□□□□□ ISO 8601 □□□	2026-03-01T10:30:45.123456
port	□□□□□□ CGrateS □□	2012, 2080
service	□□□□□□□□	rpc_json, http
request_id	JSONRPC □□ ID □□□□	1, 42, □□□□□□ null□
method	JSONRPC □□□□	SessionSv1.InitiateSession
latency_ms	□□□□□□□□□□	2.45
status	□□□□	ok, error
error	□□□□□□□□□□□□□□	INSUFFICIENT_CREDIT
src	□ IP:□□□□□□□□	10.10.1.50:45678
dst	□□ IP:□□□□CGrateS□	10.10.2.100:2012
request	□□□ JSONRPC □□□□□□□□ JSON□	{"id":1,"method":"..."}
response	□□□ JSONRPC □□□□□□□□ JSON□	{"id":1,"result":{"..."}}

## Grafana □□□

**OCS / CGrateS** □□ □□□□□□□□□□□□ JSONRPC □□□

□□□□ **JSONRPC** □□

□□□□ JSONRPC □□□□□□□□□□□□□□□□□□□□□□





□□□□□□

```
{job="cgrates-jsonrpc"} |= "method=SessionSv1.InitiateSession"
```

□□□□/□□□□□□

```
{job="cgrates-jsonrpc"} |= "Account\":"12345"
```

□□□□

```
{job="cgrates-jsonrpc"} |= "status=error"
```

□□□□

□□□□□□□□□□□□□□□□

```
max_over_time(  
  {job="cgrates-jsonrpc"}  
  |= "method="  
  | regexp "method=(?P<method>[^ ]+) latency_ms=(?P<latency>[0-  
9.]+)"  
  | __error__=""  
  | unwrap latency [ $__auto ]  
) by (method)
```

□□□□□□□□ > **100ms**□□

```
{job="cgrates-jsonrpc"}  
| regexp "latency_ms=(?P<latency>[0-9.]+)"  
| latency > 100
```

□□□□□□

**CDR** □□□

```
{job="cgrates-jsonrpc"} |= "method=CDRsV1"
```

□□□□

```
{job="cgrates-jsonrpc"} |~ "method=SessionSv1"
```

□□□□

```
{job="cgrates-jsonrpc"} |~ "method=ApierV"
```

□□□□

JSONRPC □□□□ systemd □□□ OCS/CGrateS □□□□□□

□□□□□□

```
systemctl status cgrates-jsonrpc-logger
```

□□□□□□

```
journalctl -u cgrates-jsonrpc-logger -f
```

□□□□

```
systemctl restart cgrates-jsonrpc-logger
```

□□□□

□□ **JSONRPC** □□□□

□□□JSONRPC □□□□□□□□□□

□□□□

- 確認する
- ngrep 確認
- 確認する
- Alloy 確認する

### 確認

1. 確認する

```
systemctl status cgrates-jsonrpc-logger  
journalctl -u cgrates-jsonrpc-logger -n 50
```

2. ngrep 確認

```
which ngrep
```

3. 確認する

```
tail -f /var/log/cgrates/jsonrpc.log
```

4. Alloy 確認する

```
journalctl -u alloy | grep jsonrpc
```

### 確認

確認する JSONRPC 確認する“確認”

### 確認

- 確認する JSONRPC 確認
- 確認する

### 確認

1. 確認する

```
{job="cgrates-jsonrpc"} |= "method="
```

2. latency\_ms

3.

- 
- 
- ID

1. 60
2. CGrateS
3. CGrateS

systemd Alloy

### FreeSWITCH

/var/log/freeswitch/*.log	freeswitch

## Nginx `OmniCRM`

File	Service	Log Type
<code>/var/log/nginx/access.log</code>	<code>nginx</code>	<code>access</code>
<code>/var/log/nginx/error.log</code>	<code>nginx</code>	<code>error</code>

## CGrates `JSONRPC` `OCS`

File	Service
<code>/var/log/cgrates/jsonrpc.log</code>	<code>cgrates-jsonrpc</code>

## Monitoring

### Monitoring `Grafana`

Monitoring `Loki` `Alloy`

Monitoring

- `Alloy` `Alloy`
- `Loki` `Loki`
- `Alloy` `Alloy`
- `Alloy` `Alloy` 3100 `Loki`

Monitoring

1. `Alloy` `Alloy`

```
systemctl status alloy
journalctl -u alloy -f
```

2. Loki status check

```
systemctl status loki
journalctl -u loki -f
```

3. Health check

```
curl http://<monitoring-ip>:3100/ready
```

4. Loki listening on TCP 3100

## Alloy Config

Alloy config directory: `/var/lib/alloy`

Alloy config

- Loki config
- WAL config

Alloy config

1. Loki config
2. Loki storage config 500 MB
3. Loki config
4. Loki config

## Loki Config

Loki config

Alloy config

- Loki 50 GB config
- Loki config

Alloy config

1. Loki config

```
du -sh /var/lib/loki/
```

2. `du -sh /var/lib/loki/`
3. `du -sh /var/lib/loki/`
4. `du -sh /var/lib/loki/`

component

infrastructure

Alloy

- Alloy

Alloy

1. Alloy
2. Ansible Alloy

```
ansible-playbook -i hosts/customer/hosts.yml services/all.yml -  
-limit <hostname>
```

3. Alloy

```
systemctl restart alloy
```

---

## □□□□

□□	□□	□□	□□
Loki	3100	HTTP	□□□□□□ API
Loki	9096	gRPC	□□ gRPC□□□□□□
Alloy	12345	HTTP	Alloy □□□ UI
Grafana	3000	HTTP	□□□□□

---

## □□□□

- □□□□□□ — Grafana, Prometheus, □□□□□□
- □□□□ — □□□□□□
- □□□□□□ — □□□□
- □□□□ — □□□□□□



## hosts

# hosts - hosts-file-configuration.md

## vars

```
cdrs_enabled: True           # CDR enabled
in_pool: False              # in pool
online_charging_enabled: False # OCS enabled
recording: True            # recording (AS)
populate_crm: False        # populate CRM
```

## vars (all:vars)

all:vars vars

## vars

vars

```
ansible_connection: ssh
ansible_user: root
ansible_password: password
ansible_become_password: password
```

vars SSH vars

```
ansible_ssh_private_key_file: '/path/to/key.pem'
```

vars

```
customer_name_short: omnitouch
customer_legal_name: "YKTN Lab"
site_name: YKTN
region: AU
TZ: Australia/Melbourne
```

## PLMN

```
plmn_id:
  mcc: '001'          # MNC (3 )
  mnc: '01'          # MNC (2-3 )
  mnc_longform: '001' # MNC (3 )

diameter_realm: epc.mnc{{ plmn_id.mnc_longform }}.mcc{{
plmn_id.mcc }}.3gppnetwork.org
```

Network configuration for Diameter

Network

```
network_name_short: Omni
network_name_long: Omnitouch
tac_list: [10100,100] # TAC (MME )
```

Network UE configuration > Network

## DNS

```
netplan_DNS: False # systemd-resolved netplan
DNS
manage_resolv_conf: True # False Ansible
/etc/resolv.conf
```

Network manage\_resolv\_conf False Ansible Network  
/etc/resolv.conf Network  
all:vars Network

## APT

apt\_cache\_servers

- use\_apt\_cache True False
- apt\_repo.apt\_server IP

```
# apt_cache_servers
use_apt_cache: True # APT

apt_repo:
  apt_server: "10.10.1.114" # APT
  # use_apt_cache: False
  # apt_repo_username: "omni"
  # apt_repo_password: "omni"

# (1) use_apt_cache: false /releases/
# (2) use_apt_cache: true Omnitouch
remote_apt_server: "apt.omnitouch.com.au"
remote_apt_user: "omni"
remote_apt_password: "omni"
```

APT

```
license_server_api_urls: ["https://10.10.2.150:8443/api"]
license_enforced: true
```

## MME

```
mme_dns: False # MME DNS
```

## SAEGW

mtu: 1400

# 000000

## IMS 00

ims\_dra\_support: False

# 00 DRA 00 IMS

enable\_homer: False

# 00 Homer SIP 00

## RAN 0000

```
use_nokia_monitor: True
use_casa_monitor: True
install_influxdb: True

influxdb_user: monitor
influxdb_password: "secure-password"
influxdb_organisation_name: omnitouch
influxdb_nokia_bucket_name: nokia-monitor
influxdb_casa_bucket_name: casa-monitor
influxdb_operator_token: "generated-token"
influxdb_url: http://127.0.0.1:8086

enable_pm_collection: False
enable_alarm_collection: False
enable_location_collection: False
enable_ran_status_collection: True
enable_nokia_rectifier_collection: False
collection_interval_in_seconds: 120

ran_monitor:
  sql:
    user: ran_monitor
    password: "secure-password"
    database_host: 127.0.0.1
    database_name: ran_monitor
  influxdb:
    address: 10.10.2.135
    port: 8086
  nokia:
    airscales:
      - address: 10.7.15.66
        name: site-Lab-Airscale
        port: 8080
        web_password: nemuuser
        web_username: Nemuadmin
```

□□□□□

```
firewall:
  allowed_ssh_subnets:
    - '10.0.1.0/24'
    - '10.0.0.0/24'
  allowed_ue_voice_subnets:
    - '10.0.1.0/24'
  allowed_carrier_voice_subnets:
    - '10.0.1.0/24'
  allowed_signaling_subnets:
    - '10.0.1.0/24'
```

## DNS

```
roaming_dns_servers:
  wildcard: ['10.0.99.1']
  # DNS (PLMN)
  123456: # 1
    - '10.10.2.197'
  654321: # 2
    - '10.10.0.4'
```

## SSH

```
local_users:
  usera:
    name: A
    public_key: "ssh-rsa AAAAB3Nza..."
  userb:
    name: B
    public_key: "ssh-ed25519 AAAAC3..."
```

---



```
proxmox_lxc_nameserver: "1.1.1.1" # [] LXCs

proxmoxServers:
  customer-prxmx01:
    deployment_type: lxc
    proxmoxServerAddress: 10.10.0.100
    proxmoxServerPort: 8006
    proxmoxApiTokenName: AnsibleToken
    proxmoxApiTokenSecret: "token-secret"
    proxmoxNodeName: pve01
    proxmoxLxcOsTemplate: "local:vztmpl/ubuntu-24.04-
standard_24.04-2_amd64.tar.zst"
    proxmoxLxcDefaultStorage: SSD_RAID0 # []rootfs []
```

[][][][][][]

```
dns:
  vars:
    proxmox_interface: vmbr0 # []
    gateway: 10.10.0.1 # []
    netmask: 255.255.255.0 # []
    vlanid: 100 # []
    proxmoxLxcCores: 2 # [] LXC
    proxmoxLxcMemoryMb: 4096 # [] LXC
    proxmoxLxcDiskSizeGb: 30 # [] LXC
    proxmoxLxcRootFsStorageName: SSD_RAID0 # [] LXC
    host_vm_network: vmbr1 # []
```

# VMware vCenter

```
vcenter_ip: "vcenter.example.com"
vcenter_username: "administrator@vsphere.local"
vcenter_password: "password"
vcenter_datacenter: "DC1"
vcenter_vm_template: ubuntu-24.04-model
vcenter_vm_disk_size: 50
vcenter_folder: "Omnicore"
host_vm_network: "Management"

vhosts:
  "10.0.0.23":
    vcenter_cluster_ip: 10.0.0.23
    vcenter_datastore: "datastore1 (3)"

netmask: 255.255.255.0
```

## □□□□

- IP □□□□ - □□□□□ IP □□□□
- □□□□□□ - □□□□□□□□
- □□□□□ - □□□□□□□□ group\_vars
- Netplan □□ - □□ IP □□ NIC □□
- □□□□ - □□□□□□
- APT □□□□ - □□□
- □□□□□□ - □□□□□□

## □□□□

□□□□□□□□□□□□□□□□

- **OmniCore** □□: <https://docs.omnitouch.com.au/docs/repos/OmniCore>
- **OmniCall** □□: <https://docs.omnitouch.com.au/docs/repos/OmniCall>

- **OmniCharge/OmniCRM:**

<https://docs.omnitouch.com.au/docs/repos/OmniCharge>



# 1. 設定ファイル

```
# 設定ファイル
mme:
  hosts:
    customer-mme01:
      ansible_host: 10.10.1.15

hss:
  hosts:
    customer-hss01:
      ansible_host: 10.10.2.140

# ... 設定
```

設定ファイル

# 2. group\_vars

group\_vars 設定ファイル

設定ファイルにOmniMessage SMS、TAS、SIP、Diameter

設定ファイル

# 3. APT

```
# APT
apt_repo:
  apt_server: "10.254.10.223" # IPアドレス
  use_apt_cache: false # true = false = repo
```

APT

## 4. 配置

```
# 配置
license_server_api_urls: ["https://10.10.2.150:8443/api"]
license_enforced: true
```

配置

## 5. 运行

```
ansible-playbook services/twag.yml --limit=myhost
ansible-playbook services/all.yml --limit=mme,sgw
```

```
# 运行
ansible-playbook -i hosts/customer/host_files/production.yml
services/all.yml

# 运行
ansible-playbook -i hosts/customer/host_files/production.yml
services/epc.yml
ansible-playbook -i hosts/customer/host_files/production.yml
services/ims.yml
```

## 部署

- Ansible - 配置
- 配置 - 配置
- 配置 - 配置
- IP - IP
- 配置 - 配置
- APT - 配置
- 配置 - 配置
- 配置 - Grafana Prometheus
- 配置 - Loki Alloy



□□□□□□□□□□□□□□

- **OmniCore** 4G/5G □□□□□□  
<https://docs.omnitouch.com.au/docs/repos/OmniCore>
  - OmniHSS, OmniSGW, OmniPGW, OmniUPF, OmniDRA, OmniTWAG
- **OmniCall** □□□□□□□□ <https://docs.omnitouch.com.au/docs/repos/OmniCall>
  - OmniTAS, OmniCall CSCF, OmniMessage, OmniSS7, VisualVoicemail
- **OmniCharge/OmniCRM** □□□□□□  
<https://docs.omnitouch.com.au/docs/repos/OmniCharge>
- □□□□□□ <https://docs.omnitouch.com.au/>





## 目次

```
hosts/Customer/  
├── group_vars/  
│   ├── gateways_prod/                # SIP 目次  
│   │   ├── gateway_carrier1.xml  
│   │   ├── gateway_carrier2.xml  
│   │   └── gateway_emergency.xml  
│   ├── gateways_lab/                 # 目次  
│   │   └── gateway_test.xml  
│   ├── dialplan/                    # 目次  
│   │   ├── mo_dialplan.xml          # 目次  
│   │   ├── mt_dialplan.xml          # 目次  
│   │   └── emergency.xml  
│   └── dialplan_lab/                 # 目次  
│       └── mo_dialplan.xml
```

## 目次

```
applicationserver:  
  hosts:  
    customer-tas01:  
      ansible_host: 10.10.3.60  
      gateway: 10.10.3.1  
      host_vm_network: "vubr3"  
      gateways_folder: "gateways_prod" # 目次  
      dialplan_folder: "dialplan"      # 目次 - 目次  
      "dialplan"
```

## 目次:

1. Ansible 目次 gateways\_folder: "gateways\_prod"
2. 目次 hosts/Customer/group\_vars/gateways\_prod/ 目次  
/etc/freeswitch/sip\_profiles/
3. 目次 hosts/Customer/group\_vars/dialplan/ 目次 dialplan\_folder 目次  
目次 OmniTAS 目次
4. 目次

## 目次: 目次

- `gateways_folder: "gateways_lab"`
- `gateways_folder: "gateways_prod"`
- `gateways_folder: "gateways_customer_specific"`
- `dialplan_folder: "dialplan_lab"`
- `dialplan_folder: "dialplan_prod"`

📄: 📄 <https://docs.omnitouch.com.au/docs/repos/OmniCall>

---

## 📄 3: 📄📄📄📄📄 (OmniHSS)

📄📄📄📄📄📄📄📄📄

📄📄📄

```
hosts/Customer/
├── group_vars/
│   └── hss_runtime.exs.j2      # 📄📄📄 HSS 📄📄
```

📄📄📄📄📄📄

```
omnihss:
  hosts:
    customer-hss01:
      ansible_host: 10.10.3.50
      gateway: 10.10.3.1
      host_vm_network: "vmbr3"
      hss_template_config: hss_runtime.exs.j2 # 📄 group_vars 📄
```

📄📄📄📄

📄📄📄:

1. Ansible 📄 `hss_template_config: hss_runtime.exs.j2`
2. 📄 `hosts/Customer/group_vars/hss_runtime.exs.j2` 📄📄
3. 📄 Jinja2 📄📄📄📄📄 `{{ inventory_hostname }}` 📄 `{{ plmn_id.mcc }}` 📄

4. `/etc/omnihss/runtime.exs`

5. `omnihss`

`hss_template_config` `omnihss`

参考: <https://docs.omnitouch.com.au/docs/repos/OmniCore>

## 4: **OmniMME**

OmniMME

インストール

```
hosts/Customer/
├── group_vars/
│   └── mme_runtime.exs.j2      # OmniMME 設定
```

設定

```
omnimme:
  hosts:
    customer-mme01:
      ansible_host: 10.10.3.51
      gateway: 10.10.3.1
      host_vm_network: "vubr3"
      mme_template_config: mme_runtime.exs.j2 # group_vars 参照
```

実行

手順:

- Ansible `mme_template_config: mme_runtime.exs.j2`
- `hosts/Customer/group_vars/mme_runtime.exs.j2` 参照
- Jinja2 テンプレート `{{ inventory_hostname }}` `{{ plmn_id.mcc }}` 参照
- `/etc/omnimme/runtime.exs`
- 実行

[[ mme\_template\_config ]]

[[[[: ]]]] <https://docs.omnitouch.com.au/docs/repos/OmniCore>

---

## [[[[: ]]]]

```
hosts/Customer/  
├─ host_files/  
│   └─ production.yml # [[[[[[ group_vars ]]]]]  
└─ group_vars/  
    ├─ smsc_controller.exs # OmniMessage [[[[[[  
    ├─ smsc_smpp.exs # OmniMessage SMPP [[[[[[  
    ├─ tas_runtime.exs.j2 # TAS [[[[[[  
    ├─ hss_runtime.exs.j2 # HSS [[[[[[  
    ├─ mme_runtime.exs.j2 # MME [[[[[[  
    ├─ dra_runtime.exs.j2 # DRA [[[[[[  
    ├─ pgwc_runtime.exs.j2 # PGW [[[[[[  
    ├─ dea_runtime.exs.j2 # DEA [[[[[[  
    ├─ upf_config.yaml # UPF [[  
    ├─ crm_config.yaml # CRM [[  
    ├─ stp.j2 # SS7 STP [[  
    ├─ hlr.j2 # SS7 HLR [[  
    ├─ camel.j2 # SS7 CAMEL [[  
    ├─ ipsmgw.j2 # IP-SM-GW [[  
    ├─ omnicore_smsc_ims.yaml.j2 # SMSC IMS [[  
    ├─ pytap.yaml # TAP3 [[  
    ├─ sip_profiles/ # SIP [[[[[[[[  
    │   └─ gateway_otw.xml  
    └─ dialplan/ # [[[[[[[[[[[[[[  
        ├─ mo_dialplan.xml # [[[[[[  
        ├─ mt_dialplan.xml # [[[[[[  
        └─ mo_emergency.xml # [[[[[[
```



# group\_vars

名前	グループ	説明
<code>smc_template_config</code>	omnimessage	Jinja2 テンプレート <code>smc_controller.exs</code>
<code>smc_smpp_template_config</code>	omnimessage_smpp	Jinja2 テンプレート <code>smc_smpp.exs</code>
<code>gateways_folder</code>	applicationserver	ディレクトリ <code>sip_profiles</code>
<code>dialplan_folder</code>	applicationserver	ディレクトリ <code>dialplan</code> ディレクトリ <code>dialplan</code>
<code>tas_template_config</code>	applicationserver	Jinja2 テンプレート <code>tas_runtime.exs.j2</code>
<code>hss_template_config</code>	omnihss	Jinja2 テンプレート <code>hss_runtime.exs.j2</code>
<code>mme_template_config</code>	omnimme	Jinja2 テンプレート <code>mme_runtime.exs.j2</code>
<code>dra_template_config</code>	dra	Jinja2 テンプレート <code>dra_runtime.exs.j2</code>
<code>pgwc_template_config</code>	pgwc	Jinja2 テンプレート <code>pgwc_runtime.exs.j2</code>
<code>frr_template_config</code>	omniupf	Jinja2 テンプレート <code>frr.conf.j2</code>

名前	拡張子	説明
SS7 設定	ss7	Jinja2 テンプレート stp.j2、hlr.j2、camel.j2
設定 YAML		テンプレート upf_config.yaml、crm_config.yaml

## 手順

1. **group\_vars** ディレクトリ - 作成
2. テンプレート - テンプレート `smc_template_config` と `gateways_folder`
3. テンプレート **Jinja2** - テンプレート `{{ variable_name }}` を Ansible テンプレート
4. テンプレート - テンプレート
5. テンプレート - テンプレート `group_vars` を Git

## テンプレート `group_vars`

テンプレート `group_vars` :

- テンプレート
- SIP テンプレート
- テンプレート
- Diameter テンプレート
- テンプレート

テンプレート `group_vars` :

- テンプレート IP テンプレート - テンプレート
- テンプレート - テンプレート

- `group_vars` - `group_vars`
- 

## References


- [Ansible](#) - `group_vars`
- [Ansible](#) - `group_vars`
- **OmniCall** `group_vars`: <https://docs.omnitouch.com.au/docs/repos/OmniCall> - `group_vars`
- **OmniCore** `group_vars`: <https://docs.omnitouch.com.au/docs/repos/OmniCore> - `group_vars`



## □□□□

- □□□□ **IP** □□
- □□/□□□□ `host_vm_network` □□□□□□□□□□ N/A□
- **CPU**□vCPU □□□
- **RAM**□□□□□□□□□□
- □□□□□□□□□□□□□□□□□□
- □□□□□□□□□□□□

## □□□□

- □□□□□□□□□□□□□□□□□□
- □□□□□
- □□/□□□□

## HSS Diameter □□□

- □□□□□□□□□□□□□□□□
- **Diameter** □□□□□□IP□□□□□□□□□□
- □ HSS □□□□□□□□□□ 9568□

## □□□□□□□

## □□□□□□□

□□□□ (`services/common.yml`)

- □□□□□□□□□□□□□□□□
- □□□□□□SSH □□□□□□NTP
- □□□□□□□□□□
- □□□□□□□□□□□□□□

```
ansible-playbook -i hosts/customer/host_files/production.yml
services/common.yml
```

## ユーザ (services/setup\_users.yml)

- ユーザを作成
- SSH 接続 sudo 権限
- パスワードを設定

```
ansible-playbook -i hosts/customer/host_files/production.yml
services/setup_users.yml
```

## 再起動 (services/reboot.yml)

- 再起動
- 再起動後5分待機
- ログを確認

```
ansible-playbook -i hosts/customer/host_files/production.yml
services/reboot.yml
```

## IP

### IP 生成 (util\_playbooks/ip\_plan\_generator.yml)

- IP 生成 HTML 生成
- テンプレート生成
- テンプレート生成

```
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/ip_plan_generator.yml
```

### HSS バックアップ (util\_playbooks/hss\_backup.yml)

- HSS バックアップ
- MySQL データを Ansible でバックアップ
- ログを確認

```
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/hss_backup.yml
```

📁📁📁📁 (util\_playbooks/getLocalCapture.yml)

- 📁📁📁📁📁📁📁📁📁📁📁
- 📁 /etc/localcapture/ 📁📁 pcap 📁📁
- 📁📁📁📁📁📁📁📁

```
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/getLocalCapture.yml
```

📁📁 **MTU** (util\_playbooks/updateMtu.yml)

- 📁📁📁📁📁 MTU 📁📁
- 📁📁 netplan 📁📁📁📁
- 📁📁📁📁📁📁📁📁

```
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/updateMtu.yml
```

📁📁📁📁

- 📁 **README** - 📁📁📁📁
- **Ansible** 📁📁📁📁 - 📁📁📁📁
- 📁📁📁📁📁 - 📁📁📁📁📁
- 📁📁📁📁 - 📁📁📁📁📁
- **APT** 📁📁📁📁 - 📁📁📁

# □□□□□□

## □□

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□□

- □□□□□□□□
- □□□□□□□□IP □□□□□□□□
- □□□□□□□□□□□□□□□□
- □□□□□□□□PLMN□□□□□□□□

## □□□□

□□□□□□□□□□□□

```

services/hosts/
└─ Customer_Name/
   └─ host_files/
      └─ production.yml
      └─ staging.yml
      └─ lab.yml
    
```

## □□□□□□□□

□□□□□□□□□□□□□□□□

```
# EPC []
mme:
  hosts:
    customer-mme01:
      ansible_host: 10.10.1.15
      gateway: 10.10.1.1
      host_vm_network: "vmbr1"
      mme_code: 1
      network_name_short: Customer
      tac_list: [600, 601, 602]

sgw:
  hosts:
    customer-sgw01:
      ansible_host: 10.10.1.25
      gateway: 10.10.1.1
      cdrs_enabled: true

pgwc:
  hosts:
    customer-pgw01:
      ansible_host: 10.10.1.21
      gateway: 10.10.1.1
      ip_pools:
        - '100.64.16.0/24'

# IMS []
pcscf:
  hosts:
    customer-pcscf01:
      ansible_host: 10.10.4.165

# []

license_server:
  hosts:
    customer-licenseserver:
      ansible_host: 10.10.2.150

# []
all:
  vars:
    ansible_connection: ssh
    ansible_password: password
```

```
customer_name_short: customer
plmn_id:
  mcc: '001'
  mnc: '01'
```

□□□□□□

□□□□

□□□□□□□□

```
pcscf:
  hosts:
    customer-pcscf01:
      ansible_host: 10.10.1.15      # SSH □□ IP □□
      gateway: 10.10.1.1           # □□□□
      host_vm_network: "vmbr1"     # □□□□□□□□□□ NIC □□
```

□□ □ IP □□□□□□□□□□□□□□□□ IP □□□□□□□□□□ OmniCore □□□□□□□□□□

**Proxmox** □□ `host_vm_network` □□□□□□□□□□□□□□□□□□□□ **Proxmox VM/LXC** □□

## VM □□□□

□□□□□□□□□□

```
num_cpus: 4      # CPU □□
memory_mb: 8192  # □□□□□□□□ RAM
proxmoxLxcDiskSizeGb: 50 # □□□□□GB
```

□□□□□□□□

□□□□□□□□□□□□□□□□

## MME:

```
mme_code: 1 # MME 000001-255
mme_gid: 1 # MME ID
network_name_short: Customer # 
network_name_long: Customer Network
tac_list: [600, 601, 602] #
```

**PGW:**

```
ip_pools: # IP
- '100.64.16.0/24'
- '100.64.17.0/24'
combined_CP_UP: false #
```

```
online_charging_enabled: true # OCS
tas_branch: "main" #
gateways_folder: "gateways_prod" # SIP
```

all:vars

```

all:
  vars:
    #
    ansible_connection: ssh
    ansible_password: password
    ansible_become_password: password

    #
    customer_name_short: customer
    customer_legal_name: "Customer Inc."
    site_name: "Chicago DC1"
    region: US

    # PLMN
    plmn_id:
      mcc: '001'          #
      mnc: '01'          #
      mnc_longform: '001' # MNC

    #
    network_name_short: Customer
    network_name_long: Customer Network

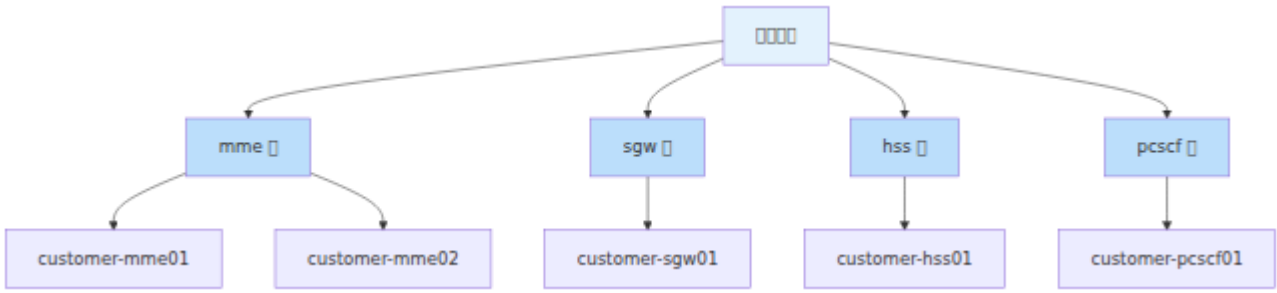
    # APT
    # apt_cache_servers
    # use_apt_cache true apt_repo.apt_server
    # IP
    apt_repo:
      apt_server: "10.254.10.223"
      apt_repo_username: "customer"
      apt_repo_password: "secure-password"
    use_apt_cache: false

    #
    TZ: America/Chicago

```

□□□□□

Ansible □□□□□□□□□□□□□□

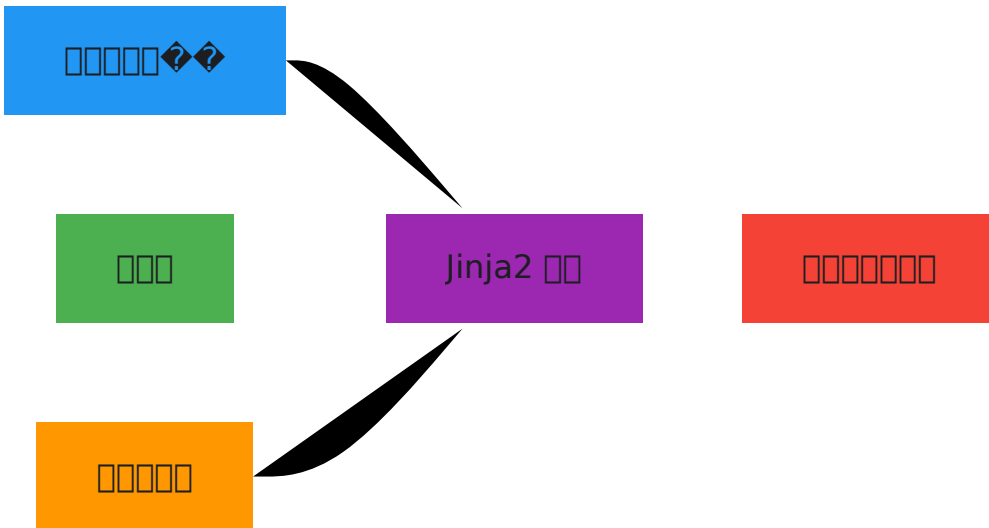


ansible mme ansible mme:hosts: ansible

## Ansible Jinja2

Ansible Jinja2 group\_vars

### Jinja2



ansible

ansible

```

plmn_id:
  mcc: '001'
  mnc: '01'
customer_name_short: acme
  
```

### Jinja2

```
# mme_config.yml.j2
network:
  plmn:
    mcc: {{ plmn_id.mcc }}
    mnc: {{ plmn_id.mnc }}
  operator: {{ customer_name_short }}
  realm: epc.mnc{{ plmn_id.mnc_longform }}.mcc{{ plmn_id.mcc
}}.3gppnetwork.org
```

□□□□□□□□

```
network:
  plmn:
    mcc: 001
    mnc: 01
  operator: acme
  realm: epc.mnc001.mcc001.3gppnetwork.org
```

## □□ Jinja2 □□

□□□□□□□□

```
{{ plmn_id.mcc }}
{{ apt_repo.apt_server }}
```

□□□□□

```
{% if online_charging_enabled %}
  charging:
    enabled: true
    ocs_ip: {{ ocs_ip }}
{% endif %}
```

□□□

```
tracking_areas:
{% for tac in tac_list %}
  - {{ tac }}
{% endfor %}
```

□□□□

```
# □□□□ 3 □
mnc{{ '%03d' | format(plmn_id.mnc|int) }}
```

## □□ **group\_vars** □□□□

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

□□□□□□□□□□

□□□□□□□□□□

□□□□□□□□□□□□□□□□□□□□□□□□

```
# EPC []
mme:
  hosts:
    customer-mme01:
      ansible_host: 10.10.1.15
      gateway: 10.10.1.1
      host_vm_network: "vmbr1"
      mme_code: 1
      mme_gid: 1
      network_name_short: Customer
      network_name_long: Customer Network
      tac_list: [600, 601, 602, 603]
      omnimme:
        sgw_selection_method: "random_peer"
        pgw_selection_method: "random_peer"

sgw:
  hosts:
    customer-sgw01:
      ansible_host: 10.10.1.25
      gateway: 10.10.1.1
      host_vm_network: "vmbr1"
      cdrs_enabled: true

pgwc:
  hosts:
    customer-pgw01:
      ansible_host: 10.10.1.21
      gateway: 10.10.1.1
      host_vm_network: "vmbr1"
      ip_pools:
        - '100.64.16.0/24'
      combined_CP_UP: false

hss:
  hosts:
    customer-hss01:
      ansible_host: 10.10.2.140
      gateway: 10.10.2.1
      host_vm_network: "vmbr2"

# IMS []
pcscf:
```

```
hosts:
  customer-pcscf01:
    ansible_host: 10.10.4.165
    gateway: 10.10.4.1
    host_vm_network: "vmbr4"

icscf:
  hosts:
    customer-icscf01:
      ansible_host: 10.10.3.55
      gateway: 10.10.3.1
      host_vm_network: "vmbr3"

scscf:
  hosts:
    customer-scscf01:
      ansible_host: 10.10.3.45
      gateway: 10.10.3.1
      host_vm_network: "vmbr3"

applicationserver:
  hosts:
    customer-as01:
      ansible_host: 10.10.3.60
      gateway: 10.10.3.1
      host_vm_network: "vmbr3"
      online_charging_enabled: false
      gateways_folder: "gateways_prod"

# [][]
license_server:
  hosts:
    customer-licenseserver:
      ansible_host: 10.10.2.150
      gateway: 10.10.2.1
      host_vm_network: "vmbr2"

monitoring:
  hosts:
    customer-oam01:
      ansible_host: 10.10.2.135
      gateway: 10.10.2.1
      host_vm_network: "vmbr2"
      num_cpus: 4
```

```
memory_mb: 8192

dns:
  hosts:
    customer-dns01:
      ansible_host: 10.10.2.177
      gateway: 10.10.2.1
      host_vm_network: "vmbr2"

# 网络
all:
  vars:
    ansible_connection: ssh
    ansible_password: password
    ansible_become_password: password

    customer_name_short: customer
    customer_legal_name: "Customer Network Inc."
    site_name: "Primary DC"
    region: US
    TZ: America/Chicago

# PLMN 网络
plmn_id:
  mcc: '001'
  mnc: '01'
  mnc_longform: '001'
  diameter_realm: epc.mnc{{ plmn_id.mnc_longform }}.mcc{{
plmn_id.mcc }}.3gppnetwork.org

# 网络
network_name_short: Customer
network_name_long: Customer Network
tac_list: [600, 601]

# APT 网络
apt_repo:
  apt_server: "10.254.10.223"
  apt_repo_username: "customer"
  apt_repo_password: "secure-password"
  use_apt_cache: false

# 网络
charging:
```

```

data:
  online_charging:
    enabled: false
  voice:
    online_charging:
      enabled: true
      domain: "mnc{{ plmn_id.mnc_longform }}.mcc{{ plmn_id.mcc
}}.3gppnetwork.org"

# firewall
firewall:
  allowed_ssh_subnets:
    - '10.0.0.0/8'
    - '192.168.0.0/16'
  allowed_ue_voice_subnets:
    - '10.0.0.0/8'
  allowed_signaling_subnets:
    - '10.0.0.0/8'

# Proxmox VM
proxmoxServers:
  customer-prxm01:
    # deployment_type vm → vm "vm"
    proxmoxServerAddress: 10.10.0.100
    proxmoxServerPort: 8006
    proxmoxApiTokenName: Customer
    proxmoxApiTokenSecret: "token-secret"
    proxmoxNodeName: pve01
    proxmoxTemplateName: ubuntu-24.04-cloud-init-template
    proxmoxTemplateId: 9000
    proxmoxTemplateUser: omnitouch # cloud-init
    proxmoxTemplatePassword: omnitouch # cloud-init
    proxmoxTemplateLocalUsers: local_users # cloud-init

# LXC proxmoxServers deployment_type: lxc
# proxmoxLxc0sTemplate

```

Proxmox VM/LXC Proxmox

## □□□□□□

□□□□□□□□□□□□□□□□□□□□

### OmniCore □□□

- **OmniCore** □□: <https://docs.omnitouch.com.au/docs/repos/OmniCore>
- **OmniHSS** - □□□□□□□□
- **OmniSGW** - □□□□□□□□□□
- **OmniPGW** - □□□□□□□□□□
- **OmniUPF** - □□□□□□
- **OmniDRA** - Diameter □□□□
- **OmniTWAG** - □□ WLAN □□□□

### OmniCall □□□

- **OmniCall** □□: <https://docs.omnitouch.com.au/docs/repos/OmniCall>
- **OmniTAS** - IMS □□□□□□VoLTE/VoNR□
- **OmniCall CSCF** - □□□□□□□□
- **OmniMessage** - □□□□
- **OmniMessage SMPP** - SMPP □□□□
- **OmniSS7** - SS7 □□□
- **VisualVoicemail** - □□□□

### OmniCharge/OmniCRM:

- **OmniCharge** □□: <https://docs.omnitouch.com.au/docs/repos/OmniCharge>

## □□□□

- **Ansible** □□□□ - □□□□□□
- □□□□ - □□□□□□□□□□□□
- □□□□□□ - □□□□□□
- **IP** □□□□ - □□□□□□ **IP** □□□□
- **Netplan** □□ - □□ **IP** □□□□□□□□

- APT 0000 - 00000
- 000000 - 00000
- 000000 - 000000

## 000

1. 000000000000000
2. 0000 PLMN 00000
3. 00 APT 00000
4. 000000000
5. 000000 group\_vars 00000
6. 00 Ansible 000000



# 🔍

## 📄

🔒 HTTPS🔒 443🔒 Omnitouch 📄

📄	IP 📄	📄
time.omnitouch.com.au	160.22.43.18	📄 1
time.omnitouch.com.au	160.22.43.66	📄 2
time.omnitouch.com.au	160.22.43.114	📄 3

📄

- 📄HTTPS (TCP/443)
- 📄160.22.43.18, 160.22.43.66, 160.22.43.114
- 📄

## DNS 📄

🔒 DNS 📄 Omnitouch 📄

📄 **DNS** 📄

- 📄 DNS 📄
- 📄 DNS 📄
  - 1.1.1.1 (Cloudflare - 📄 DNS)
  - 8.8.8.8 (Google 📄 DNS)
- 📄/📄 DNS 📄

📄 Omnitouch 📄 DNS (DoH/DoT)📄 DNS 📄 DNSSEC 📄  
📄 DNS 📄

□□□□

- □□□□
- Hosts □□□□



□□

□□	□□	□□	□□□□
<b>Prometheus</b>	□□□□□□□□	9090	15 □□□□□
<b>Loki</b>	□□□□	3100	7 □□ 50GB
<b>InfluxDB</b>	RAN □□□□□□□□	8086	30 □
<b>Grafana</b>	□□□□□□	3000	N/A
□□□□□	□□□□	9100	N/A
<b>SNMP</b> □□□	□□□□□□	9116	N/A
□□□□□	ICMP/HTTP □□	9115	N/A

---

□□□

## Prometheus

Prometheus □ 1 □□□□□□ OmniCore □□□□□□

□□□ **UID:** `omnicore_prometheus`

□□□□

名前	名前	ポート	説明
すべて	all	9100	すべての CPU 機能
MMEs	mme	9568	コア/コア
HSS	hss	9568	コア
SGW-C	sgw	9568	コア GTP-C
PGW-C	pgwc	9090	PDN コア IP
UPF コア	upf	9090	コア
Kamailio CSCF	pcscf, scscf, icscf	9090	コア
コア FreeSWITCH	applicationserver	9090	コア
DRA	dra	9568	コア
OmniMessage	omnimessage	9568	コア SMPP
OmniSS7	omniss7	8080	SS7/SIGTRAN コア
KeyDB	ocs	9121	コア/
CGrateS	ocs	2080	コア CDR

コア

項目	設定	値
SNMP (MikroTik)	IP/ポート	<code>mikrotik.hosts</code>
SNMP (iDRAC)	IP	<code>idrac.hosts</code>
SNMP (Synology)	NAS IP	<code>synology.hosts</code>
SNMP (SAF)	IP	<code>saf.hosts</code>
SNMP (Cisco)	IP	<code>cisco.hosts</code>
ICMP	IP + 8.8.8.8	なし
VMware	vCenter IP	<code>vcenter_ip</code> , <code>vcenter_password</code>
Proxmox	PVE IP	<code>proxmoxServers</code>

## Loki

Loki は Grafana Alloy を利用して

UID: `omnicore_loki`

Loki/Alloy のインストール

インストール

項目	説明	値
hostname	ホスト名	customer-mme01
component	コンポーネント	mme, cscf, ocs
unit	systemd ユニット	omnimme.service
level	ログレベル	info, error

## InfluxDB

InfluxDB データベースの構成

UID: omnicores\_influxdb

項目

名前	説明	期間
nokia-monitor	RAN モニタリング	30 日
dra	DRA モニタリング	365 日
Omnicharge_TAP3	TAP モニタリング	90 日

## Grafana

ダッシュボード

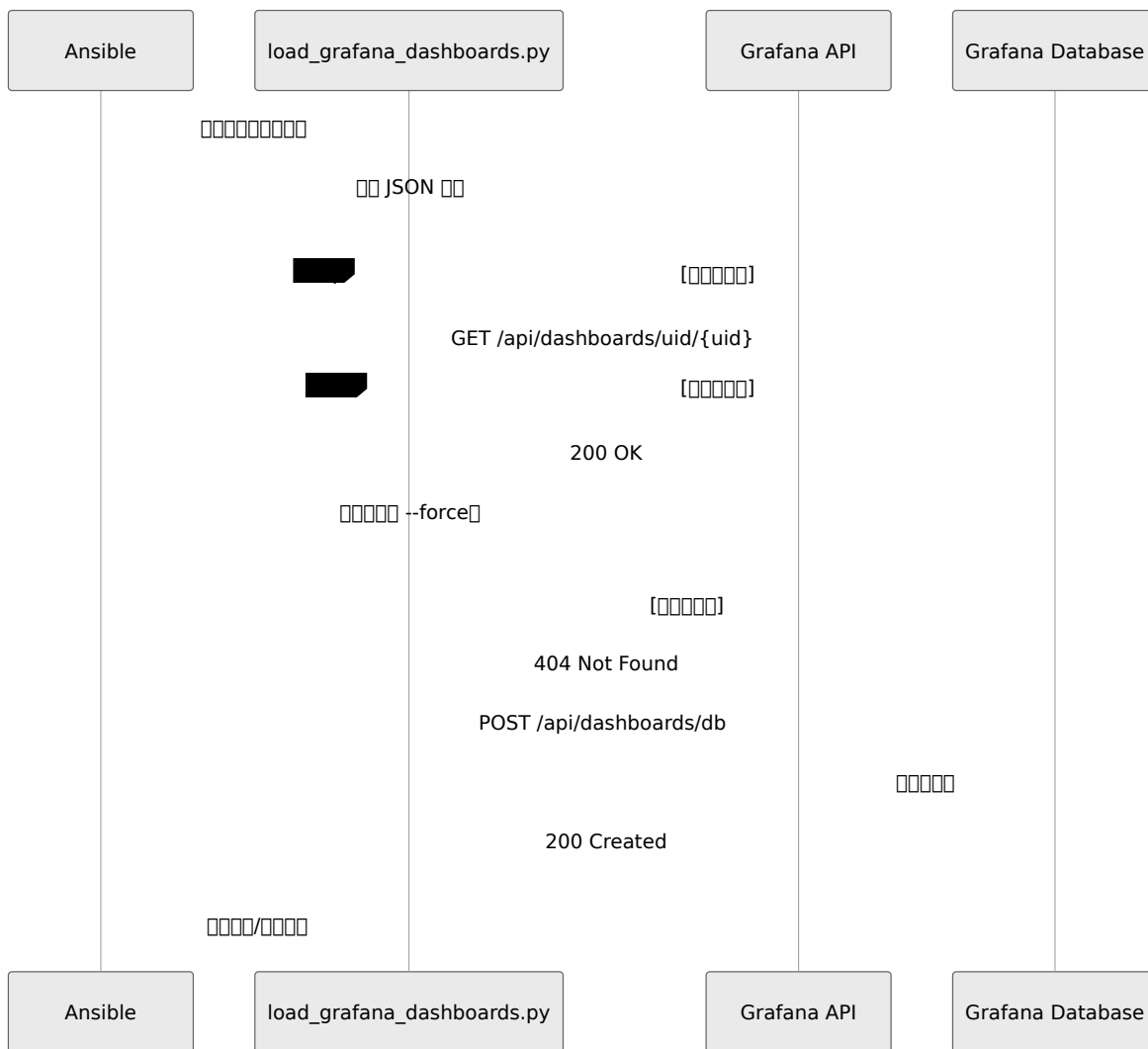
監視対象のシステム

名前	説明
<b>BSS</b>	CGrateS、KeyDB
<b>EPC</b>	MME、SGW、PGW、UPF、HSS、DRA
<b>IMS</b>	CSCF、OmniMessage
監視	SNMP
<b>RAN</b>	OmniRAN
その他	

## API

Grafana API

- Grafana UI
- API
- Ansible



[ ]

[ ] Ansible [ ]

```
# Ansible
python3 roles/monitoring/files/load_grafana_dashboards.py \
  --url http://<monitoring-ip>:3000 \
  --user admin \
  --password <grafana_admin_password> \
  --dashboards-dir roles/monitoring/templates/grafana/dashboards

#
python3 roles/monitoring/files/load_grafana_dashboards.py \
  --url http://<monitoring-ip>:3000 \
  --user admin \
  --password <grafana_admin_password> \
  --dashboards-dir roles/monitoring/templates/grafana/dashboards
\
  --force
```

JSON Ansible

```
roles/monitoring/templates/grafana/dashboards/
├── BSS/
│   ├── KeyDB_Cluster.json
│   ├── cgrates_mysql.json
│   └── cgrates_stats.json
├── EPC/
│   ├── MME_Dashboard.json
│   ├── OmniHSS.json
│   ├── OmniDRA.json
│   ├── SGW.json
│   ├── PGWs.json
│   └── ...
├── IMS/
│   ├── SMSc.json
│   ├── MMSc.json
│   └── ...
├── Infrastructure/
│   ├── Node_Exporter_Full.json
│   ├── MikroTik_Dashboard.json
│   └── ...
├── RAN/
│   ├── Nokia_Overview.json
│   ├── Nokia_Detailed.json
│   └── ...
└── Logs/
    ├── CSCF_Logs.json
    ├── MME_Logs.json
    └── ...
```

□□□□□

□□□□ Grafana □□□□□□□□□□□□□□□□□□□□

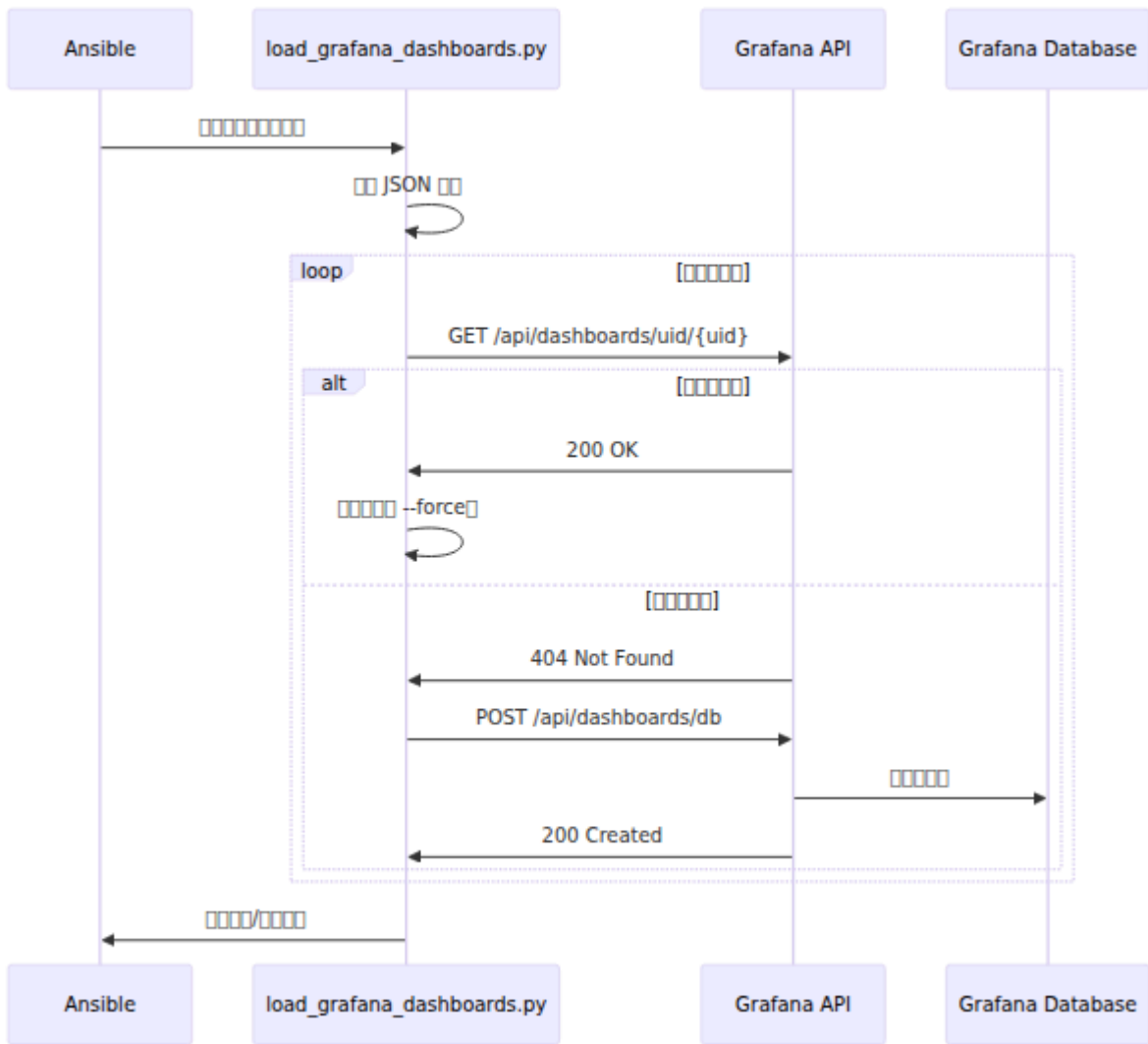
```
# □ roles/monitoring □□
python3 export_grafana.py --customer <CUSTOMER>
```

□□□□□

- □□□□□□  
roles/monitoring/templates/grafana/dashboards/{folder}/\*.json □□□□□

□□

- □□□□□  
hosts/Omnicores\_{CUSTOMER}/group\_vars/grafana/alerts/all\_alert\_rules.json
- □□□□  
hosts/Omnicores\_{CUSTOMER}/group\_vars/grafana/alerts/contact\_points.json
- □□□□□  
hosts/Omnicores\_{CUSTOMER}/group\_vars/grafana/alerts/notification\_policies.json



□□□□

- □□□□□□□□□□□□□□ version □ id □□□□□□□□
- □□□ Grafana □□□□□□□□□□□□

- `group_vars/`

`grafana/`

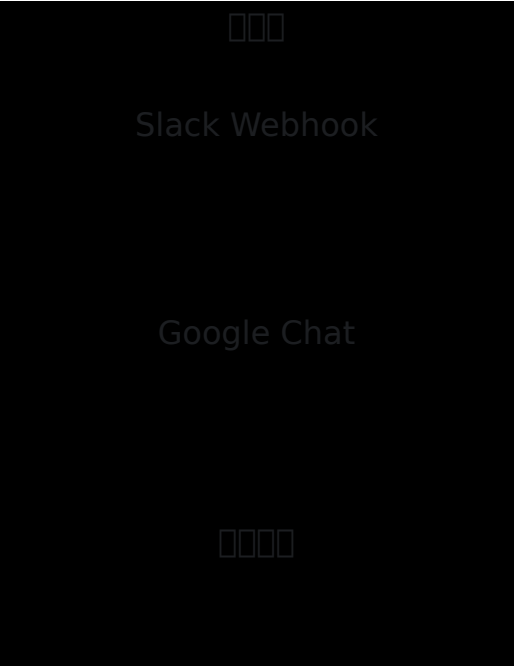
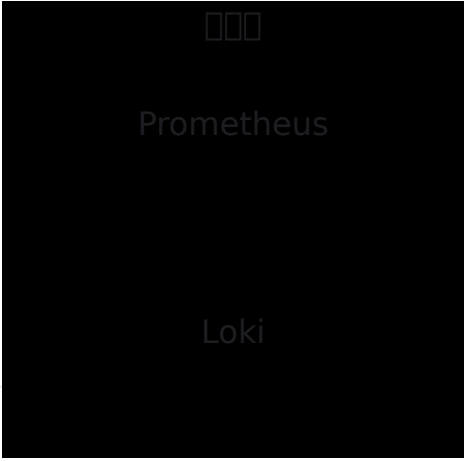
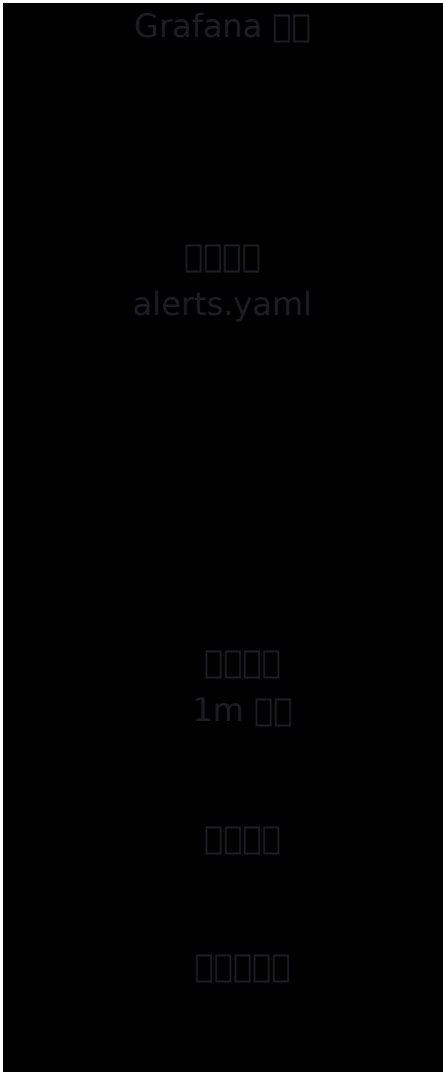
`dashboards/`

```
hosts/customer/group_vars/  
└─ grafana/  
    └─ dashboards/  
        ├── Custom_Dashboard_1.json  
        └─ Custom_Dashboard_2.json
```

---

□□

□□



## Alerting

Alerting is configured via a YAML file and the API.

```
python3 roles/monitoring/files/load_grafana_alerts.py \
  --url http://<monitoring-ip>:3000 \
  --user admin \
  --password <grafana_admin_password> \
  --alerts-file
roles/monitoring/templates/grafana/provisioning/alerting/alerts.yaml
```

## Alerting Rules

Alert Name	Group	Expression	Duration
Alerting CPU 90%	Alerting	Alerting CPU > 90%	5 min
<b>CPU</b> > 90%	Alerting	CPU > 90%	5 min
Alerting CPU 90%	Alerting	Alerting CPU > 90%	5 min
Alerting Prometheus	Alerting	Prometheus	5 min
<b>MME</b> > 40%	EPC	MME > 40%	30 min
<b>MME</b> < 0	EPC	MME < 0	5 min
<b>IMS</b> > 40%	IMS	P-CSCF > 40%	30 min
Alerting MOS < 4	IMS	Alerting MOS < 4	5 min
<b>eNodeB</b> > 1	RAN	Alerting eNB > 1	1 min
Alerting P95	EPC	P95	5 min

□□□□□□

alerts.yaml □□□□□□

```
apiVersion: 1
groups:
- orgId: 1
  name: □□□
  folder: □□□□
  interval: 1m
  rules:
- uid: alert-lowDiskSpace
  title: □□□□□□□□ 90% □□
  condition: C
  data:
- refId: A
  datasourceUid: omnicores-prometheus
  model:
    expr: |
      100 - ((node_filesystem_avail_bytes{job="Node
Exporter",mountpoint="/" } * 100)
      / node_filesystem_size_bytes{job="Node
Exporter",mountpoint="/" })
      # ... □□□□□
  noDataState: OK
  execErrState: Error
  for: 5m
  annotations:
    summary: □□□□□□□□□□ 90%
  labels:
    type: resource
```

□□□□□□

□□□□□□□□□□□□□□

```
# [] group_vars/all.yml []
monitoring_data:
  contactPoints:
    - orgId: 1
      name: default
      receivers:
        # Slack []
        - uid: slack-alerts
          type: slack
          disableResolveMessage: false
          settings:
            url: "https://hooks.slack.com/services/xxx/yyy/zzz"

        # Google Chat []
        - uid: gchat-alerts
          type: googlechat
          disableResolveMessage: false
          settings:
            url:
              "https://chat.googleapis.com/v1/spaces/xxx/messages?key=yyy"
```

[][][]

[][][][][][][][][][]

```
# templates/grafana/notification-policies.yaml.j2
apiVersion: 1

policies:
  - orgId: 1
    receiver: default
    group_by: ['alertname', 'instance']
    group_wait: 30s
    group_interval: 5m
    repeat_interval: 4h
```

---

□□□□

## Prometheus □□

□□□□□□□□ `/etc/prometheus/prometheus.yml` □

□□	□□	□□
<code>scrape_interval</code>	1m	□□□□□□□□
<code>evaluation_interval</code>	1m	□□□□□□□□□□
<code>scrape_timeout</code>	50s	□□□□□□□□

## Grafana □□

□□□□□□□□ `/etc/grafana/grafana.ini` □

□□□□□□□□□□□□□□

□□	□	□□
<code>admin_password</code>	<code>grafana_admin_password</code> var	□□□□□□□□
<code>allow_embedding</code>	true	□□□ iframe □□□
<code>provisioning</code>	<code>/etc/grafana/provisioning</code>	□□□□

## Loki □□

□□ Loki □□□□□□□□□□□□ □□□□□□□□

## InfluxDB □□

□□□□□□□□ `/etc/influxdb/influxdb.conf` □

名前	プロバイダ	ポート
nokia-monitor	autogen	30
dra	autogen	365
Omnicharge_TAP3	autogen	90

---

## ポート

名前	ポート	プロトコル	説明
Grafana	3000	HTTP	管理 UI と API
Prometheus	9090	HTTP	メトリクス収集
Loki	3100	HTTP	ログ収集
InfluxDB	8086	HTTP	InfluxDB API
SNMP	9100	HTTP	SNMP
SNMP	9116	HTTP	SNMP
SNMP	9115	HTTP	SNMP
Alloy	12345	HTTP	Alloy UI

---

## □□□□

### □□□□

1. □□ Grafana□□□□ `http://<monitoring-ip>:3000`
2. □□□ □□□ □□□□□□□□□□
3. □□□□□□□□□□□□□□□□

### □□□□

1. □□ Grafana□□□□ `http://<monitoring-ip>:3000`
2. □□ □□ □□□ **Loki** □□□□
3. □□ LogQL □□□□

```
# □□□□□□□□□□□□  
{hostname="customer-mme01"}  
  
# □□ MME □□□□□□  
{component="mme"} |~ "(?i)error"  
  
# □□□□ IMSI  
{component="hss"} |= "123456789012345"
```

### □□□□□□□□

1. □  
`roles/monitoring/templates/grafana/provisioning/alerting/alerts.ya`  
`ml` □□□□□□□□
2. □□□□□□□□□□□□□□

```
python3 roles/monitoring/files/load_grafana_alerts.py \  
  --url http://<monitoring-ip>:3000 \  
  --user admin --password <password> \  
  --alerts-file  
roles/monitoring/templates/grafana/provisioning/alerting/alerts.y  
\  
  --force
```

□□□□□

□ Grafana UI □□□□□□□□□□□□□□

```
cd roles/monitoring
python3 export_grafana.py --url http://<monitoring-ip>:3000
git add templates/grafana/
git commit -m "□□□□□□□□"
```

□□□□

## Prometheus □□□□

□□□□□□□□“□□□”□□□□□□□□ DOWN

□□□□□□

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

□□□□□

1. □□□□□□□□□□□□□□□□

```
systemctl status <service>
curl http://localhost:<port>/metrics
```

2. □□□□□□□□□□□□□□

```
curl http://<target>:<port>/metrics
```

3. □□ Prometheus □□□□□ `http://<monitoring-ip>:9090/targets`

## 목차

### 1. Grafana 설치

#### 1.1 요구사항

- 운영체제
- 포트
- 로그 JSON 형식

#### 1.2 설치

1. Grafana 패키지 다운로드 → 압축 해제 → 설치
2. Grafana 서비스 `journalctl -u grafana-server -f`로 모니터링
3. Grafana 웹 인터페이스 접근

## 2. Grafana 설정

### 2.1 데이터 소스 설정

#### 2.1.1 Prometheus

- Prometheus 서버 주소
- Prometheus 사용자 이름
- Prometheus 비밀번호

#### 2.1.2 Loki

1. Grafana 패키지 다운로드 → 압축 해제
2. Grafana 서비스 모니터링 → 압축 해제 → 설치
3. Grafana 웹 인터페이스 접근

## Grafana 모니터링 구성

### 1. Grafana 모니터링 "시시시" 구성

#### 1.1 요구사항

- Prometheus/Loki/InfluxDB 部署
- 部署 Prometheus
- 部署 Loki

部署

1. 部署 Prometheus

```
systemctl status prometheus loki influxdb
```

2. 部署 Loki

```
ss -tlnp | grep -E '9090|3100|8086'
```

3. 部署 InfluxDB

```
curl http://localhost:9090/api/v1/status/config  
curl http://localhost:3100/ready
```

---

## VIP 部署

VIP 部署 IMS EPC UE 部署





項目	単位	初期値	範囲	説明
<code>hss_url</code>	文字列	-	-	OmniHSS REST API URL [HTTPS]
<code>scrape_interval_seconds</code>	秒	30	1-3600	スクレイピング間隔
<code>ping_timeout_seconds</code>	秒	2	1-10	UE IP ping タイムアウト
<code>ping_count</code>	回数	2	1-10	ping 回数
<code>blackbox_exporter_url</code>	文字列	-	-	ping 結果を出力する URL <code>http://pcscf01:9115</code> を指定して ICMP ping
<code>subscribers</code>	文字列	-	-	サブスクリプション

設定例

項目	値	単位	初期値	説明
<code>imsi</code>	15	文字列	-	IMSI 15 桁
<code>label</code>	IMSI	文字列	IMSI	ラベル名
<code>type</code>	<code>full</code> <code>voip_only</code> <code>data_only</code>	文字列	<code>full</code>	取得するデータの種類

MSISDN HSS API 接続設定

00

000000 9550 0 /metrics 000000

0000

00	00	00
<code>vip_subscriber_service_healthy</code>	Gauge	000000000000000000000000 1000000
<code>vip_subscriber_ims_registered</code>	Gauge	00000000 S-CSCF0000 1000000 0
<code>vip_subscriber_epc_registered</code>	Gauge	0000000000 MME 000000 1000000 0
<code>vip_subscriber_ue_ip_reachable</code>	Gauge	00 UE IP 0 ping 000000 1000000 0
<code>vip_subscriber_hss_reachable</code>	Gauge	00 HSS API 000000000000 1000000 0
<code>vip_subscriber_enabled</code>	Gauge	00000000 HSS 00000000 1000000 0

0000

00	00	00
<code>vip_subscriber_ims_registration_age_seconds</code>	Gauge	0000 IMS 000000 000000000000 -10
<code>vip_subscriber_epc_registration_age_seconds</code>	Gauge	0000 EPC 000000 00000000000000 0 -10

□□□□

□□	□□	□□
<code>vip_subscriber_info</code>	Gauge	□□□ 1□□□□□□□□□□□□□□□□

□□□□□□□□□□

□□	□□	□□
<code>imsi</code>	□□ IMSI	<code>313380930011948</code>
<code>label</code>	□□□□□□	□□
<code>msisdn</code>	□□□□□□□ HSS□	<code>24724748250</code>
<code>type</code>	□□□□	<code>full</code> □ <code>voip_only</code> □ <code>data_only</code>

`vip_subscriber_info` □□□□□□□□

□□	□□	□□
<code>healthy</code>	□□□□□□	<code>1</code> □ <code>0</code>
<code>ims</code>	IMS □□□□	<code>1</code> □ <code>0</code>
<code>epc</code>	EPC □□□□	<code>1</code> □ <code>0</code>
<code>ping</code>	UE IP □□□	<code>1</code> □ <code>0</code>
<code>assigned_scscf</code>	S-CSCF □□□	<code>scscf01.ims.example.com</code>
<code>last_seen_mme</code>	MME □□□	<code>mme02.epc.example.com</code>
<code>ue_ip</code>	UE □□□ IP □□	<code>100.72.83.20</code>

□□□□

```
# VIP 健康
count(vip_subscriber_service_healthy == 1)

# 不健康 VIP 健康
count(vip_subscriber_service_healthy == 0)

# VoIP 健康 IMS 健康
vip_subscriber_ims_registered{type=~"voip_only|full"}

# 健康 EPC 健康
vip_subscriber_epc_registered{type=~"data_only|full"}

# IMS 健康 >1 健康
vip_subscriber_ims_registration_age_seconds > 3600
```

健康

**IMS VIP** 健康 健康 VIP 健康

健康 Grafana → IMS → IMS VIP 健康

健康 **URL** `/d/ims-vip-subscribers/`

健康

項目	説明
概要	概要
構成	構成
<b>IMS</b> 項目	IMS 項目
<b>EPC</b> 項目	EPC 項目
<b>UE</b> 項目	ping UE IP 項目
項目	VIP 項目
<b>VIP</b> 項目	項目
項目	項目

項目

項目 VIP 項目

**VIP** 項目

項目 VIP 項目 項目 1 項目 vip\_subscriber\_service\_healthy == 0

項目

- 項目 VIP 項目 {{ \$labels.label }} 項目
- 項目
  - VoIP 項目 MSISDN
  - 項目 IMSI
  - 項目 MSISDN 項目 IMSI

項目

- `voip_only` `MSISDN: 24724766000`
- `data_only` `IMSI: 313380930010064`
- `full` `MSISDN: 24724748250 IMSI: 313380930011948`

## 配置 VIP

### 1. 配置

```
# hosts/<customer>/group_vars/vip_subscribers.yaml
subscribers:
  - imsi: "123456789012345"
    label: "VIP"
    type: "full" # voip_only, data_only
```

### 2. 部署

```
scp vip_subscribers.yaml <monitoring-server>:/tmp/
ssh <monitoring-server> "sudo cp /tmp/vip_subscribers.yaml
/etc/prometheus/vip_subscribers.yaml && sudo systemctl restart
ims-subscriber-exporter"
```

### 3. 验证

```
curl -s http://<monitoring-server>:9550/metrics | grep "VIP"
```

---

## 相关链接

- [Prometheus](#) — Loki Alloy
- [Prometheus](#) —
- [Prometheus](#) —

# Netplan

OmniCore netplan

- (eth0)
- IP
- 

## Netplan

netplan group\_vars Jinja2 netplan\_config

```
dra:  
  hosts:  
    <hostname>:  
      ansible_host: 10.0.1.100  
      gateway: 10.0.1.1  
      netplan_config: netplan.yaml.j2
```

hosts/<customer>/group\_vars/netplan.yaml.j2

netplan.yaml.j2

```

network:
  version: 2
  ethernets:
    # ipv4 - ipv4 ansible_host gateway
    eth0:
      addresses:
        - "{{ ansible_host }}/{{ mask_cidr | default(24) }}"
      nameservers:
        addresses:
{% if 'dns' in group_names %}
          # ipv4 DNS ipv4 DNS ipv4
          - 8.8.8.8
{% else %}
          # ipv6 dns 'dns' ipv6 DNS ipv6
{% for dns_host in groups['dns'] | default([]) %}
          - {{ hostvars[dns_host]['ansible_host'] }}
{% endfor %}
{% endif %}
      search:
        - slice
      routes:
        - to: "default"
          via: "{{ gateway }}"

{% if secondary_ips is defined %}
  # ipv4 - ipv4 secondary_ips ipv4
  # ipv4ens19, ens20, ens21... (18 + loop.index)
{% for nic_name, nic_config in secondary_ips.items() %}
  ens{{ 18 + loop.index }}:
    addresses:
      - "{{ nic_config.ip_address }}/{{ mask_cidr | default(24)
    }}"
{% if nic_config.routes is defined %}
    # ipv4 ipv4 - ipv4 ipv4
    routes:
{% for route in nic_config.routes %}
      - to: "{{ route }}"
        via: "{{ nic_config.gateway }}"
{% endfor %}
{% endif %}
{% endfor %}
{% endif %}

```



□□□□

□□□□□ Ubuntu □□□□□□□□□□□□□□

- □□□□□□□□ ens19
- □□□□□□□□ ens20
- □□□□□□□□ ens21
- □□□□...

□□ Proxmox □□□□□□□□□□ NIC □□□□□□□□□□□□□□

□□□□

```
dra:
  hosts:
    <hostname>:
      ansible_host: 10.0.1.100
      gateway: 10.0.1.1
      host_vm_network: "ovsbr1"
      vlanid: "100"
      netplan_config: netplan.yaml.j2
      secondary_ips:
        public_ip:
          ip_address: 192.0.2.50
          gateway: 192.0.2.1
          host_vm_network: "vibr0"
          vlanid: "200"
          routes:
            - '198.51.100.0/24'
            - '203.0.113.0/24'
        peering_ip:
          ip_address: 172.16.50.10
          gateway: 172.16.50.1
          host_vm_network: "ovsbr2"
          vlanid: "300"
          routes:
            - '172.17.0.0/16'
```

## Netplan

```
network:
  version: 2
  ethernets:
    eth0:
      addresses:
        - "10.0.1.100/24"
      nameservers:
        addresses:
          - 10.0.1.53
        search:
          - slice
      routes:
        - to: "default"
          via: "10.0.1.1"
    ens19:
      addresses:
        - "192.0.2.50/24"
      routes:
        - to: "198.51.100.0/24"
          via: "192.0.2.1"
        - to: "203.0.113.0/24"
          via: "192.0.2.1"
    ens20:
      addresses:
        - "172.16.50.10/24"
      routes:
        - to: "172.17.0.0/16"
          via: "172.16.50.1"
```

## Proxmox

proxmox.yml NIC

1. NIC
2. NIC

## Proxmox 環境

- `host_vm_network` - NIC 名
- `vlanid` - VLAN ID

## 手順



1. Jinja2 テンプレートを作成
2. テンプレートを `/etc/netplan/01-netcfg.yaml` に保存
3. Netplan の構成を適用
4. `netplan apply` を実行
5. `ip addr show` で IP アドレスを確認

## Ansible

### IP アドレスを動的に割り当てる (DEA)

```
<hostname>:
  ansible_host: 10.0.1.100          # 本機 IP
  gateway: 10.0.1.1
  netplan_config: netplan.yaml.j2
  secondary_ips:
    diameter_roaming:
      ip_address: 192.0.2.50       # 仮想 IP
      gateway: 192.0.2.1
      host_vm_network: "vibr0"
      vlanid: "200"
      routes:
        - '198.51.100.0/24'       # 仮想ネットワーク
```

## PGW S5/S8

```
<hostname>:
  ansible_host: 10.0.2.20          # IP
  gateway: 10.0.2.1
  netplan_config: netplan.yaml.j2
  secondary_ips:
    s5s8_interface:
      ip_address: 203.0.113.17    # S5/S8 IP
      gateway: 203.0.113.1
      host_vm_network: "vmbr0"
      vlanid: "50"
```

PGW S5/S8

```
<hostname>:
  ansible_host: 10.0.1.100        #
  gateway: 10.0.1.1
  netplan_config: netplan.yaml.j2
  secondary_ips:
    data_network:
      ip_address: 10.0.2.100      #
      gateway: 10.0.2.1
      host_vm_network: "ovsbr2"
      vlanid: "200"
    backup_network:
      ip_address: 10.0.3.100      #
      gateway: 10.0.3.1
      host_vm_network: "ovsbr3"
      vlanid: "300"
```

## IP

Jinja2 IP

## Inventory

Inventory IP addresses are defined in `inventory_hostname`

```
# Secondary IP addresses
{{ secondary_ips.diameter_public_ip.ip_address }}

# Inventory hostname variables
{{ hostvars[inventory_hostname]['secondary_ips']
  ['diameter_public_ip']['ip_address'] }}

# Gateway and VLAN
{{ secondary_ips.diameter_public_ip.gateway }}
{{ secondary_ips.diameter_public_ip.vlanid }}
```

## Groups

Groups are defined in `hostvars`

```
# Dra group
{{ hostvars[groups['dra'][0]]['secondary_ips']
  ['diameter_public_ip']['ip_address'] }}

# DRA IP addresses
{% for host in groups['dra'] %}
{% if hostvars[host]['secondary_ips'] is defined %}
  - {{ hostvars[host]['secondary_ips']['diameter_public_ip']
    ['ip_address'] }}
{% endif %}
{% endfor %}
```

## DRA IP

DRA IP addresses are defined in `IP`

```
# dra_config.yaml.j2 - inventory_hostname
peers:
  - name: external_peer
    # IP
    local_ip: {{ hostvars[inventory_hostname]['secondary_ips']
['diameter_public_ip']['ip_address'] }}
    remote_ip: 198.51.100.50
    port: 3868
```

## IP

```
{% if secondary_ips is defined and
secondary_ips.diameter_public_ip is defined %}
public_ip: {{ secondary_ips.diameter_public_ip.ip_address }}
{% else %}
public_ip: {{ ansible_host }}
{% endif %}
```

SSH

```
ip link show
```

```
1: lo: <LOOPBACK,UP,LOWER_UP> ...
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> ...
3: ens19: <BROADCAST,MULTICAST,UP,LOWER_UP> ...
4: ens20: <BROADCAST,MULTICAST,UP,LOWER_UP> ...
```

## Netplan

```
cat /etc/netplan/01-netcfg.yaml
```

## Netplan

```
netplan apply
```

## Netplan

```
netplan --debug apply
```

## 

```
ip route show
```

## 

- -
- Proxmox VM/LXC -
- -

# Proxmox VM/LXC 00

00000000 Proxmox 000 OmniCore 000000000000 — services/proxmox.yml — 000  
00 deployment\_type 0000 QEMU 0000 LXC 000

00000000 VMware/HyperV 000 Vultr / AWS / GCP 0000000

000

- 000000 — 0000000000/LXC
- IP 0000
- Netplan 00
- 0000

## LXC 0 VM

LXC 000

- 000000000000
- 00000000
- 0000
- 0000000000000000
- 00000 **EPC/IMS** 00
- 0000 **UPF** 00000000/TUN 000

000 (KVM) 0

- 00000000000000
- 0000000000000/000000
- 00000000
- 0000000
- **UPF** 0000

00000

- `proxmoxServers.*.UPF`
- **LXC** `apt-cache` `dns`

`proxmoxServers.*.UPF` = `proxmoxServers.*.UPF`

`proxmoxServers.*.deployment_type` LXC `apt-cache` `dns`

## Proxmox

### 1. API

```
# Proxmox UI: → → API
# : root@pam!<TokenName>
#
```

### 2a. Cloud-Init

Proxmox Ubuntu — cloud-init SSH

`proxmox.yml`





## 2b. LXC LXC

```
# Proxmox shell  
pveam update  
pveam download local ubuntu-24.04-standard_24.04-2_amd64.tar.zst
```

```
local:vztmpl/ubuntu-24.04-standard_24.04-2_amd64.tar.zst  
proxmoxLxc0sTemplate
```

□□□□□□

□□□□□

```
all:
  vars:
    # deployment_type □□□ → □□□□ "vm"
    proxmoxServers:
      pve-node-01:
        proxmoxServerAddress: 192.168.1.100
        proxmoxServerPort: 8006
        proxmoxApiTokenName: ansible
        proxmoxApiTokenSecret: "your-token-secret-uuid"
        proxmoxNodeName: pve-node-01
        proxmoxTemplateName: ubuntu-2404-template
        proxmoxTemplateId: 9000
        proxmoxTemplateUser: omnitouch # □□□ cloud-init □□
        □□□□□□□□ local_users □
        proxmoxTemplatePassword: omnitouch # □□□ cloud-init □□□
        □□□ cloud-init □□□
        storage: local-lvm # □□
      pve-node-02:
        # ... □□□□□□□□

    local_users:
      admin_user:
        name: Admin User
        public_key: "ssh-rsa AAAA..."

mme:
  hosts:
    site-mme01:
      ansible_host: 192.168.1.10
  vars:
    proxmox_interface: vmb0
    gateway: 192.168.1.1
    netmask: 255.255.255.0
    vlanid: 100 # □□
```

## LXC 配置

```
all:
  vars:
    proxmox_lxc_nameserver: "1.1.1.1" # 指定 DNS 1.1.1.1
    proxmoxServers:
      pve-node-01:
        deployment_type: lxc
        proxmoxServerAddress: 192.168.1.100
        proxmoxServerPort: 8006
        proxmoxApiTokenName: ansible
        proxmoxApiTokenSecret: "your-token-secret-uuid"
        proxmoxNodeName: pve-node-01
        proxmoxLxcOsTemplate: "local:vztmpl/ubuntu-24.04-
standard_24.04-2_amd64.tar.zst"
        proxmoxLxcDefaultStorage: local-lvm # 指定 rootfs 存储
      pve-node-02:
        deployment_type: lxc
        # ... 配置

    local_users:
      admin_user:
        name: Admin User
        public_key: "ssh-rsa AAAA..."

dns:
  hosts:
    site-dns01:
      ansible_host: 192.168.1.20
  vars:
    proxmox_interface: vmbro
    gateway: 192.168.1.1
    netmask: 255.255.255.0
    vlanid: 100 # 指定 VLAN
    proxmoxLxcCores: 2 # 指定 CPU 核心数
    proxmoxLxcMemoryMb: 4096 # 指定内存大小
    proxmoxLxcDiskSizeGb: 30 # 指定磁盘大小
    proxmoxLxcRootFsStorageName: local-lvm # 指定 rootfs 存储
    proxmoxLxcDefaultStorage
```



## □□□□

□□□□ LXC □□□□□□□□□□□□□□□□

- □□□□□□□□VLAN □□□LXC □ IP□
- □□/□□
- □□□□□□□□ `scsi0` / LXC `rootfs`□
- □□ `secondary_ips` □□□ NIC □□□□□□□□□□□□□□□□□□

## □□

□□□□□□□□□□ Ansible □□□ + `site_name`□

## □□□□□□

□□□□ LXC □□□□□□□□□□

```
mme01 → pve-node-01 (0 % 3 = 0)
mme02 → pve-node-02 (1 % 3 = 1)
mme03 → pve-node-03 (2 % 3 = 2)
mme04 → pve-node-01 (3 % 3 = 0)
```

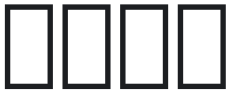
## □□□□□□

□□□□ LXC □□□□ `proxmoxLxcCores`□ `proxmoxLxcMemoryMb`□  
`proxmoxLxcDiskSizeGb`□ `proxmoxLxcRootFsStorageName`□  
`proxmoxLxc0sTemplate`□ `host_vm_network`□□□□□□□□□□□□□□□□

```
dns:
  hosts:
    site-dns01:
      ansible_host: 192.168.1.20
      proxmoxLxcDiskSizeGb: 120 # 120GB
      host_vm_network: vmbr1 # vmbr1
  vars:
    proxmox_interface: vmbr0
    gateway: 192.168.1.1
    netmask: 255.255.255.0
```

## [[ util\_playbooks/proxmox\_lxc.yml

```
proxmoxServerAddress/PROXMOX_API_TOKEN
services/proxmox.yml deployment_type: lxc
```



OmniCore `services/`



```
all.yml
├─ setup_users.yml
├─ apt_cache.yml
├─ dns.yml
├─ common.yml
├─ license_server.yml
├─ monitoring.yml
│   └─ grafana.yml
├─ epc.yml
│   ├── common.yml
│   ├── omnimme.yml
│   ├── omnisgwc.yml
│   ├── omnipgwc.yml
│   ├── upf.yml
│   ├── omnihss.yml
│   └─ omnidra.yml
├─ ims.yml
│   ├── pcscf.yml
│   ├── icscf.yml
│   ├── scscf.yml
│   ├── as.yml
│   ├── omnimessage.yml
│   ├── smsc.yml
│   └─ omnisep.yml
├─ omniepdg.yml
├─ omniss7.yml
├─ ocs.yml
├─ crm.yml
├─ ran_monitor.yml
└─ health_check.yml
```

omnimme.yml 50 all.yml  
--limit

□□□□

□□□□

□□	□□	□□
all.yml	□□□	□□□□□□□□ DNS□□□□ EPC□ IMS□ OCS□ CRM
epc.yml	□□□□□	MME□ SGW-C□ PGW-C□ UPF□ HSS□ DRA
ims.yml	□□/IMS	P/I/S-CSCF□□□□□□□ XCAP□□□□□
ocs.yml	□□	CGrateS□ KeyDB □□□ OCS □□
monitoring.yml	□□□□	Prometheus□ Grafana□□□□□□□ HOMER

# □□□□□□

□□	□□
<code>common.yml</code>	□□□□□□□□□□□□□□NTP□□□□□□
<code>setup_users.yml</code>	□□□□□□□□ SSH □□
<code>dns.yml</code>	DNS □□□□□□
<code>license_server.yml</code>	OmniCore □□□□□□
<code>netplan.yml</code>	□□□□□□
<code>firewall.yml</code>	iptables/nftables □□
<code>apt_cache.yml</code>	□□ APT □□□□□□□□

## EPC 配置

ファイル名	機能	説明
<code>omnimme.yml</code>	OmniMME	4G/LTE MME (4G)
<code>omnisgwc.yml</code>	OmniSGW-C	4G/LTE S-GW-C
<code>omnipgwc.yml</code>	OmniPGW-C	PDN 接続管理
<code>upf.yml</code> / <code>omniupf.yml</code>	OmniUPF	4G/LTE U-Plane (SGW-U/PGW-U)
<code>omnihss.yml</code> / <code>hss.yml</code>	OmniHSS	4G/LTE HSS
<code>omnidra.yml</code>	OmniDRA	Diameter 接続管理
<code>omnitwag.yml</code>	OmniTWAG	4G/LTE TWAG
<code>omniepdg.yml</code>	OmniEPDG	4G/LTE EPDG (WiFi 接続)
<code>gtp_proxy.yml</code>	GTP 代理	GTP 接続管理

# IMS 配置

ファイル名	機能	説明
<code>pcscf.yml</code>	P-CSCF	プロキシ-CSCF
<code>icscf.yml</code>	I-CSCF	インターネット-CSCF
<code>scscf.yml</code>	S-CSCF	セッション-CSCF
<code>as.yml</code>	OmniTAS	オムニTAS
<code>omnisep.yml</code>	OmniSEP	XCAP、XCAP-BSF、XCAP-URL
<code>omnimessage.yml</code>	OmniMessage	メッセージ IP
<code>smsc.yml</code>	SMSC	短メッセージサービスセンター

# 監視

ファイル名	監視対象
<code>ocs.yml</code>	ocs (CGateS + KeyDB)
<code>crm.yml</code>	crm
<code>omniss7.yml</code>	SS7/SIGTRAN
<code>homer.yml</code>	SIP/Diameter
<code>grafana.yml</code>	Grafana
<code>promtail.yml</code>	ログ Loki
<code>ran_monitor.yml</code>	RAN

## 📁📁📁📁

📁	📁
<code>proxmox.yml</code>	📁 Proxmox VE 📁📁📁📁
<code>proxmox_lxc.yml</code>	📁 LXC 📁📁📁📁/📁📁
<code>proxmox_delete.yml</code>	📁 Proxmox 📁📁/LXC

## 📁

📁	📁
<code>backup.yml</code>	📁📁📁📁📁📁
<code>reboot.yml</code>	📁📁📁📁
<code>shutdown.yml</code>	📁📁📁
<code>apt_update.yml</code>	📁📁📁📁
<code>apt_refresh_metadata.yml</code>	📁 APT 📁📁📁📁
<code>speedtest.yml</code>	📁📁📁📁📁📁

□□□□

□□	□□
restore_applicationserver.yml	□□□□ OmniTAS
restore_omnimessage_controller.yml	□□□□ OmniMessage
restore_smsc.yml	□□□□ SMSC

□□

□□□□□□

```
ansible-playbook -i hosts/customer/host_files/production.yml
services/all.yml
```

□□□□□□□□

```
# □□□□□□
ansible-playbook -i hosts/customer/host_files/production.yml
services/epc.yml

# □ IMS/□□
ansible-playbook -i hosts/customer/host_files/production.yml
services/ims.yml
```

## □□□□□□

```
# □□□ MME
ansible-playbook -i hosts/customer/host_files/production.yml
services/omnimme.yml

# □□□□□
ansible-playbook -i hosts/customer/host_files/production.yml
services/monitoring.yml
```

## □□□□□□

```
# □□□□□□□□□□ all.yml
ansible-playbook -i hosts/customer/host_files/production.yml
services/all.yml --limit mme01

# □□□□□□□□□□
ansible-playbook -i hosts/customer/host_files/production.yml
services/all.yml --limit "mme01,hss01"

# □□□□□□□□
ansible-playbook -i hosts/customer/host_files/production.yml
services/all.yml --limit mme
```

## □□□□

- □□□□ - □□□□□□□□
- □□□□□□ - □□□□□□□
- □□□□□□ - □□□□
- □□□□ - □□□□□□□□□□□□□□□□
- □□□□□□□□ - Grafana□Prometheus□□□□

# 目次

OmniCore 利用プレイブック `util_playbooks/` 目次

## 目次

ファイル名	説明
<code>proxmox.yml</code>	Proxmox のインストール
<code>proxmox_delete.yml</code>	Proxmox の削除/LXC
<code>proxmox_lxc.yml</code>	Proxmox の LXC の作成
<code>vmware.yml</code>	VMware vSphere のインストール
<code>vmware_delete.yml</code>	VMware vSphere の削除
<code>health_check.yml</code>	ヘルスチェック
<code>restore_hss.yml</code>	HSS のバックアップ/リストア
<code>restore_ocs.yml</code>	OCs KeyDB/MySQL のバックアップ
<code>ip_plan_generator.yml</code>	Mermaid の IP 計画生成
<code>get_ports.yml</code>	ポート情報の取得
<code>getLocalCapture.yml</code>	ローカルキャプチャ
<code>delete_local_user.yml</code>	ローカルユーザーの削除



```
# 実行
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/vmware.yml

# 実行制限
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/vmware.yml --limit mme

# 実行制限
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/vmware_delete.yml --limit old-host
```

実行

```
all:
  vars:
    vcenter_ip: "vcenter.example.com"
    vcenter_username: "administrator@vsphere.local"
    vcenter_password: "password"
    vcenter_datacenter: "Datacenter"
    vcenter_folder: "OmniCore"
    vcenter_vm_template: "ubuntu-2404-template"
  vhosts:
    esxi-01:
      vcenter_cluster_ip: "192.168.1.10"
      vcenter_datastore: "datastore1"
```

実行

実行 util\_playbooks/health\_check.yml

実行結果 OmniCore | OmniCall | HTML |

```
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/health_check.yml
```

実行 /tmp/health\_check\_YYYY-MM-DD HH:MM:SS.html

## Table 1

Component	Description
OmniHSS	OmniHSS Diameter server
OmniDRA	Diameter server
OmniTAS	OmniTAS CPU server
OCS	KeyDB server

## HSS Backup

Run `util_playbooks/restore_hss.yml`

OmniHSS backup and restore

```
ansible-playbook -i hosts/customer/host_files/production.yml  
util_playbooks/restore_hss.yml
```

## Table 2

File	Path	Location
hss_dump_<hostname>_<timestamp>.sql	omnihss	MySQL
hss_<hostname>_<timestamp>.tar.gz	/etc/omnihss	

# OCS 復旧

コマンド `util_playbooks/restore_ocs.yml`

このコマンドは OCS (CGrates) のバックアップを復旧し、KeyDB を再構築します。

```
ansible-playbook -i hosts/customer/host_files/production.yml  
util_playbooks/restore_ocs.yml
```

## 手順

1. OCS を停止し、CGrates と KeyDB をバックアップする。
2. AOF をバックアップする。
3. KeyDB RDB をバックアップし、KeyDB を再構築する。
4. MySQL StoreDB をバックアップする。
5. `/etc/cgrates` をバックアップする。
6. OCS を再起動する。

## バックアップファイル

名前	ファイル名	説明
KeyDB DataDB	<code>keydb_dump_&lt;hostname&gt;_&lt;timestamp&gt;.rdb</code>	KeyDB RDB ファイル
MySQL StoreDB	<code>cgrates_dump_&lt;hostname&gt;_&lt;timestamp&gt;.sql</code>	cgrates の MySQL データ
設定	<code>cgrates_&lt;hostname&gt;_&lt;timestamp&gt;.tar.gz</code>	<code>/etc/cgrates</code> のバックアップ



# 0000

000 util\_playbooks/get\_ports.yml

000000000000000000000000

```
ansible-playbook -i hosts/customer/host_files/production.yml  
util_playbooks/get_ports.yml
```

# 0000

00	00
/tmp/all_ports.csv	000000IP000000000000 CSV
./open_ports.rst	00 Sphinx 000 reStructuredText 0

# 00000

00	0???
000	00000
IP	000 ansible_host IP 00
IP 00	IPv4 0 IPv6
00	TCP 0 UDP
00	00000
00	0000

## 000000

000 util\_playbooks/getLocalCapture.yml

000000 /etc/localcapture 00000000000000000000

```
ansible-playbook -i hosts/customer/host_files/production.yml  
util_playbooks/getLocalCapture.yml
```

000 ./localCapturePcaps/<hostname>/\*.pcap

## 0000

000 util\_playbooks/delete\_local\_user.yml

00000000000000000000

```
ansible-playbook -i hosts/customer/host_files/production.yml  
util_playbooks/delete_local_user.yml
```

000 0000000000000000

## 000000000

## 0000

```
ansible-playbook -i <inventory_file> util_playbooks/<playbook>.yml
```

## オプション

オプション	説明
<code>-i &lt;inventory&gt;</code>	インベントリファイル
<code>--limit &lt;hosts&gt;</code>	実行するホストの制限
<code>-v / -vv / -vvv</code>	verbosity (静か / 普通 / 多量)
<code>--check</code>	実行せずに差分を確認する
<code>--diff</code>	差分を確認する

## 例

```
# インベントリファイル指定
ansible-playbook -i hosts/acme/host_files/production.yml
util_playbooks/health_check.yml

# ホスト制限 HSS
ansible-playbook -i hosts/acme/host_files/production.yml
util_playbooks/restore_hss.yml --limit hss01

# 多量 Verbosity IP 計画生成器
ansible-playbook -i hosts/acme/host_files/production.yml
util_playbooks/ip_plan_generator.yml -v
```

