5G Core (5GC)

يتم نشر النواة المستقلة 5G (SA) بواسطة `services/5gc.yml` playbook. يقوم دور مشترك `5gc_nf`، واحد لكل 5 NFs. تشرح هذه الصفحة هذا الدور، والتكوين الذي يقرأه كل واحد منها، و التي يبنيها.

نظرة عامة

دور واحد (`5gc_nf`) لجميع الـ NFs في 5GC. يستدعي playbook الدور مرة واحدة لكل NF ويمرر متغير `nf_package` الذي يحدد حزمة Debian، خدمة systemd، وقالب تكوين Jinja2. هذا يحافظ على تثبيت، إصدار التكوين، الترخيص، ومنطق إعادة التشغيل متطابقًا عبر كل NF.

```
- name: Setup AMF
  hosts: amf
  roles:
    - {role: 5gc_nf, nf_package: omniamf, become: yes}
```

وظائف الشبكة

كل NF له مجموعة جرد، حزمة، خدمة systemd، وقالب تكوين خاص به (`roles/5gc_nf/templates/<nf_package>.j2`).

NF	`nf_package`	مجموعة الجرد	قالب التكوين	الدور
وظيفة مستودع الشبكة	`omninrf`	`nrf`	`omninrf.j2`	اكتشاف الخدمة / تسجيل NF
إدارة الوصول والتنقل	`omniamf`	`amf`	`omniamf.j2`	إشارات N1/N2، التسجيل، إدارة الاتصال
خادم المصادقة	`omniausf`	`ausf`	`omniausf.j2`	مصادقة 5G-AKA / EAP-AKA'
وظيفة دعم الربط	`omnibsf`	`bsf`	`omnibsf.j2`	ربط PCF لجلسة PDU
وظيفة الشحن	`omnichf`	`chf`	`omnichf.j2`	شحن متقارب عبر الإنترنت / غير متصل
اختيار شريحة الشبكة	`omnissf`	`nssf`	`omnissf.j2`	اختيار NSSAI / الشريحة
وظيفة التحكم في السياسة	`omnipcf`	`pcf`	`omnipcf.j2`	قواعد السياسة وجودة الخدمة
وكيل الاتصال بالخدمة	`omniscp`	`scp`	`omniscp.j2`	توجيه رسائل SBI / الاتصالات غير المباشرة
وظيفة إدارة الجلسات	`omnismf`	`smf`	`omnismf.j2`	جلسة PDU والتحكم في UPF (N4)
إدارة البيانات الموحدة	`omniudm`	`udm`	`omniudm.j2`	بيانات الاشتراك، فك تشفير SUCI

NF	`nf_package`	مجموعة الجرد	قالب التكوين	الدور
مستودع البيانات الموحد	`omniudr`	`udr`	`omniudr.j2`	تخزين بيانات داعم لـ UDM/PCF

5 كيفية عمل دور `5gc_nf`

لكل NF، يقوم الدور بنفس التسلسل:

1. **حل `group_vars_omnicore`**: يتم تعيينه إلى `<inventory>/group_vars/`، بحيث يتم حل عمليات البحث عن التكوين بالنسبة إلى الجرد النشط.
2. **اكتشاف خوادم الترخيص**: إذا لم يتم تعيين `license_server_api_urls`، يتم اشتقاقه من مجموعة جرد `license_server` (`https://<host>:8443/api`).
3. **تثبيت المتطلبات الأساسية**: `lsb-release`، `curl`، `gpg`، `unzip`، `libglu1-mesa`، `libsctp1`.
4. **تثبيت حزمة NF** من مستودع Omnitouch APT. الأحدث بشكل افتراضي؛ قم بتثبيته مع `nf_version` (انظر تثبيت الإصدار).
5. **تكوين القالب (المُعَلم)**: يقوم بتجسيد `templates/<nf_package>.j2` إلى `/etc/<nf_package>/runtime-<deb_version>.exs`.
6. **إنشاء رابط رمزي لـ `runtime.exs`** → تكوين الإصدار النشط، ونسخ ذلك الرابط الرمزي إلى كل دليل `/opt/<nf_package>/releases/*`.
7. **إعادة تشغيل NF** (المعالج) كلما تغير التكوين النشط.

تثبيت الإصدار والعودة

يتم كتابة التكوين لكل إصدار Debian (`runtime-<deb_version>.exs`) و`runtime.exs` هو رابط رمزي للإصدار النشط. يتم الاحتفاظ بالتكوينات السابقة على القرص، لذا فإن العودة هي مجرد إعادة تثبيت إصدار الحزمة الأقدم:

```
- {role: 5gc_nf, nf_package: omniamf, nf_version: "1.2.2-1",
become: yes}
```

تجاهل `nf_version` لتتبع أحدث حزمة.

مفاتيح HNET لـ UDM (فك تشفير SUCI)

يمتلك UDM (`nf_package: omniudm`) خطوات إضافية لدعم فك تشفير SUCI (TS 33.501 §6.12). يستعيد فك التشفير SUPI من SUCI المخفية التي يقدمها UE عند التسجيل. يحتاج إلى مفاتيح الشبكة المنزلية **السرية** التي تتزاوج مع المفاتيح العامة للشبكة المنزلية على USIMs.

- تعيش المفاتيح في `group_vars/hnet_udm/` على وحدة التحكم كـ Ansible `<curve>-<id>.key` (PEM)، حيث `<curve>` هو `curve25519` (Profile A، X25519) أو `secp256r1` (Profile B، prime256v1)، و`<id>` هو معرف المفتاح العام للشبكة المنزلية.
- افتراضيًا بحيث يعمل فك `curve25519-1.key` إذا كان الدليل **فارغًا**، يتم إنشاء التشفير من الصندوق. هذه الخطوة غير متغيرة. لا يتم الكتابة فوق مفتاح موجود.
- لكل `curve25519-*.key`، يتم تصدير القيمة العامة الخام المطابقة 32 بايت (hex) إلى ملف `.pub` لتوفير USIM. تبقى ملفات `.pub` فقط في الجرد؛ يتم `*.key` نشرها إلى `/etc/omniudm/hnet/` على UDM (الجذر، `0600` الوضع).

> المفتاح الذي تم إنشاؤه تلقائيًا عشوائي ويقوم فقط بفك تشفير SUCIs من USIMs المبرمجة بمفتاحها العام `.pub` على المنزلية الشبكة المصدرة وقدمها كمفتاح عام **المطابق**. خذ ويمكنها ترك HNET التي تستخدم المخطط الفارغ إلى مفتاح SIM. لا تحتاج بطاقات USIM الدليل فارغًا.

تفاصيل كاملة وأمثلة تخطيط الملفات: المثال 8 تكوين المتغيرات الجماعية.

الوثائق ذات الصلة

- Service Playbooks - جميع playbooks الخدمة وهيكل النشر
- مجال - النواة الدائرية CS القديم (BSC، MSC، SS7)
- قوالب تكوين لكل - تكوين المتغيرات الجماعية NF ومفاتيح HNET
- كيف تتناسب المجالات معًا - معمارية النشر
- **تكوين OmniCore**: https://docs.omnitouch.com.au/docs/repos/OmniCore - ما يتم تضمينه في ملفات تكوين NF

معيار تخطيط IP لـ OmniCore

نظرة عامة

توضح هذه الوثيقة نهج تخطيط IP القياسي لنشر OmniCore. تتطلب البنية المعمارية **أربعة** **شبكات فرعية داخلية** لتقسيم وظائف الشبكة بشكل صحيح من أجل الأمان والأداء ووضوح العمليات.

متطلبات تخصيص IP

التخصيص القياسي: أربع شبكات فرعية /24

يتطلب كل نشر لـ OmniCore أربع شبكات فرعية متميزة للتواصل الداخلي:

1. **شبكة النواة الحزمية** - أول /24
2. **شبكة الإشارة** - ثاني /24
3. **شبكة IMS الداخلية** - ثالث /24
4. **شبكة IMS العامة** - رابع /24

مهم: هذه توصيات، وليست متطلبات

تخصيص الشبكات الفرعية الموضح في هذه الوثيقة هو **أفضل ممارسة موصى بها** لتنظيم نشرات OmniCore. ومع ذلك، فإن البنية المعمارية **مرنة تمامًا**:

- **جميع المضيفين في شبكة فرعية واحدة**: يمكنك وضع جميع المكونات في شبكة فرعية واحدة إذا كان ذلك يناسب احتياجات نشراتك
- **كل نوع مضيف في شبكته الفرعية الخاصة**: يمكنك إنشاء شبكات فرعية منفصلة لكل نوع من المكونات (واحدة لـ MMEs، واحدة لـ HSS، إلخ)
- **تجميعات مخصصة**: يمكنك تنظيم المضيفين في أي هيكل شبكة فرعية يتناسب مع متطلباتك المحددة

- **مزج IPs الداخلية والعامة**: يمكن لبعض المضيفين استخدام عناوين داخلية (RFC 1918) بينما يستخدم الآخرون IPs عامة، كل ذلك ضمن نفس النشر

يوفر نهج الشبكات الفرعية الأربعة الموصى به **عزل أمني، إدارة حركة المرور، ووضوح عمليات** IP مثالي، ولهذا السبب نقترحه للنشرات الإنتاجية. ومع ذلك، يجب عليك تكييف خطة لتناسب طوبولوجيا الشبكة الخاصة بك، ومساحة العناوين المتاحة، ومتطلبات التشغيل.

تحليل تقسيم الشبكة

1. شبكة النواة الحزمية (أول /24)

الغرض: عناصر خطة المستخدم وخطة التحكم الأساسية

المكونات:

- OmniMME (كيان إدارة التنقل)
- OmniSGW (البوابة الخدمية)
- OmniPGW-C (خطة التحكم في بوابة PDN)
- OmniUPF/PGW-U (وظيفة خطة المستخدم / بوابة PDN)

مثال: `10.179.1.0/24`

```
mme:
  hosts:
    customer-mme01:
      ansible_host: 10.179.1.15
      gateway: 10.179.1.1
      host_vm_network: "v400-omni-packet-core"
```

---

2. شبكة الإشارة (ثاني /24)

الغرض: إشارات قطرية، سياسة، وظائف شحن وإدارة

المكونات:

- OmniHSS (خادم المشتركين المنزلي)

- OmniCharge OCS (نظام الشحن عبر الإنترنت)
- OmniHSS PCRF (وظيفة قواعد السياسة والشحن)
- OmniDRA DRA (وكيل توجيه قطرية)
- خوادم DNS (DNS داخلي/بنية تحتية - متميز عن DNS الموجه للعملاء في قسم IMS العامة؛ كلاهما مقصود)
- خوادم TAP3/CDR
- المراقبة/OAM
- التقاط SIP
- خادم الترخيص
- مراقب RAN
- رابط تحذير Omnitouch CBC (مركز بث الخلايا) - إذا تم نشره

مثال: `10.179.2.0/24`

```
hss:
  hosts:
    omni-site-hss01:
      ansible_host: 10.179.2.140
      gateway: 10.179.2.1
      host_vm_network: "v401-omni-signalling"
```

---

3. شبكة IMS (24/ ثالث) الداخلية

الغرض: إشارات وخدمات IMS الأساسية (إشارات SIP الداخلية)

المكونات:

- OmniCSCF S-CSCF (وظيفة التحكم في جلسة المكالمات الخدمية)
- OmniCSCF I-CSCF (وظيفة التحكم في جلسة المكالمات الاستجوابية)
- OmniTAS (خادم تطبيقات الهاتف / خادم التطبيقات)
- OmniMessage (وحدة التحكم في SMS، SMPP، IMS)

مثال: `10.179.3.0/24`

```
scscf:
  hosts:
    omni-site-scscf01:
      ansible_host: 10.179.3.45
      gateway: 10.179.3.1
      host_vm_network: "v402-omni-ims-internal"
```

---

4. شبكة IMS العامة (رابع /24)

الغرض: خدمات IMS الموجهة للمستخدم، DNS والتوصيل للعملاء، SS7/القديمة الموجه

المكونات:

- OmniCSCF P-CSCF (وظيفة التحكم في جلسة المكالمات الوكيلة)
- خوادم XCAP
- خوادم البريد الصوتي المرئي
- DNS العملاء
- OmniSS7 STP (نقطة نقل إشارات SS7)
- OmniSS7 HLR (سجل الموقع المنزلي) - 2 لـ 3G/2G
- OmniSS7 IP-SM-GW (MAP SMSc)
- OmniSS7 بوابة CAMEL
- MSC (مركز التحويل المحمول)

> **ملاحظة:** عادةً ما توجد وظائف SS7/HLR، والتوصيل القديمة ذات الصلة في القسم العام. حيث تتصل بشبكات 2G/3G الخارجية عبر SIGTRAN/M3UA بدلاً من SIP IMS الداخلية.

مثال: `10.179.4.0/24`

```
pcscf:
  hosts:
    omni-site-pcscf01:
      ansible_host: 10.179.4.165
      gateway: 10.179.4.1
      host_vm_network: "v403-omni-ims-public"
```

طرق التنفيذ

يدعم OmniCore طريقتين رئيسيتين لتنفيذ هذا التقسيم الشبكي:

الطريقة 1: واجهات الشبكة الفيزيائية/الافتراضية (موصى بها للإنتاج)

استخدم NICs منفصلة أو جسور افتراضية لكل قسم شبكة. يوفر هذا أقوى عزل وهو النهج الموصى به للنشرات الإنتاجية. القاعدة هي تسمية كل جسر بعد قسمه (`v400-omni-packet-core`، `v401-omni-signalling`، `v402-omni-ims-internal`، `v403-omni-ims-public`)، صالح أيضًا حيث يتم التعامل مع (مثل `vmbr0`) على الرغم من أن جسر مشترك واحد التقسيم في مكان آخر.

مثال:

```yaml
# النواة الحزمية - v400-omni-packet-core
mme:
  hosts:
    omni-site-mme01:
      ansible_host: 10.179.1.15
      gateway: 10.179.1.1
      host_vm_network: "v400-omni-packet-core"

# الإشارة - v401-omni-signalling
hss:
  hosts:
    omni-site-hss01:
      ansible_host: 10.179.2.140
      gateway: 10.179.2.1
      host_vm_network: "v401-omni-signalling"

# IMS الداخلية - v402-omni-ims-internal
icscf:
  hosts:
    omni-site-icscf01:
      ansible_host: 10.179.3.55
      gateway: 10.179.3.1
      host_vm_network: "v402-omni-ims-internal"

# IMS العامة - v403-omni-ims-public
pcscf:
  hosts:
    omni-site-pcscf01:
      ansible_host: 10.179.4.165
      gateway: 10.179.4.1
      host_vm_network: "v403-omni-ims-public"
```

الطريقة 2: تقسيم قائم على VLAN

استخدم واجهة فيزيائية واحدة مع وسم VLAN لفصل الشبكات. هذا مناسب للنشرات الأصغر أو عندما تكون NICs الفيزيائية محدودة.

مثال:

```
# كل قسم يستخدم وسم VLAN ؛ جميع المكونات تشترك في جسر واحد (ovsbr1)
مختلف.
# applicationserver (TAS) هو مكون IMS داخلي -> VLAN 402 / 10.178.3.x
applicationserver:
  hosts:
    omni-site-sbc01:
      ansible_host: 10.178.3.213
      gateway: 10.178.3.1
      host_vm_network: "ovsbr1"
      vlanid: "402"

# dra و dns هما مكونات الإشارة -> VLAN 401 / 10.178.2.x
dra:
  hosts:
    omni-site-dra01:
      ansible_host: 10.178.2.211
      gateway: 10.178.2.1
      host_vm_network: "ovsbr1"
      vlanid: "401"

dns:
  hosts:
    omni-site-dns01:
      ansible_host: 10.178.2.178
      gateway: 10.178.2.1
      host_vm_network: "ovsbr1"
      vlanid: "401"
```

تكوين الشبكة:

- قم بتكوين VLANs على المفتاح الفيزيائي
- وسم حركة المرور بشكل مناسب على مستوى المحاكي
- قم بتوجيه حركة المرور بين VLANs عند البوابة/جدار الحماية

مثال على تخطيط VLAN:

```
VLAN 400: 10.x.1.0/24 (النواة الحزمية)
VLAN 401: 10.x.2.0/24 (الإشارة)
VLAN 402: 10.x.3.0/24 (IMS الداخلية)
VLAN 403: 10.x.4.0/24 (IMS العامة)
```

العامة IP العمل مع عناوين

نظرة عامة

تتطلب العديد من نشرات OmniCore أن تحتوي بعض المكونات على عناوين IP عامة للاتصال الخارجي، مثل:

- **DRA** - ل��شارات قطرية التجوال مع شركات النقل الخارجية
- **SGW/PGW** **للتجوال** - لحركة مرور GTP من الشركاء المتجولين
- **ePDG** - لمكالمات WiFi (نفق IPsec من UEs)
- **بوابة SMSC** - لاتصالات SMPP مع مجمعات SMS الخارجية
- **P-CSCF** (في بعض النشرات) - للتسجيل المباشر لـ UE SIP

كيفية تخصيص IPs العامة

تُعالج IPs **الداخلية** في ملفات جرد **IPs** العامة **بنفس الطريقة تمامًا كما يتم التعامل مع** العام في حقل IP المضيفين لديك. ببساطة حدد عنوان `ansible_host` مع البوابة المناسبة وقناع الشبكة.

مثال: IPs العامة للتجوال مع SGW/PGW

```yaml
sgw:
  hosts:
    # SGWs الداخلية على الشبكة الخاصة
    customer-site-sgw01:
      ansible_host: 10.4.1.25
      gateway: 10.4.1.1
      host_vm_network: "v400-omni-packet-core"

    # SGWs للتجوال مع IPs العامة
    customer-site-roaming-sgw01:
      ansible_host: 203.0.113.10
      gateway: 203.0.113.9
      netmask: 255.255.255.248       # /29 subnet
      host_vm_network: "498-public-servers"
      in_pool: False
      cdrs_enabled: True

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

مثال: DRA مع IP عام

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

مثال: ePDG مع IP عام

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

مزج IPs الداخلية والعامة

من الشائع وجود مزيج من IPs الداخلية والعامة ضمن نفس مجموعة المكونات. على سبيل المثال:

- GTP الداخلية للمواقع المحلية باستخدام SGWs
- العامة خصيصًا لحركة مرور التجوال من شركات النقل الخارجية SGWs
- الداخلية والخارجية SGWs كل من PGW-C يمكن أن تدير نفس

تتعامل بنية OmniCore مع هذا ب��لاسة - فقط قم بتكوين كل مضيف مع عناوين IP المناسبة له.

مقدمة في نشر Ansible في Omnitouch

نظرة عامة

كمنصة لأتمتة البنية التحتية لنشر حلول الشبكة Omnitouch Ansible تستخدم خدمات شبكة (4G/5G) الخلوية الكاملة بطريقة متسقة وقابلة للتكرار وأوتوماتيكية. توفر هذه الوثيقة نظرة عامة حول كيفية استغلالنا لـ Ansible لتنظيم نشرات الاتصالات المعقدة.

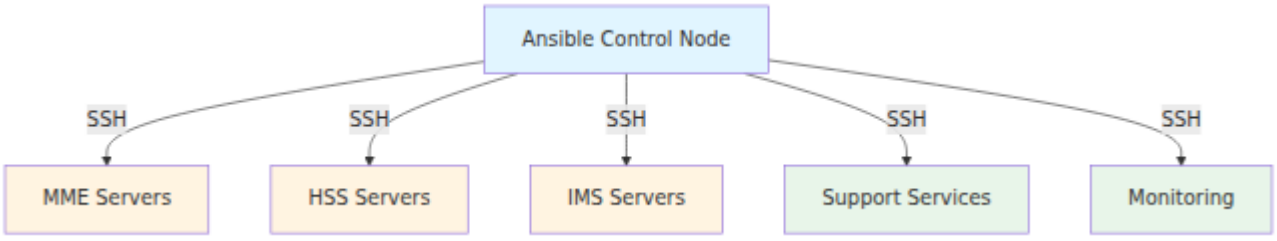
ما هو Ansible؟

Ansible هو أداة أتمتة مفتوحة المصدر تتيح لك:

- تكوين الأنظمة
- نشر البرمجيات
- تنظيم سير العمل المعقد
- إدارة البنية التحتية ككود

من أنها Ansible نهجًا تصريحيًا - حيث تصف **الحالة المرغوبة** لأنظمتك، ويتأكد Ansible يستخدم تصل إلى تلك الحالة.

كيف تستخدم Omnitouch Ansible



المفاهيم الأساسية

1. الجرد (ملفات المضيفين)

يحدد **ما** الأنظمة التي يجب إدارتها. تحتوي كل نشرية عميل على ملف مضيف يصف:

- جميع الآلات الافتراضية في الشبكة
- عناوين IP الخاصة بها
- تكوين الشبكة
- معلمات محددة للخدمة

ملفات المضيفين هي ما ستعمل معه لتعريف شبكتك.

انظر: تكوين ملف المضيفين

2. الأدوار

يحدد **كيف** يتم تكوين كل مكون. الأدوار هي وحدات قابلة لإعادة الاستخدام تحتوي على:

- مهام (خطوات للتنفيذ)
- قوالب (قوالب ملفات التكوين)
- معالجات (إجراءات يتم تشغيلها بواسطة التغييرات)
- متغيرات (قيم التكوين الافتراضية)

أمثلة على الأدوار لمكونات OmniCore: `omnihss`, `omnisgwc`, `omnipgwc`, `omnidra`, إلخ

تحدد فريق ONS هذه الأدوار. يمكنك تعديلها. ومع ذلك، هناك عمومًا طرق أنظف لإجراء أي تعديلات تحتاجها من داخل ملف المضيفين الخاص بك.

3. دفاتر التشغيل

تنظم **متى** و **أين** يتم تطبيق الأدوار:

```
- name: Setup Omnicore MME
  hosts: mme
  roles:
    - omnimme
```

نستخدم هذه بشكل أساسي كمجموعات للأدوار.

في الممارسة العملية، يطبق كل دفتر تشغيل خدمة دورًا واحدًا، ويتم سحب الإعداد المشترك لتشغيل `import_playbook` تستخدم `services/epc.yml` بشكل منفصل. حزمة الخدمة مثل `common.yml` أولاً ثم كل دفتر تشغيل خاص بالخدمة:

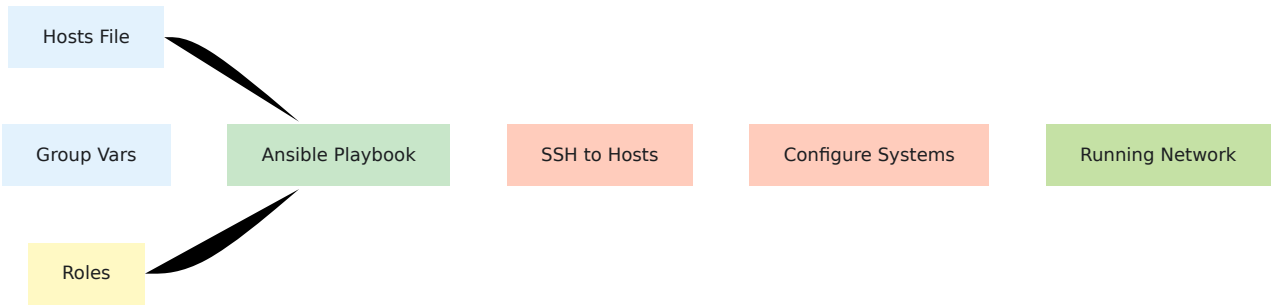
```yaml
- name: Install Common
  import_playbook: common.yml

- name: Run MME Playbook
  import_playbook: omnimme.yml
```

4. متغيرات المجموعة

يوفر **تكوينًا محددًا للعميل** يتجاوز القيم الافتراضية للدور. هنا يحدث تخصيص العميل دون تعديل الأدوار الأساسية.

انظر: متغيرات المجموعة والتكوين

هندسة النشر



عملية النشر

1. تعريف البنية التحتية

إنشاء ملف مضيف يصف طوبولوجيا شبكتك:

ملاحظة تخطيط: قبل تعريف البنية التحتية، راجع معيار تخطيط IP للحصول على إرشادات حول تقسيم الشبكة، وتخصيص عناوين IP، وتنظيم الشبكات الفرعية.

مستخدمو Proxmox: إذا كنت تنشر على Proxmox، انظر نشر VM/LXC على Proxmox لتوفير VMs/حاويات بشكل أوتوماتيكي.

انظر: تكوين ملف المضيفين و مرجع التكوين

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

2. تخصيص التكوين

تعيين المتغيرات المحددة للعميل في `group_vars`:

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

#ToDo - إضافة رابط هنا إلى مرجع التكوين للقائمة الكاملة

3. تشغيل دفاتر التشغيل

نشر الشبكة:

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

4. النشر الآلي

سوف يقوم Ansible بـ:

- إنشاء/توفير VMs (إذا كنت تستخدم تكامل Proxmox/VMware)
- تكوين الشبكات
- تثبيت حزم البرمجيات من ذاكرة التخزين المؤقت APT
- نشر كود التطبيق
- تكوين الخدمات بإعدادات العميل
- بدء الخدمات
- التحقق من النشر

المكونات الرئيسية التي نقوم بنشرها

OmniCore (منصة النواة الحزمية 4G/5G)

- **OmniHSS** - خادم المشتركين المنزلي
- **OmniSGW** - بوابة الخدمة (طبقة التحكم)
- **OmniPGW** - بوابة الحزم (طبقة التحكم)
- **OmniUPF** - وظيفة طبقة المستخدم
- **OmniDRA** - وكيل توجيه Diameter
- **OmniTWAG** - بوابة الوصول اللاسلكي الموثوق

انظر: https://docs.omnitouch.com.au/docs/repos/OmniCore

OmniCall (منصة الصوت والرسائل)

- **OmniCall CSCF** - وظيفة التحكم في جلسة المكالمات (P-CSCF، I-CSCF، S-CSCF)
- **OmniTAS** - خادم تطبيق IMS (خدمات VoLTE/VoNR)
- **OmniMessage** - مركز الرسائل القصيرة (SMS-C)
- **OmniMessage SMPP** - دعم بروتوكول SMPP
- **OmniSS7** - مكونات الإشارة SS7 (STP، HLR، CAMEL)
- **VisualVoicemail** - وظيفة البريد الصوتي

انظر: https://docs.omnitouch.com.au/docs/repos/OmniCall

OmniCharge/OmniCRM

- **منصة CRM** - إدارة علاقات العملاء، التسجيل الذاتي، الفوترة

انظر: https://docs.omnitouch.com.au/docs/repos/OmniCharge

خدمات الدعم

- **DNS** - حل DNS الشبكة
- **خادم الترخيص** - إدارة الترخيص

- **المراقبة** - Prometheus، Grafana

انظر: نظرة عامة على هندسة النشر

إدارة الحزم

نستخدم نموذج توزيع حزم هجين:

حزم APT مسبقة التجميع

توزع جميع برامج Omnitouch كحزم Debian (`.deb` files):

- تم بناؤها من المصدر في خط أنابيب CI/CD لدينا
- تم إصدارها واختبارها
- مستضافة على مستودعات الحزم

نظام ذاك��ة التخزين المؤقت APT

يمكن للعملاء الاختيار بين:

1. **ذاكرة التخزين المؤقت المحلية APT** - مرآة للحزم المطلوبة في الموقع للنشر غير المتصل
2. **المستودع العام** - الوصول المباشر إلى مستودع الحزم المستضاف من قبل Omnitouch

يوجد أيضًا `omnicore` موحد `.deb` (مبني من `packaging/`) الذي يجمع دفاتر التشغيل والأدوات لتوفير نشر ذاتي.

انظر: نظام ذاكرة التخزين المؤقت APT

إدارة الترخيص

تتطلب جميع مكونات برامج Omnitouch تراخيص صالحة يتم إدارتها من خلال خادم ترخيص مركزي:

- تتحقق المكونات من صلاحية الترخيص عند بدء التشغيل

- يتم تمكين/تعطيل الميزات بناءً على الترخيص
- يمكن أن يكون خادم الترخيص محليًا أو مستضافًا في السحابة

انظر: خادم الترخيص

فوائد هذا النهج

القابلية للتكرار

يمكن أن تقوم نفس دفاتر تشغيل Ansible بنشر:

- مختبرات التطوير
- بيئات الاختبار
- الشبكات الإنتاجية
- مواقع العملاء

الاتساق

يستخدم كل نشر نفس التكوينات المختبرة، مما يقلل من الأخطاء البشرية.

التحكم في الإصدار

يتم تعريف البنية التحتية ككود في Git:

- تتبع جميع التغييرات
- مراجعة قبل النشر
- التراجع إذا لزم الأمر

التخصيص بدون تعقيد

يمكن للعملاء تخصيص نشرهم من خلال `group_vars` دون تعديل الأدوار الأساسية.

النشر السريع

نشر شبكة خلوية كاملة في ساعات بدلاً من أيام أو أسابيع.

البدء

المتطلبات المسبقة

تحتاج إلى تثبيت التبعيات المطلوبة. يستخدم هذا المشروع، Ansible قبل تشغيل دفاتر تشغيل Python، **يتطلب** لذا **uv** لإدارة بيئة.

1. تثبيت uv

إذا لم يكن uv مثبتًا بالفعل، قم بتثبيته باستخدام أي طريقة تناسب عقد التحكم الخاص بك. انظر وثائق تثبيت uv. على سبيل المثال:

```
pipx install uv                                    # عبر pipx
curl -LsSf https://astral.sh/uv/install.sh | sh    # مثبت مستقل
```

2. تثبيت التبعيات

من جذر المستودع، قم بتشغيل:

```
uv sync
```

هذا ينشئ `.venv/` معزولة ويثبت Ansible بالإضافة إلى جميع حزم Python اللازمة (المثبتة في `uv.lock`) لأتمتة نشر Omnitouch.

3. تفعيل البيئة

قم بتفعيل البيئة الافتراضية، واحتفظ بها مفعلة لجميع أوامر Ansible:

```
source .venv/bin/activate      # Windows: .venv\Scripts\activate
```

4. تثبيت مجموعات Ansible

قم بتثبيت مجموعات Ansible التي تعتمد عليها دفاتر التشغيل (من `requirements.yml`):

```
ansible-galaxy collection install -r requirements.yml
```

5. تشغيل أوامر Ansible

مباشرة Ansible مع تفعيل البيئة، قم بتشغيل أوامر:

```
ansible --version
```

ملاحظة: قم بإلغاء تفعيل البيئة عند الانتهاء عن طريق تشغيل `deactivate`.

خطوات النشر

1. مراجعة تكوين ملف المضيفين لفهم كيفية تعريف شبكتك
2. تعلم عن متغيرات المجموعة للتخصيص
3. فهم نظام ذاكرة التخزين المؤقت APT لإدارة الحزم
4. مراجعة هندسة النشر لرؤية كيفية تناسق كل شيء
5. انشر!

الخطوات التالية

- **خطط لهيكل شبكتك وتخصيص IP** - معيار تخطيط IP
- تكوين ملف المضيفين - تعلم كيفية تعريف طوبولوجيا شبكتك
- فهم توزيع الحزم - نظام ذاكرة التخزين المؤقت APT
- خادم الترخيص - تعلم عن إدارة الترخيص
- نظرة عامة على هندسة النشر - رؤية الصورة الكاملة
- تكوين متغيرات المجموعة - تخصيص نشر الخاص بك
- دفاتر التشغيل المساعدة - أدوات تشغيلية لفحص الصحة، النسخ الاحتياطي، والصيانة

مستودع APT وتوزيع الحزم

نظرة عامة

يوفر نظام APT الخاص بـ Omnitouch توزيع الحزم لجميع عمليات النشر. يتم تقديم نوعين من المحتوى:

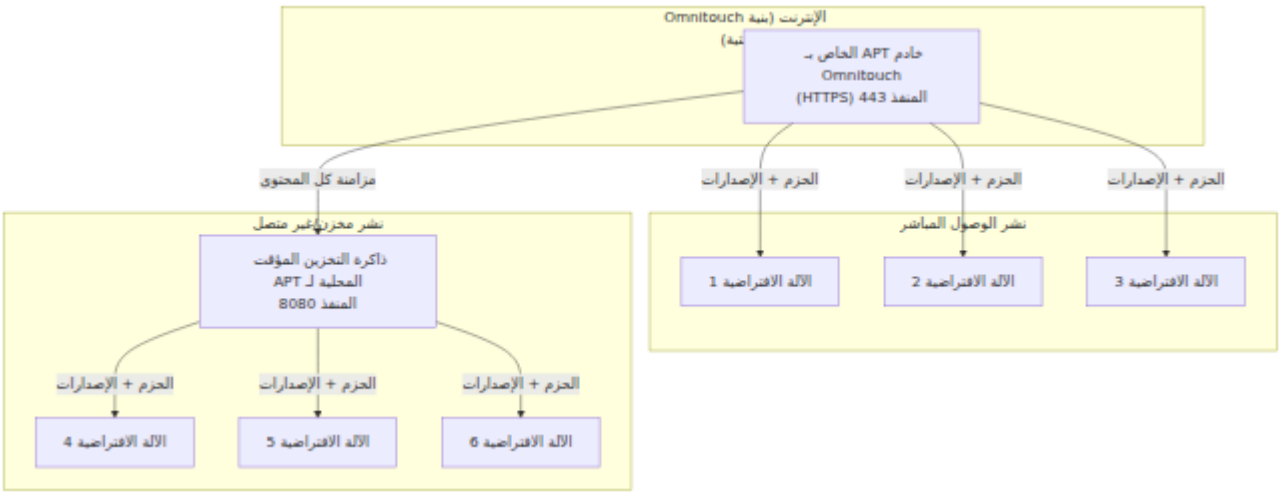
1. **حزم APT**: حزم Debian المثبتة عبر `apt install`
2. **الإصدارات الثنائية**: ثنائيات مسبقة البناء يتم تنزيلها مباشرة (مصدري Prometheus، الوكلاء، إلخ)

تتضمن صفحة الحزم على خادم APT **الإعداد الذاتي** أيضًا علامة **(OmniCore)**، والتي توفر حزمة `omnicore` (المقدمة في `/pool/main/o/omnicore/`) التي تحتوي على مجموعة أدوات الأتمتة الكاملة باستخدام Ansible للعملاء الذين يرغبون في نشر وإدارة شبكتهم بأنفسهم.

يدعم النظام نموذجين للنشر:

1. **الوصول المباشر**: تسحب الآلات الافتراضية الحزم مباشرة من خادم APT الخاص بـ Omnitouch (مثل `apt.<region>.omnitou.ch`)
2. **مرآة التخزين المحلية**: يقوم خادم محلي بمزامنة المحتوى من Omnitouch ويقدم الحزم للآلات الافتراضية (للنشر غير المتصل/المعزول)

الهيكلية



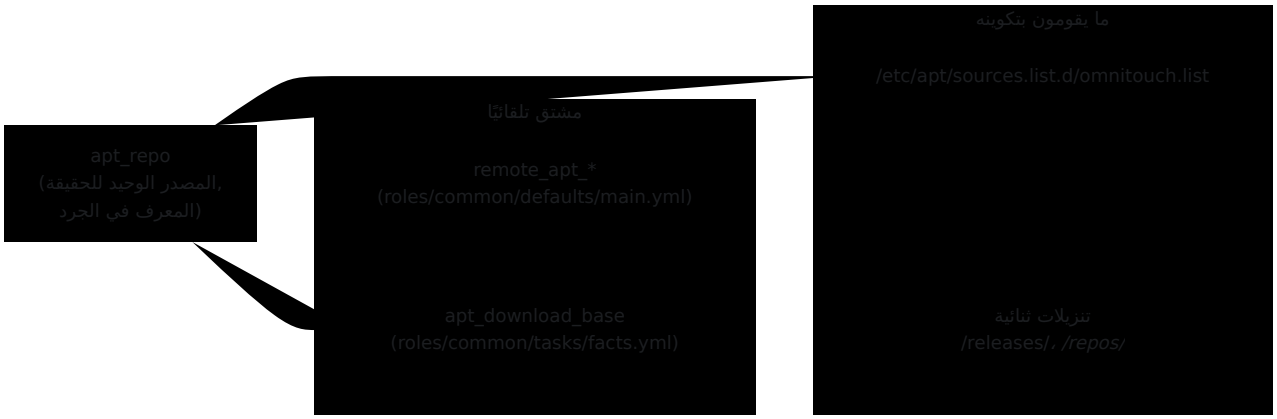
المحتوى المقدم

كل المحتوى المطلوب للنشر APT يستضيف خادم:

نوع المحتوى	الوصف	المسار
حزم Omnitouch	حزم `.deb` مخصصة (omnihss، omnimme، إلخ)	`/dists/<distro>/`
مجموعة أدوات OmniCore	حزمة `omnicore`: أتمتة Ansible الكاملة (الأدوار، دفاتر اللعب، التبعيات) للنشر الذاتي	`/pool/main/o/omnicore/`
حزم Ubuntu	حزم Ubuntu المخزنة مع جميع التبعيات	`/<distro>/pool/main/`
إصدارات GitHub	ثنائيات مسبقة البناء (Prometheus، Grafana، Homer، إلخ)	`/releases/<org>/<repo>/`
أرشيفات المصدر	أرشيفات المصدر لتطبيقات الويب (CGrateS_UI، speedtest)	`/repos/`
حزم الطرف الثالث	Galera، FRR، InfluxDB، KeyDB، إلخ	`/releases/<vendor>/`

متغيرات التكوين

هو المصدر الوحيد للحقيقة لتوزيع الحزم. قم بتعريفه مرة واحدة في جردك. `apt_repo` لتنزيل الثنائيات، أو المخطط، أو المنفذ، أو URL كل شيء آخر مشتق منه. لا تقوم بتكوين عنوان بيانات الاعتماد بشكل منفصل.


apt_repo
(المصدر الوحيد للحقيقة,
المعرف في الجرد)
مشتق تلقائيًا
remote_apt_*
(roles/common/defaults/main.yml)
apt_download_base
(roles/common/tasks/facts.yml)
ما يقومون بتكوينه
/etc/apt/sources.list.d/omnitouch.list
تنزيلات ثنائية
/releases/، /repos/

كيف يعمل الاشتقاق

الدور	المكان الذي تم تعيينه	المتغير
الكتلة الوحيدة التي تقوم بتكوينها. يتم تضمينها مباشرة في سطر مصدر APT.	جردك	`apt_repo`
مشتق من `apt_repo`. الملاذ النهائي عندما يكون `apt_repo` غائبًا: `apt.us-fl.omnitou.ch` / `443` / `https`.	`roles/common/defaults/main.yml`	`remote_apt_*`
عنوان URL الأساسي المحسوب الوحيد الذي تستخدمه كل دور لتنزيلات الثنائيات، بحيث يتفق سطر مصدر APT وتنزيلات الثنائيات دائمًا على المضيف، والمخطط، والمنفذ، وبيانات الاعتماد.	`roles/common/tasks/facts.yml`	`apt_download_base`

فإن جردًا عاديًا **يحتاج فقط إلى** `apt_repo`، نظرًا لأن كلا القيم المشتقة مبنية من نفس كتلة **`apt_repo`**. `remote_apt_*` لا تقوم بتعيين بنفسك.

> **استثناء: خادم مرآة التخزين.** المضيف في مجموعة `apt_cache_servers` هو المكان الخاصة `apt_repo` بشكل صريح. تشير `remote_apt_*` الوحيد الذي لا يزال يتم فيه تعيين

> خادم Omnitouch `remote_apt_*` به إلى *ذاكرة التخزين المؤقت المحلية*. ثم تصف قيم *العلوي* الذي يقوم بمزامنة منه (انظر الخيار 2).

المخطط والمنفذ

المنفذ `443`: يتبع المخطط المنفذ تلقائيًا: → `https`، **أي منفذ آخر** → `http`. عند حذف `apt_repo.apt_repo_port`:

- `443` الافتراضي هو **الوصول المباشر (HTTPS)**
- `8080` الافتراضي هو **وضع التخزين المؤقت (HTTP)**

متى يتم استخدام كل وضع

السيناريو	سطر مصدر APT	تنزيلات ثنائية (`apt_download_base`)
`use_apt_cache: true` (التخزين المؤقت)	`http://<cache>:8080/...` (بدون مصادقة)	`http://<cache>:8080/...` (بدون مصادقة)
`use_apt_cache: false` (مباشر)	`https://<user>:<pass>@<server>/...`	`https://<user>:<pass>@<server>:443/...`

الخيار 1: الوصول المباشر

بالنسبة لعمليات النشر التي تحتوي على اتصال بالإنترنت، تسحب الآلات الافتراضية الحزم مباشرة من خادم APT الخاص بـ Omnitouch.

متطلبات الشبكة

المصدر IP قائمة بيضاء لعنوان: يجب أن يكون عنوان IP العام الخاص بك مدرجًا في القائمة البيضاء على خادم APT الخاص بـ Omnitouch. أثناء الإعداد، قدم الشبكات الفرعية الخاصة بك إلى Omnitouch. في المقابل، ستتلقى:

- الأساسية HTTP **اسم المستخدم** و **كلمة المرور** لمصادقة

- **FQDN** لخادم APT (مثل `apt.<region>.omnitou.ch`)

متطلبات جدار الحماية: يجب السماح بالوصول الخارجي إلى نطاقات IP التالية الخاصة بـ Omnitouch:

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

الخدمات التي تتطلب الوصول إلى بنية Omnitouch التحتية:

الخدمة	المنفذ	البروتوكول	الغرض
خادم APT	443	TCP	تنزيل الحزم (HTTPS)
خادم APT	53	TCP/UDP	حل DNS لـ FQDN الخاص بـ APT
خادم الترخيص	123	UDP	مزامنة الوقت NTP للتحقق من الترخيص
خادم الترخيص	53	TCP/UDP	حل DNS للتحقق من الترخيص

تأكد من السماح بحركة مرور HTTPS (TCP/443)، وNTP (UDP/123)، وDNS (TCP+UDP/53) إلى نطاقات IP الخاصة بـ Omnitouch.

التكوين

```yaml
all:
  vars:
    use_apt_cache: false

    # المصدر الوحيد للحقيقة. تكوين تنزيل الثنائيات (remote_apt_*) و
    # عنوان URL لـ apt_download_base لا تقوم بتعيينها - مشتق من هذا تلقائيًا
    # بشكل منفصل.
    apt_repo:
      apt_server: "apt.<region>.omnitou.ch"   # FQDN المقدم من
Omnitouch
      apt_repo_username: "your-username"
      apt_repo_password: "your-password"
      # apt_repo_port: 443                    # اختياري؛ الافتراضي هو
443 (HTTPS)
```

المعلمات

مصادر حزم APT (`apt_repo`)

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`apt_repo.apt_server`	سلسلة	نعم	-	اسم مضيف أو APT خادم IP عنوان
`apt_repo.apt_repo_username`	سلسلة	نعم	-	اسم مستخدم مصادقة HTTP الأساسية
`apt_repo.apt_repo_password`	سلسلة	نعم	-	كلمة مرور مصادقة HTTP الأساسية
`apt_repo.apt_repo_port`	عدد صحيح	لا	`443`	منفذ الخادم. يتبع المخطط المنفذ (`443` → HTTPS، وإلا HTTP).

تنزيلات الثنائيات (`remote_apt_*`)

لا تقم بتعيينها في جرد الوصول المباشر. توجد `apt_repo` **هذه مشتقة تلقائيًا من** فقط حتى يمكن لخادم مرآة التخزين الإشارة إلى مصدر علوي مختلف (انظر الخيار 2). كمرجع، الافتراضي لها كما يلي:

المعلمة	مشتق ن	الملاذ
`remote_apt_server`	`apt_repo.apt_server`	`apt.us-fl.omnitou.ch`
`remote_apt_port`	`apt_repo.apt_repo_port`	`443`
`remote_apt_protocol`	المنفذ (`443` → `https`)	`https`
`remote_apt_user`	`apt_repo.apt_repo_username`	`""`
`remote_apt_password`	`apt_repo.apt_repo_password`	`""`

عام

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`use_apt_cache`	Boolean	نعم	-	`false` يجب أن تكون للوصول المباشر

أنماط URL (الوصول المباشر)

مصادر حزم APT (المعرفة في `/etc/apt/sources.list.d/omnitouch.list`):

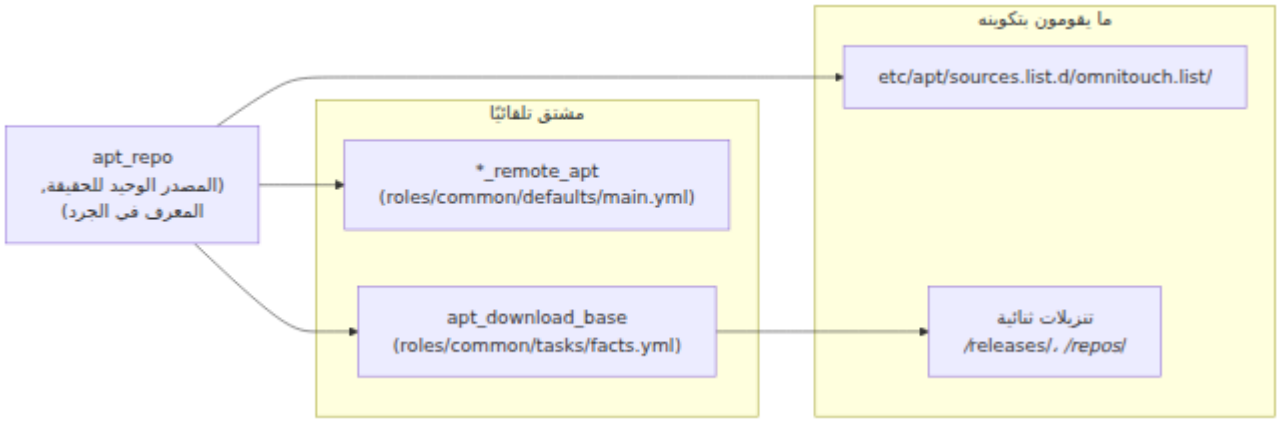
```
deb [trusted=yes] https://{username}:{password}@{apt_server}/
noble main
```

تنزيلات الثنائيات (المبنية من `apt_download_base` بواسطة مهام Ansible `get_url`):

```
https://{username}:
{password}@{apt_server}/releases/prometheus/node_exporter/node_export
1.8.1.linux-amd64.tar.gz
```

> يتم إضافة منفذ غير (HTTPS إنه الافتراضي لـ) URL يتم حذف المنفذ `443` من عنوان ما لم يكن `443` HTTP ويحول المخطط إلى `:<port>` قياسي كـ.

كيف يعمل



وتنزيلات APT الأساسية لكل من حزم HTTP تقوم الآلات الافتراضية بالمصادقة باستخدام مصادقة Omnitouch (مخزنة مسبقًا)، لذا لا تحتاج من خادم Ubuntu الثنائيات. يتم أيضًا تقديم حزم نظام Ubuntu الآلات الافتراضية إلى الوصول إلى مرايا.

توقيع الحزمة (GPG)

مفتاح توقيع المستودع ويقدم المفتاح العام المحمي في Omnitouch الخاص بـ APT ينشئ خادم `/omnitouch.gpg`. تلقائيًا بجلب `common` بالنسبة للمضيفات ذات الوصول المباشر، تقوم الدور `/etc/apt/trusted.gpg.d/omnitouch.gpg` بحيث يتحقق `apt-get update` هذا المفتاح وتثبيته في `NO_PUBKEY`. يحدث هذا تلقائيًا؛ ن بيانات التعريف الخاصة بالمستودع دون تحذيرات��
لا حاجة لإدارة المفاتيح يدويًا.

> تستخدم المضيفات التي تسحب من ذاكرة GPG. لا يستخدم وضع مرآة التخزين التحقق من `[trusted=yes]` التخزين المؤقت المحلية (انظر الخيار 2) بدلاً من ذلك.

الخيار 2: مرآة التخزين المحلية

لعمليات النشر غير المتصلة، المعزولة، أو التي تعاني من قيود النطاق الترددي، قم بنشر ذاكرة تخزين مؤقت محلية لـ Omnitouch. تقوم بمزامنة كل المحتوى من APT

الهيكلية

التكوين

قم بتعريف خادم التخزين المؤقت في ملف المضيفين الخاص بك. إنه المكان الوحيد الذي يتم فيه تعيين `remote_apt_*` بشكل صريح. تصف تلك المتغيرات خادم Omnitouch **العلوي** الذي تقوم ذاكرة التخزين المؤقت بمزامنة منه:

```yaml
apt_cache_servers:
  hosts:
    example-apt-cache:
      ansible_host: 192.168.1.100
      gateway: 192.168.1.1
  vars:
    # العلوي Omnitouch يقوم خادم التخزين المؤقت بمزامنة المحتوى من مستودع
    remote_apt_server: "apt.<region>.omnitou.ch"
    remote_apt_port: 443
    remote_apt_protocol: "https"
    remote_apt_user: "your-username"
    remote_apt_password: "your-password"

all:
  vars:
    # use_apt_cache: true  # يتم تعيينه تلقائيًا عند وجود مجموعة
apt_cache_servers
    # apt_repo.apt_server: يتم اشتقاقه تلقائيًا إلى 192.168.1.100 (أول خادم
تخزين مؤ��ت)
```

كيف يعمل:

- **خادم التخزين المؤقت** (`192.168.1.100`): يستخدم بيانات اعتماد `remote_apt_*` لمزامنة المحتوى من خادم Omnitouch العلوي عبر HTTPS.
- **جميع المضيفات الأخرى**: تشتق تلقائيًا `apt_repo.apt_server: "192.168.1.100"` وتسحب من ذاكرة التخزين المؤقت على المنفذ `8080` دون بيانات اعتماد.

المعلمات

مصادر حزم APT (`apt_repo`)

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`apt_repo.apt_server`	سلسلة	نعم	تلقائيًا	تخزين IP عنوان ت المحلي. يتم ه تلقائيًا من أول مضيف `apt_cache_s` ذا لم يتم تحديده
`apt_repo.apt_repo_username`	سلسلة	لا	-	غير مطلوب عند ستخدام التخزين مؤقت (لا حاجة للمصادقة)
`apt_repo.apt_repo_password`	سلسلة	لا	-	غير مطلوب عند ستخدام التخزين مؤقت (لا حاجة للمصادقة)
`apt_repo.apt_repo_port`	عدد صحيح	لا	`8080`	الذي تستخدمه ات للوصو�ى خادم التخزين المؤقت المحلي محل `apt_cache_p` (الذي تم إهماله

مزامنة خادم التخزين المؤقت (`remote_apt_*`)

تقوم هذه المتغيرات بتكوين كيفية مزامنة خادم التخزين المؤقت للمحتوى من Omnitouch. هذا هو السيناريو الوحيد الذي يتم فيه تعيين `remote_apt_*` يدويًا:

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`remote_apt_server`	سلسلة	نعم	-	خادم APT العلوي الخاص بـ Omnitouch للمزامنة منه
`remote_apt_port`	عدد صحيح	لا	`443`	منفذ خادم APT العلوي الخاص بـ Omnitouch
`remote_apt_protocol`	سلسلة	لا	`https`	البروتوكول لاتصال المزامنة
`remote_apt_user`	سلسلة	نعم	-	بيانات الاعتماد للمزامنة من Omnitouch
`remote_apt_password`	سلسلة	نعم	-	بيانات الاعتماد للمزامنة من Omnitouch

عام

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`use_apt_cache`	Boolean	لا	`true`	يتم تعيينه تلقائيًا إلى `true` عند وجود مجموعة `apt_cache_servers`

> يتم الآن تعيين المنفذ الذي تتصل به `apt_cache_port`. **تم إهماله:** لم يعد يستخدم المضيفات عبر `apt_repo.apt_repo_port` (الافتراضي هو `8080` عند حذفه).

أنماط URL (وضع التخزين المؤقت)

مصادر حزم APT (المعرفة في `/etc/apt/sources.list.d/omnitouch.list`):

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

تنزيلات الثنائيات (المبنية من `apt_download_base`):

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

لا تحتاج الوصول إلى التخزين المؤقت إلى بيانات اعتماد. يستخدم تكوين APT `[trusted=yes]`.

نشر التخزين المؤقت

1. **قم بتوفير خادم التخزين المؤقت** (آلة افتراضية أو حاوية LXC +50 بسعة قرص جيجابايت)
2. **قم بتشغيل برنامج إعداد التخزين المؤقت:**

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

3. **تحقق من التخزين المؤقت** عن طريق تصفح `http://192.168.1.100:8080/`

ما الذي يتم مزامنته

تقوم مرآة التخزين بمزامنة **كل المحتوى** من خادم Omnitouch. الخاص بـ APT. تختلف الآلية حسب نوع المحتوى:

- **حزم APT** (`/dists/` + `/pool/` لكل من Omnitouch وUbuntu) تتم مزامنتها باستخدام `curl` مدفوعة بفهرس مستودع Debian. تقوم المرآة بجلب ملف `Release` للحصول على اسم `Packages.gz` العلوي (وتتجاوز إذا لم يتغير)، وتحلل الملف/SHA256 لا/التي تفتقر إليها فقط أو الملف، وتقوم بتنزيل الملفات التي حجم كل حزمة. يتم التحقق من كل ملف تم تنزيله باستخدام SHA256 تتطابق مع النسخة المحلية. قبل الالتزام به SHA256.

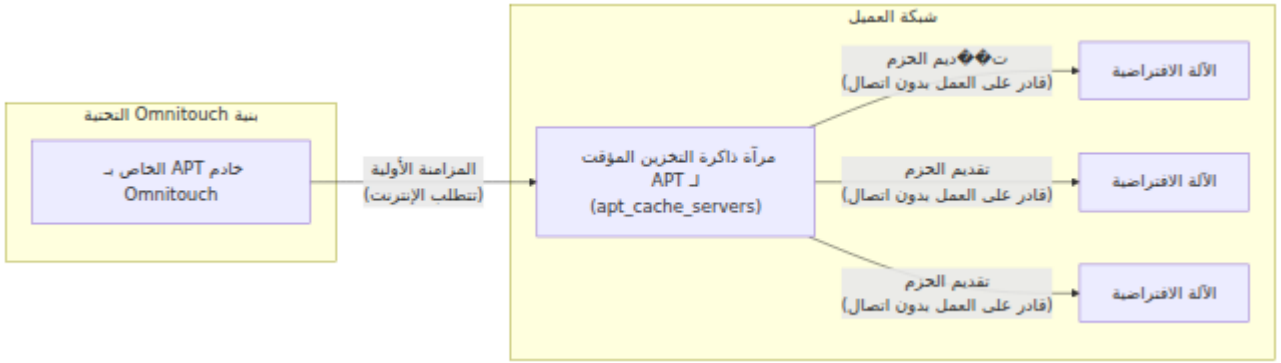
- **الإصدارات الثنائية وأرشيفات المصدر** (`/releases/`، `/repos/`) ليس لديها فهرس بأسلوب Debian، لذا تتراجع إلى `wget --recursive --timestamping`، الذي يعيد تنزيل الملفات فقط التي تحتوي على `Last-Modified` أحدث.

دلائل المحتوى المزامنة:

المسار	المحتوى
`/dists/<distro>/`	بيانات التعريف الخاصة بمستودع APT (ملفات Packages، Release)
`/pool/main/`	حزم .deb المخصصة لـ Omnitouch
`/<distro>/pool/main/`	حزم Ubuntu وجميع التبعيات
`/releases/`	إصدارات GitHub (Prometheus، Grafana، Zabbix، إلخ)
`/repos/`	أرشيفات المصدر (Erlang، Elixir، CGrateS_UI، إلخ)

بعد المزامنة الأولية، يمكن أن تقدم الذاكرة التخزينية جميع الحزم دون اتصال بالإنترنت.

كيف يعمل



الأساسية لكل المحتوى. تتم مزامنة HTTP تقوم مرآة ��لتخزين بالمصادقة باستخدام مصادقة حزم APT باستخدام `curl` ضد فهرس مستودع Debian (`Release` + `Packages.gz`)، وتنزيل الملفات فقط التي تختلف عن النسخة المحلية والتحقق من كل منها بعد التنزيل؛ SHA256 `wget --recursive --timestamping` تتراجع إلى `/repos/` و`/releases/` في كلتا الحالتين، تقوم المزامنات اللاحقة بجلب الملفات الجديدة أو المتغيرة فقط.

التكوين التلقائي

في جردك، يقوم النظام تلقائيًا `apt_cache_servers` عند وجود مجموعة:

1. لجميع المضيفات (ما لم يتم تجاوزها صراحة) `use_apt_cache: true` بتعيين
2. الخاص بأول خادم `ansible_host` من عنوان IP `apt_repo.apt_server` باشتقاق تخزين مؤقت

مثال على التكوين الأدنى

```yaml
apt_cache_servers:
  hosts:
    apt-cache-01:
      ansible_host: 192.168.1.100
      gateway: 192.168.1.1
  vars:
    # يقوم خادم التخزين المؤقت بمزامنة المحتوى من مستودع Omnitouch العلوي
    remote_apt_server: "apt.<region>.omnitou.ch"
    remote_apt_user: "your-username"
    remote_apt_password: "your-password"
```

ما يحدث تلقائيًا:

- جميع المضيفات (باستثناء خادم التخزين المؤقت) تحصل على `use_apt_cache: true`
- جميع المضيفات (باستثناء خادم التخزين المؤقت) تحصل على `apt_repo.apt_server: "192.168.1.100"`
- جميع المضيفات تسحب من `http://192.168.1.100:8080/` دون بيانات اعتماد
- خادم التخزين المؤقت يقوم بمزامنة الحزم من `https://your-username:your-password@apt.<region>.omnitou.ch/`

تجاوز السلوك التلقائي

لإجبار الوصول المباشر حتى مع تعريف خوادم التخزين المؤقت:

```
all:
  vars:
    use_apt_cache: false  # إجبار الوصول المباشر حتى مع تعريف خوادم التخزين المؤقت

    apt_repo:
      apt_server: "apt.<region>.omnitou.ch"
      apt_repo_username: "user"
      apt_repo_password: "pass"
```

ملخص التكوين

السيناريو 1: الوصول المباشر إلى خادم APT (بدون تخزين مؤقت)

تقوم جميع المضيفات بسحب الحزم مباشرة من خادم مستودع APT. فقط `apt_repo` مطلوب. يتم اشت��اق قيم `remote_apt_*` منها.

```
all:
  vars:
    use_apt_cache: false

    # المصدر الوحيد للحقيقة لكل من مصادر APT وتنزيلات الثنائيات
    apt_repo:
      apt_server: "apt.<region>.omnitou.ch"
      apt_repo_username: "user"
      apt_repo_password: "pass"
```

النتيجة: جميع المضيفات تولد `deb [trusted=yes] https://user:pass@apt.<region>.omnitou.ch/ noble main`

معرف في ملف APT السيناريو 2: خادم تخزين مؤقت المضيفين (تلقائي)

خادم التخزين المؤقت موجود في جردك وسيتم نشره/مزامنته بواسطة Ansible.

```
apt_cache_servers:
  hosts:
    cache-server:
      ansible_host: 192.168.1.100
      gateway: 192.168.1.1
  vars:
    # يقوم خادم التخزين المؤقت بمزامنة الحزم من المستودع المعتمد العلوي
    remote_apt_server: "apt.<region>.omnitou.ch"
    remote_apt_port: 443
    remote_apt_protocol: "https"
    remote_apt_user: "user"
    remote_apt_password: "pass"

# لا حاجة لتكوين في all: vars:
# كل شيء يتم اشتقاقه تلقائيًا من مجموعة apt_cache_servers
```

النتيجة:

- **خادم التخزين المؤقت**: قوم بالمزامنة من `https://user:pass@apt.<region>.omnitou.ch/`
- **جميع المضيفات الأخرى**: تولد `deb [trusted=yes] http://192.168.1.100:8080/noble noble main` (بدون بيانات اعتماد)

السيناريو 3: ذاكرة التخزين المؤقت غير موجودة APT في ملف المضيفين (يدوي)

يوجد خادم التخزين المؤقت في مكان آخر وتم إعداده بالفعل (غير مُدار بواسطة Ansible الخاص بك).

```yaml
all:
  vars:
    use_apt_cache: true

    # توجيه جميع المضيفات إلى خادم التخزين المؤقت الخارجي
    apt_repo:
      apt_server: "192.168.1.100"  # عنوان IP لخادم التخزين المؤقت الخارجي
      apt_repo_port: 8080          # عادةً ما يعمل التخزين المؤقت على المنفذ 8080

# apt_cache_servers لا حاجة لمجموعة
# remote_apt_* لا حاجة لـ (التخزين مؤقت مُعد بالفعل خارجيًا)
```

النتيجة: جميع المضيفات تولد `deb [trusted=yes] http://192.168.1.100:8080/noble noble main` (بدون بيانات اعتماد)

مثال كامل

هذا مثال كامل يعمل. يظهر تكوين خادم التخزين المؤقت ع عدة مضيفات تطبيق:

```yaml
# مجموعة خادم التخزين المؤقت APT
apt_cache_servers:
  hosts:
    example-apt-cache:
      ansible_host: 10.179.1.114
      gateway: 10.179.1.1
      host_vm_network: "vmbr0"
      num_cpus: 4
      memory_mb: 16384
      proxmoxLxcDiskSizeGb: 120
  vars:
    # يقوم خادم التخزين المؤقت بمزامنة الحزم من المستودع المعتمد العلوي
    remote_apt_server: "apt.<region>.omnitou.ch"
    remote_apt_port: 443
    remote_apt_protocol: "https"
    remote_apt_user: "customer-username"
    remote_apt_password: "customer-secure-token"

# خوادم التطبيقات
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

# التكوين العالمي
all:
  vars:
    # التكوين التلقائي: (لا حاجة لتكوين يدوي)
    # - use_apt_cache: true (ممكن تلقائيًا عند وجود apt_cache_servers)
```

```
    # - apt_repo.apt_server: "10.179.1.114" (مشتق تلقائيًا من خادم التخزين المؤقت)
```

ما يحدث أثناء النشر:

1. **خادم التخزين المؤقت (`10.179.1.114`):**
   - يستخدم `remote_apt_*` من قسم `vars:`
   - يقوم بتنزيل جميع الحزم من `https://customer-username:customer-secure-token@apt.<region>.omnitou.ch/`
   - يقدم الحزم على المنفذ 8080 عبر nginx
2. **مضيفات التطبيقات (`customer-hss01`، `customer-mme01`، `customer-dns01`):**
   - تكتشف `apt_cache_servers` تلقائيًا وجود مجموعة
   - تعيين تلقائي لـ `use_apt_cache: true`
   - اشتقاق تلقائي لـ `apt_repo.apt_server: "10.179.1.114"`
   - توليد: `deb [trusted=yes] http://10.179.1.114:8080/noble noble main`
   - تسحب جميع الحزم من خادم التخزين المؤقت (بدون حاجة لبيانات اعتماد)

تحديث التخزين المؤقت

للمزامنة مع حزم جديدة أو تحديثات:

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

تقوم هذه العملية بمزامنة كل المحتوى من خادم APT الخاص بـ Omnitouch بشكل تدريجي:

- إصدارات جديدة من حزم Omnitouch
- حزم Ubuntu جديدة وتبعياتها
- إصدارات جديدة من GitHub

- أرشيفات المصدر المحدثة

تكون المزامنات اللاحقة تدريجية: تتم مقارنة حزم APT بفهرس Debian العلوي بواسطة SHA256 (وتتخطى تمامًا إذا لم يتغير ملف `Release` العلوي)، بينما تستخدم `/releases/` و`/repos/` `wget --timestamping`. يتم تخطي الملفات غير المتغيرة الموجودة، مما يجعل إعادة المزامنة سريعة.

ملاحظة: خادم APT الخاص بـ Omnitouch هو المصدر الوحيد للحقيقة لجميع الحزم. تقوم `services/apt_cache.yml` بإنشاء ونشر الحزم عليه؛ تشغيل Omnitouch على مرايا التخزين المؤقت بعد ذلك يقوم بمزامنة ما هو منشور حاليًا.

استكشاف الأخطاء وإصلاحها

فشل تحديث APT مع 401 غير مصرح به

الأعراض:

```
فشل في جلب https://10.179.1.115/noble/dists/noble/main/binary-amd64/Packages  401  غير مصرح به
```

الأسباب المحتملة:

- تم تعريف تكوين `apt_repo` في `all: vars:` بدلاً من `apt_cache_servers: vars:`
- المضيفات تحاول الوصول إلى المستودع المعتمد مباشرة بدلاً من التخزين المؤقت
- اسم المستخدم `apt_repo_username` أو كلمة المرور `apt_repo_password` غير صحيحة
- عنوان IP المصدر غير مدرج في القائمة البيضاء على خادم APT الخاص بـ Omnitouch
- استخدام بيانات اعتماد التخزين المؤقت للوصول المباشر أو العكس

الحل:

1. **تحقق من نطاق التكوين**: تأكد من تعريف `apt_repo` مع بيانات الاعتماد في `apt_cache_servers: vars:`، وليس في `all: vars:`

2. **تحقق من وضع التخزين المؤقت**: عند استخدام التخزين المؤقت، يجب أن تتصل المضيفات بخادم التخزين المؤقت (المنفذ 8080)، وليس بالمستودع العلوي
3. **تحقق من المصادر المولدة**: على المضيف الذي فشل، تحقق من `/etc/apt/sources.list.d/omnitouch.list`
    - **الصحيح (وضع التخزين المؤقت)**: `deb [trusted=yes] http://10.179.1.114:8080/noble noble main`
    - **غير صحيح (لديه بيانات الاعتماد في المكان الخطأ)**: `deb [trusted=yes] https://user:pass@10.179.1.115/noble noble main`
4. تحقق من أن بيانات الاعتماد صحيحة لوضع النشر الخاص بك
5. تأكد من أن عنوان IP العام الخاص بك مدر�ج في القائمة البيضاء مع Omnitouch (إذا كنت تستخدم الوصول المباشر)

فشل تنزيلات الثنائيات (Node Exporter، Zabbix، إلخ.)

الأعراض: فشل برنامج Ansible في تنزيل الملفات من مسار `/releases/`

الأسباب المحتملة:

- تنزيلات الثنائيات تستمد مصادقتها من بيانات اعتماد `apt_repo` غير صحيحة (في الوضع المباشر `apt_repo`)
- على خادم مرآة التخزين، بيانات اعتماد `remote_apt_user` / `remote_apt_password` غير صحيحة
- وأي منفذ آخر يعني HTTPS المخطط أو المنفذ خاطئ. تذكر أن المنفذ `443` يعني HTTP.

الحل:

1. في الوضع المباشر، تحقق من أن `apt_repo.apt_repo_username` / `apt_repo.apt_repo_password` تتطابق مع تلك المقدمة من Omnitouch
2. على خادم مرآة التخزين، تحقق من بيانات اعتماد `remote_apt_*` العلوية
3. تأكد من أن `apt_download_base` المشتقة تستخدم المضيف/المخطط/المنفذ المتوقع (تحقق باستخدام `ansible-playbook ... -vvv` أو عن طريق فحص مهمة `get_url` الفاشلة)

خادم التخزين المؤقت لا يمكنه المزامنة

الأعراض: فشل برنامج خادم التخزين المؤقت في تنزيل الحزم

الأسباب المحتملة:

- خادم التخزين المؤقت ليس لديه وصول إلى الإنترنت
- بيانات اعتماد `remote_apt_*` غير صحيحة
- جدار الحماية يمنع الاتصالات الخارجية إلى Omnitouch

الحل:

1. تحقق من أن خادم التخزين المؤقت يمكنه الوصول إلى خادم APT العلوي الخاص بـ Omnitouch على المنفذ 443
2. تحقق من بيانات اعتماد `remote_apt_*`
3. راجع قواعد جدار الحماية للوصول الخارجي

الوثائق ذات الصلة

- تكوين ملف المضيفين: تكوين الجرد والمتغيرات
- مرجع التكوين: مرجع كامل للمعلمات
- بنية النشر: بنية النظام العامة
- نشر Proxmox: نشر خادم التخزين المؤقت كحاوية LXC

تسجيل مركزي

نظرة عامة

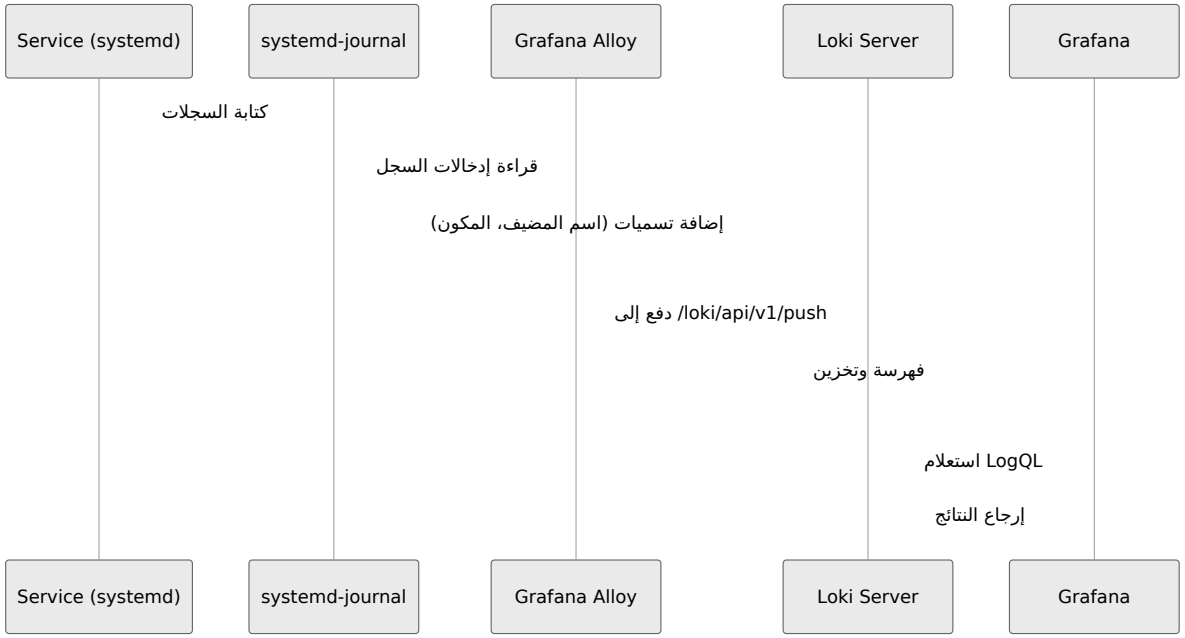
نظام تسجيل مركزي يجمع السجلات من جميع المكونات ويجعلها قابلة للبحث OmniCore يتضمن من خلال Grafana. يستخدم النظام:

- **Grafana Alloy**: وكيل جمع السجلات يتم نشره على جميع الآلات الافتراضية
- **Grafana Loki**: خادم تجميع وتخزين السجلات على مضيفي المراقبة
- **لوحات Grafana**: لوحات مسبقة البناء لكل نوع من المكونات

الهيكلية

كيف يعمل

تدفق جمع السجلات



تسميات السجل

تتضمن كل إدخال سجل هذه التسميات للتصفية والاستعلام:

التسمية	الوصف	المثال
`job`	اسم المضيف للآلة المصدر	`customer-mme01`
`hostname`	نفس اسم الوظيفة (للتوافق)	`customer-mme01`
`component`	نوع المكون من مجموعة الجرد	`mme`, `pcscf`, `ocs`
`unit`	اسم وحدة systemd	`omnimme.service`
`level`	شدة السجل	`info`, `warning`, `error`
`service`	معرف Syslog	`omnimme`

التكوين

النشر التلقائي

يتم تكوين التسجيل تلقائيًا عندما:

1. توجد مجموعة `monitoring` في جردك
2. يتم تشغيل الدور المشترك على المضيفين المستهدفين

لا يتطلب الأمر أي تكوين إضافي لوظائف أساسية.

متطلبات الجرد

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

عند تعريف مجموعة `monitoring`:

- **مضيفو المراقبة**: تشغيل خادم Loki (يستقبل ويخزن السجلات)
- **جميع المضيفين الآخرين**: تشغيل وكيل Alloy (يجمع ويرسل السجلات)

تكوين الاحتفاظ

يتم تكوين Loki من فترة الأقد�مع الاحتفاظ القائم على الوقت. يقوم الضاغط بحذف السجلات الاحتفاظ:

الوصف	الافتراضي	الإعداد
يتم حذف السجلات الأقدم من فترة الاحتفاظ	72 ساعة (3 أيام)	الاحتفاظ القائم على الوقت

لتخصيص فترة الاحتفاظ، قم بتجاوز هذه المتغير في جردك:

```yaml
all:
  vars:
    loki_retention_period: "168h"  # على سبيل المثال 7 أيام بالساعات
```

تكوين ذاكرة التخزين المؤقت لـ Alloy

يقوم Alloy بتخزين السجلات محليًا في سجل الكتابة المسبق (WAL) عندما يكون Loki غير متاح. يتم التخلص من القطع المخزنة لمنع نمو القرص غير المحدود:

الإعداد	الافتراضي	الوصف
الحد الأقصى لعمر القطعة في WAL	1 ساعة	يتم إسقاط قطع السجل المخزنة التي تزيد عن 1 ساعة

عندما يكون Loki غير متاح لفترة أطول من عمر القطعة، يتم إسقاط أقدم السجلات المخزنة.

لوحات Grafana

يتم توفير لوحات مسبقة البناء تلقائيًا لكل نوع من المكونات.

اللوحات المتاحة

اللوحة	الموقع	الوصف
سجلات CSCF	السجلات ← سجلات CSCF	سجلات P-CSCF، S-CSCF، I-CSCF (OmniCSCF)
سجلات MME	السجلات ← سجلات MME	أحداث MME للتوصيل/الفصل والأخطاء
سجلات SGW	السجلات ← سجلات SGW	أحداث جلسة SGW وحامل البيانات
سجلات PGW	السجلات ← سجلات PGW	أحداث PDN وجلسة PGW
سجلات HSS	السجلات ← سجلات HSS	رسائل Diameter HSS وأحداث المصادقة
سجلات OmniMessage	السجلات ← سجلات OmniMessage	أحداث SMPP وتسليم SMS
سجلات OCS/CGrateS	السجلات ← سجلات OCS/CGrateS	أحداث الشحن وحركة مرور JSONRPC
مكالمات RPC لـ CGrateS	مكالمات ← السجلات RPC لـ CGrateS	البحث ومطابقة طلبات/ردود RPC مع الكمون

ميزات اللوحة

تتضمن كل لوحة:

- **صندوق البحث**: بحث نصي حر عبر جميع السجلات
- **رسوم بيانية لمعدل السجل**: الحجم بمرور الوقت حسب المضيف والشدة
- **أقسام المكونات**: عرض مجمع (على سبيل المثال، P-CSCF، S-CSCF، I-CSCF)
- **تصفية الأخطاء**: عرض مسبق التصفية للأخطاء والإخفاقات
- **ذيل حي**: بث سجلات في الوقت الحقيقي (تحديث كل 10 ثواني)

استخدام اللوحات

1. انتقل إلى Grafana (عادةً `http://<monitoring-host>:3000`)
2. انتقل إلى **لوحات المعلومات** ← **السجلات** ← اختر المكون
3. استخدم متغير **البحث** لتصفية السجلات
4. اضبط نطاق الوقت حسب الحاجة

استعلام السجلات

أساسيات LogQL

يستخدم Loki LogQL. الصيغة الأساسية للاستعلام:

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

الاستعلامات الشائعة

جميع السجلات من مضيف معين:

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

جميع سجلات مكون MME:

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

البحث عن الأخطاء عبر جميع CSCFs:

```
{component=~"pcscf|scscf|icscf"} |~ "(?i)error"
```

تصفية حسب وحدة systemd:

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

دمج الفلاتر:

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

استعلامات معدل السجل

حجم السجل لكل مكون:

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

معدل الأخطاء لـ CSCF:

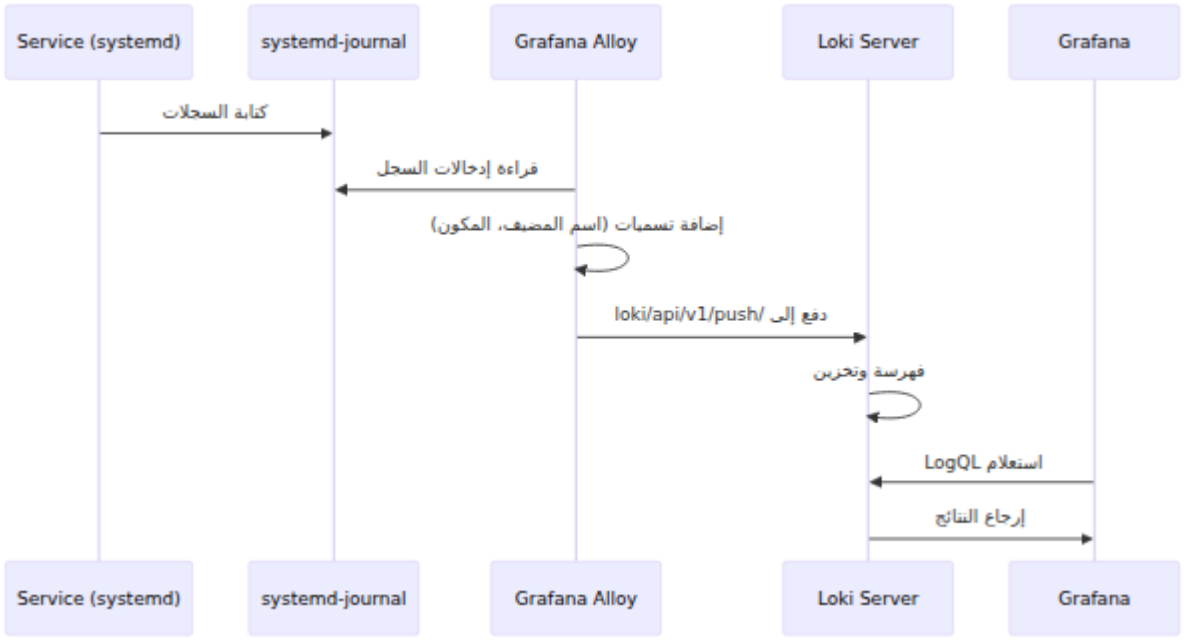
```
sum(rate({component=~"pcscf|scscf|icscf"} |~ "(?i)error" [5m]))
```

تسجيل JSONRPC لـ CGrateS

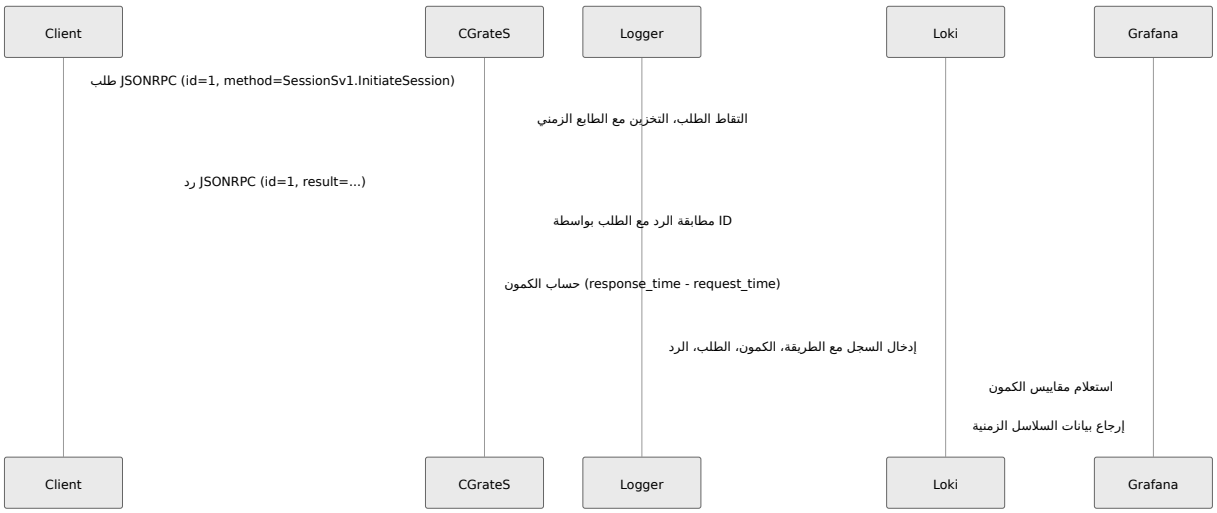
بالنسبة لنشر OCS/CGrateS، يتم التقاط حركة مرور JSONRPC /وتسجيلها مع مطابقة الطلب الرد وتتبع الكمون لتحليل الأداء واستكشاف الأخطاء وإصلاحها.

يتم تعريف مسجل JSONRPC في دور `cgrates` ويتم جمع ملف السجل الخاص به (`/var/log/cgrates/jsonrpc.log`) بواسطة Alloy على كل من مجموعات مضيفي `ocs` و `cgrates`.

الهيكلية



تدفق الطلب/الرد



ما يتم التقاطه

يتم التقاط حركة المرور على هذه المنافذ لـ CGrateS:

المنفذ	اسم الخدمة	البروتوكول	الوصف
2012	`rpc_json`	TCP	JSONRPC الخام عبر TCP
2080	`http`	HTTP	واجهة برمجة التطبيقات HTTP (تستبعد `/health` و `/metrics`)

يقوم المسجل تلقائيًا بتصفية:

- عمليات سحب مقاييس Prometheus (`GET /metrics`)
- طلبات فحص الصحة (`GET /health`)
- الردود المضغوطة باستخدام Gzip (بيانات ثنائية)

تنسيق السجل

تكتب سجلات JSONRPC إلى `/var/log/cgrates/jsonrpc.log`. يمثل كل إدخال سجل **زوج الطلب/الرد المكتمل** مع قياس الكمون:

```
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":{...}}
```

حقول السجل

الحقل	الوصف	المثال
`timestamp`	الطابع الزمني ISO 8601 عند استلام الرد	`2026-03-01T10:30:45.123456`
`port`	منفذ CGrateS الذي تم الطلب إليه	`2012`, `2080`
`service`	اسم الخدمة للمنفذ	`rpc_json`, `http`
`request_id`	معرف طلب JSONRPC للمطابقة	`1`, `42`, (فارغ للفراغ)
`method`	اسم طريقة JSONRPC	`SessionSv1.InitiateSession`
`latency_ms`	الكمون في جولة كاملة بالمللي ثانية	`2.45`
`status`	حالة النتيجة	`ok`, `error`
`error`	رسالة الخطأ إذا كانت الحالة خطأ	`INSUFFICIENT_CREDIT`
`src`	عنوان IP:المنفذ (العميل)	`10.10.1.50:45678`
`dst`	عنوان IP:المنفذ (CGrateS)	`10.10.2.100:2012`
`request`	الحمولة الكاملة لطلب JSONRPC (JSON مضغوط)	`{"id":1,"method":"..."}`
`response`	الحمولة الكاملة لرد JSONRPC (JSON مضغوط)	`{"id":1,"result":{...}}`

لوحة Grafana

تتضمن لوحة **سجلات OCS / CGrateS** لوحات مخصصة لتحليل JSONRPC.

حسب الطريقة JSONRPC الكمون لـ

رسم بياني زمني يظهر الكمون لكل طريقة JSONRPC. مفيد لتحديد بمرور الوقت:

- الطرق البطيئة التي قد تحتاج إلى تحسين
- ارتفاعات الكمون التي تشير إلى مشاكل في النظام
- اتجاهات الأداء بمرور الوقت

يعرض الشريط الأسطوري متوسط وأقصى كمون لكل طريقة.

حركة مرور JSONRPC

مع JSONRPC لوحة سجلات تعرض جميع أزواج الطلب/الرد لـ:

- الطابع الزمني
- اسم الطريقة
- الكمون
- الحمولة الكاملة للطلب والرد (منسقة)

تعمق في الطريقة

بواسطة طريقة معينة باستخدام متغير **الطريقة** في أعلى JSONRPC قم بتصفية حركة مرور لرؤية حركة (على سبيل المثال، `SessionSv1.InitiateSession`) اللوحة. أدخل اسم الطريقة المرور الخاصة بتلك الطريقة فقط.

لوحة مكالمات CGrateS لـ RPC

مكالمات تقدم لوحة **CGrateS لـ RPC** أدوات مركزة لاستكشاف أخطاء مكالمات API ومطابقة الطلبات مع الردود.

حالة الاستخدام: العثور على مكالمة فاشلة

عندما يبلغ المستخدم "حاولت الاتصال بـ 1234 وفشل":

1. **CGrateS لـ RPC** افتح **لوحات الم��لومات** ← **السجلات** ← **مكالمات**
2. في حقل **البحث**، أدخل رقم الهاتف أو معرف الحساب: `1234`
3. اضبط **الحالة** على **خطأ** لرؤية المكالمات الفاشلة فقط
4. جميع المكالمات المطابقة مع الطلب/الرد الكامل **RPC** تعرض لوحة **بحث مكالمات**

ما أعاده) يعرض كل إدخال سجل الحمولة الكاملة للطلب (ما تم إرساله) والحمولة الرد CGrateS)، مما يسهل تحديد الخطأ بالضبط.

متغيرات اللوحة

المتغير	الغرض	المثال
البحث	بحث نصي حر في الحمولة الطلب/الرد	`1234`, `Account123`, `INSUFFICIENT_CREDIT`
الطريقة	تصفية حسب اسم طريقة RPC	`SessionSv1.InitiateSession`, `CDRsV1.ProcessEvent`
الحالة	تصفية حسب حالة النتيجة	الكل، النجاح، الخطأ

لوحات المعلومات

إحصائيات ملخصة (الصف العلوي):

- **إجمالي مكالمات RPC**: عدد جميع المكالمات في نطاق الوقت
- **الأخطاء**: عدد المكالمات الفاشلة (ملونة: أخضر=0، أصفر=1-9، أحمر=10+)
- **متوسط الكمون**: متوسط الوقت المستغرق (ملون وفقًا للعتبات)
- **أقصى كمون**: أعلى مكالمة كمون

الكمون لمكالمات RPC حسب الطريقة: سلسلة زمنية تظهر الكمون لكل طريقة. مرر فوق لرؤية القيم المحددة.

بحث مكالمات RPC: اللوحة الرئيسية للسجلات تعرض المكالمات المطابقة. انقر على أي إدخال للتوسع ورؤية JSON الطلب/الرد الكامل.

مكالمات RPC الفاشلة: عرض مسبق التصفية يظهر فقط المكالمات `status=error`.

تحليل الطريقة: مخططات دائرية تظهر توزيع المكالمات وتوزيع الأخطاء حسب الطريقة.

استعلام سجلات JSONRPC

استعلامات أساسية

جميع حركة مرور JSONRPC:

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

تصفية حسب الطريقة:

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

البحث في الحمولة الطلب/الرد:

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

الأخطاء فقط:

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

تحليل الكمون

استخراج الكمون كمعيار (للرسوم البيانية):

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

البحث عن الطلبات البطيئة (الكمون > 100 مللي ثانية):

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

استعلامات محددة للطريقة

معالجة CDR:

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

إدارة الجلسات:

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

عمليات الحساب:

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

إدارة الخدمة

يعمل مسجل JSONRPC كخدمة systemd على مضيفي OCS/CGrateS.

تحقق من حالة الخدمة:

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

عرض سجلات الخدمة:

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

إعادة تشغيل الخدمة:

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

استكشاف الأخطاء وإصلاحها

عدم ظهور سجلات JSONRPC

تظهر عدم وجود بيانات JSONRPC **الأعراض**: لوحة حركة مرور

الأسباب المحتملة:

- خدمة المسجل غير قيد التشغيل
- عدم تثبيت ngrep
- عدم تطابق واجهة الشبكة
- لا يجمع ملف السجل Alloy

الحل:

1. تحقق من حالة خدمة المسجل:

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

2. تحقق من تثبيت ngrep:

```
which ngrep
```

3. تحقق من كتابة ملف السجل:

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

4. تحقق من أن Alloy يجمع الملف:

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

تظهر لوحة الكمون عدم وجود بيانات

لوحة الكمون لـ **الأعراض**: "لا توجد بيانات JSONRPC" تظهر حسب الطريقة"

الأسباب المحتملة:

- عدم وجود حركة مرور JSONRPC في نطاق الوقت المحدد
- عدم تطابق تنسيق السجل (لا يتطابق regex)

الحل:

1. تحقق من وجود السجلات لنطاق الوقت:

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

2. تحقق من أن حقل `latency_ms` موجود في السجلات
3. قم بتوسيع نطاق الوقت

عدد الطلبات المعلقة مرتفع

الأعراض: تظهر سجلات التصحيح العديد من الطلبات المخزنة ولكن القليل من الاكتمالات المسجلة

الأسباب المحتملة:

- طلبات بدون ردود (تم فصل العميل)
- تم التقاط الرد قبل الطلب (ترتيب الحزم)
- عدم تطابق معرف الطلب بين الطلب والرد

الحل:

1. يقوم المسجل تلقائيًا بتنظيف الطلبات المعلقة التي تزيد عن 60 ثانية
2. تحقق من وجود مشكلات في الشبكة بين العميل و CGrateS
3. تحقق من أن CGrateS تستجيب للطلبات

سجلات الملفات

بالإضافة إلى سجلات نظام journal، يقوم Alloy بجمع سجلات ملفات محددة:

سجلات FreeSWITCH (خوادم التطبيقات)

تسمية الوظيفة	المسار
`freeswitch`	`/var/log/freeswitch/freeswitch.log`
`freeswitch-structured`	`/var/log/omnitas-freeswitch.log`

سجلات Nginx (OmniCRM)

تسمية النوع	تسمية الوظيفة	المسار
`access`	`nginx`	`/var/log/nginx/access.log`
`error`	`nginx`	`/var/log/nginx/error.log`

JSONRPC لـ CGrateS (OCS)

تسمية الوظيفة	المسار
`cgrates-jsonrpc`	`/var/log/cgrates/jsonrpc.log`

استكشاف الأخطاء وإصلاحها

عدم ظهور السجلات في Grafana

الأعراض: لا توجد سجلات مرئية في لوحات Loki

الأسباب المحتملة:

- غير قيد التشغيل على المضيف المصدر Alloy خدمة
- غير قيد التشغيل على مضيف المراقبة Loki خدمة
- حظر الاتصال الشبكي بين المضيفين
- على المنفذ Loki 3100 الوصول إلى Alloy لا يمكن لـ

الحل:

1. على المضيف المصدر Alloy تحقق من حالة:

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

2. تحقق من حالة Loki على مضيف المراقبة:

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

3. اختبر الاتصال من المصدر إلى المراقبة:

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

4. تحقق من أن جدار الحماية يسمح بـ TCP 3100 من جميع المضيفين إلى المراقبة

استخدام Alloy العالي للقرص

الأعراض: يستهلك مساحة قرص كبيرة `/var/lib/alloy`

الأسباب المحتملة:

- Loki غير متاح لفترة طويلة
- امتلاء ذاكرة التخزين المؤقت لـ WAL

الحل:

1. تحقق من اتصال Loki (انظر أعلاه)
2. إذا كان Loki معطلاً، يتم تخزين السجلات محليًا في WAL
3. بمجرد أن يصبح Loki متاحًا، يتم إرسال السجلات المخزنة تلقائيًا
4. يتم إسقاط القطع المخزنة التي تزيد عن 1 ساعة (حسب التصميم)

امتلاء تخزين Loki

الأعراض: يتوقف Loki عن قبول السجلات، القرص ممتلئ على خادم المراقبة

الأسباب المحتملة:

- حجم السجل يتجاوز القرص المتاح قبل أن تحذف فترة الاحتفاظ السجلات القديمة
- عدم تشغيل ضغط الاحتفاظ

الحل:

1. تحقق من استخدام تخزين Loki:

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

2. تحقق من أن الضاغط قيد التشغيل (تحقق من سجلات Loki)
3. قم بتشغيل الضغط يدويًا إذا لزم الأمر عن طريق إعادة تشغيل Loki
4. ضع في اعتبارك زيادة تخصيص القرص أو تقليل فترة الاحتفاظ

فقدان تسميات المكونات

الأعراض: تظهر السجلات ولكن تسمية `component` هي `infrastructure` بدلاً من القيمة المتوقعة

الأسباب المحتملة:

- المضيف ليس في مجموعة الجرد المتوقعة
- لم يتم تجديد تكوين Alloy بعد تغيير الجرد

الحل:

1. تحقق من أن المضيف في مجموعة الجرد الصحيحة
2. أعد تشغيل Ansible لتجديد تكوين Alloy:

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

3. أعد تشغيل Alloy:

```
systemctl restart alloy
```

منافذ الخدمة

الخدمة	المنفذ	البروتوكول	الوصف
Loki	3100	HTTP	واجهة إدخال السجل واستعلام API
Loki	9096	gRPC	gRPC داخلي (غير مستخدم خارجيًا)
Alloy	12345	HTTP	وواجهة المستخدم Alloy مقاييس
Grafana	3000	HTTP	الوصول إلى لوحة المعلومات

الوثائق ذات الصلة

- المراقبة والملاحظة: Grafana، Prometheus، لوحات المعلومات، والتنبيهات
- معمارية النشر: الهيكلية العامة للنظام
- تكوين ملف المضيفين: تكوين الجرد
- مرجع التكوين: مرجع كامل للمعلمات

النواة المعتمدة على الدوائر (CS)

يتم نشر مجال الدوائر المعتمدة على الدوائر القديمة (2G/3G) بواسطة (الصوت والرسائل القصيرة `services/cs.yml` playbook. يقوم بتوفير BSC ومفتاح OmniMSC، الموجهة للراديو، وstack الإشارات SS7/SIGTRAN.

نظرة عامة

تشغل `cs.yml` ثلاثة أدوار، كل منها ضد مجموعة جرد خاصة بها:

الوصف	الحزم	مجموعة الجرد	الدور	المكون
وحدة التحكم في المحطة الأساسية + بوابة الوسائط (للاختبار والتحقق من 2G/3G)	BSC, MGW	`bsc`	`circuit_switched/bsc`	BSC
مفتاح مركز التحويل المحمول	`omnimsc`	`msc`	`circuit_switched/omnimsc`	MSC
stack SS7/SIGTRAN (STP، HLR، SMSC، CAMEL، مختبر)	`omniss7`	`ss7`	`omniss7`	SS7

```
- name: Setup MSC
  hosts: msc
  roles:
    - {role: circuit_switched/omnimsc, become: yes}
```

BSC (`circuit_switched/bsc`)

يقوم بتثبيت **BSC وبوابة الوسائط (MGW)** من خادم Omnitouch APT للاختبار والتحقق من الدوائر المعتمدة. يقوم الدور بتكوين كلا الخدمتين.

- يقوم بإنشاء ملفات تكوين BSC و MGW من قوالب الدور.
- يقوم بتثبيت تجاوز systemd لخدمة MGW التي **تعطل جدولة وحدة المعالجة المركزية في الوقت الحقيقي**. يقوم بإزالة `CPUSchedulingPolicy` و `Priority` و `AmbientCapabilities`. لا تدعم الآلات الافتراضية الجدولة في الوقت الحقيقي.
- يقوم المعالجات بإعادة تشغيل كل خدمة. ثم **يتحقق** كل معالج من أن الخدمة تصل إلى `active` (6 محاولات، بفاصل 5 ثواني) قبل أن يستمر.

MSC (`circuit_switched/omnimsc`)

يقوم بتثبيت حزمة `omnimsc` ويقوم بتكوينها مثل NF 5GC:

- يحل `group_vars_omnicore` ويستخرج `license_server_api_urls` من مجموعة `license_server`.
- يقوم بإنشاء `runtime.exs.j2` المدمج إلى `/etc/omnimsc/`. إذا تم تعيين `omnimsc_template_config`، فإنه يقوم بإنشاء قالب مخصص من `group_vars/` بدلاً من ذلك.
- يضمن وجود دليل إخراج CDR (`omnimsc_cdr_output_dir`، الافتراضي `/var/lib/omnimsc/cdr`).
- يقوم بإنشاء شهادة TLS موقعة ذاتياً لواجهة الويب / API.
- يستخدم نمط التكوين النسخي + `runtime.exs`، مع إعادة التشغيل عند التغيير.

SS7 (`omniss7`)

يتم، **SS7** ويقوم بإنشاء **قالب تكوين واحد لكل دور** `omniss7` دور واحد يقوم بتثبيت حزمة اختياره بواسطة متغير المضيف `ss7_role`:

`ss7_role`	القالب	الوظيفة
`stp`	`stp.j2`	نقطة نقل الإشارة (توجيه SS7)
`hlr`	`hlr.j2`	سجل الموقع المنزلي
`smsc`	`ipsmgw.j2`	SMSC / IP-SM-GW
`camel`	`camel.j2`	التحكم في خدمة CAMEL
`tester`	`tester.j2`	وكيل اختبار / تحميل الإشارات

محددات إضافية:

- تعيين `ss7_protocol: m2pa` يقوم بإنشاء `m2pa.j2` (نقطة نهاية M2PA) بدلاً من قالب `tester`.
- تعيين `ss7_template_config` (مسار تحت `group_vars/`) يتجاوز **جميع** القوالب المدمجة بتكوين محدد للمضيف.

مثل المكونات الأخرى في Elixir، يستخدم `omniss7` نمط التكوين النسخي + `runtime.exs` ويرتبط بالرمز في `/opt/omniss7/releases/*`.

الوثائق ذات الصلة

- Service Playbooks - جميع playbooks الخدمة وهيكل النشر
- 5G Core - 5 النواة المستقلة 5G والدور المشترك `gc_nf`
- Group Variables Configuration - قوالب التكوين المخصصة (مثل `ss7_template_config`، `omnimsc_template_config`)
- **تكوين OmniCore**: https://docs.omnitouch.com.au/docs/repos/OmniCore - ما يجب أن يتواجد في ملفات تكوين المكونات

مرجع التكوين

نظرة عامة

يوفر هذا المستند مرجعًا شاملاً لتكوين نشرات OmniCore. يتم تعريف التكوين بشكل أساسي في ملفات جرد المضيفين مع الحد الأدنى من تجاوزات group_vars المطلوبة للنشرات الحديثة.

للحصول على وثائق محددة للمنتج، انظر:

- **OmniCore**: https://docs.omnitouch.com.au/docs/repos/OmniCore
- **OmniCall**: https://docs.omnitouch.com.au/docs/repos/OmniCall
- **OmniCharge**: https://docs.omnitouch.com.au/docs/repos/OmniCharge

نهج التكوين

تستخدم نشرات OmniCore الحديثة نموذج تكوين مبسط:

المبدأ الرئيسي: يتم تعريف معظ��م التكوين مباشرة في ملف المضيفين. تتعامل القيم الافتراضية للدور مع العديد من الإعدادات. ومع ذلك، يتم تكوين بعض المكونات حصريًا من `group_vars` هو المثال الملحوظ (انظر أدناه). OmniCRM. وليس لديها قيمة افتراضية للدور في تلك الحالات، فإن القيمة المفقودة تفشل التشغيل بصوت عالٍ بدلاً من العودة إلى قيمة افتراضية.

> **ملاحظة أمان**: العديد من القيم الافتراضية للدور هي عناصر غير آمنة مخصصة للاستخدام في المختبر فقط ويجب تجاوزها لأي نشر حقيقي. تشمل الأمثلة كلمات مرور قاعدة البيانات (مثل `sql_pass: password`, `omnitouch2024`) وبيانات اعتماد مستودع APT (`omni`/`omni`). تأكد دائمًا من تعيين أسرار حقيقية وفريدة في جردك أو group_vars قبل النشر.

تخطيط الشبكة

للحصول على إرشادات حول IP **قبل تكوين المضيفين**، راجع معيار تخطيط:

- استراتيجيات تقسيم الشبكة
- تخصيص عنوان IP
- تنظيم الشبكة الفرعية
- التعامل مع IP العامة

معلمات المضيف الشائعة

#ToDo - من فقط قل للتحقق hosts-file-configuration.md لهذا

علامات الخدمة المحددة

```
cdrs_enabled: True                  # تمكين توليد CDR
in_pool: False                      # استبعاد من مجموعة التوازن
populate_crm: False                 # ملء CRM بالبيانات الأولية
```

المتغيرات العالمية (all:vars)

يحتوي قسم `all:vars` على إعدادات على مستوى النشر. تستخدم النشرات الحديثة متغيرات عالمية قليلة، مع تغطية العديد من الإعدادات بواسطة القيم الافتراضية للدور. لاحظ الاستثناءات أدناه (OmniCRM، Grafana SMTP) حيث يجب تقديم التكوين بشكل صريح لأنه لا توجد قيمة افتراضية للدور.

المتغيرات العالمية الأساسية

المصادقة والوصول

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

بدلاً من كلمات المرور SSH **بديل**: استخدم مفاتيح:

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

هوية العميل

```
customer_name_short: omnitouch
customer_legal_name: "Customer Lab"
site_name: Customer-Site
region: AU
TZ: Australia/Melbourne
```

تكوين PLMN

```
plmn_id:
  mcc: '001'                # رمز الدولة المتنقلة (3 أرقام)
  mnc: '01'                 # رمز الشبكة المتنقلة (2-3 أرقام)
  mnc_longform: '001'       # MNC مزود بالصفر (دائمًا 3 أرقام)

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

الغرض: يحدد شبكتك المتنقلة بشكل فريد. يستخدم لبناء مجال Diameter.

أسماء الشبكات

```
network_name_short: Omni
network_name_long: Omnitouch
tac_list: [10100,100]             # قائمة TAC الافتراضية (يمكن تجاوزها لكل
MME)
```

في الإعدادات > الشبكة المتنقلة UE **معروض**: أسماء الشبكات المعروضة على أجهزة.

تكوين DNS

```
manage_resolv_conf: True          # تعيين إلى False لمنع Ansible من إدارة /etc/resolv.conf
```

ملاحظة: عندما يتم تعيين `manage_resolv_conf` إلى `False`، لن يقوم Ansible بكتابة `/etc/resolv.conf` على هذا المضيف. هذا مفيد للمضيفين الذين يحتاجون إلى تكوين DNS مخصص أو يتم إدارتهم بواسطة أنظمة خارجية. يمكن تعيينه لكل مضيف في الجرد أو عالميًا في `all:vars`.

تكوين مستودع APT

هو المصدر الوحيد للحقيقة. قم بتعريفه مرة واحدة. يتم اشتقاق متغيرات تحميل `apt_repo` منه تلقائيًا. لا تقوم بتعيينها بشكل `apt_download_base` وURL و`remote_apt_*` الثنائيات منفصل في جرد عادي.

القيم الافتراضية التلقائية: عندما يتم تعريف مجموعة `apt_cache_servers` مع المضيفين:

- `use_apt_cache` يتم تعيين افتراضيًا إلى `True` (ما لم يتم تعيينه صراحة إلى `False`)
- `apt_repo.apt_server` يتم تعيين افتراضيًا إلى عنوان IP لأحد خوادم التخزين المؤقت الأولى

```
use_apt_cache: False              # الوصول المباشر إلى المستودع (True = التخزين المؤقت المحلي)

apt_repo:
  apt_server: "apt.<region>.omnitou.ch"  # FQDN المقدمة من Omnitouch (للتخزين المؤقت أو عنوان IP)
  apt_repo_username: "your-username"     # مطلوب للوصول المباشر
  apt_repo_password: "your-password"     # مطلوب للوصول المباشر
  # apt_repo_port: 443                   # 443 اختياري؛ الافتراضي هو (HTTPS).
                                         # 443 :يتبع المخطط المنفذ -> https، وإلا http.
```

في وضع **HTTP / 8080** للوصول المباشر و **HTTPS / 443** المخطط والمنفذ الافتراضيان هما يقوم بتعيين (`apt_cache_servers` group) التخزين المؤقت. فقط خادم التخزين المؤقت العلوي الذي يتزامن منه Omnitouch يدويًا، لوصف خادم `remote_apt_*`.

انظر: نظام تخزين APT

خادم الترخيص

```
# في الجرد عند عدم `license_server` اختياري: يتم اشتقاقه تلقائيًا من مجموعة تعيينه.
# يصبح كل مضيف في تلك المجموعة https://<ansible_host>:8443/api.
license_server_api_urls: ["https://10.10.2.150:8443/api"]
license_enforced: false          # القيمة الافتراضية الفعالة هي false (roles/omnihss/defaults/main.yml)
```

ملاحظة: إذا كانت مجموعة `license_server` موجودة في الجرد، يتم اشتقاق `license_server_api_urls` تلقائيًا من أعضائها؛ تحتاج فقط إلى تعيينها يدويًا لتجاوز هذا السلوك.

انظر: خادم الترخيص

إعدادات MME

يختار MME أقران S-GW وP-GW باستخدام طريقة قابلة للتكوين، يتم تعيينها لكل بوابة تحت `omnimme`. يمكن أن تكون الطريقة قيمة واحدة أو مقسمة حسب حركة المرور `home`/`roaming`. القيم الصالحة هي `random_peer` (اختيار من الأقران) و`dns_lookup` (حل الأقران عبر DNS المكونة).

```
omnimme:
  sgw:
    selection_method: random_peer        # أو، قيمة واحدة...
    # selection_method:
    #   home: random_peer
    #   roaming: dns_lookup
  pgw:
    selection_method:
      home: random_peer
      roaming: dns_lookup
```

إعدادات AMF

```yaml
# قائمة tac_list تقود أيضًا supported_ta_list لـ AMF. كل TAC يتم هنا
# يتعين AMF إذا لم يتم تعيينه، فإن NG. المتصلة أثناء إعداد gNB الإعلان عنه للأجهزة
# على TAC 1 أي يرفض مما فقط، gNB يبث لا TAC 1.
tac_list: [1, 3]

# تعطيل تشفير NAS (فرض NEA0). هو الافتراضي true لأن omniamf
# >=20260727 يتفاوض على NEA2 ولكن لا يمكنه فك تشفير NAS المشفر في الاتجاه
الصاعد (الوضع الأمني)
# 5 تعيين إلى false. مكتمل والرسائل اللاحقة)، مما يتسبب في فشل كل تسجيل
في الاتجاه الصاعد يعمل NEA2 بمجرد أن يؤكد البائع أن فك تشفير
# ضد UEs سلامة تتأثر لا .الحقيقية NAS (NIA2).
amf_force_null_ciphering: true
```

إعدادات SAEGW

```yaml
mtu: 1400                          # الحد الأقصى لوحدة النقل
```

إعدادات OmniHSS

تقوم OmniHSS ببناء قائمة الأقران Diameter تستخدم مجموعات. تلقائيًا من الجرد `mme` و `dra` أي منها يتم استخدامها يعتمد على) `applicationserver` و `pcscf` و `icscf` و `scscf` و `pgwc` و `packet_core_dra_support` / `ims_dra_support`). تقوم `exclude_diameter_peers` بإسقاط المضيفين الفرديين من تلك القائمة الناتجة، للأقران الموجودة في الجرد ولكن لا ينبغي أن تكون أقران Diameter لـ HSS.

```yaml
omnihss:
  exclude_diameter_peers:          # اختياري؛ الافتراضي هو [] (لا شيء
مستبعد)
    - customer-as01
    - customer-as02
```

- يتم مطابقة الإدخالات ضد `inventory_hostname`، وليس FQDN أو IP Diameter.
- ينطبق الاستبعاد على كل مجموعة تم اكتشافها تلقائيًا، لذا تغطي قائمة واحدة مضيفًا بغض النظر عن المجموعة التي يجلس فيها.
- يجب أن توجد الأسماء في الجرد. تؤكد الدور ذلك قبل النشر، لذا فإن خطأ مطبعي يفشل التشغيل بصوت عالٍ بدلاً من ترك النظير في `/etc/omnihss/runtime.exs`

بصمت.

- `vars:` لكل مضيف، أو تحت (`omnihss` قم بتعيينه حيث يتم تعيين باقي قاموس Ansible نظرًا لأن .(لمجموعة يستبدل بدلاً من دمج القواميس، يجب أن تعيش في `omnihss` نفس كتلة مثل المفاتيح الأخرى.
- تحت SIP S-CSCF فقط. قائمة أقران Diameter يؤثر ذلك على قائمة الأقران `config :hss, ims:` غير ذات صلة ولا يتم تصفيتها بواسطة هذه المفتاح.

> على مضيف يزيله من `dra_peer_exclude: True` على آلية خاصة به: تعيين DRA يحتوي قائمة أقران OmniDRA. استبعاد مضيف من HSS الاثنين مستقلان. لا يستبعده من DRA، أو العكس.

إعدادات OmniCRM

لا توجد ; (`hosts/` تحت) للعميل `group_vars` من OmniCRM يتم الحصول على تكوين **حصريًا** قيم افتراضية مدمجة في الدور لأسرار الموقع.

تثبيت حزم :(القيمة الافتراضية للدور، `omnicrm_deploy_mode: package`) المسار المفضل `omnicrm-api` / `omnicrm-ui` العامة من `apt_repo`، قالب `crm_config.yaml` إلى `/etc/omnicrm/crm_config.yaml`، مزامنة `deployment_assets/`، و`systemctl restart omnicrm-api`. يأتي علامة UI من كتلة `ui:` (runtime `GET /crm/public/ui-config`). من بناء **لا شيء** يحتاج هذا المسار إلى yarn من **ولا شيء** على المضيف `crm_env_override`.

أرشيف git + بناء + بيئة المضيف yarn؛ (`omnicrm_deploy_mode: source`) المسار القديم `crm_env_override` مطلوب لهذا الوضع فقط.

```
omnicrm_deploy_mode: package      # package (default) | source
omnicrm_package_version: ""       # اختياري تثبيت إصدار apt لـ omnicrm-api/ui
crm_config.yaml                   # API + ui: config (group_vars; مطلوب)
deployment_assets/                # شعارات / رموز اختيارية لـ /etc/omnicrm/deployment_assets
crm_env_override: my-site.env     # مطلوب فقط ؟؟وضع المصدر / النسخة الاحتياطية المحلية للواجهة
cell_sites: cellSites.txt         # اختياري: بيانات موقع الخلية للواجهة
message_templates: constants.js   # اختياري: قوالب رسائل البث الخلوي
site_data: site_data.json         # اختياري: حدود جغرافية لبث الخلية
```

إعدادات Grafana SMTP

تستخدمها دور المراقبة لإرسال بريد إلكتروني لتنبيه Grafana. `grafana_smtp_user` و `grafana_smtp_password` **مطلوبان** (لا توجد قيمة افتراضية: يجب تعيينهما)؛ `grafana_smtp_host` الافتراضي هو `in-v3.mailjet.com:587`. انظر `roles/monitoring/templates/grafana/grafana.ini`.

```
grafana_smtp_user: "smtp-user"            # مطلوب، لا توجد قيمة افتراضية
grafana_smtp_password: "smtp-password"    # مطلوب، لا توجد قيمة افتراضية
grafana_smtp_host: "in-v3.mailjet.com:587"  # اختياري، هذا هو الافتراضي
```

إعدادات IMS

```
ims_dra_support: False              # توجيه IMS عبر DRA
enable_homer: False                 # تمكين التقاط SIP Homer
```

تكوين مراقب RAN

```
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
```

تكوين جدار الحماية

```
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'
```

خوادم DNS للتجوال

```
roaming_dns_servers:
  wildcard: ['10.0.99.1']
  # DNS محدد للناقل (إلى استنادًا PLMN)
  123456:                          # 1 مثال الناقل
    - '10.10.2.197'
  654321:                          # 2 مثال الناقل
    - '10.10.0.4'
```

المستخدمون المحليون (مفاتيح SSH)

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

تكوين Hypervisor

Proxmox

يتم نشر الموقع كـ VMs أو حاويات LXC، يتم اختيارها لكل إدخال `proxmoxServers` عبر `deployment_type` (القيمة الافتراضية `vm`). يجب أن تتفق جميع الإدخالات في موقع واحد؛ الخلط يفشل التحقق. انظر نشر Proxmox للحصول على الشرح الكامل.

موقع VM

```
proxmoxServers:
  customer-prxmx01:
    # تم حذف deployment_type → هو الافتراضي "vm"
    proxmoxServerAddress: 10.10.0.100
    proxmoxServerPort: 8006
    proxmoxApiTokenName: AnsibleToken
    proxmoxApiTokenSecret: "token-secret"
    proxmoxNodeName: pve01
    proxmoxTemplateName: ubuntu-24.04-cloud-init-template
    proxmoxTemplateId: 9000
    proxmoxTemplateUser: omnitouch        # اسم المستخدم الاختياري لـ cloud-init؛ الافتراضي هو أول مف local_users
    proxmoxTemplatePassword: omnitouch    # كلمة المرور الاختيارية لـ cloud-init؛ الافتراضي هو اسم مستخدم cloud-init
    storage: SSD_RAID0            # اختياري، التخزين الافتراضي لـ VM
```

موقع LXC

```yaml
proxmox_lxc_nameserver: "1.1.1.1"  # اختياري، يتم حقنه في LXCs عند الإنشاء

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   # اختياري، النسخة الاحتياطية لـ rootfs
```

تجاوزات على مستوى المجموعة (كلا النوعين)

```yaml
dns:
  vars:
    proxmox_interface: vmbr0              # مطلوب: الجسر
    gateway: 10.10.0.1                    # مطلوب
    netmask: 255.255.255.0                # مطلوب
    vlanid: 100                           # اختياري
    proxmoxLxcCores: 2                    # اختياري (LXC فقط)
    proxmoxLxcMemoryMb: 4096              # اختياري (LXC فقط)
    proxmoxLxcDiskSizeGb: 30              # اختياري (LXC فقط)
    proxmoxLxcRootFsStorageName: SSD_RAID0 # اختياري (LXC فقط)
    host_vm_network: vmbr1                # تجاوز الجسر الاختياري
```

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
```

الوثائق ذات الصلة

- معيار تخطيط IP - إرشادات حول بنية الشبكة وتخصيص IP
- تكوين ملف المضيفين - كيفية هيكلة ملفات المضيفين
- تكوين المتغيرات الجماعية group_vars - متى وكيف تستخدم
- تكوين Netplan - عناوين IP متعددة وإعدادات NIC الثانوية
- بنية النشر - كيفية تفاعل المكونات
- نظام تخزين APT - إدارة الحزم
- خادم الترخيص - تكوين الترخيص

وثائق المنتج

للحصول على أدلة تشغيل مفصلة وتكوين متقدم:

- **مكونات OmniCore**: https://docs.omnitouch.com.au/docs/repos/OmniCore

- **مكونات OmniCall**: https://docs.omnitouch.com.au/docs/repos/OmniCall
- **OmniCharge/OmniCRM**: https://docs.omnitouch.com.au/docs/repos/OmniCharge

نظرة عامة على بنية النشر

نظرة عامة

يوفر هذا المستند رؤية كاملة لكيفية نشر برنامج الشبكة الخلوية لخدمات Omnitouch باستخدام Ansible، 4 موضحًا كيف تتناسب جميع المكونات معًا لإنشاء شبكة G/5G تعمل.

للحصول على إرشادات تفصيلية حول موضع المكونات، وإرشادات تخصيص IP راجع معيار تخطيط IP العام، والتعامل مع IP، عنوان IP.

مثال كامل على النشر

0. توفير البنية التحتية (اختياري)

قبل التكوين VMs/LXCs، قم بتوفير Proxmox بالنسبة لنشر:

```
# نشر VMs و/أو حاويات LXC على Proxmox (نوع النشر لكل مضيف يحدد أيهما)
ansible-playbook -i hosts/Customer/hosts.yml
util_playbooks/proxmox.yml
```

راجع: نشر VM/LXC على Proxmox

1. تعريف البنية التحتية (ملف المضيفين)

```yaml
# تعريف ما يجب نشره وأين
mme:
  hosts:
    customer-mme01:
      ansible_host: 10.10.1.15

hss:
  hosts:
    customer-hss01:
      ansible_host: 10.10.2.140

# ... جميع المكونات الأخرى
```

راجع: تكوين ملف المضيفين

2. التخصيص (group_vars)

يخزن مجلد `group_vars` أي تجاوزات تكوين مطلوبة على مستوى المضيف أو الموقع أو الشبكة.

على سبيل المثال، يحتوي مجلد على تكوين OmniMessage SMSc الخاص بك. تعيش خطوط Diameter هنا أيضًا الخاص بك يعيش هنا. كل منطق توجيه TAS التي يتصل بها SIP.

يتم تكوين OmniCRM هنا أيضًا `group_vars` إعداداته خاصة بـ. فقط. يتم استنساخ مصدره من باستخدام بيانات اعتماد (`delegate_to: localhost`) Ansible على وحدة التحكم GitHub وحدة التحكم الخاصة بها قبل مزامنتها مع الهدف، بحيث لا تهبط أي بيانات اعتماد على المضيف الهدف.

راجع: تكوين المتغيرات الجماعية

3. توزيع الحزم (APT Cache)

```yaml
# تكوين مكان الحصول على الحزم
apt_repo:
  apt_server: "10.254.10.223"  # عنوان IP لخادم التخزين المؤقت أو خادم المستودع المباشر
use_apt_cache: false  # true = استخدام التخزين المؤقت المحلي، false = الوصول المباشر إلى المستودع
```

راجع: نظام APT Cache

4. تكوين الترخيص

```yaml
# توجيه المكونات إلى خادم الترخيص
license_server_api_urls: ["https://10.10.2.150:8443/api"]
license_enforced: true
```

راجع: خادم الترخيص

5. تنفيذ النشر

تتعامل `services/twag.yml`. يمكنك نشر مكونات فردية. على سبيل المثال، قم بتشغيل `services/all.yml` مع كل شيء. استخدم `--limit=myhost` أو `--limit=mmee,sgw` لتحديد المضيفين الذين تعمل عليهم مجموعة التشغيل.

```
# نشر الشبكة الكاملة
ansible-playbook -i hosts/customer/host_files/production.yml
services/all.yml

# أو نشر مكونات محددة
ansible-playbook -i hosts/customer/host_files/production.yml
services/epc.yml
ansible-playbook -i hosts/customer/host_files/production.yml
services/ims.yml
```

نماذج الن��ر

هناك طريقتان للحصول على مجموعة أدوات OmniCore وتشغيلها على وحدة التحكم.

النموذج A: استنساخ المستودع على وحدة التحكم

النموذج التقليدي: استنساخ هذا المستودع على وحدة التحكم Ansible الخاصة بك، احتفظ بمخزونك تحت `hosts/`، هذا وقم بتشغيل مجموعات التشغيل مباشرة من النسخة المستنسخة. هو النموذج المفترض في مثال النشر أعلاه.

النموذج B: تثبيت `omnicore` .deb ذاتيًا

تم تعبئة OmniCore أيضًا كحزمة Debian ذاتية الاحتواء `omnicore`، بحيث يمكن لوحدة التحكم تثبيت مجموعة الأدوات من apt بدلاً من استنساخ المستودع.

- **ما الذي يتم تثبيته.** الحزمة (`name: omnicore`، انظر `packaging/nfpm.yaml`) تحتوي على أدوار Ansible، ومجموعات التشغيل، ومجموعات *vendored*، ومترجم Python مضمن (`/opt/omnicore/python`) وبيت العجلات غير المتصل بالإنترنت إلى `/opt/omnicore`. تقوم خطوة ما بعد التثبيت (`packaging/postinstall.sh`) بإنشاء بيئة افتراضية ذاتية الاحت��اء (`/opt/omnicore/venv`) وترتبط بيت العجلات + المترجم المضمن من `ansible*` بحيث تقوم وحدة التحكم بتشغيل النسخ، `/usr/local/bin` عبر `PATH` المضمنة بـ Ansible الخاصة بها: لا Python ومكتبات Ansible من `uv.lock` الدقيقة من التوزيعة، لا Python فقط من النظام، لا شبكة. يتم سحب الأدوات النظامية الحقيقية (`git`، `rsync`، `openssh-client`) بواسطة apt. الحزمة هي **amd64** وتستهدف **Ubuntu 22.04+** (glibc ≥ 2.35).
- **كيف يتم بناؤها.** يقوم `packaging/build.sh` بتهيئة المستودع إلى `/opt/omnicore` (باستثناء `hosts/`، `.git/`، البيئات الافتراضية وملفات البناء)، يجمع المترجم المستقل + بيت `requirements.yml`، يورد مجموعات Ansible من العجلات من الاعتماديات المقفلة لـ Python، يقوم بتشغيل فحص سري مغلق، ثم يبني `.deb` باستخدام nfpm.
- **كيف يتم نشرها.** يقوم `.github/workflows/build-deb.yml` بتشغيل `build.sh` في كل دفع إلى `main` وينشر `.deb` (بالإضافة إلى `_metadata.json`) كإصدار (كملحق) GitHub. يقوم برنامج `discover_github_debs.py` الخاص بخادم Omnitouch باكتشاف هذا الأصل الإصدار عبر منظمة apt ويضيفه إلى aptly. لا حاجة لتغييرات في دور apt.

تدفق العميل:

```
# 1. تثبيت مجموعة الأدوات من apt
apt-get install omnicore

# 2. توفير مخزونك الخاص
#    (لا تحتوي الحزمة على مخزون - إنه يعيش خارج /opt/omnicore)
vi /etc/omnicore/inventory

# 3. تشغيل مجموعات التشغيل من مجموعة الأدوات المثبتة
cd /opt/omnicore
ansible-playbook -i /etc/omnicore/inventory services/all.yml
```

منفصل عن مجموعة الأدوات (`/etc/omnicore/inventory`) نظرًا لأن المخزون يعيش في المثبتة (`/opt/omnicore`)، فإن ترقية الحزمة لا تلمس أبدًا تعريف البنية التحتية أو التخصيص.

الوثائق ذات الصلة

- البدء - مقدمة في نشر Ansible
- مجموعات تشغيل الخدمة - **مرجع مجموعة التشغيل والتسلسل الهرمي**
- تعريف البنية التحتية - تكوين ملف المضيفين
- معيار تخطيط IP - **بنية الشبكة وتخصيص IP**
- التخصيص - تكوين المتغيرات الجماعية
- إدارة الحزم - نظام APT Cache
- إدارة الترخيص - خادم الترخيص
- المراقبة والرؤية - Grafana، Prometheus، ولوحات المعلومات، التنبيهات
- جمع سجلات - التسجيل المركزي Loki وAlloy

وثائق المنتج

للحصول على معلومات مفصلة حول تكوين كل مكون:

- **OmniCore** (4G/5G Packet Core): https://docs.omnitouch.com.au/docs/repos/OmniCore

    - OmniHSS، OmniSGW، OmniPGW، OmniUPF، OmniDRA، OmniTWAG

- **OmniCall** (الصوت والرسائل): https://docs.omnitouch.com.au/docs/repos/OmniCall
    - OmniTAS، OmniCall CSCF، OmniMessage، OmniSS7، VisualVoicemail
- **OmniCharge/OmniCRM** (الفوترة): https://docs.omnitouch.com.au/docs/repos/OmniCharge
- **الوثائق الرئيسية**: https://docs.omnitouch.com.au/

جدار الحماية حسب الدور

يوفر جدار الحماية حسب الدور لكل مضيف مجموعة قواعد `iptables` و `ip6tables` مستمدة من نموذج إعلاني واحد لحركة مرور الشبكة، بدلاً من قواعد مكتوبة يدوياً لكل مضيف. يصف ملف واحد كل تدفق مسموح به في النواة (من يمكنه الوصول إلى أي دور، وعلى أي بروتوكول ومنفذ، ولماذا). من هذا النموذج، تنتج المنصة شيئين يتفقان دائماً مع بعضهما البعض:

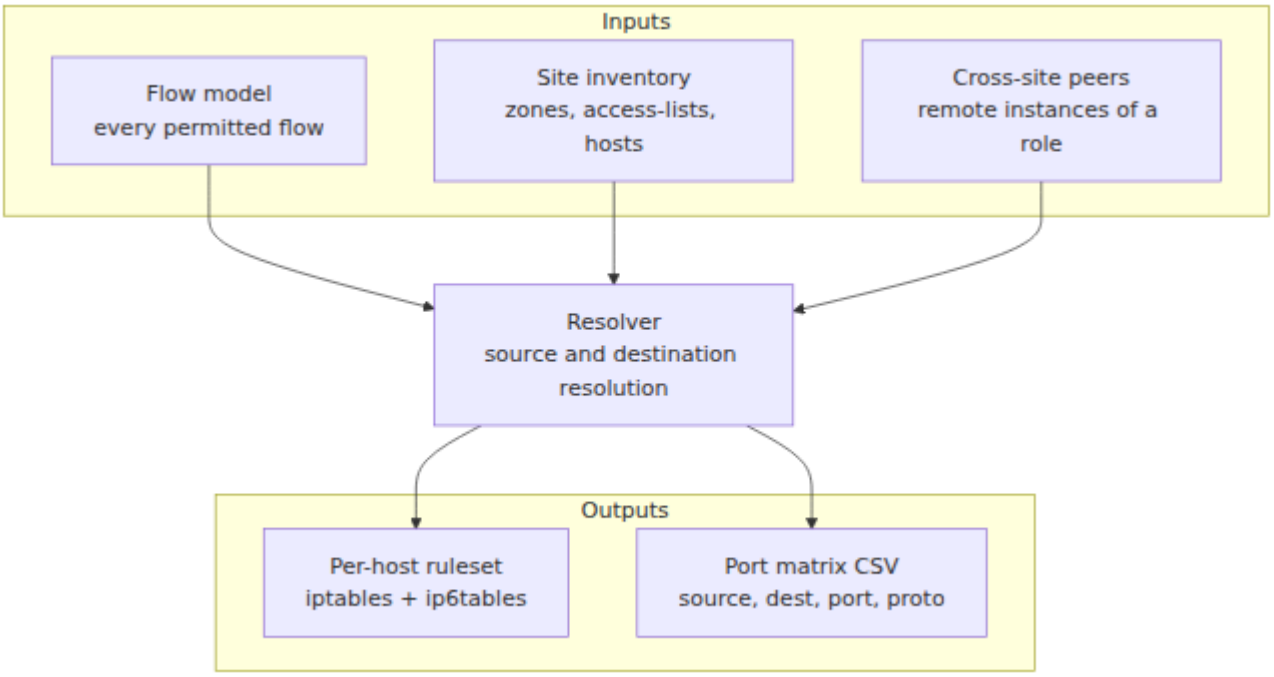
- **مجموعة القواعد المفروضة** المطبقة على كل مضيف، و
- **مصفوفة الاتصالات / المنافذ** (CSV) للمراجعة ولتسليمها إلى جدار حماية أعلى.

تعمل القواعد في أحد وضعين: **مراقبة** (تسجيل كل تدفق، عدم إسقاط أي شيء) أو **فرض** (افتراضي-الرفض). الحجب متعمد وآمن، وكل تطبيق محمي من خلال استعادة تلقائية على المضيف حتى لا يمكن لمجموعة قواعد خاطئة أن تقفل المشغلين.

جدول المحتويات

نظرة عامة على المعمارية

نموذج واحد بالإضافة إلى تغذية جرد كل موقع تغذي محللاً واحداً. ينتج المحلل كل من القواعد على المضيف ومصفوفة المنافذ، لذا لا يمكن أن تصف الوثائق شيئاً مختلفاً عما يتم فرضه فعلياً.



يتم تسجيل الحركة المرورية التي تم إسقاطها وتدقيقها من خلال خط أنابيب الرؤية القياسي:

نموذج التدفق

نموذج التدفق هو المصدر الوحيد للحقيقة حول **من يمكنه الوصول إلى ماذا**. يحتوي على جزئين: مجموعة من الافتراضات الهيكلية (التي لا تحمل أي عنوان موقع)، وقائمة التدفقات. لا تقول التدفق شيئًا عن أرقام المنافذ مباشرة - بل تسمي **خدمة** مكشوفة من قبل الدور الوجهة (انظر فهرس الخدمات)، ويتم البحث عن المنفذ من إعلان ذلك الدور:

```
- id: '13'                          # stable label, appears in each
rule's comment
  dst: hss                          # inventory group the rule is
applied to
  service: diameter                 # named service exposed by the
hss role (-> 3868/tcp+sctp)
  purpose: 'Diameter'               # description, shown in the
matrix
  sources: [{ zone: internal }]     # who may reach it (list of
source tokens)
```

المعامل	النوع	مطلوب	الافتراضي	الوصف
`id`	سلسلة	نعم	-	معرف ثابت للتدفق. يظهر في تعليق كل قاعدة تم إنشاؤها كـ `ans:<dst>:<id>` بحيث يمكن تتبع قاعدة حية إلى النموذج.
`dst`	سلسلة	نعم	-	مجموعة الجرد التي تقع عليها القاعدة، أو الوجهات الزائفة `all` (كل مضيف) و `five_g_sbi` (مجموعات وظائف الشبكة 5G). ينتج التدفق الذي لا يتم نشر مجموعة وجهته في موقع ما لا قواعد ولا صفوف في المصفوفة هناك.
`service`	سلسلة	نعم*	-	خدمة مسماة تم إعلانها من قبل الدور الوجهة (انظر فهرس الخدمات). يتم البحث عن المنفذ (المنافذ) من إعلان الدور، لذا لا تعيد التدفق أبداً ذكر رقم منفذ.
`proto_ports`	قائمة	نعم*	-	البديل الحرفي لـ `service`، للتدفقات غير المرتبطة بدور واحد (على سبيل المثال تدفقات `all`-host). كل إدخال هو `PORT/proto` (`tcp`، `udp`، `sctp`؛ قد يكون المنفذ نطاق `LO-HI`) أو بروتوكول عاري (`icmp`، `esp`، `proto 89`).
`purpose`	سلسلة	نعم	-	وصف إنساني للتدفق. يظهر في عمود `service` من مصفوفة المنافذ.
`sources`	قائمة	نعم	-	المصادر المسموح بها، كقائمة من رموز المصدر. انظر أنواع

المعامل	النوع	مطلوب	الافتراضي	الوصف
				المصادر.

* **إما** يوفر التدفق `service` **أو** `proto_ports`، وليس كلاهما.

أنواع المصادر

للتدفق هو رمز ذو مفتاح واحد يتم حله ضد تكوين الموقع وجردها. `sources` كل إدخال في قائمة لا يساهم الرمز الذي لا يتم حله إلى أي شيء (على سبيل المثال قائمة وصول فارغة) في أي قاعدة: لا يتم إصداره أبداً كعنصر نائب.

الرمز	يحل إلى
`{ zone: <name> }`	CIDRs من تلك المنطقة الطائرة (`firewall.zones.<name>`).
`{ list: <name> }`	قائمة وصول المشغل، على سبيل المثال `allowed_oam_subnets`.
`{ group: <name> }`	عناوين مضيف مجموعة الجرد، بالإضافة إلى أقران ذلك الدور عبر المواقع. انظر الأقران عبر المواقع.
`{ ue_pools: true }`	مجموعات UE / المشتركين في الموقع (`firewall.ue_pools`).
`{ dra_peers: true }`	كل عنوان تم تكوينه في قائمة الأقران الخاصة بمضيف DRA.
`{ roaming: true }`	نطاقات التجوال (IR.21 / GRX-IPX). حالياً أي مصدر: انظر تحذيرات تشغيلية.
`{ literal: "..." }`	مصدر نصي حر غير مشتق من الجرد.

فهرس الخدمات

أرقام المنافذ **تعود للدور الذي يفتحها**، وليس للنموذج المركزي. كل دور يكشف عن المنافذ في دليل دوره، مفاتيحاً بمجموعات الجرد التي يخدمها `firewall_services.yml` يشحن ملف هذا يحافظ على تعريف المنفذ بجوار .(يخدم عد�� مجموعات، لذا يعلن كل منها `cscf` دور مثل) الكود الذي يفتحه:

```
# roles/omnihss/firewall_services.yml
firewall_services:
  hss:
    diameter: [3868/tcp, 3868/sctp]   # S6a / Cx-Dx / Sh
    api_oam:  8080/tcp                # HSS API (OAM)
    api_nf:   8443/tcp                # HSS API (queried by other
NFs)
    postgres: 5432/tcp                # logical replication
between HSS peers
```

لكل دور في فهرس واحد. يتم البحث عن `firewall_services.yml` يجمع المحلل كل لذلك التدفق، لذا يعبر النموذج فقط عن **العلاقة** `dst` للتدفق في الفهرس لمجموعة `service:` بينما **المنفذ** يعيش مع الدور. أضف أو (`hss` الخاصة بـ `diameter` من يمكنه الوصول إلى خدمة) الخاصة به؛ لا يتغير شيء في `firewall_services.yml` انقل مستمعاً في دور وقم بتحديث النموذج المركزي.

تحتوي هذه الانقسام على شبكة أمان تفتقر إليها النموذج المركزي فقط: **فحص الانحراف** يقارن ما تعلنه الأدوار ضد ما يستمع إليه المضيفون فعلياً.

```
ansible-playbook -i hosts/<Deployment>/host_files/<Site>.yml \
    util_playbooks/firewall_services_audit.yml
```

لكل مضيف، يتم الإبلاغ عن:

- **غير معلن** - مستمع حي على منفذ لم يعلنه أي دور (إما إضافته إلى `firewall_services.yml` للدور، أو أنه مستمع غير متوقع)، و
- **غير مستمع** - خدمة معلنة ليست فعلياً قيد التشغيل.

يتم تخطي المضيفون الذين لا تعلن مجموعاتهم عن أي خدمات، لذا فإن الفحص مفيد أثناء ترحيل الأدوار إلى الفهرس تدريجياً بدلاً من جميعها دفعة واحدة.

منافذه MME **المستمعون الشرطيون.** يتم فتح بعض المنافذ فقط عندما يتم نشر دور آخر: يفتح LCS (SBc-AP `29168/sctp`، SLs `9082/sctp`) فقط عندما يكون مضيف `omnilcs` في الجرد. لا يمكن أن تعبر الإعلانات الثابتة عن هذا الشرط، لذا يتم الإشارة إلى مثل هذه المنافذ في `firewall_services.yml` للدور ولكن لا يتم إعلانها. في موقع يقوم بتشغيل الدور المعتمد، يتم الإبلاغ عنها كغير معلنة، وهو ما يعد تحذيراً صحيحاً للمراجعة والسماح بها.

تكوين الموقع

يعلن كل موقع عن كتلة `firewall` تحت `all.vars` في جرده. لا توجد بدائل للعناوين: يجب على الموقع وصف شبكته الخاصة. أي شيء لا يعلنه الموقع يكون ببساطة فارغاً، ولا يتم وراثة أي عنوان من بيئة أخرى.

المثال أدناه مكتمل تماماً بحيث يظهر كل حقل قيمة تمثيلية. تستخدم العناوين نطاقات توثيق (كل طائرة هي شبكة فرعية خاصة بها ضمن الشبكة الفائقة) وتخطيط متماسك لكل طائرة، وهو ما يجعل المناطق مفيدة: فهي تسمح لتدفق واحد بالسماح، على سبيل المثال، `internal`، فقط لطائرة IMS بدلاً من النواة بالكامل.

```
firewall:
  allowed_oam_subnets:           ['10.8.0.0/24']       #
management source (SSH, UI, scrape, mgmt APIs)
  allowed_ran_subnets:           ['10.20.0.0/16']      # RAN
signalling source (S1AP, NGAP, GTP-U, Abis)
  allowed_ue_voice_subnets:      ['100.64.0.0/16']     # external
UE SIP into the P-CSCF
  allowed_carrier_voice_subnets: ['203.0.113.0/24']    # carrier
interconnect SIP
  zones:                                               # plane
zones: each plane's real subnet(s)
    internal:    ['10.8.0.0/16']                       #   trusted
core supernet
    mobile_core: ['10.8.1.0/24']                       #   mobile
core NFs (EPC + 5GC + CS core)
    core_svc:    ['10.8.2.0/24']                       #   core
services plane (HSS, DRA, OCS, OAM)
    ims:       ['10.8.3.0/24']                         #   IMS
plane (CSCF, TAS)
    ue_svc:    ['10.8.4.0/24']                         #   UE-
reachable plane (P-CSCF, DNS, ePDG)
    core_v6:  ['2001:db8:0:1::/64']                    #   IPv6
addresses of core NFs (not UEs)
  ue_pools: ['100.64.0.0/16', '2001:db8:64::/48']      # UE /
subscriber IP ranges (v4 and v6)
  extra_rules:                                         # manual
accept rules (see below)
    - { proto: tcp, dport: '8443', source: '203.0.113.10/32',
comment: 'partner API' }
```

هو إلزامي. يمكن ترك كل قائمة أخرى فارغة (`[]`) في موقع `allowed_oam_subnets` فقط، وموقع بدون نواة يحدد RAN `allowed_ran_subnets: []` يفتقر إلى تلك الوظيفة: موقع بدون UE يحدد `ue_pools: []`. تنتج القائمة IPv6 يحدد `core_v6: []`، وموقع بدون مجموعات الفارغة ببساطة عدم وجود قواعد للتدفقات التي تستخدمها (انظر عمود `Required` أدناه).

المعامل	النوع	مطلوب	الافتراضي	الوصف
`allowed_oam_subnets`	قائمة	نعم	-	الشبكات رعية لمصدر لإدارة. تحكم SSH، واجهة OmniWeb جمع Prometheـ وواجهات برمجة التطبيقات الإدارية. إذا كانت فارغة، وقف المولد ع خطأ، لأن قيمة فارغة تترك الإدارة بدون مصدر مسموح به سبب إغلاقاً تحت فرض.
`allowed_ran_subnets`	قائمة	لا	`[]`	الشبكات رعية لمصدر الإشارة RA (S1AP، NGAP، GT U، Abis). كن أن تكون فارغة في موقع بدون RAN.
`allowed_ue_voice_subnets`	قائمة	لا	`[]`	الشبكات رعية لمصدر رجي لـ SIP

المعامل	النوع	مطلوب	الافتراضي	الوصف
				UE إلى P-CSCF.
`allowed_carrier_voice_subnets`	قائمة	لا	`[]`	الشبكات رعية لمصدر لتواصل SIP ن الشركات
`zones`	خريطة	نعم	-	مناطق لائرات التي بط كل اسم ئرة بشبكتها الفرعية قيقية. انظر معاملات المنطقة.
`ue_pools`	قائمة	لا	`[]`	نطاقات IP UE / المشتركين. ار��ة إذا لم يكن لدى الموقع أي منها.
`extra_rules`	قائمة	لا	`[]`	قواعد قبول دوية مطبقة على كل ضيف. انظر معاملات القاعدة اليدوية.

معاملات المنطقة

قم بتعيين الطائرات التي يستخدمها الموقع فقط؛ CIDRs. تربط كل منطقة اسم طائرة بقائمة من اترك الباقي فارغاً.

المنطقة	الوصف
`internal`	الشبكة الفائقة الأساسية الموثوقة. أوسع مصدر إشارة داخلي.
`mobile_core`	وظائف الشبكة الأساسية المتنقلة عبر الأجيال: EPC (MME، S/P-GW، UPF)، 5GC (AMF، SMF، وشبكة SBI)، وCS core (MSC، SS7/SIGTRAN).
`ims`	طائرة IMS (I-CSCF، S-CSCF، TAS).
`ue_svc`	طائرة الخدمات القابلة للوصول من UE (P-CSCF، DNS، ePDG).
`core_svc`	طائرة خدمات النواة (HSS، DRA، OCS، OAM).
`core_v6`	عناوين خدمات النواة IPv6. فارغة إذا لم يكن لدى الموقع نواة IPv6.

معاملات القاعدة اليدوية

لكل مضيف `firewall_extra_rules` و (عالمية، لجميع المضيفين) `extra_rules` تضيف `:` على `source` قواعد قبول خام للحالات التي لا يغطيها النموذج. يتم تطبيق الذي يحتوي على IPv6.

المعامل	النوع	مطلوب	الافتراضي	الوصف
`proto`	سلسلة	لا	-	البروتوكول، على سبيل المثال `tcp`، `udp`، `sctp`.
`dport`	سلسلة	لا	-	منفذ الوجهة أو النطاق.
`source`	سلسلة	لا	-	المصدر. يتم تطبيق قاعدة على CIDR في القيمة التي تحتوي على IPv6 `:`.
`comment`	سلسلة	لا	`override`	تسمية مسجلة في تعليق القاعدة.

أوضاع التشغيل

يتم التحكم في وضع التطبيق بواسطة `firewall_mode`.

الوضع	السلوك
`monitor`	يتم تثبيت كل قاعدة قبول، تظل سياسة `INPUT` على `ACCEPT`، ويتم تسجيل حركة المرور المتساقطة (كـ `FW-AUDIT`) أي شيء. استخدم هذا للتحقق من النموذج مقابل حركة المرور الحقيقية قبل فرضه ثم السماح بها. لا يتم إسقاط أي
`enforce`	يتم تسجيل حركة المرور المتساقطة (كـ `FW-DROP`) ثم إسقاطها، ويتم تعيين سياسات `INPUT` و `FORWARD` إلى `DROP`.

تسقط حركة المرور. أي قيمة `enforce` **الحجب هو اختيار آمن: فقط القيمة الدقيقة** أخرى، بما في ذلك متغير غير مضبوط أو خطأ مطبعي، يتم التعامل معها كـ مراقبة، لذا لا يمكن لمضيف أن يبدأ في إسقاط حركة المرور عن طريق الخطأ. هذا ينطبق على ثلاثة مستويات: في جردها، `firewall_mode: monitor` تعلن المواقع عن `monitor`، الافتراضي للمنصة هو ويعالج المولد أي شيء ليس `enforce` كـ مراقبة. يتطلب فرض تغييراً متعمداً.

```
all:
  vars:
    firewall_mode: monitor   # change to 'enforce' only after the
audit log confirms coverage
```

أو تجاوزها لكل مجموعة أو لكل `all.vars`، لكل موقع تحت `firewall_mode` يمكن تعيين مضيف من خلال تسلسل المتغيرات العادي.

الأقران عبر المواقع

تقوم الأدوار التي تتحد عبر المواقع (CSCF، TAS، OmniMessage، تكرار HSS، OCS) بإدراج أقرانها البعيدة في ملف الأقران الخاص بالموقع (على سبيل المثال `prod_master_peers.yml`). يتم دمج تلك العناوين البعيدة في حل `{ group: <role> }`، لذا فإن التدفق الذي يسمح بدور يغطي تلقائياً كل من الحالات المحلية وعبر المواقع له تحت سياسة واحدة متطابقة. يتم استخدام نفس الحل للقواعد ولمصفوفة المنافذ، لذا تظهر الأقران في كلاهما.

حل العناوين

يتم حل المضيف إلى **كل عنوان تم الإعلان عنه معه**، وليس فقط إلى عنوانه الرئيسي. هذا يعني `ansible_host` بالإضافة إلى كل `secondary_ips.<name>.ip_address` في الجرد. على UPF الثانوية: يجيب NIC تربط العقدة الخاصة بمسار البيانات وعقد الإشارة الخدمات ببطاقات على عنوانه العام. تغطي كل من SIGTRAN ويبدأ STP ويتلقى، `n3` على عنوانه N3 GTP-U القواعد والمصفوفة كل تلك العناوين، كمصادر وكوجهات، لذا فإن نظيراً يسمح لعقدة بواسطة العنوان المقدم له هو السماح بالعنوان الذي تستخدمه الحركة فعلياً.

المضيف الذي ينتمي إلى **عدة مجموعات جرد** يتلقى **اتحاد** كل تدفقات كل مجموعة وخدماتها المعلنة. تكشف العقدة المتقاربة عن كل شيء لجميع الأدوار التي تعمل بها، ومجموعة المستمعين المتوقعة لها (المستخدمة في فحص الانحراف) هي الاتحاد عبر مجموعاتها.

يمكن أن تسرد NIC، لأن الحل يستخدم كل عنوان معلن بدلاً من تتبع أي خدمة ترتبط بأي عنوان (على سبيل المثال N3 GTP-U المعروض على المصفوفة خدمة على عنوان لا ترتبط به فعلياً UPF الخاص بـ N6). هذا هو تقريبي متعمد: مصفوفة كاملة ولكنها واسعة قليلاً آمنة لتسليمها إلى كسر SIGTRAN إلى جدار حماية أعلى، بينما يؤدي حذف عنوان مسار بيانات حقيقي أو عنوان NIC الحركة بشكل صامت. إذا كانت الدقة حسب العنوان مطلوبة لاحقاً، يمكن أن تسمي الخدمة الذي ترتبط به؛ حتى ذلك الحين، يبقى الحل مبنياً على الحقائق وكاملاً.

مصفوفة الاتصالات / المنافذ

مصفوفة المنافذ هي نموذج التدفق المحلول ضد جرد الموقع�� مسطحة إلى صف واحد لكل مصدر، مضيف وجهة، ومنفذ/بروتوكول واحد. إنها الشكل للمراجعة أو لتسليمها إلى جدار حماية أعلى. قم بإنشائها باستخدام:

```
ansible-playbook -i hosts/<Deployment>/host_files/<Site>.yml \
    util_playbooks/render_port_matrix.yml
```

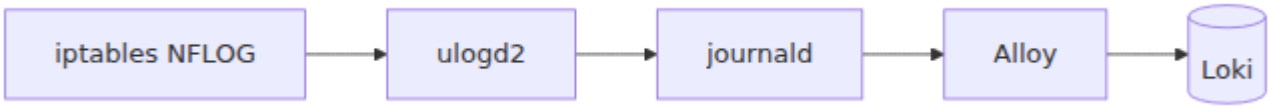
هذا يكتب `generated/firewall-port-matrix-exploded.csv` (لكل موقع، غير ملتزم). الأعمدة هي:

العمود	الوصف
`id`	معرف التدفق.
`service`	غرض التدفق.
`source`	CIDR أو عنوان مصدر تم حله.
`destination_role`	مجموعة الجرد الوجهة.
`destination_host`	اسم المضيف الوجهة.
`destination_ip`	عنوان المضيف الوجهة.
`port`	منفذ الوجهة أو النطاق (فارغ لإدخالات البروتوكول فقط مثل ICMP أو ESP).
`protocol`	البروتوكول.

يظهر الصف فقط عندما يتم نشر وجهته في الموقع ويحل مصدره إلى عنوان حقيقي. الأدوار غير المنشورة والمصادر غير المحلولة لا تنتج أي صفوف.

حماية من الإغلاق

تتم حماية تطبيق القواعد بواسطة استعادة تلقائية، يتم التحكم فيها بواسطة `firewall_rollback_guard` (افتراضيًا مفعل) و `firewall_rollback_timeout`:



1. يتم التقاط مجموعة القواعد الحية.
2. يتم تسليح مؤقت على المضيف سيستعيد اللقطة. يبقى هذا حتى يتم قطع جلسة الإدارة.
3. يتم تطبيق القواعد الجديدة.
4. يعيد المتحكم إنشاء اتصال جديد ويجري فحصاً تافهاً. إذا كانت القواعد الجديدة قد أقفلت عليه، فإن هذا يفشل ويتوقف التشغيل لذلك المضيف.
5. إذا تم تأكيد الاتصال، يتم إلغاء تسليح المؤقت. إذا لم يكن كذلك، يتم تشغيل المؤقت ويستعيد مجموعة القواعد العاملة دون أي إجراء من المشغل.

المعامل	النوع	مط��وب	الافتراضي	الوصف
`firewall_rollback_guard`	بولياني	لا	`true`	يمكّن اللقطة، المؤقت، وفحص إعادة الاتصال حول كل تطبيق.
`firewall_rollback_timeout`	عدد صحيح	لا	`120`	الثواني التي ينتظرها المؤقت قبل استعادة اللقطة تلقائياً.

IPv6 والدعم المزدوج

ويتم تطبيق القاعدة على IPv6، يعني `:` في القيمة CIDR: يتم استنتاج عائلة العنوان لكل IPv6 لا تنتج أي قواعد CIDRs IPv4 وبالتالي، فإن منطقة أو قائمة تحتوي فقط على `ip6tables`. IPv6 المناطق والمجموعات الخاصة بـ IPv6 لذلك المصدر. تترك المواقع التي لا تحتوي على نواة إلى IPv6 لـ `INPUT` تحت فرض، يتم تعيين سياسة (`core_v6: []` على سبيل المثال) فارغة في وضع المراقبة قبل فرض مضيف يصل إليه فقط من IPv6 لذا تحقق من حركة مرور، `DROP` فقط IPv4 خلال مصدر.

التسجيل

وتحمل عبر خ�� NFLOG يتم تسجيل الحركة المرورية التي تم تدقيقها وإسقاطها باستخدام الأنابيب القياسي إلى Loki. يتم استخدام ثلاثة بادئات:

البادئة	المعنى
`FW-FLOW` / `FW-FLOW6`	اتصال جديد تم رؤيته في وضع المراقبة. يبني خريطة التدفق الحية للحركة المرورية المستخدمة فعلياً.
`FW-AUDIT` / `FW-AUDIT6`	حركة المرور المتساقطة في وضع المراقبة: سيتم إسقاطها تحت الفرض، ولكنها مسموح بها هنا. راجع هذه قبل فرضها.
`FW-DROP` / `FW-DROP6`	حركة المرور المتساقطة التي تم إسقاطها فعلياً تحت الفرض.

المثال Loki استعلامات:

```
# الحركة المرورية التي سيتم حظرها بمجرد فرض هذا المضيف
{job="firewall"} |= "FW-AUDIT"

# الحركة المرورية التي يتم إسقاطها تحت الفرض، حسب المضيف
sum by (host) (count_over_time({job="firewall"} |= "FW-DROP"
[5m]))
```

تحذيرات تشغيلية

- **التجوال والمصادر النصية الحرة حالياً أي مصدر.** يتم إصدار التدفقات التي يكون مصدرها `roaming` أو `literal` بدون قيود على المصدر (`0.0.0.0/0`). اعتبرها مفتوحة حتى يتم حل نطاقات التجوال إلى CIDRs حقيقية.
- **يتم قبول ICMP من أي مصدر.** يسمح مجموعة القواعد المولدة بـ ICMP بدون شروط، وهو أوسع من نطاق الإدارة المعروض لتدفق ping.
- **لا تفرض مضيفاً في طبقة المستخدم دون قواعد توجيه.** تحت الفرض، يتم تعيين سياسة `FORWARD` إلى `DROP`، ولا يتم إنشاء أي قواعد توجيه. سيفشل فرض على دون إضافة قواعد توجيه في كسر طبقة المستخدم. تحقق أولاً SGW أو PGW أو UPF.
- **قائمة الإدارة خشنة.** تحكم `allowed_oam_subnets` واجهة المستخدم، SSH، جمع، وواجهات برمجة التطبيقات الإدارية معاً. يحتاج الموقع الذي يجمع من شبكة مختلفة عن تلك التي يدير منها إلى إدخال `extra_rules` لفصلها.

تكوين متغيرات المجموعة

نظرة عامة

دليل `group_vars` هو المكان الذي تخزن فيه ملفات التكوين المخصصة التي تتجاوز القوالب الافتراضية.

هنا تعيش تكويناتك الخاصة بالعملاء: خطوط SIP، قواعد توجيه Diameter، منطق توجيه SMS، خطط الاتصال، وأي تخصيصات أخرى حيث لا تريد استخدام التكوين الافتراضي. إنه موجود في `group_vars`.

الموقع: `hosts/{Customer}/group_vars/`

تحل الأدوار هذا الدليل في وقت التشغيل كـ `group_vars_omnicore` - دليل `group_vars/` الذي يقع بجوار دليل الجرد (`{{ inventory_dir.rsplit('/', 1)[0] }}/group_vars/`). في أي مكان يشير فيه مسار الملف أدناه إلى `{{ group_vars_omnicore }}`، فإنه يشير إلى هذا الدليل.

كيف يعمل

تمتلك أدوار Ansible قوالب تكوين افتراضية. لتخصيص ��شر معين، ضع ملفاتك المخصصة في `group_vars` وارجع إليها في ملف المضيفين الخاص بك.

```
التكوين ← group_vars (إذا تم تحديده) ← قالب الدور الافتراضي ← تجاوز المنشور
```

المثال 1: قالب تكوين مخصص (OmniMessage)

بعض المكونات تقبل قوالب تكوين مخصصة Jinja2.

هيكل الملف

```
hosts/Customer/
└── group_vars/
    └── smsc_controller.exs        # قالب تكوينك المخصص
```

الإشارة في ملف المضيفين

```yaml
omnimessage:
  hosts:
    customer-smsc-controller01:
      ansible_host: 10.10.3.219
      gateway: 10.10.3.1
      host_vm_network: "vmbr3"
      smsc_template_config: smsc_controller.exs   # الإشارة إلى اسم ملف القالب الخاص بك في group_vars
```

ماذا يحدث:

1. يجد Ansible `smsc_template_config: smsc_controller.exs`
2. يبحث في `hosts/Customer/group_vars/smsc_controller.exs`
3. يقوم بتطبيقه باستخدام Jinja2 (يمكن استخدام `{{ inventory_hostname }}`، `{{ plmn_id.mcc }}`، إلخ)
4. ينشر إلى `/etc/omnimessage/runtime.exs`
5. يعيد تشغيل الخدمة

بدون `smsc_template_config`، القالب الافتراضي من الدور يتم استخدا.

تفاصيل التكوين: انظر https://docs.omnitouch.com.au/docs/repos/OmniCall

---

المثال 2: مجموعات ملفات التكوين (وخطط الاتصال OmniTAS بوابات)

بعض المكونات تستخدم أدلة من ملفات التكوين.

هيكل الملف

```
hosts/Customer/
└── group_vars/
    ├── gateways_prod/              # تكوينات بوابة SIP
    │   ├── gateway_carrier1.xml
    │   ├── gateway_carrier2.xml
    │   └── gateway_emergency.xml
    ├── gateways_lab/               # بوابات المختبر
    │   └── gateway_test.xml
    ├── dialplan/                   # قواعد توجيه المكالمات
(افتراضي)
    │   ├── mo_dialplan.xml         # منشأ من الهاتف المحمول
(صادر)
    │   ├── mt_dialplan.xml         # منتهي من الهاتف المحمول
(وارد)
    │   └── emergency.xml
    └── dialplan_lab/               # خطط الاتصال في المختبر
        └── mo_dialplan.xml
```

الإشارة في ملف المضيفين

```yaml
applicationserver:
  hosts:
    customer-tas01:
      ansible_host: 10.10.3.60
      gateway: 10.10.3.1
      host_vm_network: "vmbr3"
      gateways_folder: "gateways_prod"   # الإشارة إلى مجلد البوابة الخاص بك لاستخدامه على هذا المضيف
      dialplan_folder: "dialplan"         # اختياري - الافتراضي هو "dialplan" إذا لم يتم تعيينه
```

ماذا يحدث:

1. يجد Ansible `gateways_folder: "gateways_prod"`
2. ينسخ جميع الملفات من `hosts/Customer/group_vars/gateways_prod/` إلى `/etc/freeswitch/sip_profiles/`

3. ينسخ جميع الملفات من `hosts/Customer/group_vars/dialplan/` (أو المجلد المحدد بواسطة `dialplan_folder`) إلى دليل قوالب OmniTAS
4. تقوم الخدمات بتحميل التكوينات

بيئات مختلفة: استخدم مجلدات مختلفة لكل بيئة:

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

تفاصيل التكوين: انظر https://docs.omnitouch.com.au/docs/repos/OmniCall

---

المثال 3: قالب تكوين مخصص (OmniHSS)

يقبل خادم المشتركين المنزلي قوالب تكوين وقت التشغيل المخصصة.

هيكل الملف

```
hosts/Customer/
└── group_vars/
    └── hss_runtime.exs.j2        # قالب تكوين HSS الخاص بك المخصص
```

الإشارة في ملف المضيفين

```
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   # الإشارة إلى اسم ملف القالب الخاص بك في group_vars
```

ماذا يحدث:

1. يجد Ansible `hss_template_config: hss_runtime.exs.j2`
2. يبحث في `hosts/Customer/group_vars/hss_runtime.exs.j2`
3. يقوم بتطبيقه باستخدام Jinja2 (يمكن استخدام `{{ inventory_hostname }}`، `{{ plmn_id.mcc }}`، إلخ)
4. ينشر إلى `/etc/omnihss/runtime.exs`
5. يعيد تشغيل الخدمة

بدون `hss_template_config`، يتم استخدام القالب الافتراضي من الدور.

تفاصيل التكوين: انظر https://docs.omnitouch.com.au/docs/repos/OmniCore

---

المثال 4: قالب تكوين مخصص (OmniMME)

تقبل وحدة إدارة الحركة قوالب تكوين وقت التشغيل المخصصة.

هيكل الملف

```
hosts/Customer/
└── group_vars/
    └── mme_runtime.exs.j2        # قالب تكوين MME لمخصص الخاص بك
```

الإشارة في ملف المضيفين

```
omnimme:
  hosts:
    customer-mme01:
      ansible_host: 10.10.3.51
      gateway: 10.10.3.1
      host_vm_network: "vmbr3"
      mme_template_config: mme_runtime.exs.j2   # الإشارة إلى اسم ملف القالب الخاص بك في group_vars
```

ماذا يحدث:

1. يجد Ansible `mme_template_config: mme_runtime.exs.j2`
2. يبحث في `hosts/Customer/group_vars/mme_runtime.exs.j2`
3. يقوم بتطبيقه باستخدام Jinja2 (يمكن استخدام `{{ inventory_hostname }}`، `{{ plmn_id.mcc }}`، إلخ)
4. ينشر إلى `/etc/omnimme/runtime.exs`
5. يعيد تشغيل الخدمة

بدون `mme_template_config`، يتم استخدام القالب الافتراضي من الدور.

تفاصيل التكوين: انظر https://docs.omnitouch.com.au/docs/repos/OmniCore

---

المثال 5: ملفات التكوين المطلوبة (OmniCRM)

يتم الحصول على OmniCRM من **حصريًا** group_vars - تم. لا يوجد دور افتراضي للرجوع إليه. تصميم تكوين API من شحن بدلاً عالٍ بصوت التشغيل يفشل بحيث الدور في مدمج غير ليكون أسرار موقع آخر بصمت.

وضع النشر المفضل: حزم apt

بتثبيت حزم Ansible يقوم، (الإعداد الافتراضي للدور) `omnicrm_deploy_mode: package` مع `apt_repo` العامة من `omnicrm-ui` و `omnicrm-api`، ويقوم بتطبيق التكوين تحت

لا يتم إعادة بناء واجهة systemctl. عبر `omnicrm-api` ويعيد تشغيل `/etc/omnicrm/`، يتم) `crm_config.yaml` في `ui:` المستخدم على المضيف؛ يأتي العلامة التجارية من كتلة تقديمها كـ `GET /crm/public/ui-config`).

هيكل الملف

```
hosts/Customer/
└── group_vars/
    ├── crm_config.yaml          # تكوين API + ui: (مطلوب)
    ├── deployment_assets/       # شعارات / أيقونات اختيارية →
/etc/omnicrm/deployment_assets
    │   └── logos/
    ├── customer_crm.env         # اختياري؛ فقط لوضع
omnicrm_deploy_mode: source المحلية المستخدم واجهة تطوير أو
    ├── cellSites.txt            # بيانات مواقع الخلايا الاختيارية
    ├── constants.js             # قوالب رسائل البث الخلوي
الاختيارية
    └── site_data.json           # حدود جغرافية لبث الخلايا
الاختيارية
```

الإشارة في ملف ��لمضيفين

```
omnicrm:
  hosts:
    customer-crm01:
      ansible_host: 10.10.3.70
      gateway: 10.10.3.1
      host_vm_network: "vmbr3"
      omnicrm_deploy_mode: package          # الافتراضي؛ استخدم source
للـ git+yarn القديم
      # omnicrm_package_version: "20260810...."  # تثبيت اختياري
      logo_light: logo-light.png            # يتم نسخه إلى
deployment_assets/logos/
      logo_dark: logo-dark.png
      login_background: login-bg.jpg        # →
deployment_assets/login/auth-one-bg.jpg (وقت التشغيل)
      login_splash: splash.jpg              # →
deployment_assets/login/login-side.jpg (يحافظ على امتداد المصدر)
      ssl_fqdn: portal.example.com
      # crm_env_override: customer_crm.env  # مطلوب فقط لوضع المصدر
      cell_sites: cellSites.txt             # اختياري
      message_templates: constants.js       # اختياري
      site_data: site_data.json             # اختياري
```

ماذا يحدث (وضع الحزمة):

1. يقوم Apt بتثبيت `omnicrm-api` و `omnicrm-ui`.
2. يقوم بتطبيق `{{ group_vars_omnicore }}/crm_config.yaml` إلى `/etc/omnicrm/crm_config.yaml` (**مطلوب**).
3. يقوم بمزامنة `deployment_assets/` (و `logo_*` / `login_*` ية ��الاختي) إلى `/etc/omnicrm/deployment_assets/`.
4. يعيد تشغيل `omnicrm-api` يقوم ؛ nginx بتقديم واجهة المستخدم من `/usr/share/omnicrm-ui`.

انظر (`GET /crm/public/assets/login/…` يتم تقديم شاشة تسجيل الدخول / الخلفية عبر `ui.branding.login_splash` / `login_background` في `crm_config.yaml`). لا يتم دمجها في واجهة المستخدم `.deb`. يحتفظ النسخ بامتداد المصدر (`splash.jpg` → `login/login-side.jpg`); ثم يقوم API بتقديم أي ملف `login-side.*` موجود.

ماذا يحدث (وضع المصدر، القديم):

1. يقوم بتطبيق `crm_config.yaml` إلى `/etc/omnicrm/OmniCRM-API/`.
2. يقوم بتطبيق `{{ crm_env_override }}` إلى واجهة المستخدم `.env` (**مطلوب** في وضع المصدر).
3. يقوم ببناء واجهة المستخدم على المضيف باستخدام yarn.

مهم: يجب أن يكون `crm_config.yaml` موجودًا. بالنسبة لوضع الحزمة، قم بتضمين كتلة `ui:` واضبط `ui.assets_dir: '/etc/omnicrm/deployment_assets'`.

تفاصيل التكوين: انظر https://docs.omnitouch.com.au/docs/repos/OmniCore

---

المثال 6: دليل بيانات التزويد (OmniHSS)

تقرأ خدمة تزويد OmniHSS (`services/provision_omnihss.yml`) بيانات المشتركين والملفات الشخصية من دليل مخصص داخل `omnihss/` group_vars.

هيكل الملف

```
hosts/Customer/
└── group_vars/
    └── omnihss/
        ├── subscribers.csv              # المشتركين للتزويد
        ├── apn_identifiers.json
        ├── apn_qos_profiles.json
        ├── apn_profiles.json
        ├── eir_rules.json
        ├── ims_profiles.json
        ├── epc_profiles.json
        ├── roaming_rules.json
        ├── roaming_profiles.json
        ├── flow_information_rules.json
        ├── charging_rules.json
        └── pcrf_profiles.json
```

ماذا يحدث:

1. تقرأ الخدمة `{{ group_vars_omnicore }}/omnihss/subscribers.csv` لقائمة المشتركين للتزويد.
2. تقرأ الملفات JSON المصاحبة من نفس دليل `omnihss/` (`apn_identifiers.json`، `apn_qos_profiles.json`، `apn_profiles.json`، `eir_rules.json`، `ims_profiles.json`، `epc_profiles.json`، `roaming_rules.json`، `roaming_profiles.json`، `flow_information_rules.json`، `charging_rules.json`، `pcrf_profiles.json`) لتعريف بيانات APN و QoS و IMS و EPC و roaming و PCRF و charging و flow المطبقة على هؤلاء المشتركين.

تفاصيل التكوين: انظر https://docs.omnitouch.com.au/docs/repos/OmniCore

---

المثال 7: بيانات اعتماد SMTP لـ Grafana (المراقبة)

تقوم قالب `grafana.ini` لدور المراقبة بتوصيل إعدادات البريد الإلكتروني (SMTP) لتنبيهات Grafana. تم تمرير اسم المستخدم وكلمة المرور من group_vars. لا توجد **قيمة افتراضية للدور** لتصميمها عمداً لتكون غير مدمجة، بحيث يفشل التشغيل بصوت عالٍ بدلاً من شحن بيانات الاعتماد.

الإشارة في ملف المضيفين

```
monitoring:
  hosts:
    customer-oam01:
      ansible_host: 10.10.3.135
      gateway: 10.10.3.1
      host_vm_network: "vmbr3"
      grafana_smtp_user: "smtp-user"          # مطلوب - لا يوجد افتراضي
      grafana_smtp_password: "smtp-password"  # مطلوب - لا يوجد افتراضي
      grafana_smtp_host: "in-v3.mailjet.com:587"  # اختياري - الافتراضي هو in-v3.mailjet.com:587
```

ماذا يحدث:

1. يتم تطبيق `grafana_smtp_user` و `grafana_smtp_password` في قسم `[smtp]` من `/etc/grafana/grafana.ini`. لا يوجد بديل - كلاهما مطلوب.
2. `grafana_smtp_host` اختياري ويفترض القيمة الافتراضية `in-v3.mailjet.com:587` إذا لم يتم تعيينه.

تفاصيل التكوين: انظر https://docs.omnitouch.com.au/docs/repos/OmniCore

---

المثال 8: مفاتيح HNET لفك تشفير SUCI في UDM

تقوم OmniUDM بفك تشفير SUCI (TS 33.501 §6.12): تستعيد SUPI من SUCI المخفية التي يقدمها UE **عامة** عند التسجيل. يتطلب ذلك مفاتيح **خاصة** للشبكة المنزلية تتزاوج مع مفاتيح للشبكة المنزلية المجهزة على USIMs. تعيش هذه المفاتيح الخاصة في دليل مخصص `/etc/omniudm/hnet/` على مضيف UDM ويتم نشرها إلى group_vars داخل `hnet_udm/` بواسطة دور `gc_nf` 5.

هيكل الملف

```
hosts/Customer/
└── group_vars/
    └── hnet_udm/
        ├── curve25519-1.key       # مفتاح خاص X25519، PEM (معرف المفتاح العام للشبكة المنزلية 1)
        ├── curve25519-1.pub       # قيمة عامة خام 32 بايت (hex) - لتجهيز USIM
        ├── secp256r1-2.key        # مفتاح خاص EC (prime256v1)، PEM (معرف المفتاح العام HNET 2)
        └── ...
```

تسمية الملفات: `<curve>-<id>.key`، حيث `<id>` هو معرف المفتاح العام للشبكة المنزلية الذي يتطابق مع SUCI، و `<curve>` هو `curve25519` (الملف الشخصي A، X25519) أو `secp256r1` (الملف الشخصي B، prime256v1). اسم الملف هو كيف يتم ربط المفتاح بمعرف مفتاح HNET على USIM الخاص به. يجب أن يتطابق مع المعرف المبرمج على USIM.

ماذا يحدث:

1. يضمن الدور أن `{{ group_vars_omnicore }}/hnet_udm/` موجود على وحدة تحكم Ansible (تم إنشاؤه باستخدام `become: false`: تبقى المفاتيح على وحدة التحكم ولا يتم إنشاؤها أبداً على الجهاز البعيد).
2. يقوم بفحص الدليل بحثًا عن ملفات `*.key`.
3. **إذا لم يتم العثور على أي منها، يتم إنشاء مفتاح افتراضي `curve25519-1.key` (X25519) تلقائيًا** باستخدام `openssl genpkey`، بحيث يعمل فك تشفير SUCI هذا من الصندوق. idempotent (`creates:` guard): لا يتم الكتابة فوق مفتاح موجود.
4. يتم تصدير القيمة العامة الخام المطابقة 32 بايت إلى `curve25519-*.key`، لكل ملف `.pub` (hex) لتجهيز USIM. تبقى ملفات `.pub` في الجرد ولا يتم نشرها إلى UDM.
5. يتم نسخ جميع ملفات `*.key` إلى `/etc/omniudm/hnet/` على UDM (الوضع `0600`، مملوكة للجذر)، ويتم إعادة تشغيل `omniudm`.

> **مهم:** المفتاح الذي تم إنشاؤه تلقائيًا عشوائي وسيقوم فقط بفك تشفير SUCIs من USIMs المبرمجة بمفتاحها العام `.pub` خذ، SIM. بالنسبة لمختبر تتحكم فيه في برمجة **المطابق**. إذا كانت SIMs على USIM (ID 1) المصدرة وقم بتجهيزها كمفتاح عام للشبكة المنزلية الخاصة بك تستخدم مخطط HNET فلا حاجة لمفتاح، (بدون إخفاء) null ويمكن أن يبقى هذا الدليل فارغًا.

لتصدير مفتاح عام من مفتاح خاص موجود يدويًا:

```
openssl pkey -in curve25519-1.key -pubout -outform DER | tail -c
32 | xxd -p -c 32
```

تفاصيل التكوين: انظر https://docs.omnitouch.com.au/docs/repos/OmniCore

---

مثال هيكل الدليل في العالم الحقيقي

```
hosts/Customer/
├── host_files/
│   └── production.yml          # ملف المضيفين يشير إلى ملفات
group_vars
└── group_vars/
    ├── smsc_controller.exs     # قالب مخصص لـ OmniMessage
    ├── smsc_smpp.exs           # قالب مخصص لـ OmniMessage SMPP
    ├── tas_runtime.exs.j2      # قالب مخصص لـ TAS
    ├── hss_runtime.exs.j2      # قالب مخصص لـ HSS
    ├── mme_runtime.exs.j2      # قالب مخصص لـ MME
    ├── dra_runtime.exs.j2      # قالب مخصص لـ DRA
    ├── pgwc_runtime.exs.j2     # قالب مخصص لـ PGW
    ├── dea_runtime.exs.j2      # قالب مخصص لـ DEA
    ├── upf_config.yaml         # تكوين UPF
    ├── crm_config.yaml         # تكوين CRM
    ├── stp.j2                  # قالب SS7 STP
    ├── hlr.j2                  # قالب SS7 HLR
    ├── camel.j2                # قالب SS7 CAMEL
    ├── ipsmgw.j2               # قالب IP-SM-GW
    ├── omnicore_smsc_ims.yaml.j2  # تكوين SMSC IMS
    ├── pytap.yaml              # تكوين TAP3
    ├── sip_profiles/           # بوابات SIP (مجلد)
    │   └── gateway_otw.xml
    ├── dialplan/               # قواعد توجيه المكالمات (مجلد)
    │   ├── mo_dialplan.xml     # منشأ من الهاتف المحمول
    │   ├── mt_dialplan.xml     # منتهي من الهاتف المحمول
    │   └── mo_emergency.xml    # توجيه الطوارئ
    ├── omnihss/                # بيانات تزويد OmniHSS (مجلد)
    │   ├── subscribers.csv         # المشتركين للتزويد
    │   ├── apn_identifiers.json
    │   ├── apn_qos_profiles.json
    │   ├── apn_profiles.json
    │   ├── eir_rules.json
    │   ├── ims_profiles.json
    │   ├── epc_profiles.json
    │   ├── roaming_rules.json
    │   ├── roaming_profiles.json
    │   ├── flow_information_rules.json
    │   ├── charging_rules.json
    │   └── pcrf_profiles.json
```

```
    └── hnet_udm/                # مفاتيح فك تشفير SUCI في UDM
(مجلد)
        ├── curve25519-1.key     # مفتاح خاص X25519 (معرف
1 المفتاح العام للشبكة المنزلية)
        └── curve25519-1.pub     # قيمة عامة خام (hex) لتجهيز
USIM
```

المعلمات الشائعة التي تشير إلى

group_vars

المعلمة	المكون	الإشارات
`smsc_template_config`	omnimessage	ملف قالب Jinja2 (مثل `smsc_controller.exs`)
`smsc_smpp_template_config`	omnimessage_smpp	ملف قالب Jinja2 (مثل `smsc_smpp.exs`)
`gateways_folder`	applicationserver	اسم المجلد (مثل `gateways_prod`)
`dialplan_folder`	applicationserver	اسم المجلد (مثل `dialplan`) - الافتراضي هو `dialplan` إذا لم يتم تعيينه
`tas_runtime_template`	applicationserver	ملف قالب Jinja2 (مثل `tas_runtime.exs.j2`) - الافتراضي هو `tas_runtime.exs.j2` إذا لم يتم تعيينه
`hss_template_config`	omnihss	ملف قالب Jinja2 (مثل `hss_runtime.exs.j2`)
`mme_template_config`	omnimme	ملف قالب Jinja2 (مثل `mme_runtime.exs.j2`)
`dra_template_config`	dra	ملف قالب Jinja2 (مثل `dra_runtime.exs.j2`)
`pgwc_template_config`	pgwc	ملف قالب Jinja2 (مثل `pgwc_runtime.exs.j2`)

المعلمة	المكون	الإشارات
`frr_template_config`	omniupf	ملف قالب Jinja2 (مثل `frr.conf.j2`)
قوالب SS7	ss7 (أدوار متنوعة)	ملفات قالب Jinja2 (مثل `stp.j2`, `hlr.j2`, `camel.j2`)
`crm_config.yaml`	omnicrm	ملف تكوين API - **مطلوب**، يتم الحصول عليه فقط من `{{ group_vars_omnicore }}/crm_config.yaml` (لا يوجد دور افتراضي)
`crm_env_override`	omnicrm	اسم ملف واجهة المستخدم .env (مثل `customer_crm.env`) - **مطلوب**، لا يوجد دور افتراضي
`cell_sites`	omnicrm	ملف بيانات مواقع الخلايا الاختيارية (مثل `cellSites.txt`)
`message_templates`	omnicrm	قوالب رسائل البث الخلوي الاختيارية (مثل `constants.js`)
`site_data`	omnicrm	حدود جغرافية لبث الخلايا الاختيارية (مثل `site_data.json`)
`grafana_smtp_user`	monitoring	مستخدم Grafana SMTP - **مطلوب**، لا يوجد دور افتراضي

المعلمة	المكون	الإشارات
`grafana_smtp_password`	monitoring	كلمة مرور Grafana SMTP - **مطلوب**، لا يوجد دور افتراضي
`grafana_smtp_host`	monitoring	مضيف Grafana SMTP اختياري - الافتراضي هو `in-v3.mailjet.com:587`
تزويد OmniHSS	omnihss	دليل `omnihss/` مع `subscribers.csv` + ملفات JSON للملف الشخصي
مفاتيح HNET	5gc_nf (omniudm)	دليل `hnet_udm/` لمفاتيح خاصة `<curve>-<id>.key` لفك تشفير SUCI - يتم إنشاء مفتاح افتراضي `curve25519-1.key` تلقائيًا إذا كان فارغًا
ملفات YAML للتكوين	مكونات متنوعة	ملفات تكوين مباشرة (مثل `upf_config.yaml`, `crm_config.yaml`)

النقاط الرئيسية

1. **تحتوي group_vars على تخصيصات** - تجاوزات للتكوينات الافتراضية
2. **الإشارة بالاسم** - استخدم معلمات مثل `smsc_template_config` أو `gateways_folder`
3. **تدعم القوالب Jinja2** - الوصول إلى أي متغير Ansible باستخدام `{{ variable_name }}`

4. يتم نسخ جميع الملفات في المجلدات المشار إليها - **تقوم المجلدات بنشر كل شيء**
5. قم بالتزام جميع group_vars إلى Git - **تحكم في الإصدار لكل شيء**

group_vars متى تستخدم

✅ استخدم group_vars لـ:

- قوالب تكوين مخصصة للمكونات
- تعريفات بوابة SIP
- خطط توجيه المكالمات
- قواعد توجيه Diameter
- إعدادات خاصة بالعميل تتجاوز الافتراضات

❌ لا تستخدم group_vars لـ:

- استخدم ملف المضيفين - (أسماء المضيفين، IP عناوين) تكوين المضيف الأساسي
- الاختبارات لمرة واحدة - استخدم متغيرات محددة للمضيف في ملف المضيفين
- إذا كانت دائمة group_vars التغييرات المؤقتة - قم بالتعديل على الهدف، والتزم إلى

الوثائق ذات الصلة

- مرجع التكوين - جميع معلمات المضيفين وما تقوم به
- تكوين ملف المضيفين - كيفية هيكلة ملفات المضيفين
- **تكوين OmniCall**: https://docs.omnitouch.com.au/docs/repos/OmniCall - ما يجب أن يكون في ملفات التكوين
- **تكوين OmniCore**: https://docs.omnitouch.com.au/docs/repos/OmniCore - تفاصيل تكوين المكونات

تكوين ملف المضيفين

نظرة عامة

ملف المضيفين (يسمى أيضًا ملف الجرد) هو الوثيقة المركزية للتكوين التي تحدد نشر شبكة الهاتف المحمول بالكامل. يحدد:

- ما هي وظائف الشبكة التي يجب نشرها
- أين تعمل (عناوين IP، شرائح الشبكة)
- كيف يتم تكوينها (معلمات محددة للخدمة)
- إعدادات محددة للعميل (PLMN، الميزات، بيانات الاعتماد)

موقع الملف

توجد ملفات المضيفين تحت الدليل الرئيسي `hosts/` (المحدد كـ `inventory` في `ansible.cfg`)، منظمة حسب العميل:

```
hosts/
└── Customer_Name/
    └── host_files/
        ├── Production.yml
        ├── Staging.yml
        └── Customer_Lab.yml
```

أسماء الملفات عشوائية. يوجد ملف واحد لكل نشر، وعادة ما يكون مسمى باسم النشر أو الموقع (مثل `Production.yml`، `Staging.yml`) بدلاً من نظام ثابت `production.yml`/`staging.yml`/`lab.yml`.

مثال على هيكل ملف المضيفين

يوضح هذا المثال المبسط الأقسام الرئيسية:

```yaml
# مكونات 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'

# مكونات IMS
pcscf:
  hosts:
    customer-pcscf01:
      ansible_host: 10.10.4.165

# خدمات الدعم
license_server:
  hosts:
    customer-licenseserver:
      ansible_host: 10.10.2.150

# المتغيرات العالمية
all:
  vars:
    ansible_connection: ssh
    ansible_password: password
```

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

معلمات المضيفين الشائعة

تكوين الشبكة

يتضمن كل مضيف عادةً:

```
pcscf:
  hosts:
    customer-pcscf01:
      ansible_host: 10.10.1.15      # عنوان IP للوصول عبر SSH
      gateway: 10.10.1.1             # البوابة الافتراضية
      host_vm_network: "vmbr1"       # اسم واجهة الشبكة لاستخدامها على Hypervisor
```

ملاحظة: للحصول على إرشادات حول تخطيط عنوان IP واستراتيجيات تقسيم الشبكة، راجع معيار تخطيط IP الذي يحدد بنية الشبكة الموصى بها بأربعة شرائح لنشر OmniCore.

مستخدمو Proxmox: تحدد معلمة `host_vm_network` الجسر الذي يجب استخدامه. راجع نشر Proxmox VM/LXC للتوفير التلقائي.

تخصيص موارد VM

للخدمات التي تحتاج إلى موارد محددة:

```
num_cpus: 4                    # نوى المعالج
memory_mb: 8192                # الذاكرة العشوائية بالميغابايت
proxmoxLxcDiskSizeGb: 50       # حجم القرص بالجيجابايت
```

معلمات محددة للخدمة

لكل وظيفة شبكة معلماتها الخاصة. أمثلة:

MME:

```
mme_code: 1                      # معرف MME (1-255)
mme_gid: 1                       # معرف مجموعة MME
network_name_short: Customer     # اسم الشبكة (يظهر على الهواتف)
network_name_long: Customer Network
tac_list: [600, 601, 602]      # رموز منطقة التتبع
```

PGW:

```
ip_pools:                        # تجمعات IP للمشتركين
  - '100.64.16.0/24'
  - '100.64.17.0/24'
combined_CP_UP: false            # فصل التحكم / خطة المستخدم
```

للحصول على شرح مفصل لما يتحكم فيه كل متغير، راجع: مرجع التكوين

خادم التطبيقات:

```
online_charging_enabled: true  # تمكين تكامل OCS
tas_branch: "main"             # فرع البرمجيات للنشر
gateways_folder: "gateways_prod"  # تكوين بوابة SIP
```

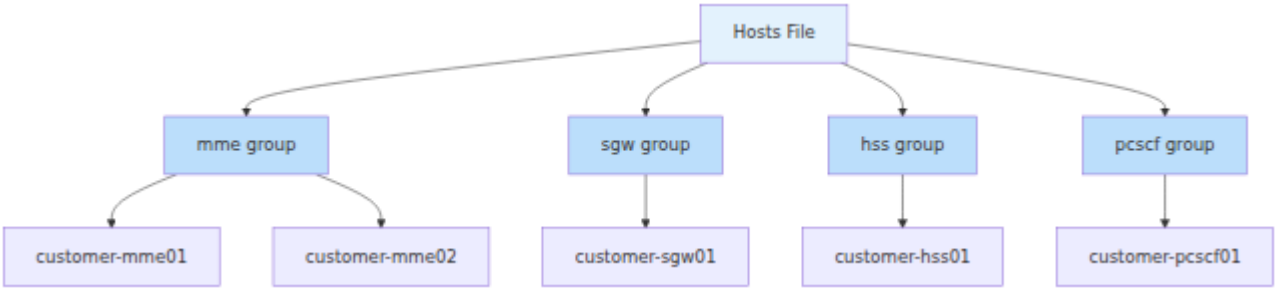
قسم المتغيرات العالمية

يحتوي قسم `all:vars` على إعدادات تنطبق على النشر بالكامل:

```yaml
all:
  vars:
    # المصادقة
    ansible_connection: ssh
    ansible_password: password
    ansible_become_password: password

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

    # معرف PLMN (شبكة الهاتف المحمو)
    plmn_id:
      mcc: '001'              # رمز الدولة المحمول
      mnc: '01'               # رمز شبكة الهاتف المحمول
      mnc_longform: '001'     # MNC مع صفر إضافي

    # أسماء الشبكة
    network_name_short: Customer
    network_name_long: Customer Network

    # مستودع APT
    # ملاحظة: إذا تم تعريف مجموعة apt_cache_servers مع المضيفين،
    # فإن use_apt_cache يكون افتراضيًا true و apt_repo.apt_server
    # لأحد خوادم التخزين المؤقت IP افتراضيًا يكون عنوان
    apt_repo:
      apt_server: "10.254.10.223"
      apt_repo_username: "customer"
      apt_repo_password: "secure-password"
    use_apt_cache: false

    # المنطقة الزمنية
    TZ: America/Chicago
```

فهم مجموعات المضيفين

تنظم Ansible المضيفين في مجموعات تتوافق مع الأدوار:



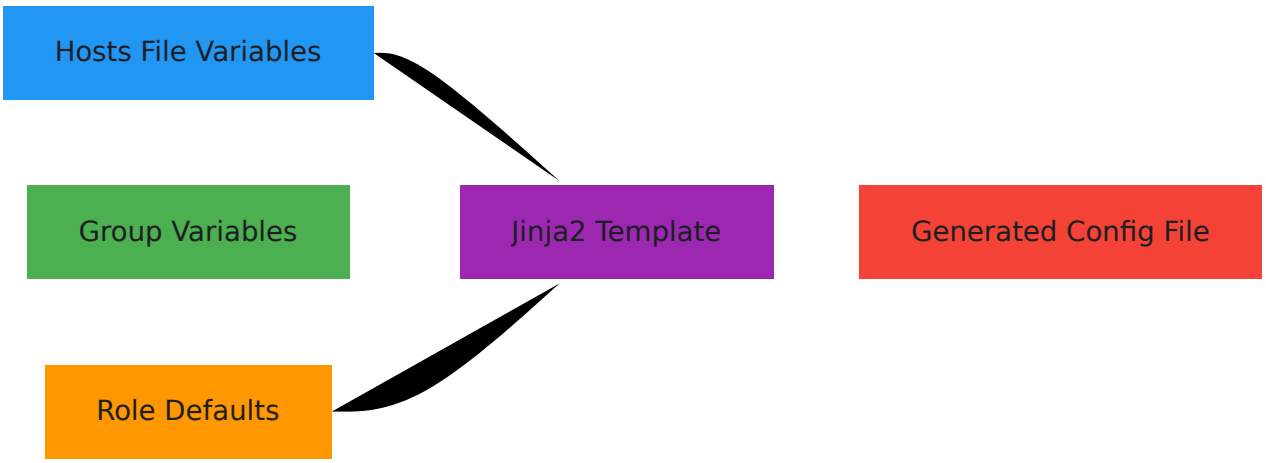
فإنه ينطبق على جميع المضيفين في قسم `mme`، عند تشغيل كتاب اللعب الذي يستهدف `mme:hosts:`.

يوضح الرسم البياني أعلاه عددًا قليلاً فقط من المجموعات. توجد العديد من المجموعات الأخرى، بما في ذلك `ocs`، `crm`، `smsc`، `omnimessage`، `dra`، `xcap`، `homer`، `cbc`، ومجموعات 5GC `amf`، `smf`، و `upf`، من بين آخرين.

التكوين باستخدام قوالب Jinja2

تستخدم Ansible **قوالب Jinja2** لإنشاء ملفات التكوين من المتغيرات المحددة في ملف المضيفين و group_vars.

كيف تعمل Jinja2



مثال على استخدام القالب

يحدد ملف المضيفين:

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

قالب Jinja2 (في الدور):

```
# 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
```

ملف التكوين الناتج:

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

أنماط Jinja2 الشائعة

الوصول إلى المتغيرات المتداخلة:

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

المنطق الشرطي:

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

الحلقات:

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

التنسيق:

```
# إضافة صفر إلى 3 أرقام
mnc{{ '%03d' | format(plmn_id.mnc|int) }}
```

group_vars تجاوز المتغيرات باستخدام

بينما يحدد ملف المضيفين البنية التحتية وإعدادات المضيف المحددة، يمكن أن تتجاوز `group_vars` الافتراضات لمجموعات المضيفين.

راجع: تكوين متغيرات المجموعة

مثال كامل على ملف المضيفين

هذا المثال الأكثر اكتمالاً يحتوي على بيانات حساسة مخفية:

```yaml
# EPC Core
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 Core
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"

# خدمات الدعم
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"

# المتغيرات العالمية
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

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

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

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

    # تكوين الشحن
    charging:
```

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

    # قواعد الجدار الناري
    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'

    # تكوين Hypervisor (مثال Proxmox VM)
    proxmoxServers:
      customer-prxmx01:
        # الافتراضي هو → تم حذف نوع النشر "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        # اسم مستخدم cloud-
init اختياري؛ الافتراضي هو أول مفتاح local_users
        proxmoxTemplatePassword: omnitouch    # كلمة مرور cloud-init
اختيارية؛ الافتراضي هو اسم مستخدم cloud-init

    # لموقع LXC، قم بتعيين deployment_type: lxc و
    # proxmoxLxcOsTemplate في كل إدخال في proxmoxServers بدلاً من ذلك.
```

راجع نشر Proxmox VM/LXC للحصول على تفاصيل إعداد Proxmox والتكوين الكاملة.

مراجع وثائق المنتج

للحصول على تكوين مفصل لكل مكون، راجع الوثائق الرسمية للمنتج:

مكونات OmniCore:

- **وثائق OmniCore**: https://docs.omnitouch.com.au/docs/repos/OmniCore
- **OmniHSS** - خادم المشتركين المنزلي
- **OmniSGW** - بوابة الخدمة (خطة التحكم)
- **OmniPGW** - بوابة الحزمة (خطة التحكم)
- **OmniUPF** - وظيفة خطة المستخدم
- **OmniDRA** - وكيل توجيه Diameter
- **OmniTWAG** - بوابة الوصول WLAN الموثوقة

مكونات OmniCall:

- **وثائق OmniCall**: https://docs.omnitouch.com.au/docs/repos/OmniCall
- **OmniTAS** - خادم تطبيق IMS (VoLTE/VoNR)
- **OmniCall CSCF** - وظائف التحكم في جلسة المكالمات
- **OmniMessage** - مركز الرسائل ��لقصيرة
- **OmniMessage SMPP** - دعم بروتوكول SMPP
- **OmniSS7** - مجموعة إشارات SS7
- **VisualVoicemail** - البريد الصوتي

OmniCharge/OmniCRM:

- **وثائق OmniCharge**: https://docs.omnitouch.com.au/docs/repos/OmniCharge

الوثائق ذات الصلة

- مقدمة في نشر Ansible - عملية النشر العامة
- مرجع التكوين - **دليل كامل لجميع متغيرات التكوين**
- تكوين متغيرات المجموعة - تجاوز التكوينات الافتراضية

- معيار تخطيط IP - **إرشادات بنية الشبكة وتخصيص IP**
- تكوين Netplan - **عناوين IP المتقدم وتكوين الشبكة الثانوية**
- نظام تخزين APT - توزيع الحزم
- خادم الترخيص - إدارة الترخيص
- نظرة عامة على بنية النشر - عرض كامل للنظام

الخطوات التالية

1. أنشئ ملف المضيفين الخاص بك بناءً على هذا القالب
2. حدد PLMN وهويتك الشبكية
3. قم APT تكوين وصول مستودع
4. إعداد خادم الترخيص
5. تخصيص باستخدام group_vars حسب الحاجة
6. نشر باستخدام كتب لعب Ansible

خادم الترخيص

نظرة عامة

يدير خادم الترخيص تفعيل الميزات لجميع مكونات Omnitouch. يتحقق كل مكون من ترخيصه عند بدء التشغيل وبشكل دوري أثناء التشغيل.

الإعداد

1. التعريف في ملف المضيفين

```
license_server:
  hosts:
    customer-licenseserver:
      ansible_host: 10.10.2.150
      gateway: 10.10.2.1
      host_vm_network: "vmbr2"

all:
  vars:
    customer_legal_name: "اسم العميل"
    license_server_api_urls: ["https://10.10.2.150:8443/api"]
```

> **ملاحظة:** `license_enforced` هو **نطاق HSS**، وليس تبديل تطبيق عالمي. يتم تعريفه حيث يكون افتراضيًا `roles/omnihss/defaults/main.yml` (`omnihss` فقط لدور HLD تحت قم بتعيينه و يتم قراءته حاليًا فقط بواسطة مولد (`false` ؛ لا يستهلكه أي دور آخر. إذا كنت بح��جة إليه omnihss متغيرات مجموعة.

2. توفير ملف الترخيص

ضع `license.json` (المقدم من Omnitouch) في `hosts/Customer/group_vars/`

3. النشر

```
ansible-playbook -i hosts/Customer/host_files/production.yml
services/license_server.yml
```

مثل) تكون أدلة الجرد الحقيقية مكتوبة بحروف كبيرة ومسبوقة `hosts/Omnicore_Customer/`)، مثل) وملف المضيف يسمى باسم الموقع `hosts/Omnicore_Customer/host_files/Production.yml`).

https://license_server . للتحقق من حالة جميع التراخيص، انتقل إلى

إزالة ترخيص

لإزالة ترخيص، استخدم `util_playbooks/delete_license.yml`.

متطلبات الشبكة

تكوين جدار الحماية

يجب تكوين جدران الحماية في مواقع العملاء للسماح بحركة مرور HTTPS (المنفذ 443) إلى خوادم التحقق من الترخيص الخاصة بـ Omnitouch:

اسم المضيف	الغرض
1.time.omnitouch.com.au	خادم التحقق من الترخيص 1
2.time.omnitouch.com.au	خادم التحقق من الترخيص 2
3.time.omnitouch.com.au	خادم التحقق من الترخيص 3

قواعد الخروج المطلوبة:

- البروتوكول: HTTPS (TCP/443)
- الوجهة: أسماء المضيفين أعلاه، يتم حلها عبر DNS
- الاتجاه: صادر

> **حل عبر DNS: لا تقم بتشفير عناوين IP.** دائمًا اسمح لهذه المضيفين بواسطة FQDN (أو بواسطة العناوين التي يقوم جدار الحماية الخاص بك بحلها ديناميكيًا). يمكن أن تتغير عناوين IP الأساسية دون إشعار. القوائم البيضاء لعناوين IP الثابتة غير مدعومة وستؤدي في النهاية إلى كسر التحقق من الترخيص.

متطلبات DNS

يتطلب خادم الترخيص الخاصة بـ DNS فعال للتواصل مع بنية التحقق من الترخيص وجود حل Omnitouch.

تكوين DNS المطلوب:

- يجب أن يكون لخادم الترخيص الوصول إلى خوادم DNS العامة
- قم بتكوين DNS لاستخدام أحد الخيارات التالية:
  - 1.1.1.1 (Cloudflare - يدعم DNS الآمن)
  - 8.8.8.8 (Google Public DNS)
- لا تستخدم خوادم DNS الداخلية/الشركات لخادم الترخيص

ملاحظة: تستخدم خوادم ترخيص Omnitouch DNS الآمن (DoH/DoT). يض��ن استخدام بواسطة DNS ويمنع المشكلات مع اعتراض DNSSEC العامة التحقق الصحيح من DNS خوادم الأجهزة الأمنية.

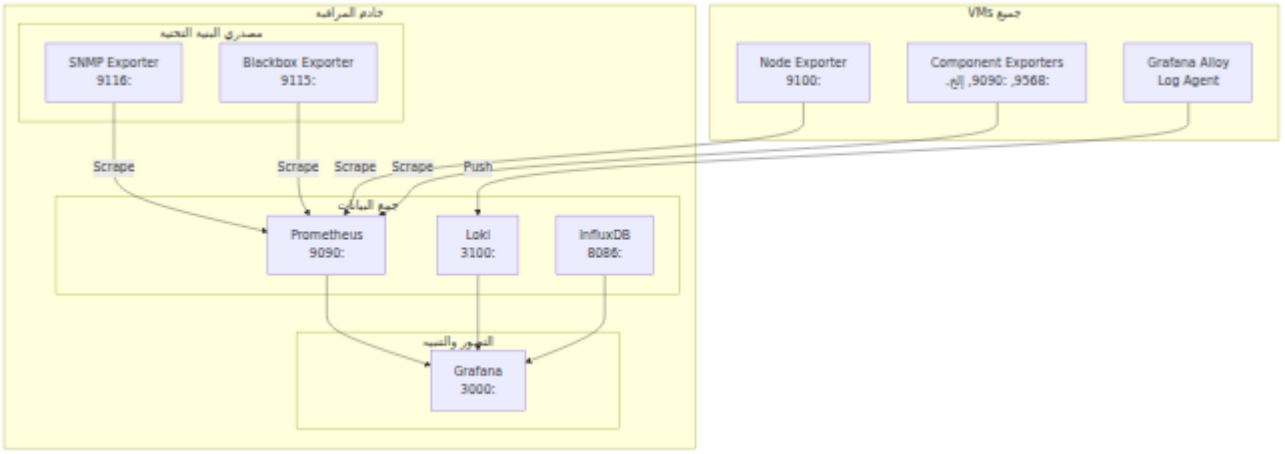
الوثائق ذات الصلة

- مرجع التكوين
- تكوين ملف المضيفين

المراقبة والمراقبة

نظرة عامة

يتضمن OmniCore، مجموعة شاملة من أدوات المراقبة والمراقبة التي توفر جمع المقاييس، وتجميع السجلات، والتصور، والتنبيه. يتم نشر النظام تلقائيًا بواسطة دور المراقبة.



المكونات

المكون	الغرض	المنفذ	الاحتفاظ بالبيانات
Prometheus	تخزين مقاييس السلاسل الزمنية	9090	30 5 يومًا أو جيجابايت
Loki	تجميع السجلات	3100	50 7 أيام أو جيجابايت
InfluxDB	مقاييس RAN (نوكيا)	8086	30 يومًا
Grafana	التصور والتنبيه	3000	غير متاح
Node Exporter	مقاييس المضيف	9100	غير متاح
SNMP Exporter	مقاييس أجهزة الشبكة	9116	غير متاح
Blackbox Exporter	اختبارات ICMP/HTTP	9115	غير متاح

مصادر البيانات

Prometheus

يقوم Prometheus بجمع المقاييس من جميع مكونات OmniCore كل دقيقة واحدة.

معرف مصدر البيانات: `omnicore_prometheus`

الأهداف المجمعة

اسم الوظيفة	مجموعة الجرد	المنفذ	المقاييس
`Node Exporter`	`all`	9100	وحدة المعالجة المركزية للمضيف، الذاكرة، القرص، الشبكة
`MMEs`	`mme`	9568	عدد الجلسات، معدلات الإرفاق/الفصل
`HSS`	`hss`	9568	طلبات المصادقة، عمليات البحث عن المشتركين
`SGW-C`	`sgw`	9568	عدد الحاملات، رسائل GTP-C
`PGW-C`	`pgwc`	9090	اتصالات PDN، تخصيصات IP
`UPF Standalone`	`upf`	9090	عدد الحزم، عرض النطاق
`CSCF`	`pcscf`, `scscf`, `icscf`	9090	التسجيلات، المعاملات
`ApplicationServer FreeSWITCH`	`applicationserver`	9090	عدد المكالمات، استخدام القناة
`DRA`	`dra`	9568	قرارات التوجيه، حالة النظراء
`OmniMessage`	`omnimessage`	9568	جلسات SMS، تسليم SMPP
`OmniSS7`	`omniss7`	8080	مقاييس بوابة SS7/SIGTRAN

اسم الوظيفة	مجموعة الجرد	المنفذ	المقاييس
`KeyDB`	`ocs`	9121	استخدام الذاكرة، العمليات/ثانية
`CGrateS`	`ocs`	2080	التصنيف، الشحن، إحصائيات CDR

مصدري البنية التحتية

المصدر	الأهداف	متغير التكوين
SNMP (MikroTik)	أجهزة التوجيه/المفاتيح	`mikrotik.hosts`
SNMP (iDRAC)	خوادم Dell	`idrac.hosts`
SNMP (Synology)	أجهزة NAS	`synology.hosts`
SNMP (SAF)	روابط الميكروويف	`saf.hosts`
SNMP (Cisco)	المفاتيح	`cisco.hosts`
Blackbox ICMP	جميع المضيفين + 8.8.8.8	تلقائي
VMware	مجموعات vCenter	`vcenter_ip`, `vcenter_password`
Proxmox	مجموعات PVE	`proxmoxServers`

Loki

يتلقى Loki السجلات من وكلاء Grafana Alloy الذين يعملون على جميع VMs.

معرف مصدر البيانات: `omnicore_loki`

انظر Loki/Alloy. للحصول على تفاصيل تكوين التسجيل المركزي

تسميات السجلات

التسمية	الوصف	المثال
`hostname`	اسم مضيف VM المصدر	`customer-mme01`
`component`	نوع المكون	`mme`, `cscf`, `ocs`
`unit`	اسم وحدة systemd	`omnimme.service`
`level`	شدة السجل	`info`, `error`

InfluxDB

تقوم InfluxDB بتخزين بيانات السلاسل الزمنية لحالات استخدام محددة تتطلب احتفاظًا أطول أو أنماط استعلام مختلفة.

معرف مصدر البيانات: `omnicore_influxdb`

قواعد البيانات

قاعدة البيانات	الغرض	الاحتفاظ
`nokia-monitor`	مقاييس أداء RAN من نوكيا	30 يومًا
`dra`	إحصائيات توجيه DRA	365 يومًا
`Omnicharge_TAP3`	مقاييس ملف TAP للتجوال	90 يومًا

لوحات معلومات Grafana

تنظيم لوحة المعلومات

تُنظم لوحات المعلومات في مجلدات حسب المنطقة الوظيفية:

المجلد	المحتويات
BSS	لوحات معلومات CGrateS، KeyDB، الشحن
EPC	لوحات معلومات MME، SGW، PGW، UPF، HSS، DRA
IMS	لوحات معلومات CSCF، خادم التطبيق، OmniMessage
Infrastructure	Node Exporter، SNMP، مراقبة الشبكة
RAN	لوحات معلومات أداء نوكيا/OmniRAN
Logs	عارضو السجلات الخاصة بالمكونات

تحميل لوحة المعلومات (استنادًا إلى API)

بدلاً من توفيرها من خلال Grafana يتم تحميل لوحات ا��معلومات عبر واجهة برمجة تطبيقات الملفات. يتيح ذلك لوحات المعلومات أن تكون:

- معدلة من خلال واجهة مستخدم Grafana
- معدلة عبر API
- خاضعة للتحكم في الإصدارات في مستودع Ansible

تحميل لوحات المعلومات

يتم تحميل لوحات المعلومات تلقائيًا أثناء نشر Ansible. لإعادة التحميل يدويًا:

```
# من وحدة التحكم 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

# تحديث اللوحات الموجودة بالقوة
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
```

ملفات مصدر لوحة المعلومات

يتم تخزين ملفات JSON الخاصة بلوحة المعلومات في مستودع 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
    └── ...
```

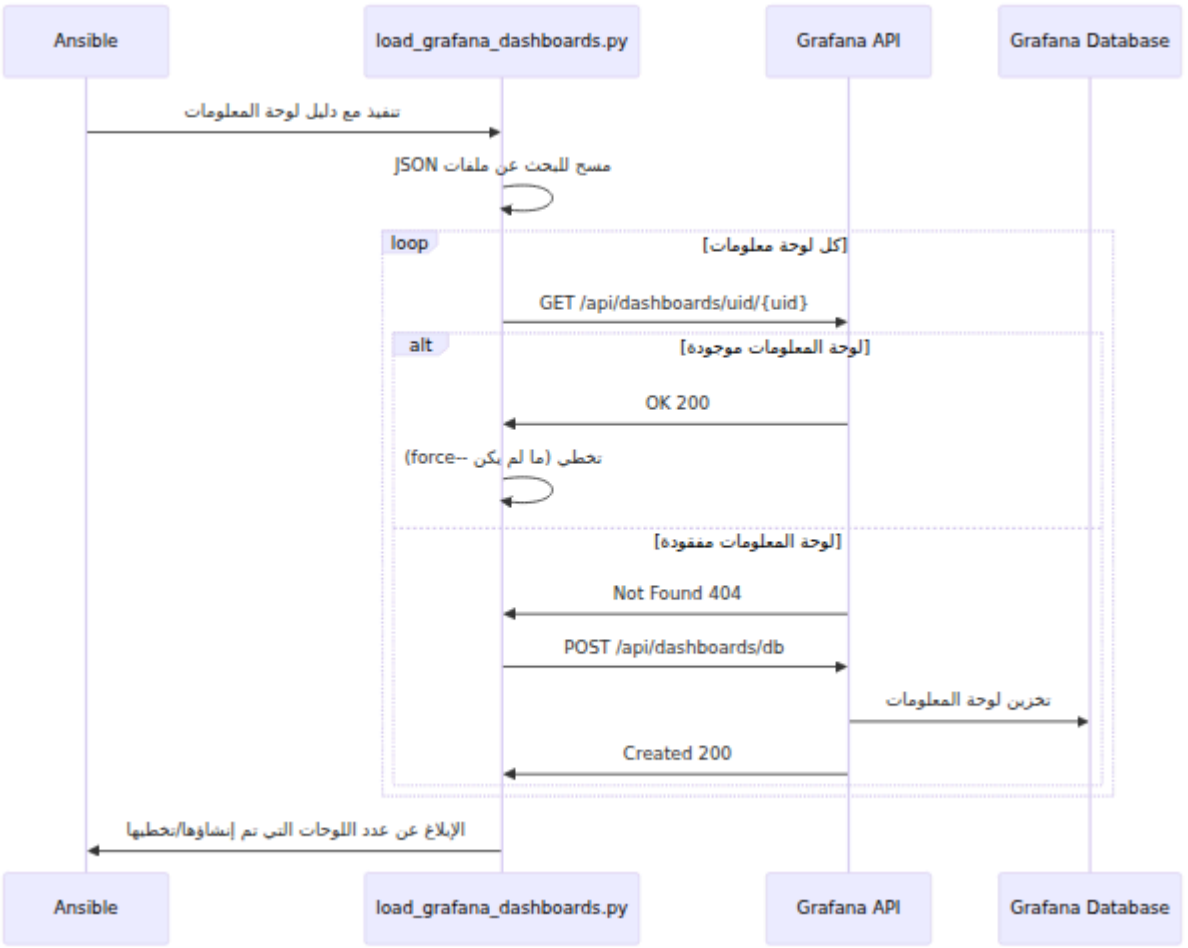
النسخ الاحتياطي للوحة المعلومات

قيد التشغيل إلى المستودع من أجل Grafana قم بتصدير لوحات المعلومات والتنبيهات من مثيل التحكم في الإصدارات:

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

هذا يصدر:

- جميع لوحات المعلومات إلى `roles/monitoring/templates/grafana/dashboards/{folder}/*.json` (مشاركة عبر العملاء)
- قواعد التنبيه إلى `hosts/Omnicore_{CUSTOMER}/group_vars/grafana/alerts/all_alert_rules.json`
- نقاط الاتصال إلى `hosts/Omnicore_{CUSTOMER}/group_vars/grafana/alerts/contact_points.json`
- سياسات الإشعار إلى `hosts/Omnicore_{CUSTOMER}/group_vars/grafana/alerts/notification_policies.json`



سلوك التصدير

- يتجاهل الحقول المتقلبة مثل (`id` و `version`) يكتب فقط الملفات التي تغيرت

- يحافظ Grafana على هيكل المجلدات الذي يتطابق مع تنظيم
- يطابق عناوين لوحات المعلومات بأسماء ملفات متسقة

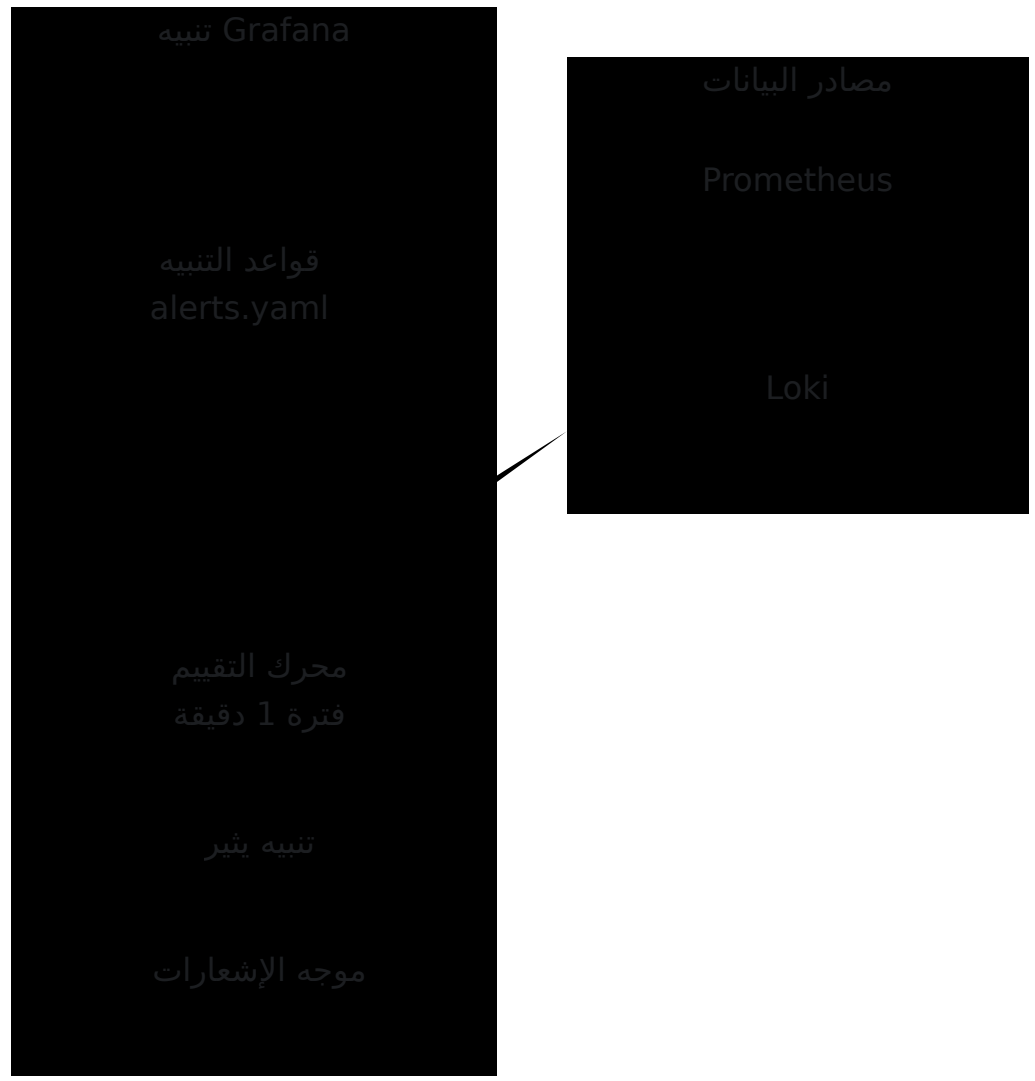
لوحات المعلومات المخصصة

يمكن وضع لوحات المعلومات الخاصة بالعميل في `group_vars/grafana/dashboards/` وسيتم تحميلها جنبًا إلى جنب مع لوحات المعلومات القياسية:

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

التنبيه

الهيكلية


تنبيه Grafana
مصادر البيانات
Prometheus
Loki
قواعد التنبيه
alerts.yaml
محرك التقييم
فترة 1 دقيقة
تنبيه يثير
موجه الإشعارات
نقاط الاتصال
Slack Webhook
Google Chat
البريد الإلكتروني
Pushover

تحميل التنبيهات

يتم تحميل التنبيهات عبر واجهة برمجة تطبيقات Grafana من صادرات كل عميل بدلاً من ملف YAML ثابت واحد. في وقت النشر، يتم إزالة ملف توفير التنبيه الثابت (`/etc/grafana/provisioning/alerting/alerts.yaml`) بحيث يمكن تعديل التنبيهات عبر واجهة المستخدم/API.

المسار الرئيسي يحمل قواعد التنبيه الخاصة بكل عميل، ونقاط الاتصال، وسياسات الإشعار من `hosts/Omnicore_<CUSTOMER>/group_vars/grafana/alerts/` باستخدام `load_grafana_from_export.py`:

```
python3 roles/monitoring/files/load_grafana_from_export.py \
    --url http://<monitoring-ip>:3000 \
    --user admin \
    --password <grafana_admin_password> \
    --alerts-dir
hosts/Omnicore_<CUSTOMER>/group_vars/grafana/alerts \
    --dashboards-dir roles/monitoring/templates/grafana/dashboards
\
    --force
```

إذا لم توجد تنبيهات خاصة بالعميل، فإن الدور يتراجع إلى تحميل قالب الدور `alerts.yaml` عبر `load_grafana_alerts.py`:

```
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
```

قواعد التنبيه الافتراضية

التنبيه	المجلد	الشرط	لمدة
استخدام مساحة القرص 90%	البنية التحتية	نظام الملفات الجذر > 90%	5 دقائق
CPU 90 فوق%	البنية التحتية	استخدام CPU > 90%	5 دقائق
حمل النظام فوق 90%	البنية التحتية	متوسط الحمل > 90%	5 دقائق
المضيف متوقف	البنية التحتية	الهدف غير Prometheus متاح	5 دقائق
انخفاض 40% في مشتركين MME	EPC	انخفاض عدد الجلسات 40%	30 ثانية
مشتركين على 0 MME	EPC	لا يوجد مشتركين متصلين	5 دقائق
انخفاض 40% في مشتركين IMS	IMS	انخفاض تسجيلات P-CSCF 40%	30 ثانية
أقل من 4 MOS متوسط	IMS	جودة الصوت متدهورة	5 دقائق
انخفاض عدد eNodeB	RAN	انخفاض عدد eNB المتصلة	1 دقيقة
تأخير استجابة القطر مرتفع	EPC	ارتفاع زمن الاستجابة P95	5 دقائق

هيكل قاعدة التنبيه

قواعد التنبيه في `alerts.yaml`:

```yaml
apiVersion: 1
groups:
  - orgId: 1
    name: مجموعة التنبيهات
    folder: البنية التحتية
    interval: 1m
    rules:
      - uid: alert-lowDiskSpace
        title: استخدام مساحة القرص 90% على المضيف
        condition: C
        data:
          - refId: A
            datasourceUid: omnicore_prometheus
            model:
              expr: |
                100 - ((node_filesystem_avail_bytes{job="Node
Exporter",mountpoint="/"} * 100)
                / node_filesystem_size_bytes{job="Node
Exporter",mountpoint="/"})
          # ... تعبيرات العتبة
        noDataState: OK
        execErrState: Error
        for: 5m
        annotations:
          summary: تجاوز استخدام مساحة القرص 90% على المضيف
        labels:
          type: resource
```

تكوين نقاط الاتصال

تُكوَّن نقاط الاتصال عبر متغيرات الجرد:

```yaml
# في group_vars/all.yml أو متغيرات خاصة بالعميل
monitoring_data:
  contactPoints:
    - orgId: 1
      name: الافتراضي
      receivers:
        # تكامل Slack
        - uid: slack-alerts
          type: slack
          disableResolveMessage: false
          settings:
            url: "https://hooks.slack.com/services/xxx/yyy/zzz"

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

تشمل أنواع نقاط الاتصال المدعومة Slack وGoogle Chat والبريد الإلكتروني وPushover (انظر `roles/monitoring/templates/grafana/contact-points.yaml.j2`). يتم تكوين Pushover عبر كتلة `notifications.pushover` (مع `enabled` و`api_token` و`user_key`).

سيا��ات الإشعار

توجه سياسات الإشعار التنبيهات إلى نقاط الاتصال المناسبة:

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

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

مرجع التكوين

تكوين Prometheus

يقع في `/etc/prometheus/prometheus.yml` على خادم المراقبة.

المعلمة	الافتراضي	الوصف
`scrape_interval`	1m	كم مرة يتم جمع الأهداف
`evaluation_interval`	1m	كم مرة يتم تقييم قواعد التسجيل
`scrape_timeout`	50s	مهلة طلبات الجمع

تكوين Grafana

يقع في `/etc/grafana/grafana.ini` على خادم المراقبة.

الإعدادات الرئيسية التي تم تكوينها بواسطة دور المراقبة:

الإعداد	القيمة	الوصف
`allow_embedding`	true	تمكين التضمين في iframes
`provisioning`	`/etc/grafana/provisioning`	دليل التوفير

`admin_password` يتم ترك سطر) `grafana.ini` لم يتم تعيين كلمة مرور المسؤول في الذي يتم تمريره إلى ، `grafana_admin_password` بدلاً من ذلك، يتم تطبيقها عبر متغير .(معلقًا مهام محمل API (محملات لوحة المعلومات/التنبيه) أثناء النشر.

تنبيه البريد الإلكتروني (SMTP)

يتطلب تنبيه البريد الإلكتروني إعداد بيانات اعتماد SMTP في `group_vars` (انظر `roles/monitoring/templates/grafana/grafana.ini`):

المتغير	مطلوب	الافتراضي	الوصف
`grafana_smtp_user`	نعم	*(لا شيء)*	اسم مستخدم SMTP. لا يوجد افتراضي. إذا لم يتم تعيينه، يفشل النشر بصوت عالٍ بدلاً من إرسال سر.
`grafana_smtp_password`	نعم	*(لا شيء)*	كلمة مرور SMTP. لا يوجد افتراضي. إذا لم يتم تعيينه، يفشل النشر.
`grafana_smtp_host`	لا	`in-v3.mailjet.com:587`	مضيف SMTP:المنفذ.
`grafana_smtp_enabled`	لا	`true`	ما إذا كان SMTP مفعلًا.

يتم اشتقاق `from_address` و`from_name` من `customer_legal_name` (مثل `alerts+<customer_legal_name>@omnitouch.com.au` و`Omnitouch Alerts - <customer_legal_name>`).

تكوين Loki

انظر التسجيل المركزي للحصول على إعدادات الاحتفاظ والتخزين الخاصة بـ Loki.

تكوين InfluxDB

يقع في `/etc/influxdb/influxdb.conf` على خادم المراقبة.

قاعدة البيانات	سياسة الاحتفاظ	المدة
`nokia-monitor`	`autogen`	30 يومًا
`dra`	`autogen`	365 يومًا
`Omnicharge_TAP3`	`autogen`	90 يومًا

منافذ الخدمة

الخدمة	المنفذ	البروتوكول	الوصف
Grafana	3000	HTTP	واجهة لوحة المعلومات و API
Prometheus	9090	HTTP	تخزين المقاييس والاستعلام
Loki	3100	HTTP	استيعاب السجلات والاستعلام
InfluxDB	8086	HTTP	واجهة برمجة تطبيقات InfluxDB
Node Exporter	9100	HTTP	مقاييس المضيف
SNMP Exporter	9116	HTTP	وكيل مقاييس SNMP
Blackbox Exporter	9115	HTTP	مقاييس الاختبار
Alloy	12345	HTTP	واجهة Alloy والمقاييس

العمليات الشائعة

عر�ض المقاييس

1. افتح Grafana على `http://<monitoring-ip>:3000`
2. انتقل إلى **لوحات المعلومات** واختر المجلد المناسب
3. استخدم محدد نطاق الوقت لضبط نافذة العرض

البحث في السجلات

1. افتح Grafana على `http://<monitoring-ip>:3000`
2. انتقل إلى **Loki** واختر مصدر بيانات **استكشاف**
3. استخدم استعلامات LogQL:

```
# جميع السجلات من مضيف معين
{hostname="customer-mme01"}

# الأخطاء MME من جميع مكونات
{component="mme"} |~ "(?i)error"

# البحث عن IMSI معين
{component="hss"} |= "123456789012345"
```

إنشاء تنبيهات مخصصة

تدفق العمل الموصى به هو إنشاء/تعديل التنبيهات في واجهة Grafana، ثم تصديرها إلى دليل `group_vars/grafana/alerts/` الخاص بالعميل (انظر النسخ الاحتياطي للوحة المعلومات) بحيث تكون خاضعة للتحكم في الإصدارات وتُعاد تطبيقها في النشر التالي.

لتحميل التنبيهات الخاصة بالعميل يدويًا:

```
python3 roles/monitoring/files/load_grafana_from_export.py \
    --url http://<monitoring-ip>:3000 \
    --user admin --password <password> \
    --alerts-dir
hosts/Omnicore_<CUSTOMER>/group_vars/grafana/alerts \
    --dashboards-dir roles/monitoring/templates/grafana/dashboards
\
    --force
```

يتم استخدام قالب الدور `roles/monitoring/templates/grafana/provisioning/alerting/alerts.yaml` فقط كخيار احتياطي عندما لا يكون لدى العميل أي تنبيهات مصدرة.

النسخ الاحتياطي للوحات المعلومات

بعد إجراء تغييرات في واجهة Grafana، قم بتصديرها إلى المستودع:

```
cd roles/monitoring
python3 export_grafana.py --customer <CUSTOMER> --url
http://<monitoring-ip>:3000
git add ../../hosts/Omnicore_<CUSTOMER>/group_vars/grafana/
git commit -m "تحديث التنبيهات من الإنتاج"
```

تكون حجة `--customer` مطلوبة. يقوم هذا البرنامج النصي بتصدير قواعد التنبيه، ونقاط الاتصال، وسياسات الإشعار إلى دليل `group_vars/grafana/alerts/` الخاص بالعميل؛ لم يعد يصدر لوحات المعلومات، والتي تتم مشاركتها عبر العملاء في دليل الدور (`roles/monitoring/templates/grafana/dashboards/`).

استكشا�ء الأخطاء وإصلاحها

متوقف Prometheus الهدف

الأعراض: تظهر لوحة المعلومات "لا توجد بيانات" أو حالة الهدف تظهر متوقفة

الأسباب المحتملة:

- الخدمة غير قيد التشغيل على المضيف المستهدف
- جدار الحماية يحظر منفذ المصدر
- حل اسم المضيف غير صحيح

الحل:

1. تحقق مما إذا كانت الخدمة تعمل على الهدف:

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

2. تحقق من الاتصال من خادم المراقبة:

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

3. تحقق من صفحة أهداف Prometheus: `http://<monitoring-ip>:9090/targets`

لوحة المعلومات لا تُحمّل

الأعراض: تظهر لوحة المعلومات رمز تحميل أو خطأ

الأسباب المحتملة:

- فشل اتصال مصدر البيانات
- مهلة الاستعلام
- تلف JSON لوحة المعلومات

الحل:

1. اختبار مصدر البيانات في Grafana: **التكوين** → **مصادر البيانات** → **اختبار**
2. تحقق من سجلات Grafana: `journalctl -u grafana-server -f`
3. حاول إعادة تحميل لوحة المعلومات من المستودع

التنبيهات لا تُطلق

الأعراض: الشرط مستوفى ولكن لم يتم تلقي إشعار

الأسباب المحتملة:

- التنبيه متوقف
- تكوين نقطة الاتصال غير صحيح
- سياسة الإشعار لا تطابق

الحل:

1. تحقق من حالة التنبيه في Grafana: **التنبيه** → **قواعد التنبيه**
2. تحقق من اختبار نقطة الاتصال: **التنبيه** → **نقاط الاتصال** → **اختبار**
3. مراجعة توجيه سياسة الإشعار

Grafana لا يمكنها الاتصال بمصدر البيانات

الأعراض: Grafana في "بوابة سيئة" أو مهلة الاتصال

الأسباب المحتملة:

- خدمة Prometheus/Loki/InfluxDB متوقفة
- جدار الحماية يحظر اتصالات localhost
- الخدمة مرتبطة بواجهة خاطئة

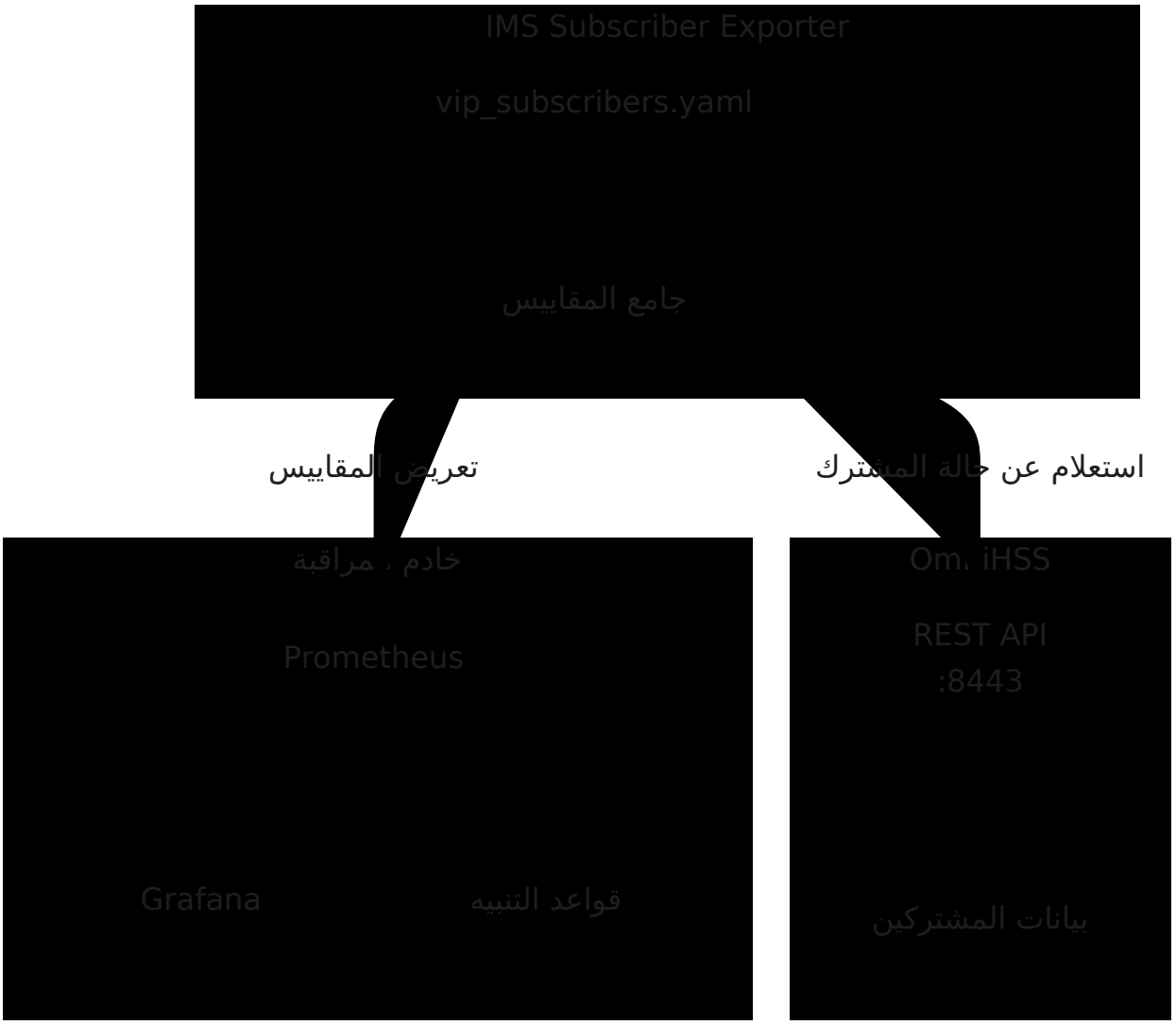
الحل:

1. تحقق من حالة الخدمة:
   ```
   systemctl status prometheus loki influxdb
   ```
2. تحقق من أن الخدمة تستمع:
   ```
   ss -tlnp | grep -E '9090|3100|8086'
   ```
3. اختبار الاتصال المحلي:
   ```
   curl http://localhost:9090/api/v1/status/config
   curl http://localhost:3100/ready
   ```

مراقبة المشتركين VIP

تتبع حالة في الوقت الفعلي للمشتركين الحرجين. يراقب النظام VIP تقدم مراقبة المشتركين للمشتركين ذوي الأولوية العالية مثل خدمات UE وقابلية الوصول إلى EPC، وإرفاق IMS، تسجيل الطوارئ، والمستشفيات، والبنية التحتية الرئيسية.

الهيكلية



أنواع الخدمات

يتم تصنيف المشتركين حسب نوع الخدمة المتوقع، مما يحدد تقييم الصحة:

النوع	مطلوب IMS	مطلوب EPC	حالة الاستخدام
`full`	نعم	نعم	مشتركون عاديون مع صوت وبيانات
`voip_only`	نعم	لا	أجهزة VoIP فقط (هواتف IP، هواتف برمجية)
`data_only`	لا	نعم	أجهزة بيانات فقط (أجهزة توجيه، IoT، M2M)

تقييم الصحة:

- **full**: صحي عندما يكون كل من IMS مسجلاً (`assigned_scscf` ليس فارغًا) وEPC (`last_seen_mme` ليس فارغًا) متصلًا
- **voip_only**: صحي عندما يكون IMS مسجلاً (`assigned_scscf` ليس فارغًا)
- **data_only**: صحي عندما يكون EPC متصلًا (`last_seen_mme` ليس فارغًا)

التكوين

يتم تكوين المشتركين VIP في متغيرات مجموعة المضيف:

الملف: `hosts/<customer>/group_vars/vip_subscribers.yaml`

```yaml
vip_subscriber_monitoring:
  hss_url: "https://10.80.12.140:8443"

  # إعدادات المراقبة
  scrape_interval_seconds: 30
  ping_timeout_seconds: 2
  ping_count: 2

  # المشتركين الذين يجب مراقبتهم
  subscribers:
    # مشتركون بخدمة كام��ة (IMS + EPC)
    - imsi: "001010000000001"
      label: "خدمات الطوارئ"
      type: "full"

    - imsi: "001010000000002"
      label: "البنية التحتية الحرجة"
      type: "full"

    # خدمات بيانات فقط (أجهزة توجيه، IoT - لا يتوقع IMS)
    - imsi: "001010000000003"
      label: "جهاز التوجيه في الموقع"
      type: "data_only"

    # خدمات VoIP فقط (يتوقع IMS، لا حامل بيانات)
    - imsi: "001010000000004"
      label: "هواتف IP"
      type: "voip_only"
```

معلمات التكوين

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`hss_url`	سلسلة	نعم	-	برمجة URL عنوان تطبيقات OmniHS (HTTPS)
`scrape_interval_seconds`	عدد صحيح	لا	30	رة يتم استعلام حالة المشترك
`ping_timeout_seconds`	عدد صحيح	لا	2	IP p مهلة لاختبارات UE
`ping_count`	عدد صحيح	لا	2	سلة ping عدد حزم
`blackbox_exporter_url`	سلسلة	لا	-	لمصدر URL عنوان ختبارات blackbox ping (مثل `http://pcscf01` ICM يستخدم اختبار محلي ping بدلاً من
`subscribers`	قائمة	نعم	-	ة بالمشتركين الذين يجب مراقبتهم

معلمات المشترك

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`imsi`	سلسلة	نعم	-	IMSI (15 رقمًا) المشترك
`label`	سلسلة	لا	IMSI	اسم قابل للقراءة البشرية للعرض
`type`	سلسلة	لا	`full`	نوع الخدمة: `full`، `voip_only`، أو `data_only`

ولا تحتاج إلى HSS تلقائيًا من استجابة واجهة برمجة التطبيقات MSISDN **ملاحظة**: يتم استرداد تكوينها.

المقاييس

يعرض المصدر المقاييس على المنفذ 9550 في `/metrics`.

مقاييس الحالة

المقياس	النوع	الوصف
`vip_subscriber_service_healthy`	Gauge	1 إذا كان المشترك يلبي معايير الصحة لنوع خدمته، 0 خلاف ذلك
`vip_subscriber_ims_registered`	Gauge	1 إذا كان لدى المشترك S-CSCF المعين، 0 خلاف ذلك
`vip_subscriber_epc_registered`	Gauge	1 إذا كان لدى المشترك إرفاق نشط، 0 خلاف ذلك MME
`vip_subscriber_ue_ip_reachable`	Gauge	1 إذا استجاب IP UE لاختبار ping، 0 خلاف ذلك
`vip_subscriber_hss_reachable`	Gauge	1 إذا أعادت واجهة برمجة التطبيقات HSS بيانات صالحة، 0 خلاف ذلك
`vip_subscriber_enabled`	Gauge	1 إذا كان حساب المشترك مفعلًا في HSS، 0 خلاف ذلك

مقاييس العمر

المقياس	النوع	الوصف
`vip_subscriber_ims_registration_age_seconds`	Gauge	ثوانٍ منذ آخر تسجيل IMS (-1 إذا لم يكن مسجلاً)
`vip_subscriber_epc_registration_age_seconds`	Gauge	ثوانٍ منذ آخر تحديث موقع EPC (-1 إذا لم يكن متصلًا)

مقياس المعلومات

المقياس	النوع	الوصف
`vip_subscriber_info`	Gauge	دائمًا 1، يحمل جميع الحالة والمعلومات كعلامات

التسميات على جميع المقاييس:

التسمية	الوصف	المثال
`imsi`	IMSI المشترك	`001010000000002`
`label`	اسم قابل للقراءة البشرية	`البنية التحتية الحرجة`
`msisdn`	رقم الهاتف (من HSS)	`61400000002`
`type`	نوع الخدمة	`full`، `voip_only`، `data_only`

تسميات إضافية على `vip_subscriber_info`:

التسمية	الوصف	المثال
`healthy`	حالة صحة الخدمة	`1` أو `0`
`ims`	حالة تسجيل IMS	`1` أو `0`
`epc`	حالة إرفاق EPC	`1` أو `0`
`ping`	قابلية الوصول إلى IP UE	`1` أو `0`
`assigned_scscf`	اسم مضيف S-CSCF	`scscf01.ims.example.com`
`last_seen_mme`	اسم مضيف MME	`mme02.epc.example.com`
`ue_ip`	عنوان IP المعين لـ UE	`100.72.83.20`

استعلامات المثال

```
# عدد المشتركين VIP الصحيين
count(vip_subscriber_service_healthy == 1)

# عدد المشتركين VIP غير الصحيين
count(vip_subscriber_service_healthy == 0)

# حالة تسجيل IMS لخدمات VoIP
vip_subscriber_ims_registered{type=~"voip_only|full"}

# حالة إرفاق EPC لخدمات البيانات
vip_subscriber_epc_registered{type=~"data_only|full"}

# المشتركين الذين لديهم تسجيل IMS (<1 ساعة) قديم
vip_subscriber_ims_registration_age_seconds > 3600
```

لوحة المعلومات

توفر لوحة معلومات **VIP IMS** رؤية في الوقت الفعلي لحالة المشتركين **مشتركي** VIP.

الموقع: Grafana → IMS → مشتركي VIP IMS

عنوان URL للوحة المعلومات: `/d/ims-vip-subscribers/`

الألواح

اللوحة	الوصف
الخدمات الصحية	عدد المشتركين الذين يلبيون معايير الصحة
الخدمات غير الصحية	عدد المشتركين الذين يفشلون في معايير الصحة
IMS مسجل	عدد المشتركين الذين لديهم تسجيل IMS نشط
EPC مسجل	عدد المشتركين الذين لديهم إرفاق EPC نشط
UE قابلية الوصول إلى	عدد المشتركين الذين لديهم IP UE قابل للاختبار
إجمالي المراقبين	العدد الإجمالي للمشتركين VIP المكونين
VIP حالة المشترك	جدول يعرض جميع المشتركين مع أعمدة الحالة
صحة الخدمة مع مرور الوقت	خط زمني للحالة يظهر تاريخ الصحة

التنبيه

تثير قواعد التنبيه عندما تصبح خدمات المشتركين VIP غير صحية.

خدمة المشترك VIP غير صحية

التنبيه: `VIP Subscriber Service Unhealthy` **لمدة**: 1 دقيقة **الخطورة**: حرجة
الشرط: `vip_subscriber_service_healthy == 0`

التعليقات التوضيحية:

- **الملخص**: `VIP Subscriber {{ $labels.label }} غير صحي`
- **الوصف**: يتضمن نوع الخدمة والمعرف المناسب:
  - خدمات VoIP: الاسم و IMSISDN
  - خدمات البيانات: الاسم و IMSI

    - الاسم و IMSI وMSISDN الخدمات الكاملة:

أمثلة على أوصاف التنبيهات:

- `خدمة هواتف IP (voip_only) متوقفة. MSISDN: 61400000001`
- `خدمة جهاز التوجيه في الموقع (data_only) متوقفة. IMSI: 001010000000003`
- `خدمة البنية التحتية الحرجة (full) متوقفة. MSISDN: 61400000002 IMSI: 001010000000002`

إضافة مشتركين VIP جدد

1. تحرير ملف التكوين:

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

2. نشر التكوين المحدث:

```
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. تحقق من ظهور المشترك الجديد في المقاييس:

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

الوثائق ذات الصلة

- تفاصيل تكوين Alloy وLoki: التسجيل المركزي

- معمارية النشر: المعمارية العامة للنظام
- تكوين ملف المضيفين: تكوين الجرد

تكوين Netplan

نظرة عامة

يمكن لـ OmniCore تكوين واجهات الشبكة تلقائيًا على الأجهزة الافتراضية الموزعة باستخدام netplan. هذا مفيد لـ:

- إعداد واجهة الإدارة الأساسية (eth0)
- إضافة واجهات ثانوية لعناوين IP المخصصة لحركة المرور، أو اتصالات التبادل العامة
- تكوين المسارات الثابتة لوجهات محددة

تمكين تكوين Netplan

لتمكين تكوين netplan التلقائي لمضيف، أضف المتغير `netplan_config` الذي يشير إلى قالب Jinja2 في مجلد `group_vars` الخاص بك:

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

سيتم الحصول على القالب من `hosts/<customer>/group_vars/netplan.yaml.j2`.

مرجع القالب

لا يوجد `netplan.yaml.j2` مشترك أو قياسي. يقوم كل عميل بإنشاء خاص به وتخ��لف هذه القوالب - `netplan.yaml.j2` في `hosts/<customer>/group_vars/`، عند DNS، اسم الواجهة الأساسية، مجال البحث وتسمية الواجهة الثانوية تختلف حسب العميل. إدخال عميل جديد، أنشئ قالبًا خاصًا بالعميل يتناسب مع تخطيط واجهته بدلاً من نسخ قالب عميل آخر حرفيًا.

فيما يلي قالب عميل واحد كمثال. إنه توضيحي، وليس مواصفة - العملاء الآخرون يختلفون. تم إضافة تعليقات هنا للتوضيح:

```
network:
  version: 2
  ethernets:
    # تستخدم - الواجهة الأساسية ansible_host و gateway من الجرد
    eth0:
      addresses:
      - "{{ ansible_host }}/{{ mask_cidr | default(24) }}"
      nameservers:
        addresses:
{% if 'dns' in group_names %}
        # إذا كان هذا المضيف هو خادم DNS، استخدم DNS الخارجي لتجنب
الاعتماد الدائري
        - 8.8.8.8
{% else %}
        # خلاف ذلك، استخدم خوادم DNS من مجموعة 'dns' في الجرد
{% for dns_host in groups['dns'] | default([]) %}
        - {{ hostvars[dns_host]['ansible_host'] }}
{% endfor %}
{% endif %}
        # مجال البحث DNS - لا خاص بالعميل (يختلف حسب العميل؛ بعضهم
يحددون أي شيء)
        search:
        - customer.example
      routes:
      - to: "default"
        via: "{{ gateway }}"

{% if secondary_ips is defined %}
    # حلقة عبر الواجهات الثانوية dict secondary_ips من الجرد
    # هذا المثال يسميها ens19، ens20، ens21... (18 + loop.index).
هذا
    # خاص بالقالب؛ انظر "تسمية الواجهة" أدناه.
{% for nic_name, nic_config in secondary_ips.items() %}
    ens{{ 18 + loop.index }}:
      addresses:
      # القناع يأتي من mask_cidr الخاص بكل NIC (يكون افتراضيًا 24)
      - "{{ nic_config.ip_address }}/{{ nic_config.mask_cidr |
default(24) }}"
{% if nic_config.routes is defined %}
      # مسارات ثابتة لهذه الواجهة - كل مسار يستخدم بوابة هذه
الواجهة
      routes:
{% for route in nic_config.routes %}
```

```
      - to: "{{ route }}"
        via: "{{ nic_config.gateway }}"
{% endfor %}
{% endif %}
{% endfor %}
{% endif %}
```

النقاط الرئيسية:

- `ansible_host` و `gateway` تأتي من إدخال الجرد الخاص بالمضيف
- `dns` ديناميكيًا من المضيفين في مجموعة DNS يتم سحب خوادم
- مجال البحث DNS (يختلف؛ بعض العملاء يتجاهلونه خاص بالعمي)
- تختلف تسمية الواجهة الثانوية حسب العميل (بعضهم يستخدم `ens19`، `ens20`، ...؛ انظر أدناه)
- يمكن أن يكون لكل IP ثانوي قناع خاص به (`mask_cidr`)، وبوابة، ومسارات ثابتة

تكوين الواجهة الأساسية

يتم تكوين الواجهة الأساسية (eth0) تلقائيًا باستخدام:

- `ansible_host` - عنوان IP
- `gateway` - البوابة الافتراضية
- `mask_cidr` - قناع الشبكة (يكون افتراضيًا 24)

يتم تعيين خوادم DNS تلقائيًا إلى:

- المضيفين في مجموعة `dns` (تستخدم عناوين IP الخاصة بهم `ansible_host`)
- تتراجع إلى DNS `8.8.8.8` إذا كان المضيف هو نفسه خادم

الرابط MTU (`link_mtu`)

يمكن لدور netplan، **الرابط** للواجهة الأساسية بشكل مستقل عن قالب **MTU** تثبيت `common`. عند تعيينه، فإنه يقوم بإنشاء drop-in netplan عبر متغير الجرد `link_mtu` على الواجهة الأساسية للمضيف ويطبقه. هذا موجود (`/etc/netplan/999-link-mtu.yaml`) لأن الأجهزة الافتراضية تأتي من قالب VMware/Proxmox مع أي MTU تم شحنه من مجموعة

مما يؤدي إلى عدم تطابق مع بقية النسيج ويؤدي إلى فقدان، (أحيانًا jumbo/9000) المنفذ المصدر الحزم الكبيرة SCTP/SIP عندما يكون PMTUD معطلًا.

بدمج جميع netplan عمداً بـ `-999` بحيث يتم فرزها **أخيرًا**: يقوم drop-in تم تسمية بترتيب أبجدي مع فوز آخر ملف لكل مفتاح، ويمكن أن تترك عمليات `/etc/netplan/*.yaml` التهيئة ملفات متنافسة تحدد `mtu` على نفس الواجهة (`80` ملف تكوين قديم `-ansible-config.yaml` `99` وملف `-netcfg-vmware.yaml` لمحرك VMware). سيتم تجاوز drop-in ذو الرقم الأقل بهدوء بواسطة تلك.

```
all:
  vars:
    link_mtu: 1500   # تثبيت NIC 1500 على مضيف لكل الأساسي
```

لتطبيقه على مستوى الموقع، أو لكل مضيف للتجاوز. إذا لم يتم تعريف `all.vars` قم بتعيينه في `link_mtu`، فإن الدور يترك MTU الواجهة دون تغيير.

> `link_mtu` **ليس** `mtu`**.** المتغير `mtu` على المستوى الأعلى هو MTU الحمولة GTP/tunnel رابط. سيكون MTU **ليس**، الذي تستهلكه أدوار packet-core (مثل 1400/1430/1300) الواجهة MTU المخصص لـ `link_mtu` الفيزيائي لتلك القيم خاطئًا. استخدم دائمًا NIC تعيين.

الواجهات الثانوية

استخدم، (العامة، التبادل، إلخ IP عناوين) للمضيفين الذين يحتاجون إلى واجهات شبكة إضافية تكوين `secondary_ips`.

المخطط

```
secondary_ips:
  <logical_name>:
    ip_address: <ip_address>
    mask_cidr: <prefix_length>   # قناع لكل - اختياري NIC، 24 افتراضيًا
    gateway: <gateway_ip>
    host_vm_network: <proxmox_bridge>
    vlanid: <vlan_id>
    nic_name: <interface_name>   # تجاوز اسم الواجهة المحدد - اختياري
                                 # (مدعوم من القوالب التي تحترم ذلك)
    routes:                      # مسارات ثابتة عبر هذه الواجهة - اختياري
      - '<destination_cidr>'
      - '<destination_cidr>'
    table_routes:                # متقدم، خاص بالقالب - اختياري
      - { to: '<cidr>', via: '<gateway>', table: <table_id> }
    routing_policy:              # متقدم، خاص بالقالب - اختياري
      from: '<source_cidr>'
      via: '<gateway>'
      table: <table_id>
      priority: <priority>       # 100 افتراضيًا، اختياري
```

> يحدد طول البادئة لتلك الواجهة الثانوية الواحدة. ي��م قراءته من المفتاح `mask_cidr` على `mask_cidr`، وليس من (`nic_config.mask_cidr`) الثانوي IP داخل إدخال المستوى الأعلى. إذا تم تجاهله، فإنه يكون افتراضيًا إلى `/24`.

> تمكّن التوجيه حسب السياسة/المصدر ولكن يتم `routing_policy` و `table_routes` عرضها فقط بواسطة القوالب التي تنفذها. التوجيه المتقدم خاص بالقالب. تأكد من أن للعميل يحترم هذه المفاتيح قبل استخدامها `netplan.yaml.j2`.

تسمية الواجهة

تسمية الواجهة الثانوية خاصة بالقالب - ليست موحدة عبر العملاء:

- بعض العملاء يسميون الواجهات حسب الفهرس: `ens19`، `ens20`، `ens21`، ... (`18 + loop.index`). عند إضافة Proxmox يتطابق هذا مع أسماء الواجهات المعينة بواسطة NICs إضافية إلى VM.
- صراحة عند تعيينه، ويتراجعون إلى `nic_config.nic_name` يستخدم عملاء آخرون خلاف ذلك. (مثل اسم قائم على الفهرس `ens(34 + loop.index)`)

يمكنك تثبيت واجهة ثانوية على اسم محدد عن طريق تعيين `nic_name`، إذا كان القالب يحترم `nic_name` على إدخال IP الخاص بالقالب الثانوي؛ خلاف ذلك، يتم تطبيق مخطط الفهرس.

مثال على التكوين

```yaml
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
          mask_cidr: 28
          gateway: 192.0.2.1
          host_vm_network: "vmbr0"
          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 الناتج الناتج عن

تم عرض التكوين أعلاه باستخدام القالب المثال، مما ينتج عنه (مجال البحث واسم DNS خاص بالعميل `ens19`/`ens20`):

```
network:
  version: 2
  ethernets:
    eth0:
      addresses:
      - "10.0.1.100/24"
      nameservers:
        addresses:
        - 10.0.1.53
        search:
        - customer.example
      routes:
      - to: "default"
        via: "10.0.1.1"
    ens19:
      addresses:
      - "192.0.2.50/28"
      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"
```

تظهر واجهة `public_ip` لأن إدخالها حدد `28/` كـ `mask_cidr: 28`، بينما تتراجع `peering_ip` إلى الافتراضي `24/`.

تكامل Proxmox

عند استخدام playbook `proxmox.yml`، يتم إنشاء NICs على الثانوية تلقائيًا VM:

1. تتم إضافة NICs الثانوية أثناء التهيئة الأولية **للأجهزة الافتراضية الجديدة**
2. تتم إضافة NICs الثانوية ويتم إعادة تشغيل VM **للأجهزة الافتراضية الموجودة** لتطبيق التغييرات

تستخدم تكوين Proxmox:

- `host_vm_network` - الجسر الذي يتم توصيل NIC به
- `vlanid` - علامة VLAN للواجهة

كيف يعمل

1. يتم عمل نسخة احتياطية من تكوينات netplan الموجودة في `/etc/netplan` ثم يتم إزالتها لتجنب التعارضات (`*.bak`)
2. يتم عرض قالب Jinja2 الخاص بالعميل إلى `/etc/netplan/01-netcfg.yaml` (وضع `0600`)
3. يقوم `netplan try --timeout 30` بتطبيق التكوين مع التراجع التلقائي إذا فقد الاتصال
4. يؤكد `wait_for_connection` أن المضيف ظل قابلًا للوصول
5. يقوم `netplan apply` بتأكيد التكوين

حالات الاستخدام الشائعة

Diameter Edge Agent (DEA) مع IP عام

```
<hostname>:
  ansible_host: 10.0.1.100          # IP الإدارة الداخلية
  gateway: 10.0.1.1
  netplan_config: netplan.yaml.j2
  secondary_ips:
    diameter_roaming:
      ip_address: 192.0.2.50        # IP عام لشركاء التجوال
      gateway: 192.0.2.1
      host_vm_network: "vmbr0"
      vlanid: "200"
      routes:
        - '198.51.100.0/24'         # شبكة شريك التجوال
```

PGW مع واجهة S5/S8

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

خادم متعدد الاتصال مع شبكات إدارة وبيانات منفصلة

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

الثانوية في القوالب IPs الإشارة إلى

يمكنك الإشارة إلى عناوين IP الثانوية في قوالب Jinja2 وملفات التكوين الأخرى.

على نفس المضيف

عند تكوين خدمة على نفس المضيف الذي يحتوي على IPs أو ثانوية، يمكنك الإشارة مباشرة استخدام `inventory_hostname`:

```
# الإشارة ��باشرة (الأبسط)
{{ secondary_ips.diameter_public_ip.ip_address }}

# أو صراحة عبر inventory_hostname (نفس النتيجة)
{{ hostvars[inventory_hostname]['secondary_ips']
['diameter_public_ip']['ip_address'] }}

# الوصول إلى خصائص أخرى
{{ secondary_ips.diameter_public_ip.gateway }}
{{ secondary_ips.diameter_public_ip.vlanid }}
```

من مضيف آخر

عندما تحتاج إلى الإشارة إلى IP ثانوي لمضيف *مختلف* (على سبيل المثال، تكوين اتصال نظير)، استخدم `hostvars` مع اسم المضيف المستهدف:

```
# الإشارة إلى المضيف الأول في مجموعة dra
{{ hostvars[groups['dra'][0]]['secondary_ips']
['diameter_public_ip']['ip_address'] }}

# حلقة عبر جميع مضيفي DRA والحصول على عناوين IP الخاصة بهم
{% for host in groups['dra'] %}
{% if hostvars[host]['secondary_ips'] is defined %}
  - {{ hostvars[host]['secondary_ips']['diameter_public_ip']
['ip_address'] }}
{% endif %}
{% endfor %}
```

مثال: تكوين نظير DRA

تكوين نظير diameter للربط إلى IP العام الخاص به:

```
# في dra_config.yaml.j2 - استخدم inventory_hostname للمضيف الحالي
peers:
  - name: external_peer
    # الربط إلى IP diameter العام لهذا المضيف
    local_ip: {{ hostvars[inventory_hostname]['secondary_ips']
['diameter_public_ip']['ip_address'] }}
    remote_ip: 198.51.100.50
    port: 3868
```

التحقق مما إذا كانت IPs الثانوية موجودة

تحقق دائمًا مما إذا كانت المتغيرات موجودة قبل استخدامها:

```
{% 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 %}
```

استكشاف الأخطاء وإصلاحها

تحقق من أسماء الواجهات

قم بتسجيل الدخول إلى VM وتحقق من أسماء الواجهات:

```
ip link show
```

الناتج المتوقع لجهاز افتراضي به واجهتين ثانويتين:

```
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> ...
```

تحقق من تكوين Netplan

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

تطبيق Netplan يدويًا

```
netplan apply
```

تصحيح Netplan

```
netplan --debug apply
```

تحقق من المسارات

```
ip route show
```

الوثائق ذات الصلة

- تكوين ملف المضيفين - إعداد جرد المضيف
- نشر Proxmox VM/LXC - تهيئة VM
- مرجع التكوين - جميع متغيرات التكوين

عمليات النسخ المتماثل و العقد في OmniHSS

تخزن OmniHSS بيانات المشتركين، والمصادقة (AuC)، وبيانات IMS في قاعدة بيانات محلية على كل عقدة. PostgreSQL **النسخ المتماثل المنطقي ثنائي الاتجاه في** يشكل هذا **شبكة نشطة-نشطة**: كل عقدة هي رئيسية كاملة للقراءة/الكتابة، والكتابة على أي عقدة تنتشر إلى جميع العقد الأخرى. لا يوجد رئيس/احتياطي. يمكن لأي عقدة أن تخدم حركة مرور Cx/Dx، و S6a في أي، وتقبل التهيئة وقت.

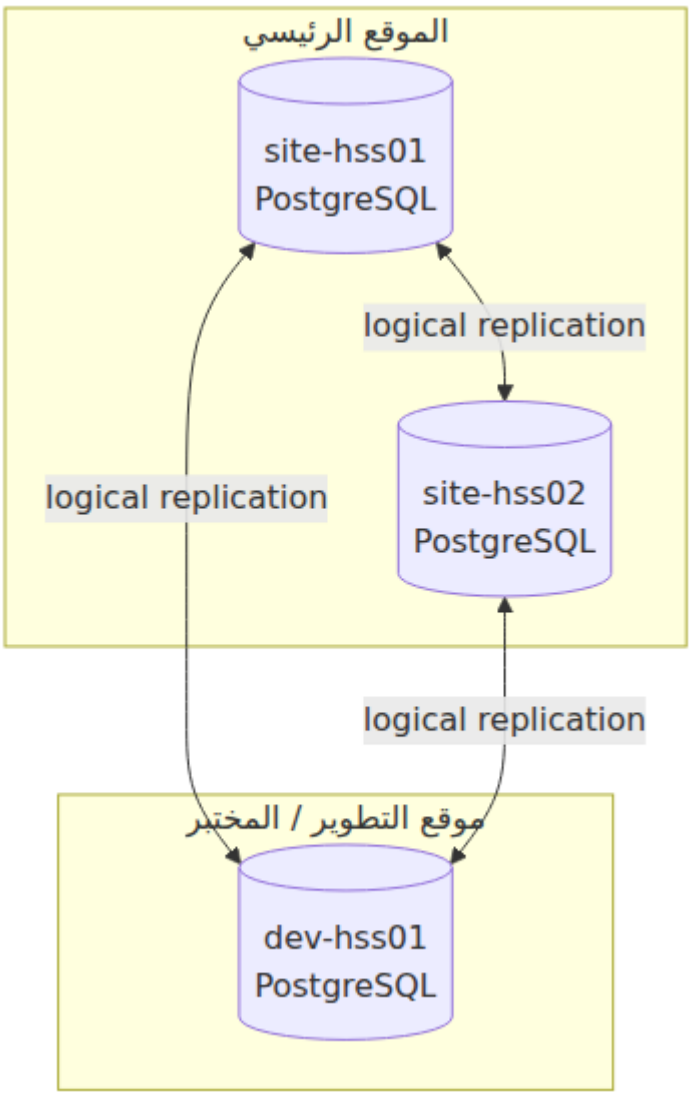
يغطي هذا الدليل نموذج النسخ المتماثل. يشرح كيفية إضافة عقدة OmniHSS جديدة وتحميلها بالبيانات. كما يشرح كيفية التحقق من الصحة، وإصلاح، وإزالة الأقران.

جدول المحتويات

نظرة عامة على المعمارية

تعمل كل عقدة OmniHSS على تشغيل مثيل PostgreSQL جميع **تنشر** خاص بها. كل عقدة جداول التطبيق الخاصة بها و **تشترك** في نشر كل نظير. `origin = none` يمنع حلقات النسخ

المتماثل. على جداول الحالة المتغيرة بسرعة، تحل الشبكة تعارضات الكتابة **آخر كتابة تفوز (LWW)** من خلال مقارنة `updated_at`.

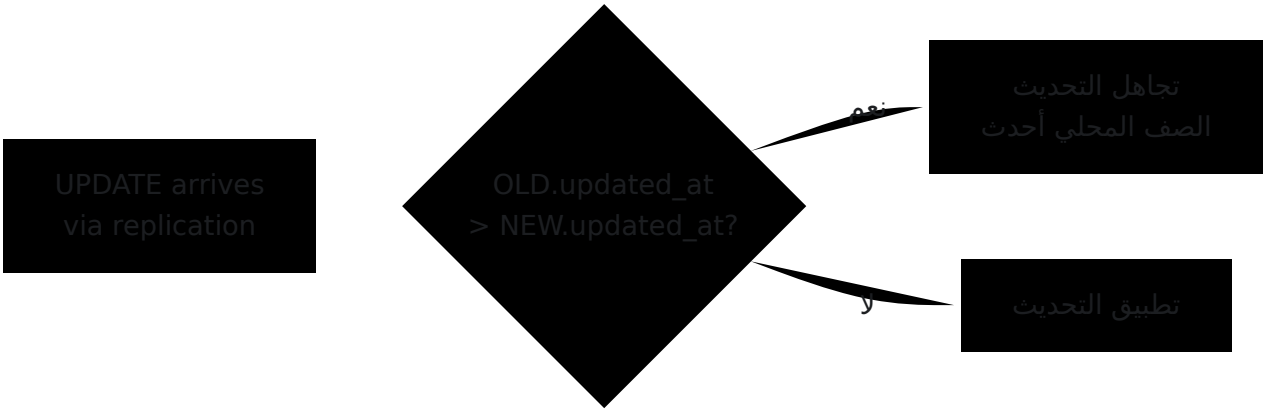


عقد، تحتفظ كل عقدة بـ `N` كل رابط أعلاه هو اشتراك أحادي الاتجاه (واحد لكل اتجاه). لشبكة من `N-1` اشتراكات. تستهلك الأقران `N-1` من فتحات النسخ المتماثل الخاصة بنشرها.

ما الذي يتم نسخه متماثلاً

فئة الجدول	أمثلة	معالجة التعارض
البيانات الرئيسية / المهيأة	`subscriber`, `key_set` (AuC), `sim`, `msisdn`	لا حاجة: يتم تهيئتها من مكان واحد في كل مرة
الحالة الديناميكية	`subscriber_state`, `pdn_session`, `lte_call`, `ims_subscription`	آخر كتابة تفوز على `updated_at`
محاسبة الهجرة	`schema_migrations`	**مستبعدة** من النسخ المتماثل

تقوم كل عقدة) `schema_migrations` **باستثناء** `public` يغطي النشر كل جدول في مخطط عندما تقوم عقدتان بتحديث نفس صف الحالة في وقت واحد، .(بتشغيل هجراتها بشكل مستقل بدلاً من أي تحديث وصل أخيرًا `updated_at` على أحدث LWW تحافظ مشغلات.



كيفية اكتشاف الأقران

بناء قائمة الأقران تلقائيًا من مصدرين OmniHSS عندما يتم تمكين النسخ المتماثل، يقوم دور:

1. المعرفة OmniHSS **المحلية**: عقد `hss` **المضيفون الآخرون في مجموعة جرد** (لا `hss01` / `hss02` على سبيل المثال، زوج محلي) في نفس ملف الجرد/المضيف ؛ يكفي أن تكون عضواً في `connected_omnihss` تحتاج هذه إلى أن تكون مدرجة في المجموعة.

2. **قائمة** `connected_omnihss` في ملف الأقران البعيد للموقع: عقد OmniHSS المعرفة في ملف مضيف مختلف (مواقع أخرى). يمكن اكتشاف الأقران عبر الملفات الخاصة بآخر `hss` فقط هنا، لأن جرد واحد ليس لديه معرفة بمجموعة.

يقوم الدور بتصفية ا��أقران من المجموعة المحلية إلى المضيفين الذين هم مشاركون في النسخ المتماثل لـ OmniHSS: أولئك الذين تكون قيمة `omnihss.replication.enabled` لديهم `true`. العقدة التي ليست علامتها `true` لا يتم اختيارها كقرين نسخ متماثل.

يستبعد الدور المضيف الحالي من قائمة أقرانه تلقائيًا، بحيث يمكن مشاركة نفس كتلة `connected_omnihss` من قبل كل موقع دون محاولة عقدة النسخ من نفسها.

تسمية الكائنات

تسمي كل عقدة كائنات النسخ المتماثل الخاصة بها بحيث لا تتداخل شبكة بأي حجم:

كائن	الاسم	تم إنشاؤه في	فريد لكل
النشر	`<self>_pub`	هذه العقدة	العقدة
الاشتراك	`sub_from_<source>`	هذه العقدة (المشترك)	(المشترك)
فتحة النسخ المتماثل	`<subscriber>_from_<source>`	المصدر (الناشر)	(المشترك، الزوج (المصدر

الإعداد) يجب أن يتضمن اسم الفتحة **المشترك**. اعتبر ما يحدث إذا كان يتضمن فقط المصدر ستقوم عقدتان تشتركان في PostgreSQL، ا��افتراضي لـ .(حيث ترث الفتحة اسم الاشتراك نفس المصدر بمحاولة إنشاء فتحة بنفس الاسم على ذلك المصدر. ستفشل الاشتراك الثانية بعد ذلك مع *"فتحة النسخ المتماثل موجودة بالفعل"*. إن تضمين المشترك يعطي كل رابط فتحة فريدة عالميًا.

السلامة `--limit` المصالحة و

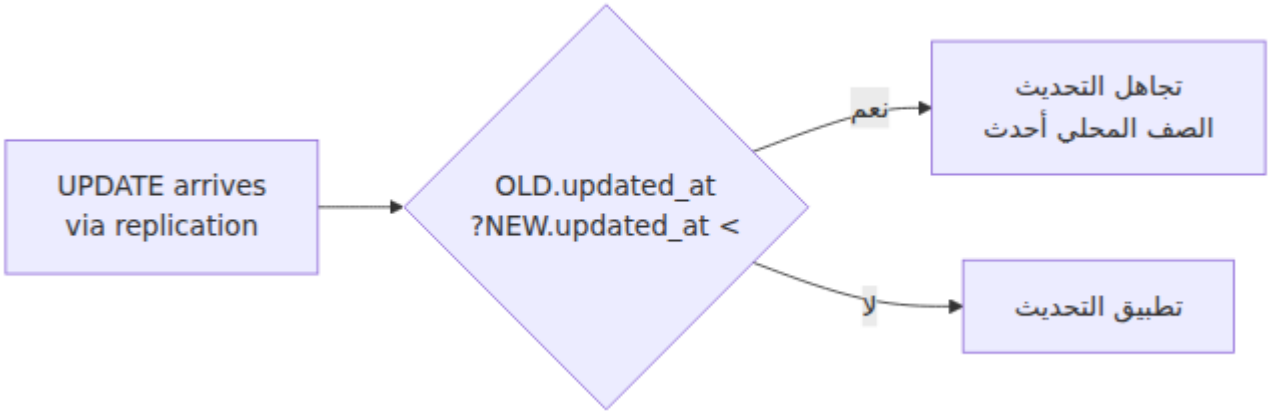
كل تشغيل للدور **يصلح** كائنات النسخ المتماثل للعقدة مقابل مجموعة الأقران الصالحة الحالية (مجموعة `hss` المحلية + `connected_omnihss`، مطروحًا منها الذات):

- تنظف المنشورات القديمة/المعاد (`<self>_pub` يتم إسقاط المنشورات الأخرى غير تسميتها تلقائيًا).
- يتم تعطيل الاشتراكات التي لم يعد نظيرها في المجموعة الصالحة، ويتم فصلها عن فتحتها البعيدة، ويتم إسقاطها.
- يتم إسقاط الفتحات المنطقية غير النشطة التي لا تغذي نظيرًا صالحًا حاليًا. لا يتم لمس الفتحات النشطة أبدًا.

قائمة الشبكة **السلطوية**: يقوم التشغيل التالي بإزالة `connected_omnihss` هذ�� يجعل من كل عقدة. تمنع خاصيتان من الأمان هذا (ذات الصلة `hss` ومن مجموعات) قرين تم إزالته منها من أن يكون مدمرًا:

- **`--limit` آمن.** يتم اشتقاق المجموعة الصالحة من `groups['hss']` و`connected_omnihss`، بغض النظر عن *الكاملة* عضوية الجرد والتي تعكس دائمًا `--limit`. تشغيل `--limit hss01` يصلح فقط `hss01`. ترى `hss01` ولكن لا تزال أبدًا بإزالة `--limit` كصالحين وتحافظ على اشتراكاتهم. لا يقوم `hss02`/`hss03` الأقران المستبعدة.
- **المجموعات الفارغة لا تقوم بالإزالة.** يتم تخطي خطوات الإزالة تمامًا بدلاً من مسح جميع النسخ المتماثل إذا كانت مجموعة الأقران الصالحة تحسب إلى فارغة (غير محملة `connected_omnihss` واحد و HSS سبيل المثال، جرد جزئي مع).

> **تحذير القائمة السلطوية:** نظرًا لأن الإزالة تلقائية، يقوم التشغيل بإزالة قرين عبر الملفات تأكد دائمًا من أن `connected_omnihss` يكون جزءًا حقيقيًا من الشبكة ولكنه مفقود من `connected_omnihss` مكتمل لكل قرين عبر الملفات.



التكوين

تمكين النسخ المتماثل

يتم التحكم في النسخ المتماثل لكل نشر بواسطة علامة `omnihss.replication.enabled`. وهي `false` بشكل افتراضي. قم بتعيينها في جرد الموقع أو group_vars:

```yaml
omnihss:
  database_type: "postgres"
  database_host: "localhost"
  database_port: 5432
  database_name: "omnihss"
  database_username: "hss"
  database_password: "password"
  replication:
    enabled: true
```

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`database_type`	سلسلة	لا	`postgres`	قاعدة البيانات الخلفية. ينطبق النسخ المتماثل فقط على `postgres`.
`database_host`	سلسلة	لا	`localhost`	مضيف قاعدة البيانات. يتم تكوين النسخ المتماثل فقط عندما تكون قاعدة البيانات محلية (`localhost`).
`database_port`	عدد صحيح	لا	`5432`	منفذ PostgreSQL، يستخدم أيضًا كمنفذ افتراضي للاتصالات بالأقران.
`database_name`	سلسلة	لا	`omnihss`	اسم قاعدة البيانات. يجب أن يتطابق عبر جميع الأقران.
`database_username`	سلسلة	نعم	`hss`	دور تسجيل الدخول. تم إنشاؤه مع خاصية `REPLICATION` حتى يتمكن من قيادة النسخ المتماثل المنطقي.
`database_password`	سلسلة	نعم	`password`	كلمة المرور لدور تسجيل الدخول. يجب أن تتطابق عبر جميع الأقران (تستخدم في سلاسل اتصال الأقران).

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`replication.enabled`	منطقي	لا	`false`	مفتاح رئيسي. عندما يتم `true`، تكون تكوين PostgreSQL للنسخ المتماثل المنطقي ويتم إنشاء المنشورات/الاشتراكات.

يقوم الدور بتطبيق إعدادات PostgreSQL، `true` هي `replication.enabled` عندما تكون التالية على المثيل المحلي:

الإعداد	القيمة
`wal_level`	`logical`
`max_wal_senders`	`10`
`max_replication_slots`	`10`
`listen_addresses`	`*`
`max_slot_wal_keep_size`	`10GB` (يمكن تجاوزها عبر `omnihss.replication.max_slot_wal_keep_size`)

القيمة	الإعداد

يتم تحديث `pg_hba.conf` للسماح بعنوان IP لكل من اتصالات (`32/`) لكل نظير و `all` `replication`.

إعلان الأقران عبر المواقع (`connected_omnihss`)

تُعلن عقد OmniHSS البعيدة في ملف الأقران البعيد المشار إليه بواسطة متغير `remote_peers_file` الخاص بكل موقع (عادةً `prod_master_peers.yml`):

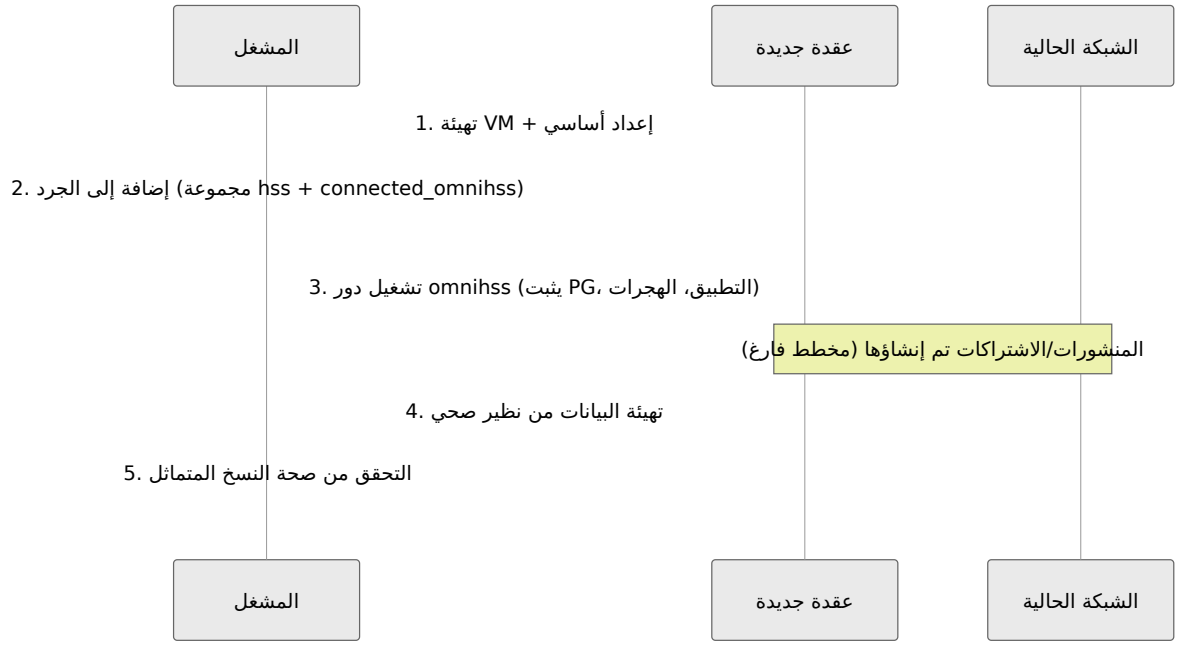
```
connected_omnihss:
  - name: site-hss01
    host: 10.4.2.142
    port: 5432
  - name: site-hss02
    host: 10.4.2.143
    port: 5432
  - name: dev-hss01
    host: 10.4.10.142
    port: 5432
```

المعلمة	النوع	مطلوب	الافتراضي	الوصف
`name`	سلسلة	نعم	-	للنظير. هذا `inventory_hostname` يسمي الاشتراك (`sub_from_<name>`) ويحل نشر النظير (`<name>_pub`)، لذا يجب أن **يتطابق** مع اسم المضيف الفعلي للنظير.
`host`	سلسلة	نعم	-	عنوان IP للنظير، يمكن الوصول إليه من هذه العقدة عبر شبكة الإشارة.
`port`	عدد صحيح	لا	`5432`	منفذ PostgreSQL للنظير. يعود إلى `database_port`.

نظرًا لأن ملف الأقران البعيد يتم تحميله بواسطة كل موقع، يجب أن تحتوي القائمة الكاملة لـ `connected_omnihss` في الشبكة. تتخطى كل موقع تلقائيًا عقدها OmniHSS على **كل** عقد المحلية (`hss`) لأنها مغطاة بالفعل بواسطة مجموعة وتتخطى نفسها.

إضافة عقدة OmniHSS جديدة

إضافة عقدة لها مرحلتان متميزتان: **الانضمام إلى الشبكة** (حتى تتدفق التغييرات في كلا الاتجاهين) و **تحميل البيانات الموجودة** على العقدة الجديدة. تهم المرحلة الثانية لأن العقدة الجديدة تمامًا تبدأ بقاعدة بيانات فارغة. انظر تحميل عقدة جديدة بالبيانات.



الخطوة 1: تهيئة المضيف

قم بتهيئة VM والشبكة الأساسية كما هو الحال في أي عقدة OmniCore (انظر نشر Proxmox). يجب أن تصل العقدة إلى منفذ PostgreSQL عبر شبكة (بشكل افتراضي `5432`) لكل نظير. ويجب أن تتمكن الأقران من الوصول إليها الإشارة،.

الخطوة 2: إضافة العقدة إلى الجرد

أضف المضيف الجديد إلى ملف مضيف الموقع تحت مجموعة `hss:`:

```
hss:
  hosts:
    site-hss01:
      ansible_host: 10.4.2.142
      gateway: 10.4.2.1
      host_vm_network: "v401-omni-signalling"
    site-hss03:        # عقدة جديدة
      ansible_host: 10.4.2.144
      gateway: 10.4.2.1
      host_vm_network: "v401-omni-signalling"
```

في ملف الأقران البعيد حتى تتعلم **المواقع الأخرى** عنه `connected_omnihss` ثم أضفه إلى أيضًا:

```
connected_omnihss:
  - name: site-hss03
    host: 10.4.2.144
    port: 5432
  # ... إدخالات موجودة ...
```

الخطوة 3: تشغيل دور OmniHSS

قم بتشغيل كتاب خدمة OmniHSS ضد المضيف الجديد. يقوم هذا بتثبيت PostgreSQL و ويش��ل هجرات قاعدة البيانات (إنشاء مخطط **فارغ**)، ويقوم بتكوين النسخ OmniHSS، المتماثل المنطقي، ويقوم بإنشاء النشر بالإضافة إلى اشتراك لكل نظير:

```
ansible-playbook -i hosts/Customer_Name/host_files/Production.yml
\
  services/omnihss.yml --limit site-hss03
```

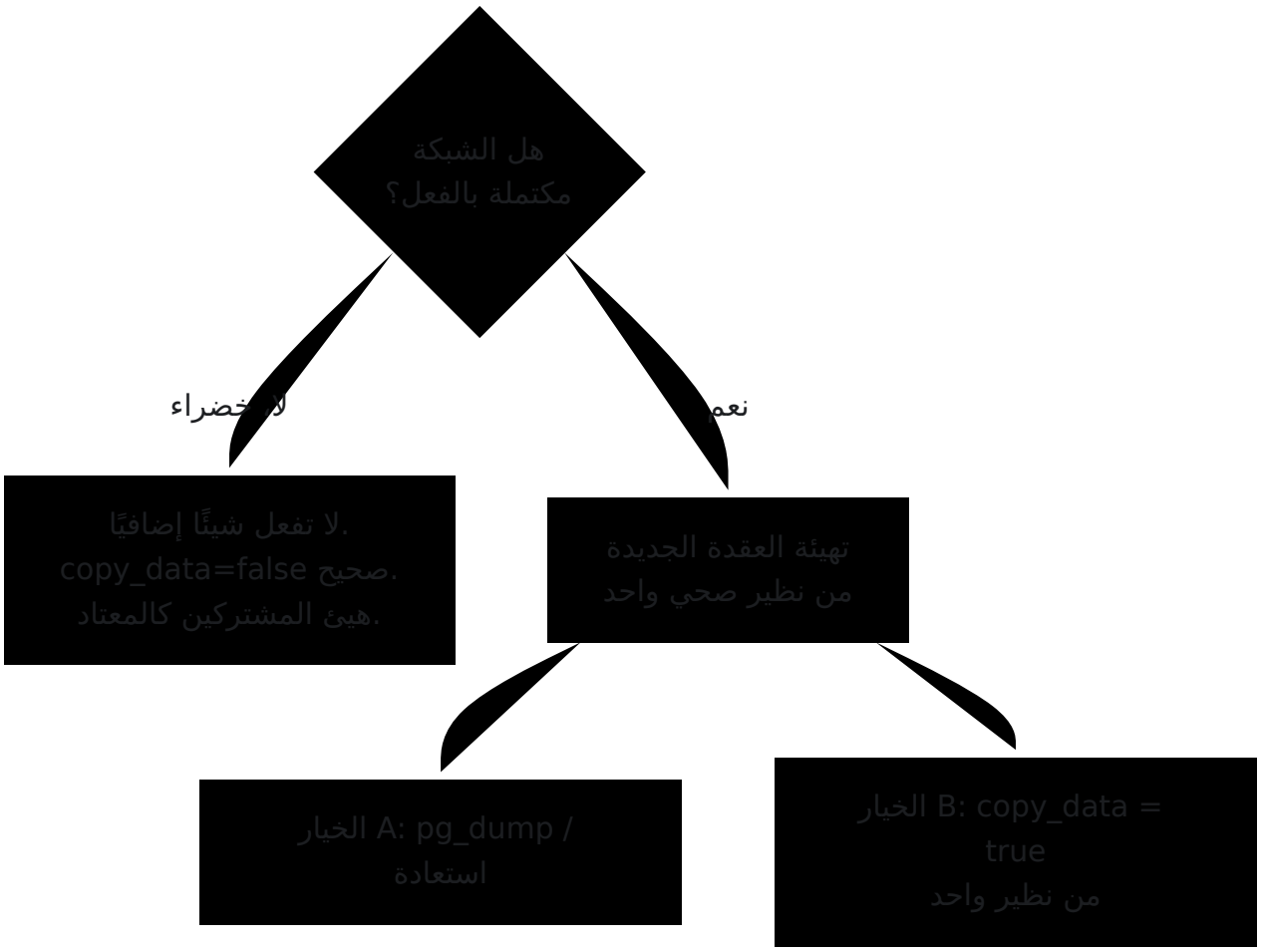
أعد تشغيل الدور على الأقران الحالية (أو دع التشغيل المجدول التالي يتعامل معها) حتى تفتح `pg_hba.conf` للعقدة الجديدة وتقوم بإنشاء اشتراكها مرة أخرى. الاشتراكات غير متغيرة: `ALTER SUBSCRIPTION ... REFRESH PUBLICATION` يكتشف الدور ويخطي الاشتراكات الموجودة، ويصدر حتى تلتقط أي جداول جديدة.

في هذه المرحلة، أصبحت العقدة الجديدة عضوًا في الشبكة وتتلقى **جميع الكتابات المستقبلية**. لكنها لا تحتوي بعد على قاعدة المشتركين الموجودة. تابع لتحميل بياناتها.

تحميل عقدة جديدة بالبيانات

هذا صحيح لشبكة **خضراء** (تبدأ `copy_data = false`. يقوم الدور بإنشاء الاشتراكات الآلية مع كل عقدة فارغة في نفس الوقت: لا يوجد شيء للنسخ، والنسخ سيؤدي إلى تعارضات صفوف مكررة). ولكن هذا يعني أن العقدة التي تنضم إلى شبكة **مكتملة بالفعل** تبقى فارغة باستثناء التغييرات التي تحدث بعد انضمامها.

اختر المسار الذي يتناسب مع حالتك:



شبكة خضراء

إذا كنت تقوم بإنشاء جميع عقد OmniHSS معًا دون بيانات مسبقة، فلا تحتاج إلى التهيئة. قم بتمكين النسخ المتماثل، وشغل الدور عبر جميع العقد، وقم بتهيئة المشتركين بشكل طبيعي. كل كتابة تتكرر إلى جميع الأعضاء.

الخيار A: تهيئة عبر النسخ والاستعادة (موصى به)

خذ لقطة من نظير صحي واستعدها على العقدة الجديدة. ثم دع النسخ المتماثل يحمل التغييرات المستمرة. يعيد هذا استخدام أدوات النسخ الاحتياطي/الاستعادة القياسية وهو النهج الأكثر توقعًا، بما في ذلك لمجموعات البيانات الكبيرة.

1. لقاعدة بيانات `pg_dump` **قم بعمل نسخة احتياطية من نظير صحي.** ينتج هذا بحيث لا ، `--no-publications --no-subscriptions` تم أخذه مع OmniHSS (يتم نقل كائنات النسخ المتماثل على آلة التحكم:

```
ansible-playbook -i
hosts/Customer_Name/host_files/Production.yml \
  services/backup.yml --limit site-hss01
```

يتم حفظ النسخة الاحتياطية كـ `backups/<Site_Name>/hss_dump_<hostname>_<timestamp>.sql`.

2. يتوقف كتاب الاستعادة عن OmniHSS، ويسقط **استعادة على العقدة الجديدة.** ويعيد إنشاء قاعدة البيانات، ويستعيد النسخة الاحتياطية، ويعيد تشغيل الخدمة. يأخذ نسخة احتياطية قبل الاستعادة من الهدف أولاً:

```
ansible-playbook -i
hosts/Customer_Name/host_files/Production.yml \
  util_playbooks/restore_hss.yml --limit site-hss03
```

من الخطوة 1 واترك مسار SQL عند المطالبة، قدم المسار إلى النسخة الاحتياطية (لأن العقدة الجديدة لديها تكوينها الخاص من تشغيل الدور) التكوين فارغًا.

3. **تأكيد تدفق النسخ المتماثل.** الآن تطبق الاشتراكات الموجودة للعقدة الجديدة (`copy_data = false`) التغييرات على القمة من النسخة الاحتياطية المستعادة. قم بتشغيل أدناه التحقق.

> **نافذة الصيانة:** هناك نافذة صغيرة بين النسخة الاحتياطية واللحظة التي يلحق فيها النسخ المتماثل. خلال ذلك، لم تعد تغييرات التهيئة على الأقران موجودة على العقدة الجديدة. تتغير بيانات المشترك الرئيسية بشكل نادر، وتتعافى الحالة الديناميكية (`subscriber_state`, `pdn_session`, `lte_call`, `ims_subscription`) من حركة المرور الحية، لذا فإن نافذة النشاط المنخفضة كافية. لا تقم بتهيئة مشتركين جدد خلال التهيئة.

الخيار B: ت��يئة باستخدام كتاب التهيئة (موصى به)

يقوم كتاب الخدمة `seed_hss_replication.yml` بالتهيئة من **مصدر** نظير مختار واحد دون عيوب النسخ الخام. يعيد إنشاء الاشتراك كاشتراك فقط للتدفق (`copy_data = false`، لا يوجد جداول متزامنة). ثم يقوم بملء الصفوف الموجودة في **الجدول الرئيسي** بتحميل آمن من التعارض. `subscriber_state` 1:1 هيكلية بيانات إنها جزء من الملء: **هو** (كل مشترك يشير إلى واحد بواسطة FK، تم إنشاؤه مرة واحدة عند التهيئة). تركه يعني أن كل مشترك تم ملؤه يصبح على الهدف. لا يمكن للتدفق إصلاح ذلك، لأن النسخ المتماثل المنطقي S6a و API يتيمًا، مما يكسر يسقط بهدوء التحديثات للصفوف التي لا توجد. لا يقوم كتاب الخدمة عمدًا بنسخ الجداول المؤقتة

لتلك الجداول الساخنة (مفتاح `COPY` يمكن أن يتداخل `pdn_session`, `lte_call`). الحقيقية مكرر) مع الكتابات المتزامنة على نظام حي ويؤدي إلى حلقات تعطل لجداول التزامن. يتم إعادة ملئها من النسخ المتماثل الحي وحركة المرور الحية بدلاً من ذلك.

تفاعلي (موصى به): اترك المصدر. سيجد كتاب الخدمة بعد ذلك الأهداف الفارغة، ويستجوب الأقران عن الأقران التي تحمل البيانات، ويطلب منك الاختيار:

```
ansible-playbook -i hosts/Customer_Name/host_files/Production.yml
\
  util_playbooks/seed_hss_replication.yml \
  --limit site-hss01,site-hss02
```

```
الأقران التي تتوفر منها البيانات للتهيئة:
  1) dev-hss01 (10.4.10.142) - 838 مشترك
أدخل رقم المصدر للتهيئة منه
```

صريح: قم بتسمية المصدر مباشرة. هذا يجبر على إعادة التهيئة حتى لو كانت الهدف تحتوي بالفعل على بيانات (استخدم للتعافي):

```
ansible-playbook -i hosts/Customer_Name/host_files/Production.yml
\
  util_playbooks/seed_hss_replication.yml \
  --limit site-hss01 \
  -e "seed_source_name=dev-hss01 seed_source_host=10.4.10.142"
```

انظر HSS Replication Seed للحصول على مرجع كامل للمعلمات والسلوك.

A للعقدة المستهدفة، وتهيئة من مصدر واحد فقط. يفضل الخيار `--limit` دائمًا استخدم لمجموعات البيانات الكبيرة جدًا.

لاحظ) تتم الخطوات اليدوية المعادلة على العقدة المستهدفة. أولاً، قم بإنشاء الاشتراك المتدفق ثم قم بملء الجداول الرئيسية بأمان من `slot_name` تسمية الكائنات انظر، الصريح لكل رابط). التعارض مع تخطي الجداول الساخنة:

```
-- 1. تدفق جميع الجداول، لا نسخ جماعي
DROP SUBSCRIPTION IF EXISTS sub_from_dev_hss01;
CREATE SUBSCRIPTION sub_from_dev_hss01
  CONNECTION 'host=10.4.10.142 port=5432 dbname=omnihss user=hss
password=<omnihss.database_password>'
  PUBLICATION dev_hss01_pub
  WITH (copy_data = false, origin = none,
        slot_name = 'site_hss01_from_dev_hss01');
```

```
# 2. ملء الجداول الرئيسية فقط (تشغيل على الهدف؛ كل جدول -t
باستثناء
#    يجب تضمين - schema_migrations، pdn_session، lte_call
subscriber_state،
#    يشير: إلى subscriber_state_id أو يصبح كل مشترك تم ملؤه يتيمًا
صف لا يصل أبدًا، مما يكسر
#    ولا يمكن للنسخ المتماثل إصلاحه لأن النسخ المتماثل API/S6a،
المنطقي
#    يسقط بهدوء التحديثات للصفوف المفقودة).
#    يجب أن يكون التحميل مختومًا بأصل النسخ المتماثل: origin=none
#    تقوم الاشتراكات الموقعة بتصفية التغييرات الموقعة عند الناشر،
لذا فإن الملء يكون
#    مثل WAL غير مرئي للشبكة. يتم فك تشفير التحميل غير الموقعة من
أي كتابة محلية
#    وتعيد صدى الإدخالات إلى الأقران الذين يحملون الصفوف بالفعل،
#    مما يؤدي إلى تأخير عمال تطبيقهم بسبب المفتاح المكرر.
sudo -u postgres psql -d omnihss -c \
  "SELECT pg_replication_origin_create('omnihss_seed_load');"
{ echo "SELECT
pg_replication_origin_session_setup('omnihss_seed_load');"
  PGPASSWORD=<pw> pg_dump -h 10.4.10.142 -U hss -d omnihss \
    --data-only --inserts --on-conflict-do-nothing --disable-
triggers \
    -t subscriber -t subscriber_state -t key_set -t sim -t msisdn
... ; } \
  | sudo -u postgres psql -d omnihss
sudo -u postgres psql -d omnihss -c \
  "SELECT pg_replication_origin_drop('omnihss_seed_load');"
```

التحقق من النسخ المتماثل

دور التشغيل يقوم بإجراء فحص صحي تلقائي بعد الإعداد، ويمكن أيضًا تشغيله بمفرده. تؤكد عملية التحقق أن:

- كل اشتراك لديه عملية عامل حية.
- كل فتحة تكرار منطقية نشطة.
- تأخير WAL لكل فتحة أقل من 100 ميجابايت.
- حالة WAL لكل فتحة هي `reserved` أو `extended` (وليس `lost`).

أو، (تتم عملية التحقق في النهاية) OmniHSS للتحقق من الصحة عند الطلب، أعد تشغيل دور تحقق مباشرة على عقدة:

```
# Subscription workers - pid should be non-null for each
sudo -u postgres psql -d omnihss -c \
  "SELECT subname, pid IS NOT NULL AS worker_alive FROM
pg_stat_subscription;"

# Replication slots - active should be true, wal_status
reserved/extended
sudo -u postgres psql -d omnihss -c \
  "SELECT slot_name, active, wal_status, \
     pg_size_pretty(pg_wal_lsn_diff(pg_current_wal_lsn(),
confirmed_flush_lsn)) AS lag \
   FROM pg_replication_slots WHERE slot_type = 'logical';"
```

العرض	المعنى
`worker_alive = false`	عامل الاشتراك متوقف: النظير غير متاح أو بيانات الاعتماد/سلسلة الاتصال خاطئة.
`active = false` (slot)	لا يوجد مشترك يستهلك الفتحة: اشتراك النظير متوقف أو تم إسقاطه.
تأخير متزايد `lag`	النظير بطيء أو الرابط متدهور. WAL يتراكم ويستهلك القرص.
`wal_status = lost`	الفتحة تأخرت كثيرًا وتمت إزالة WAL بواسطة PostgreSQL. يجب عليك **إعادة تهيئة** النظير (انظر تحميل عقدة جديدة بالبيانات).

للعقد **HSS Postgres** أيضًا لوحة **تكرار** (`health_check.yml`) تتضمن تقرير الفحص الصحي التي تمكين التكرار فيها، مما يظهر حالة عامل كل اشتراك وحالة كل فتحة نشطة وحالة WAL في لمحة.

سلامة البيانات

تطبيق التكرار المنطقي لا يفرض المفاتيح الخارجية. يعمل عامل التطبيق مع `session_replication_role = replica`، مما يقمع المحفزات الداخلية التي تنفذ قيود RI `FOREIGN KEY`. لذلك، لا يقوم PostgreSQL من صف يصل عبر التكرار **بإعادة التحقق أبدًا** مقابل والده. تعطي قيود FK سلامة *لكل عقدة*، وليس سلامة *عبر الشبكة بأكملها*.

في شبكة نشطة-نشطة تفتح ثغرة لا يمكن لأي عقدة واحدة أن تلتقطها:

- العقدة A تحذف `epc_profile` X. تمر `ON DELETE RESTRICT` لأن A ليس لديها في تلك اللحظة X مشترك محلي يشير إلى.
- العقدة B (أو خطوة توفير) تشير إلى المشترك 203 إلى X. لا يزال X تمر هذه لأن B موجودًا على B.
- تتقارب كل عقدة على "X" **و** "X → تتكرر كلا التغييرين وتطبق بدون فحوصات FK. 203" الخاصة بأي عقدة محليًا FK محذوف": مرجع عالق لم تنتهك.

غير قادر على بناء بيانات الاشتراك HSS يجعل `subscriber.epc_profile_id` مرجع عالق على ULR/AIR، `DIAMETER_ERROR_USER_UNKNOWN (5001)` بـ HSS لذلك المشترك. ثم يجيب

على الرغم من أن صف المشترك موجود ومفعل. تجعل نافذة المخطط قبل البيانات (أدناه) الأمر أكثر احتمالًا، لأن المعاملات المتجاوزة يمكن أن تفصل إنشاء/حذف الوالد عن إعادة توجيه الطفل.

الكشف (ليس التنفيذ)

على مسار التطبيق إلى إيقاف الشبكة عند أي شذوذ، لذا بدلاً من ذلك نحن FKs سوف يؤدي فرض **نكتشف وننبه**. تقوم كل عقدة بتثبيت دالة `SECURITY DEFINER`، `omnihss_fk_orphan_count()`، ذات عمود واحد تشير قيمتها إلى والد FK التي تحسب كل مفقود عبر جميع الجداول. يقوم `postgres_exporter` بتحديدها، وتنبه Grafana عندما تكون أكبر من صفر (انظر المقاييس).

قم بتشغيل عملية المسح يدويًا في أي وقت:

```
sudo -u postgres psql -d omnihss -tAc "SELECT
omnihss_fk_orphan_count();"   # 0 = clean
```

غير EPC مثال: المشتركين الذين يشيرون إلى ملف) لتحديد الصفوف المخالفة لمرجع معين موجود):

```
SELECT s.imsi, s.epc_profile_id
FROM subscriber s
LEFT JOIN epc_profile e ON e.id = s.epc_profile_id
WHERE s.epc_profile_id IS NOT NULL AND e.id IS NULL;
```

قم بإصلاح ذلك عن طريق إعادة توجيه الصف إلى والد صالح (تحد��ث على صف موجود يتكرر بشكل نظيف عبر الشبكة) أو إزالته.

منع انحراف المخطط

كان المحفز الجذري للحادث الذي دفع إلى هذا القسم هو **ترحيل المخطط الذي تم تطبيقه على عقدة واحدة بينما كانت نظيراتها لا تزال على المخطط القديم**، ثم كتابة بيانات جديدة ضد الحقول الجديدة. قامت المصدر بنشر صفوف لم تتمكن نظيرات المخطط القديم من تطبيقها، مما تسبب في أخطاء تطبيق و**معاملات متجاوزة**، وبالتالي تباين.

- OmniHSS **قم بترحيل الشبكة بأكملها معًا.** قم بتطبيق الترحيلات على كل عقدة قبل التوفير ضد الحقول الجديدة. لا تكتب بيانات المخطط الجديد بينما أي نظير متأخر.

- تقوم العقدة بتصدير **راقب الانحراف.** `omnihss_schema_max_version` / `omnihss_schema_migration_count` لكل عقدة. قارنها عبر الشبكة. العقدة المتأخرة عن نظيراتها هي ترحيل نصف مطبق.
- **راقب عدادات الأخطاء.** يعني ارتفاع `omnihss_subscription_error_apply_error_count` / `_sync_error_count` أن عامل التطبيق يفشل (وربما يتجاوز). يتم تنبيه "أخطاء تطبيق/مزامنة HSS" على هذا.

إزالة نظير

تؤدي إزالة نظير إلى إسقاط الاشتراك **من** ذلك النظير وإزالة إدخاله في `pg_hba.conf`. تبقى البيانات التي تم تكرارها بالفعل؛ فقط التغييرات المستقبلية تتوقف عن التدفق. يجب تأكيد الإزالة بشكل صريح.

على العقدة التي تقوم بإزالة النظير *منها*:

```
DROP SUBSCRIPTION IF EXISTS sub_from_<peer_name>;
```

ثم قم بإزالة سطر (أسطر) النظير من `pg_hba.conf` وأعد تحميل PostgreSQL:

```
sudo -u postgres psql -c 'SELECT pg_reload_conf();'
```

لا يزال النظير الذي يتم إزالته يحتفظ باشتراكه إلى هذه العقدة. إذا كان يجب إزالة الرابط بالكامل، قم بإزالة ذلك الاشتراك هناك أيضًا. ثم قم بإزالة العقدة من `connected_omnihss` ومجموعة `hss` حتى لا يعيد تشغيل الدور التالي إنشائها.

النسخ الاحتياطي والاستعادة

تستخدم النسخ الاحتياطي والاستعادة في OmniHSS تفريغات PostgreSQL القياسية:

العملية	الكتابة	الملاحظات
النسخ الاحتياطي	`services/backup.yml`	ينتج `hss_dump_<host>_<ts>.sql` (قاعدة البيانات) و `hss_<host>_<ts>.tar.gz` (تكوين `/etc/omnihss`) تحت `backups/<Site_Name>/`. يستثني التفريغ المنشورات/الاشتراكات.
الاستعادة	`util_playbooks/restore_hss.yml`	يطلب تفريغ SQL و/أو أرشيف التكوين. يوقف OmniHSS، يأخذ نسخة احتياطية قبل الاستعادة، يسقط/يعيد إنشاء قاعدة البيانات، يستعيد، ويعيد التشغيل.

انظر كتب التشغيل المساعدة لعملية النسخ الاحتياطي/الاستعادة الأوسع.

المقاييس

تشغل كل عقدة OmniHSS `postgres_exporter`، مما يعرض مقاييس PostgreSQL (بما في ذلك التكرار) إلى Prometheus. السلاسل المفيدة لصحة التكرار:

الوصف: ما إذا كانت كل فتحة تكرار منطقية لديها مستهلك نشط. **المقياس**: `pg_replication_slots_active` **النوع**: Gauge **التسميات**:

- `slot_name`: فتحة التكرار (واحدة لكل نظير مشترك)

الوصف: تأخير موضع WAL لكل فتحة. يظهر مدى تأخر نظير مستهلك. **المقياس**: `pg_stat_replication_*` / **النوع**: Gauge تأخير الفتحة:

استعلامات المثال:

```
# أي فتحة تكرار منطقية غير نشطة (تنبيه إذا كانت > 0)
count(pg_replication_slots_active == 0)

# يمكن التحقق من الاشتراكات التي يجب أن توجد مقابل العمال النشطين
# ضد مخرجات كتاب التشغيل للتحقق أثناء النشر.
```

مقاييس OmniHSS المخصصة

يقوم `postgres_exporter` على كل عقدة أيضًا بتصدير مقاييس صحة التكرار والسلامة الخاصة بـ OmniHSS عبر ملف استعلام مخصص (`roles/omnihss/files/postgres_exporter_queries.yaml`). يقوم Prometheus بجمع هذه تحت وظيفة `HSS Postgres`. **تنبيهات** **HSS** تدعم قواعد التنبيه في مجموعة (`roles/monitoring/templates/grafana/rules/`):

المقياس	النوع	ى
`omnihss_subscription_worker_alive{subname}`	Gauge	ن 1 امل ليق عمل
`omnihss_replication_slot_active{slot_name}`	Gauge	ن 1 ظير هلك تحة
`omnihss_replication_slot_wal_status_code{slot_name}`	Gauge	ر=0 ممتد =غير جوز **قود**
`omnihss_replication_slot_lag_bytes{slot_name}`	Gauge	يتات WA تفظ تحة
`omnihss_subscription_error_apply_error_count{subname}`	Counter	طاء امل ليق كمية
`omnihss_subscription_error_sync_error_count{subname}`	Counter	طاء امنة

المقياس	النوع	ى
		اول كمية
`omnihss_integrity_fk_orphans`	Gauge	وف ذات فتاح رجي الق ب أن ن 0)
`omnihss_schema_max_version` / `omnihss_schema_migration_count`	Gauge	لامة مياه الية حيل بق / العد

انظر المراقبة والرصد لخط أنابيب المقاييس ولوحات المعلومات.

استكشاف الأخطاء وإصلاحها

العقدة الجديدة تبقى فارغة بعد الانضمام

الأعراض: العمال التكراريون نشطون والفتحات نشطة، لكن العقدة الجديدة ليس لديها (أو لديها القليل من) المشتركين.

الأسباب المحتملة:

- انضمت العقدة إلى شبكة مكتظة بالفعل ولم يتم تهيئتها أبدًا. تستخدم الاشتراكات `copy_data = false`، لذا تصل فقط التغييرات بعد الانضمام.

الحل:

1. قم بتهيئة العقدة من نظير صحي. انظر تحميل عقدة جديدة بالبيانات.

عامل الاشتراك غير نشط

الأعراض: `pid IS NULL` لاشتراك في `pg_stat_subscription`.

الأسباب المحتملة:

- النظير غير متاح على منفذ PostgreSQL (جدار ناري / توجيه).
- عنوان IP للنظير مفقود من `pg_hba.conf` لهذه العقدة.
- بيانات اعتماد خاطئة في سلسلة الاتصال (`database_username` / `database_password` تختلف بين العقد).

الحل:

1. تأكد من إمكانية الوصول عبر TCP `5432` إلى النظير على.
2. تحقق من ظهور النظير في `connected_omnihss` (أو مجموعة `hss` وأعد (المحلية تشغيل الدور حتى يتم تحديث `pg_hba.conf`.
3. تأكد من أن `database_username`/`database_password` متطابقة عبر جميع العقد.

فتحة التكرار تظهر `wal_status = lost`

الأعراض: حالة `wal_status` لفتحة ما هي `lost`؛ النظير المقابل متأخر كثيرًا.

الأسباب المحتملة:

- كان النظير متوقفًا أو غير متاح لفترة طويلة بحيث تمت إزالة WAL المطلوبة.

الحل:

1. قم بإعادة تهيئة النظير المتأثر من عقدة صحية. انظر تحميل عقدة جديدة بالبيانات.

أخطاء مفتاح مكرر أثناء النسخ الأولي

الأعراض: أخطاء في تطبيق التغييرات مباشرة بعد إنشاء اشتراك مع `copy_data = true`.

الأسباب المحتملة:

- تم تشغيل النسخ الأولي من أكثر من نظير، مما أدى إلى نسخ نفس الصفوف مرتين.

الحل:

1. قم بالتهيئة من **نظير واحد فقط** مع `copy_data = true` ؛ احتفظ بجميع الاشتراكات الأخرى عند `copy_data = false`.

تحديثات حالة متضاربة بين المواقع

الأعراض: يبدو أن صف الحالة (مثل `subscriber_state`) "يتأرجح" بين القيم المحددة في مواقع مختلفة.

الأسباب المحتملة:

- تحديثات متزامنة لنفس الصف في موقعين.

الحل: هذا متوقع ويتم التعامل معه. يحافظ المحفز الذي يفوز في آخر كتابة على التحديث مع أحدث `updated_at`. تأكد من تزامن الساعات (NTP) عبر جميع المواقع حتى يكون ترتيب `updated_at` ذا معنى.

نشر Proxmox VM/LXC

يقوم كتاب اللعب Proxmox. على OmniCore تقوم الغالبية العظمى من عملائنا بتشغيل مجموعة بناءً على LXC وحاويات QEMU VMs بتوفير كل من ، `util_playbooks/proxmox.yml`، الواحد (`proxmoxServers` على إدخال) يمكنك تعيين التبديل لكل موقع. `deployment_type` تبديل بينما تبقى المضيفات الفردية LXC وتجاوزه لكل مضيف. يتيح ذلك لموقع ما أن يعمل في الغالب كـ كاملة VMs كـ (مثل UPF).

نحن لا نزال ندعم VMware وHyperV والسحابة (Vultr / AWS / GCP) للنشر.

انظر أيضًا:

- تعريف: تكوين ملف المضيفين للنشر VMs/LXCs
- معيار تخطيط IP
- تكوين Netplan
- معمارية النشر

مقابل VM LXC

حاويات LXC:

- خفيفة الوزن، تشارك نواة المضيف
- بدء تشغيل سريع، تكلفة منخفضة
- عزل محدود
- لا يمكن تشغيل نوى مخصصة أو وحدات نواة
- **غير مناسبة لنشر EPC/IMS الإنتاجي**
- **لا يمكن تشغيل UPF** (يتطلب وحدات نواة / أجهزة TUN)

VMs (KVM):

- افتراضية كاملة مع نواة مخصصة
- عزل كامل، تشغيل وحدات نواة / شبكات مخصصة
- تكلفة موارد أعلى
- **موصى بها للإنتاج**
- **مطلوبة لنشر UPF**

حالات الاستخدام:

- **VMs**: مواقع الإنتاج، UPF، جميع وظائف الشبكة
- **LXC**: بيئات مختبر / اختبار، خدمات خفيفة الوزن (apt-cache، مراقبة، dns)

اختيار نوع النشر

يتم حل `deployment_type` لكل مضيف بهذا الترتيب:

1. **لكل مضيف** `deployment_type` (المعين تحت اسم المضيف): يفوز إذا كان موجودًا.
2. **لكل موقع** `deployment_type` على إدخال `proxmoxServers` الذي يختاره التوزيع الدائري للمضيف.
3. **افتراضي** `vm` إذا لم يتم تعيين أي منهما.

النمط الشائع هو تعيين الافتراضي للموقع على إدخال `proxmoxServers` (مثل `deployment_type: lxc`) وتجاوز فقط المضيفات التي يجب أن تختلف. المثال القياسي هو مختبر يعمل كل شيء كـ LXC باستثناء UPF، الذي يحتاج إلى وحدات نواة / TUN ويجب أن يكون VM:

```
all:
  vars:
    proxmoxServers:
      pve-node-01:
        deployment_type: lxc      # الافتراضي للموقع → كل شيء هو LXC
        # ...

upf:
  hosts:
    site-upf01:
      ansible_host: 192.168.1.24
      deployment_type: vm         # هذا المضيف هو → تجاوز لكل مضيف VM
```

كقاعدة، حافظ على الافتراضي للموقع متسقًا وتجاوز فقط الاستثناءات. عندما تقوم بتوزيع VM عبر العديد من المضيفات، يعتمد وضع العقد على فهرس المضيف ومن الصعب التفكير فيه وLXC.

يؤدي كتاب اللعب فقط إلى الفشل الصارم في حالة حدوث تصادمات نوع مع الموارد الموجودة: إذا كانت هناك مورد مع اسم المضيف المستهدف موجود بالفعل كنوع خاطئ (مثل LXC ولكن وجود)

فإنه يتوقف مع خطأ وا��ح. انظر ،(أو العكس ،`deployment_type: vm` المضيف يحل إلى حراس النوع.

إعداد Proxmox

1. إنشاء رمز API

```
# واجهة Proxmox: رموز ← الأذونات ← مركز البيانات API
# إنشاء رمز: root@pam!<TokenName>
# انسخ سر الرمز (يظهر مرة واحدة)
# `قم بإلغاء تحديد مربع `فصل الامتيازات
```

2a. إنشاء قالب VM Cloud-Init (فقط للمواقع VM)

يقوم بتنزيل صورة سحابية لأوبونتو ويخلق Proxmox. قم بتشغيل هذا البرنامج النصي على مضيف قالبًا نظيفًا. يقوم `proxmox.yml` بحقن بيانات اعتماد cloud-init ومفاتيح SSH في وقت النسخ (انظر الملاحظات أدناه).

```
#!/bin/bash
set -e

TEMPLATE_ID=9000
IMAGE_URL="https://cloud-images.ubuntu.com/noble/current/noble-
server-cloudimg-amd64.img"
IMAGE="noble-server-cloudimg-amd64.img"

cd /var/lib/vz/template/iso
wget -N "$IMAGE_URL"

qm destroy $TEMPLATE_ID --purge 2>/dev/null || true

qm create $TEMPLATE_ID --name ubuntu-2404-template --memory 2048 -
-cores 2 --net0 virtio,bridge=vmbr0
qm importdisk $TEMPLATE_ID $IMAGE local-lvm
qm set $TEMPLATE_ID --scsihw virtio-scsi-pci --scsi0 local-
lvm:vm-${TEMPLATE_ID}-disk-0
qm set $TEMPLATE_ID --ide2 local-lvm:cloudinit
qm set $TEMPLATE_ID --boot c --bootdisk scsi0
qm set $TEMPLATE_ID --vga std
qm set $TEMPLATE_ID --agent enabled=1
qm template $TEMPLATE_ID
```

ملاحظات:

- بتعيين `proxmox.yml` القالب لا يحتوي على مستخدمين أو كلمات مرور مشفرة. يقوم في وقت النسخ `sshkeys` و`cipassword` و`ciuser`.
- أولوية مستخدم cloud-init: `proxmoxTemplateUser` (على إدخال `proxmoxServers` المطابق) → أول مفتاح `local_users`.
- أولوية كلمة مرور cloud-init: `proxmoxTemplatePassword` (على إدخال `proxmoxServers` المطابق) → مستخدم cloud-init (أي القيمة المختارة أعلاه).
- يتم حقن مفاتيح SSH من كل إدخال `local_users.*.public_key` (المصادقة القائمة على المفاتيح هي مسار تسجيل الدخول المقصود).
- قم بتعيين `proxmoxTemplateUser` (ويمكن اختيارياً `proxmoxTemplatePassword`) على كل إدخال `proxmoxServers` كلما تم إعداد القالب مع مستخدم cloud-init ليس هو أول إدخال `local_users`، بحيث يتم اختيار الحساب الصحيح.
- `--vga std` يضمن أن وحدة التحكم في الويب Proxmox تعمل.

- علامة `-N` على wget تقوم بتنزيل فقط إذا كانت أحدث من النسخة المحلية.

بديل: قالب يدوي من ISO

إذا لم تكن الصور السحابية متاحة أو كنت بحاجة إلى تثبيت مخصص:

الخطوة 1: إنشاء VM عبر واجهة الويب

- إنشاء VM → معرف جديدة VM 9000، الاسم: ubuntu-2404-template
- نظام التشغيل: تحميل ISO خادم أوبونتو أو استخدام ISO موجود
- النظام: الافتراضي (وحدة التحكم SCSI: VirtIO SCSI)
- الأقراص: 32GB، الناقل: SCSI
- وحدة المعالجة المركزية: 2 نوى
- الذاكرة: 2048 ميجابايت
- الشبكة: VirtIO، جسر vmbr0
- بدء VM وتثبيت خادم أوبونتو

الخطوة 2: داخل VM - تنظيف وتحضير

```
# تثبيت cloud-init
sudo apt update
sudo apt install cloud-init qemu-guest-agent -y

# تنظيف البيانات الخاصة بالآلة
sudo cloud-init clean
sudo rm -f /etc/machine-id /var/lib/dbus/machine-id
sudo rm -f /etc/ssh/ssh_host_*
sudo truncate -s 0 /etc/hostname
sudo truncate -s 0 /etc/hosts

# مسح تاريخ bash وإيقاف التشغيل
history -c
sudo poweroff
```

الخطوة 3: إضافة Cloud-Init وتحويل إلى قالب

- اخ��ر VM → محرك → إضافة → الأجهزة CloudInit (اختر التخزين مثل local-lvm)
- وكيل → الخيارات → الأجهزة QEMU → تمكين
- تحويل إلى قالب → انقر بزر الماوس الأيمن على VM

- لا تقم **بتعيين** مستخدم / كلمة مرور cloud-init. يقوم `proxmox.yml` على القالب بحقن هذه في وقت النسخ من `proxmoxTemplateUser` / `proxmoxTemplatePassword` (مع تراجع `local_users` لاسم المستخدم).

2b. تنزيل قالب نظام تشغيل LXC (فقط LXC للمواقع)

```
# على قشرة عقدة Proxmox:
pveam update
pveam download local ubuntu-24.04-standard_24.04-2_amd64.tar.zst
```

مسار الحجم الناتج (مثل `local:vztmpl/ubuntu-24.04-standard_24.04-2_amd64.tar.zst`) هو ما تقوم بتكوينه كـ `proxmoxLxcOsTemplate` أدناه.

تكوين ملف المضيفين

نشر VM

```
all:
  vars:
    # تم حذف deployment_type → هو الافتراضي "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        # اسم مستخدم cloud-init اختياري؛ الافتراضي هو أول مفتاح local_users
        proxmoxTemplatePassword: omnitouch    # كلمة مرور cloud-init اختياري؛ الافتراضي هو اسم مستخدم cloud-init
        storage: local-lvm  # اختياري
      pve-node-02:
        # ... تكوين العقدة الثانية

    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: vmbr0
    gateway: 192.168.1.1
    mask_cidr: 24  # 24 اختياري، الافتراضي
    vlanid: 100  # اختياري
```

نشر LXC

```yaml
all:
  vars:
    proxmox_lxc_nameserver: "1.1.1.1"   # 1.1.1.1 اختياري، الافتراضي
    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  # تراجع اختياري، rootfs
      pve-node-02:
        deployment_type: lxc
        # ... نفس الشكل

    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: vmbr0
    gateway: 192.168.1.1
    mask_cidr: 24                        # 24 اختياري، الافتراضي
    vlanid: 100                         # اختياري
    proxmoxLxcCores: 2                  # تجاوز اختياري
    proxmoxLxcMemoryMb: 4096            # تجاوز اختياري
    proxmoxLxcDiskSizeGb: 30            # تجاوز اختياري
    proxmoxLxcRootFsStorageName: local-lvm  # اختياري، يتجاوز التخزين
الافتراضي للمضيف proxmoxLxcDefaultStorage
```

الاستخدام

توفير (VM أو LXC)

```
ansible-playbook -i hosts/Customer/site.yml
util_playbooks/proxmox.yml
```

حذف

```
ansible-playbook -i hosts/Customer/site.yml
util_playbooks/proxmox_delete.yml
```

يتعامل كتاب اللعب للحذف مع كل من VMs وLXCs تلقائيًا، بغض النظر عن `deployment_type`.

سلوك كتاب اللعب

حراس النوع

- إذا كان هناك VM ولكن موجود مع اسم المضيف المستهدف `deployment_type: lxc` → يفشل كتاب اللعب مع خطأ واضح.
- الحالة العكسية (LXC موجود، الموقع مُعد كـ VM) → نفس الشيء.
- مجموعة UPF على موقع LXC → تحذير فقط (لن يعمل UPF ليس فشلًا صارمًا، لكن).

مورد جديد

- حسب فهرس المضيف داخل `proxmoxServers` يقوم بالتوزيع الدائري عبر إدخالات المجموعة، ما لم يقم المضيف بتعيين `proxmox_server:` (انظر لتثبيت إدخال معين التوزيع عبر العقد).
- يستعلم عن واجهة برمجة تطبيقات Proxmox للحصول على VMID التالي المتاح.
- **VM**: يتم استنساخه من `proxmoxTemplateId`، ويقوم بتعيين مستخدم cloud-init / من `proxmoxTemplateUser` (مستخدم cloud-init = كلمة المرور / مفاتيح SSH ؛ كلمة مرور cloud-init = `local_users` إدخال الخادم إذا تم تعيينه، وإلا أول مفتاح

`proxmoxTemplatePassword` إذا تم تعيينها، وإلا اسم مستخدم cloud-init)، يقوم بتغيير حجم `scsi0`، ويبدأ.

- **LXC**: يتم إنشاؤه من `proxmoxLxcOsTemplate`، rootfs على `proxmoxLxcRootFsStorageName` (المجموعة) ← `proxmoxLxcDefaultStorage` (الخادم) ← `storage` ← `local-lvm` (التراجع النهائي)، يقوم بحقن جميع `local_users.*.public_key` عبر `ssh-public-keys`، `0` ويبدأ. كلمة مرور الجذر مشفرة كـ `mn1Touch@`. جميع LXCs غير مميزة (`unprivileged: 1`) مع تمكين التعشيش (`features: nesting=1`).

مورد موجود

تقوم كل من VM وLXC بتحديث المسارات بمقارنة الحالية مقابل المرغوبة وتطبيق التغييرات:

- تكوين الشبكة (الجسر، علامة VLAN، IP لـ LXC)
- النوى / الذاكرة
- حجم القرص (VM `scsi0` / LXC `rootfs`)
- NICs الثانوية من `secondary_ips` يتم إعادة تشغيل المورد إذا تم تغيير الشبكة أو الموارد.

> **OmniUPF:** تعالج الأدوات إدخال `secondary_ips` المسمى `n3` (أو `data` / `gtp`) كـ NIC وحدة معالجة مركزية للواجهة الرئيسية، بحيث لا يمكن أن يؤدي الحمل الثقيل على مسار البيانات إلى تجويع PFCP/API/SSH (التي تبقى على IP الرئيسي).

وضع العلامات

يتم وضع علامة ع��ى كل مورد بأسماء مجموعته في Ansible + `site_name`.

التوزيع عبر العقد

بشكل افتراضي، تقوم الأدوات بتوزيع المضيفات عبر إدخالات `proxmoxServers` بالتوزيع الدائري، مع نفس المنطق لـ VMs وLXCs. **داخل مجموعة دوره**، الفهرس هو موضع المضيف في `proxmoxServers` (بترتيب الإعلان):

```
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)
```

> **إضافة عقدة ثانية تعيد توزيع كل شيء.** نظرًا لأن التوزيع يعتمد على الموضع، فإن الانتقال من إدخال `proxmoxServers` واحد إلى اثنين يغير مكان وجود مجموعات متعددة المضيفين الموجودة (المضيف-0 → الإدخال-0، المضيف-1 → الإدخال-1، ...). قم بتثبيت المضيفات التي لا تريد نقلها (أدناه) قبل إعادة التشغيل ضد أسطول مختلط.

تثبيت مضيف على عقدة معينة

لفرض `proxmoxServers` إلى **اسم إدخال** `proxmox_server:` قم بتعيين المفتاح الاختياري مضيف على تلك العقدة بغض النظر عن فهرس مجموعته. عند عدم تعيينه، ينطبق التوزيع الدائري تفشل التشغيل مع خطأ يسرد الأسماء الصالحة (لذا `proxmoxServers` أعلاه. قيمة ليست مفتاح لا يمكن أن يؤدي خطأ مطبعي إلى التوزيع الدائري في مكان آخر).

```yaml
device_tester:
  hosts:
    site-device-tester01:
      ansible_host: 10.179.2.50
      proxmox_server: nicklab-prxmx01   # تثبيت على هذا الإدخال من proxmoxServers (→ عقدته)
      # إذا كان تخزين القرص لتلك العقدة يختلف عن اسم التخزين العالمي proxmoxLxcRootFsStorageName،
      # (يتم قراءة التخزين عالميًا، وليس من إدخال الخادم) قم بتجاوزه هنا أيضًا:
      proxmoxLxcRootFsStorageName: local-lvm
```

للمجموعة لتثبيت كل مضيف في `vars:` على مستوى `proxmox_server` يمكن أيضًا تعيين مجموعة على عقدة واحدة.

تجاوزات لكل مضيف

(تثبيت العقدة: انظر التوزيع عبر العقد) يمكن تعيين `proxmox_server` و`deployment_type` ع��ى مستوى المجموعة وأي من مفاتيح LXC (`proxmoxLxcCores`، `proxmoxLxcMemoryMb`، `proxmoxLxcDiskSizeGb`، `proxmoxLxcRootFsStorageName`، `proxmoxLxcOsTemplate`، `host_vm_network`) لكل مضيف تحت اسم المضيف.

```yaml
dns:
  hosts:
    site-dns01:
      ansible_host: 192.168.1.20
      deployment_type: vm          # تجاوز لكل مضيف للافتراضي للموقع
      proxmoxLxcDiskSizeGb: 120    # تجاوز لكل مضيف
      host_vm_network: vmbr1       # تجاوز جسر لكل مضيف
  vars:
    proxmox_interface: vmbr0
    gateway: 192.168.1.1
    mask_cidr: 24  # 24 اختياري، الافتراضي
```

كتيبات الخدمة

تقوم كتيبات الخدمة بنشر وتكوين بنية OmniCore التحتية. تقع في دليل `services/`.

تسلسل كتيبات اللعب

تم هيكلة كتيبات اللعب بشكل هرمي لتمكين نشر سريع وموجه:

```
all.yml
├── setup_users.yml
├── apt_cache.yml          # فقط عندما يتم تعريف مجموعة
apt_cache_servers
├── dns.yml
├── common.yml
├── license_server.yml
├── monitoring.yml
│   └── grafana.yml
├── device_tester.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
│   └── omnisep.yml
├── omniepdg.yml
├── omnitwag.yml
├── omniss7.yml
├── ocs.yml
├── crm.yml
├── ran_monitor.yml
├── 5gc.yml
└── ../util_playbooks/health_check.yml
```

ثوانٍ. يستغرق (`omnimme.yml` مثل) **قم بتشغيل ما تحتاجه فقط.** يستغرق نشر مكون واحد `--limit` عبر 50 مضيفًا وقتًا أطول ولكنه يتعامل مع كل شيء. استخدم `all.yml` تشغيل لتضييق النطاق أكثر.

مرجع سريع

كتيبات اللعب على المستوى الأعلى

كتيب اللعب	النطاق	الوصف
`all.yml`	الشبكة الكاملة	المستخدمون: نشر كامل، DNS، مختبر، المراقبة، الأجهزة، EPC، IMS، ePDG، TWAG، SS7، OCS، CRM، مراقب RAN، 5G Core، وفحص صحي نهائي (من `util_playbooks/`)
`epc.yml`	Packet Core	MME، SGW-C، PGW-C، UPF، HSS، DRA
`ims.yml`	الصوت/IMS	P/I/S-CSCF، خادم التطبيقات، الرسائل، OmniSEP (XCAP/الامتيازات/BSF)
`ocs.yml`	الشحن	CGrateS، مجموعة KeyDB، مزامنة OCS
`monitoring.yml`	المراقبة	Prometheus، Grafana، المصدّرين، HOMER

كتيبات بنية تحتية

كتيب اللعب	الوصف
`common.yml`	تكوين نظام التشغيل الأساسي، الحزم، NTP، وكلاء التسجيل
`setup_users.yml`	حسابات المستخدمين المحليين ومفاتيح SSH
`dns.yml`	نشر خادم DNS
`license_server.yml`	خادم ترخيص OmniCore
`does_netplan_exist.yml`	تحقق مما إذا كان netplan مثبتًا على كل مضيف
`update_mtu.yml`	تعيين وتطبيق MTU الواجهة عبر netplan
`firewall.yml`	قواعد iptables/nftables
`apt_cache.yml`	مرآة APT محلية (إذا كانت مفعلة)

مكونات EPC

الوصف	المكون	كتيب اللعب
كيان إدارة التنقل (4G)	OmniMME	`omnimme.yml`
بوابة الخدمة لطبقة التحكم	OmniSGW-C	`omnisgwc.yml`
بوابة PDN لطبقة التحكم	OmniPGW-C	`omnipgwc.yml`
وظيفة طبقة المستخدم (SGW-U/PGW-U مجمعة)	OmniUPF	`upf.yml` / `omniupf.yml`
خادم المشتركين المنزليين	OmniHSS	`omnihss.yml` / `hss.yml`
وكيل توجيه Diameter	OmniDRA	`omnidra.yml`
بوابة الوصول اللاسلكي الموثوق	OmniTWAG	`omnitwag.yml`
بوابة بيانات الحزمة المتطورة (WiFi Calling)	OmniEPDG	`omniepdg.yml`
وكيل حركة مرور GTP	GTP Proxy	`gtp_proxy.yml`

مكونات IMS

كتيب اللعب	المكون	الوصف
`pcscf.yml`	P-CSCF	وظيفة التحكم في جلسة المكالمات الوكيلة
`icscf.yml`	I-CSCF	CSCF المستجوب
`scscf.yml`	S-CSCF	CSCF الخادم
`as.yml`	OmniTAS	خادم تطبيقات الهاتف
`omnisep.yml`	OmniSEP	البريد، BSF، خادم الامتيازات، XCAP الصوتي المرئي
`xcap.yml`	OmniSEP	(ينشر `omnisep.yml` الاسم القديم لـ نفس دور `omnisep`)
`omnimessage.yml`	OmniMessage / IMS SMSC	وحدة تحكم SMS-over-IP، SMPP، و IMS SMSC (يتم نشرها عبر دور `cscf` مع `cscf_type: smsc_ims`)

5G Core

`gc_nf` المشترك مع `gc.yml` يتم تثبيت كل وظيفة شبكة بواسطة دور 5. يتم نشره عبر دور 5. `nf_package` لكل NF. راجع 5G Core للحصول على تفاصيل الدور، تكوين كل NF، تثبيت الإصدار، ومفاتيح UDM HNET/SUCI.

الوصف	المكون	كتيب اللعب
ينشر جميع وظائف الشبكة المستقلة 5G	5G Core	`5gc.yml`
وظيفة مستودع الشبكة	OmniNRF	
وظيفة إدارة الوصول والتنقل	OmniAMF	
وظيفة خادم المصادقة	OmniAUSF	
وظيفة دعم الربط	OmniBSF	
وظيفة الشحن	OmniCHF	
وظيفة اختيار شريحة الشبكة	OmniNSSF	
وظيفة التحكم في السياسات	OmniPCF	
وكيل الاتصال بالخدمة	OmniSCP	
وظيفة إدارة الجلسات	OmniSMF	
إدارة البيانات الموحدة	OmniUDM	
مستودع البيانات الموحد	OmniUDR	

الدائرة المبدلة

يتم نشره عبر `cs.yml` (الأدوار تحت `circuit_switched/`). راجع Circuit-Switched Core للحصول على تفاصيل أدوار BSC و MSC و SS7.

كتيب اللعب	المكون	الوصف
`cs.yml`	الدائرة المبدلة	ينشر مجال CS القديم
	BSC	وحدة التحكم في المحطة الأساسية (`circuit_switched/bsc`)
	OmniMSC	مركز التحويل المحمول (`circuit_switched/omnimsc`)
	SS7	مجموعة SS7/SIGTRAN (`omniss7` role)

الخدمات الداعمة

كتيب اللعب	الوصف
`ocs.yml`	نظام الشحن عبر الإنترنت (CGrateS + KeyDB)
`crm.yml`	بوابة إدارة العملاء (OmniCRM؛ التكوين الآن في `group_vars/`)
`omniss7.yml`	بوابة SS7/SIGTRAN
`homer.yml`	التقاط حزم SIP/Diameter
`grafana.yml`	لوحات معلومات Grafana والتنبيه
`ran_monitor.yml`	تكامل مراقبة RAN
`omniroam.yml`	مجموعة تجوال OmniRoam (واجهة TAP3/RAP، واجهة برمجة تطبيقات Elixir، مولد)
`omnilcs.yml`	خدمات موقع OmniLCS وبث الخلايا
`device_tester.yml`	منصة اختبار وكيل هاتف OmniWeb (أجهزة الاختبار)

توفير Proxmox (`proxmox.yml`, `proxmox_delete.yml`) موجود تحت `util_playbooks/`. راجع Proxmox VM/LXC و نشر Utility Playbooks.

العمليات

كتيب اللعب	الوصف
`backup.yml`	نسخ احتياطي لقواعد البيانات والتكوينات
`reboot.yml`	إعادة تشغيل مضبوطة للمضيفين
`shutdown.yml`	إيقاف تشغيل سلس
`apt_update.yml`	تحديث الحزم
`apt_refresh_metadata.yml`	تحديث بيانات ا��تعريف الخاصة بذاكرة APT
`speedtest.yml`	اختبار عرض النطاق الترددي للشبكة

كتيبات الاستعادة

كتيب اللعب	الوصف
`restore_applicationserver.yml`	استعادة OmniTAS من النسخة الاحتياطية
`restore_smsc.yml`	استعادة SMSC من النسخة الاحتياطية: وحدة تحكم OmniMessage، بوابة SMPP، وعقد IMS SMSc

> ملاحظة: `health_check.yml`، `restore_hss.yml`، و `restore_ocs.yml` في **ليست**. يقوم Utility Playbooks. راجع `util_playbooks/`. إنها موجودة في `services/`. بتشغيل الفحص الصحي كـ `../util_playbooks/health_check.yml` `all.yml`.

الاستخدام

نشر كل شيء

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

نشر نظام فرعي محدد

```
# فقط نواة الحزمة
ansible-playbook -i hosts/customer/host_files/production.yml
services/epc.yml

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

نشر مكون واحد

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

# تحديث المراقبة فقط
ansible-playbook -i hosts/customer/host_files/production.yml
services/monitoring.yml
```

الحد من مضيفين محددين

```
# تشغيل all.yml ولكن فقط على مضيف واحد
ansible-playbook -i hosts/customer/host_files/production.yml
services/all.yml --limit mme01

# التشغيل على مضيفين محددين متعددين
ansible-playbook -i hosts/customer/host_files/production.yml
services/all.yml --limit "mme01,hss01"

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

الوثائق ذات الصلة

- معمارية النشر - سير العمل العام للنشر
- تكوين ملف المضيف - تعريف البنية التحتية
- تكوين المتغيرات الجماعية - التخصيص
- كتيبات الأدوات - أدوات التشغيل (فحص صحي، استعادة، إلخ.)
- المراقبة والرؤية - Grafana، Prometheus، التنبيه

كتيبات الأدوات

توفر كتيبات الأدوات أدوات تشغيلية لإدارة بنية OmniCore التحتية المنفذة. تقع هذه الكتيبات في `util_playbooks/` ويمكن تشغيلها بشكل مستقل لأداء مهام الصيانة واستكشاف الأخطاء الشائعة.

مرجع سريع

كتيب الأدوات	الغرض
`proxmox.yml`	توفير الآلات الافتراضية و/أو حاويات LXC على Proxmox (حسب `deployment_type`)
`proxmox_delete.yml`	حذف الآلات الافتراضية/حاويات LXC على Proxmox
`proxmox_restart_hung.yml`	فرض إعادة تشغيل الآلات الافتراضية/حاويات LXC عبر واجهة برمجة تطبيقات Proxmox عندما لا تستجيب SSH
`vmware.yml`	توفير الآلات الافتراضية على VMware vSphere
`vmware_delete.yml`	حذف الآلات الافتراضية على VMware vSphere
`health_check.yml`	إنشاء تقرير شامل عن صحة جميع الخدمات
`restore_hss.yml`	استعادة قاعدة بيانات HSS و/أو التكوين من النسخة الاحتياطية
`seed_hss_replication.yml`	تهيئة عقدة OmniHSS فارغة من نظير مأهول (تحميل بيانات النسخ الأولية)
`audit_hss_drift.yml`	تدقيق شبكة OmniHSS لاكتشاف انحراف البيانات (عدد الصفوف + تجزئة المحتوى لكل جدول مرجعي عبر جميع العقد)؛ قراءة فقط
`reconcile_hss_drift.yml`	تسوية انحراف بيانات شبكة OmniHSS عن طريق ملء الصفوف المفقودة عبر العقد (تشغيل جاف افتراضي)
`restore_ocs.yml`	استعادة OCS KeyDB وMySQL من النسخة والتكوين الاحتياطية
`ip_plan_generator.yml`	إنشاء وثائق الشبكة مع مخططات Mermaid

كتيب الأدوات	الغرض
`get_ports.yml`	تدقيق المنافذ المفتوحة والخدمات المستمعة عبر جميع المضيفين
`getLocalCapture.yml`	استرجاع ملفات التقاط الحزم من المضيفين
`delete_local_user.yml`	إزالة حساب مستخدم محلي من جميع المضيفين
`clear_mnesia.yml`	مسح قاعدة بيانات Mnesia لـ OmniSS7 / OmniMessage وإعادة تشغيلها (تدميري)
`delete_license.yml`	حذف ملف الترخيص على خادم الترخيص وإعادة تشغيله

توفير الآلات الافتراضية

تقوم كتيبات توفير الآلات الافتراضية بإنشاء وإدارة الآلات الافتراضية على منصات المحاكاة الافتراضية. تدعم جميع الكتيبات `--limit` لاستهداف مضيفين أو مجموعات معينة.

للحصول على تكوين مفصل راجع نشر Proxmox.

Proxmox

الملفات: `util_playbooks/proxmox.yml`, `util_playbooks/proxmox_delete.yml`

يقوم `proxmox.yml` بتوفير كل من الآلات الافتراضية وحاويات LXC؛ يتم تحديد ما إذا كان `deployment_type` المحلولة (تجاوز لكل مضيف ← إدخال `proxmoxServers` ← الافتراضي `vm`). راجع نشر Proxmox بواسطة LXC المضيف المعطى يصبح آلة افتراضية أو حاوية.

```
# توفير (آلات افتراضية و/أو حاويات LXC، حسب deployment_type)
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/proxmox.yml

# توفير مجموعة معينة فقط
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/proxmox.yml --limit mme

# حذف الآلات الافتراضية/حاويات LXC
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/proxmox_delete.yml --limit old-host
```

إعادة تشغيل المضيفين المعلقين على Proxmox

الملف: `util_playbooks/proxmox_restart_hung.yml`

يتحقق من استجابة SSH لكل مضيف ويفرض إعادة تشغيل أي آلات افتراضية أو حاويات LXC غير مستجيبة عبر واجهة برمجة تطبيقات Proxmox. يتم إيقاف تشغيل آلات QEMU بشكل قسري (أو إعادة تشغيلها إذا كانت قيد التشغيل بالفعل) وإعادة تشغيلها؛ يتم بدء تشغيل حاويات LXC. يتم تخطي المضيفين الذين يستجيبون لـ SSH.

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

# تقييد إلى مضيف أو مجموعة واحدة
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/proxmox_restart_hung.yml --limit hss01
```

VMware vSphere

الملفات: `util_playbooks/vmware.yml`, `util_playbooks/vmware_delete.yml`

تتضمن كلا ال`util_playbooks/vm/vmware/` تتبين ملفات المهام لكل آلة افتراضية تحت (`create_individual_vm.yml` و `delete_individual_vm.yml`).
`vmware.yml` ينتظر أيضًا SSH هو غلاف رقيق حول مهمة الحذف؛ بينما `vmware_delete.yml` القديمة ويطبق netplan بعد التوفير، ثم يزيل ملفات netplan على الآلة الافتراضية الجديدة.

المتطلبات المسبقة: يتطلب تثبيت تبعيات Python (`uv sync`).

الاستخدام:

```
# توفير الآلات الافتراضية
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/vmware.yml

# توفير مجموعة معينة فقط
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/vmware.yml --limit mme

# حذف الآلات الافتراضية
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/vmware_delete.yml --limit old-host
```

المتغيرات المطلوبة:

```yaml
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"
```

Vultr

الملفات: `util_playbooks/vm/vultr/vmProvisionVultr.yml`، `util_playbooks/vm/vultr/vmTeardownVultr.yml`، `util_playbooks/vm/vultr/vmCommonSetup.yml`

يوفر وينزع الحالات على Vultr عبر واجهة برمجة تطبيقات Vultr. يقوم `vmProvisionVultr.yml` بإنشاء الحالات، و `vmTeardownVultr.yml` بإزالتها، و `vmCommonSetup.yml` بأداء إعداد قاعدة بعد التوفير (DNS، إلخ، إعادة التشغيل). هي الثلاثة كتيبات نقطة دخول على مستوى عالٍ بدلاً من تضمينات المهام. تقرأ رمز API من متغير البيئة

من نفس الدليل، لذا قم `cloud-config.yaml` ويشير كتيب التوفير إلى `VULTR_API_TOKEN`، بتشغيلها من `util_playbooks/vm/vultr/`.

```
export VULTR_API_TOKEN=<token>
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/vm/vultr/vmProvisionVultr.yml
```

فحص الصحة

الملف: `util_playbooks/health_check.yml`

ينشئ تقرير صحة شامل بتنسيق HTML يغطي جميع خدمات OmniCore وOmniCall المنفذة.

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

الإخراج: `/tmp/health_check_YYYY-MM-DD HH:MM:SS.html`

المعلومات المجمعة

المكون	البيانات المجمعة
جميع الخدمات	حالة الخدمة، الإصدار، وقت التشغيل
OmniHSS	حالة قاعدة البيانات، اتصالات نظير Diameter
OmniDRA	اتصالات نظير Diameter والحالة
OmniTAS	المكالمات النشطة، الجلسات، التسجيلات، استخدام وحدة المعالجة المركزية
OCS	حالة تكرار KeyDB

استعادة HSS

الملف: `util_playbooks/restore_hss.yml`

يستعيد OmniHSS من ملفات النسخ الاحتياطي. يدعم استعادة قاعدة البيانات فقط، أو التكوين فقط، أو كليهما.

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

تنسيقات ملفات النسخ الاحتياطي

النوع	نمط اسم الملف	المحتويات
قاعدة البيانات	`hss_dump_<hostname>_<timestamp>.sql`	تفريغ قاعدة بيانات OmniHSS (الافتراضي `omnihss`)
التكوين	`hss_<hostname>_<timestamp>.tar.gz`	أرشيف من دليل `/etc/omnihss`

يتم عمل نسخة احتياطية من الحالة الحالية لـ HSS (قاعدة البيانات و/أو التكوين) قبل تشغيل الاستعادة.

المطالبات

المطالبة	مطلوب	الوصف
مسار تفريغ HSS SQL	لا	المسار الكامل إلى ملف تفريغ HSS SQL (اتركه فارغًا لتخطي استعادة قاعدة البيانات)
مسار ملف تكوين HSS tar.gz	لا	المسار الكامل إلى ملف تكوين HSS tar.gz (اتركه فارغًا لتخطي استعادة التكوين)
تأكيد	نعم	لتأكيد عملية الاستعادة على المضيفين المذكورين اكتب `yes`

تهيئة تكرار HSS

الملف: `util_playbooks/seed_hss_replication.yml`

تنضم إلى شبكة تكرار نشطة نشطة OmniHSS يقوم بتحميل البيانات الأولية لمرة واحدة لعقدة فارغة. يعيد `copy_data = false` يتم إنشاء الاشتراكات مع) مأهولة بالفعل. تظهر عقدة جديدة بحيث، `copy_data = true` هذا الكتيب إنشاء الاشتراك إلى نظير مصدر ��احد مختار مع تقوم PostgreSQL بنسخ البيانات الموجودة ثم تواصل البث دون أي فجوة.

يعمل في أحد وضعين:

تفاعلي (افتراضي): اترك متغيرات `seed_source_*` فارغة. يكتشف الكتيب أي هدف محدود فارغ. ثم يستفسر عن الأقران الصالحين للعثور على الأقران التي تحتوي على بيانات، ويقوم بإدراجها مع عدد المشتركين، ويطلب منك اختيار مصدر. يقوم بتغذية الأهداف الفارغة فقط.

```
ansible-playbook -i hosts/customer/host_files/production.yml \
  util_playbooks/seed_hss_replication.yml \
  --limit site-hss01,site-hss02
```

```
الأقران التي تحتوي على بيانات متاحة للتغذية منها:
  1) dev-hss01 (10.4.10.142) - 838 مشترك
أدخل رقم المصدر للتغذية منه
```

صريح: مرر المصدر مباشرة لتخطي القائمة. هذا **يجب** إعادة التغذية حتى لو كانت الهدف تحتوي بالفعل على بيانات (استخدم لاستعادة).

```
ansible-playbook -i hosts/customer/host_files/production.yml \
  util_playbooks/seed_hss_replication.yml \
  --limit site-hss01 \
  -e "seed_source_name=dev-hss01 seed_source_host=10.4.10.142"
```

المعلمات

المعلمة	مطلوب	الافتراضي	الوصف
`seed_source_name`	لا	(مطالب)	اسم مضيف الجرد للنظير المصدر المأهول. يستخدم لتسمية الاشتراك (`sub_from_<name>`) وحل نشره (`<name>_pub`). تركه في وضع التفاعل.
`seed_source_host`	لا	(مطالب)	عنوان IP للنظير المصدر، قابل للوصول على منفذ PostgreSQL الخاص به. اتركه في وضع التفاعل.
`seed_source_port`	لا	منفذ `omnihss.database_port` المستهدف (5432)	منفذ PostgreSQL على النظير المصدر.

> يجب تقديم كل من `seed_source_name` و `seed_source_host` معًا لاستخدام الوضع الصريح؛ عدم تقديم أي منهما يؤدي إلى تفعيل وضع التفاعل.

السلوك

1. يعد عدد المشتركين على كل هدف محدود لتحديد أي منها فارغ.
2. يستفسر عن الأقران الصالحين (مجموعة `hss` المحلية + `connected_omnihss`) للعثور على الأقران التي تحمل بيانات، ويقو�ً بإدراجها. **وضع التفاعل:** يطلب تأكيد `yes` ويطلب اختيارًا. **وضع الصريح:**
3. يضمن أن عنوان IP في `pg_hba.conf` المصدر مسموح به الخاص بالهدف.
4. يقوم بإسقاط وإعادة إنشاء الاشتراك إلى المصدر `WITH (copy_data = false)` (اسم `slot_name` لكل رابط هو `<target>_from_<source>`). يقوم ببث **جميع** الجداول ولكن لا يقوم بأداء مزامنة جداول جماعية.
5. يقوم بملء الصفوف الموجودة في **جدول الماستر** من المصدر باستخدام تحميل آمن من حيث التعارض (`pg_dump --data-only --inserts --on-conflict-do-nothing --disable-triggers`). يتم تضمين `subscriber_state`: كل مشترك يشير إلى واحد بواسطة FK، لذا فإن تخطيه سيجعل كل مشترك تم ملؤه يتيمًا. يتم **عدم** ملء جداول الجلسات المؤقتة (`pdn_session`، `lte_call`).
6. يبلغ عن عدد المشتركين على العقدة التي تم تغذيتها.

> **لماذا يتم تخطي جداول الجلسات:** إنها جداول نشطة؛ قد يتصادم `COPY` الجماعي لها على نظام حي (مفتاح مكرر) مع كتابات متزامنة ويؤدي إلى تعطل جداول المزامنة. يتم إعادة ملئها من النسخ المتماثل المتدفق وحركة المرور الحية بدلاً من ذلك. `subscriber_state` مختلف: يتم إدراج الصفوف فقط في وقت التوفير (1:1 مع المشترك) ويتم تحديثها فقط آمن. بدونها، تشير مشتركات الهدف إلى `--on-conflict-do-nothing` لاحقًا، لذا فإن صفوف الحالة التي لا تصل أبدًا، حيث تقوم النسخ المنطقية بإسقاط تحديثات الصفوف المفقودة بصمت.

> **مهم:** دائمًا إلى العقدة (أو العقد) المستهدفة. في وضع التفاعل، يتم تخطي `--limit` الأهداف غير الفارغة تلقائيًا. راجع تكرار OmniHSS وعمليات العقد للحصول على سير العمل الكامل.

تدقيق انحراف HSS

الملفات: `util_playbooks/audit_hss_drift.yml` (مستقل) و `roles/omnihss/tasks/audit_drift.yml` (التنفيذ المشترك).

يبلغ عن **انحراف البيانات** عبر شبكة OmniHSS النشطة النشطة. يتم إنشاء اشتراكات النسخ المنطقية مع `copy_data = false`، رابط الشبكة التي انحرفت قب�ً البيانا، لذا فإن (اختلافات وقت التغذية) أو عبر كتابة مباشرة/غير مكررة لا تتم تسويتها أبدًا ولا يتم تنبيهها. تظهر

لكل موقع رؤيتها لأن الشبكة تمتد Prometheus النسخ 0 تأخير بينما تختلف البيانات، ولا يمكن لـ عبر موقعين.

(`connected_omnihss` المحلية + أقران `hss` مجموعة) يتصل التدقيق بكل عضو في الشبكة غير المعتمد على الترتيب md5 وبالنسبة لكل جدول مرجعي/تكوين، يقارن عدد الصفوف الدقيق **و** لالتقاط انحراف (عدد + أقصى إصدار) `schema_migrations` للمحتويات الكاملة. كما يقارن المخطط. يتم استبعاد جداول الحالة الساخنة (`subscriber_state`, `pdn_session`, `lte_call`): فهي تختلف بشكل مشروع تحت حركة المرور الحية. يتم تضمين فحص صحي هيكلي واحد فوق المقارنة عبر العقد: `__orphaned_subscribers__`، عدد المشتركين لكل عقدة يؤدي عدد غير صفري على **أي** عقدة إلى فشل. `subscriber_state` الذين تفتقر صفوفهم إلى `subscriber` بدون التدقيق حتى عندما تتفق كل عقدة. قد يؤدي تغذية نسخ `subscriber_state` إلى يتيم كل عقدة بشكل متطابق، وهو ما لا يمكن أن تراه المقارنة النقية للأقران، لذا Postgres عبر العقد. يتم مركز المقارنة على عداء واحد يصل إلى كل قاعدة بيانات `--limit`. فإن تشغيل واحد يغطي الشبكة الكاملة عبر الموقعين من أي جرد، وهو آمن لـ

(عندما يتم تمكين النسخ) `omnihss` في نهاية دور **HSS. يتم تشغيله تلقائيًا بعد كل نشر لـ** يتم تشغيل نفس التدقيق كتحقق من التناسق بعد النشر، وبشكل افتراضي **يفشل التشغيل** إذا تم العثور على انحراف، لذا فإن الشبكة المعطلة من المستحيل تفويتها أثناء النشر. إنه غير تفاعلي (لا المجمع، `services/all.yml` لنشر `free` توقف): يجب أن يكون التدقيق آمنًا تحت استراتيجية حيث ستؤدي التوقف إلى حدوث خطأ. في ذلك النشر المجمع، يتم كتم التدقيق داخل الدور ويتم تشغيل التدقيق بدلاً من ذلك مرة واحدة في كتيب **خطّي** مخصص في نهاية `services/deploy_nfs.yml`. يتم التحكم في السلوك بواسطة متغيري دور:

المتغير	الافتراضي	التأثير
`omnihss_audit_drift`	`true`	تشغيل التدقيق في نهاية الدور
`omnihss_audit_fail_on_drift`	`true`	فشل التشغيل في حالة الانحراف؛ قم بتعيين `false` للتقرير فقط (cron/CI غير تفاعلي)

التدقيق هو **قراءة فقط** في كلتا الحالتين. الافتراضي لكتيب المستقل هو التقرير ثم **الفشل** في حالة الانحراف (حتى يمكنه التحكم في cron/CI):

```
# عند الطلب
ansible-playbook -i hosts/<inv>/host_files/<site>.yml
util_playbooks/audit_hss_drift.yml
# تقرير فقط، دائمًا ينجح
ansible-playbook -i ... util_playbooks/audit_hss_drift.yml -e
audit_fail_on_drift=false
# تعطيل التدقيق التلقائي لنشر واحد
ansible-playbook -i ... services/omnihss.yml -e
omnihss_audit_drift=false
```

HSS تسوية انحراف

الملف: `util_playbooks/reconcile_hss_drift.yml`

يعيد جداول المرجع المتباينة إلى التزامن من خلال جعل كل عقدة تحمل **اتحاد** الصفوف: لكل جدول متباين يقوم بملء الصفوف التي تفتقر إليها العقدة من أقرانها، باستخدام نفس الأسلوب الآمن من حيث التعارض كما في كتيب التغذية (`pg_dump --data-only --inserts --on-conflict-do-nothing --disable-triggers`).

- **غير تدميري**: يتم إدراج الصفوف فقط (تخطي التعارضات). لا يتم حذف أي شيء أو تحديثه. يحل "تفتقر العقدة إلى صفوف"؛ لا يحل "نفس المفتاح الأساسي، قيم مختلفة" أو يزيل الصفوف التي يجب ألا توجد. تحتاج تلك إلى قرار بشري.
- **تفاعلي بشكل افتراضي**: يقوم بالمسح، ويطبع الخطة، و**يسأل** قبل (للمتابعة `yes`) الكتابة؛ أي شيء آخر يتسبب في الإلغاء دون تغييرات. استخدم `-e dry_run=true` للنظر فقط، أو `-e assume_yes=true` للتشغيلات غير التفاعلية.
- تتم الكتابات محليًا على كل هدف كـ superuser لـ postgres (لـ `--disable-triggers`); يتم قراءة الأقران فقط، كمستخدم التطبيق.
- **اتحاد متماثل، وليس موثوقًا للإنتاج**: تشغيله ضد موقع يسحب *صفوف ذلك* ال��وقع إلى الآخرين. لا يتعامل مع الإنتاج كالأستاذ، لذا فإن تشغيله ضد جرد المختبر سيؤدي إلى نشر صفوف المختبر نحو الإنتاج. إذا كان يجب أن يكون الإنتاج هو المصدر الحقيقي، قم بتشغيله من جرد الإنتاج (أو اطلب وضع الموثوقية).
- **عبر المواقع:** يمكن أن تكون العقدة هدفًا فقط عندما تكون في الكتيب، لذا قم بتشغيل هذا ضد **كل** جرد موقع لتغطية كل عقدة. يقوم إعادة المسح في النهاية بالإبلاغ عن أي شيء لا يزال يحتاج إلى تشغيل.

```
# تفاعلي (افتراضي): يسأل قبل الكتابة
ansible-playbook -i hosts/<inv>/host_files/<site>.yml
util_playbooks/reconcile_hss_drift.yml
# نظر فقط، لا يكتب أبدًا
ansible-playbook -i ... util_playbooks/reconcile_hss_drift.yml -e
dry_run=true
# تطبيق غير تفاعلي (CI)
ansible-playbook -i ... util_playbooks/reconcile_hss_drift.yml -e
assume_yes=true
```

> أولاً لرؤية ما انحرف، ومرة أخرى بعد التسوية (ضد `audit_hss_drift.yml`) قم بتشغيل كلا الجردين) لتأكيد التقارب. انظر تك��ار وعمليات العقد OmniHSS.

استعادة OCS

الملف: `util_playbooks/restore_ocs.yml`

يستعيد OCS (CGrateS) من ملفات النسخ الاحتياطي. يتعامل مع تكرار KeyDB متعدد الماستر من خلال الاستعادة إلى عقدة واحدة والسماح للنسخ بالتزامن إلى الآخرين.

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

عملية الاستعادة

1. يوقف CGrateS وKeyDB على جميع عقد OCS
2. يمسح ملفات AOF لمنع التعارض مع البيانات المستعادة
3. يستعيد KeyDB RDB إلى العقدة الأولى، يبدأ KeyDB، ويترك النسخ تتزامن
4. يستعيد MySQL StoreDB (اختياري)
5. يستعيد تكوين `/etc/cgrates` (اختياري)
6. يبدأ CGrateS على جميع العقد

تنسيقات ملفات النسخ الاحتياطي

النوع	نمط اسم الملف	المحتويات
قاعدة بيانات KeyDB	`keydb_dump_<hostname>_<timestamp>.rdb`	لقطة RDB لـ KeyDB
قاعدة بيانات MySQL StoreDB	`cgrates_dump_<hostname>_<timestamp>.sql`	تفريغ MySQL لقاعدة بيانات `cgrates`
التكوين	`cgrates_<hostname>_<timestamp>.tar.gz`	أرشيف من دليل `/etc/cgrates`

المطالبات

المطالبة	مطلوب	الوصف
مسار تفريغ KeyDB RDB	نعم	المسار الكامل إلى ملف النسخ الاحتياطي KeyDB RDB
مسار تفريغ MySQL SQL	لا	المسار الكامل إلى تفريغ SQL لـ CGrateS (تخطي للحفاظ على StoreDB الحالي)
مسار ملف تكوين tar.gz	لا	المسار الكامل إلى النسخة الاحتياطية للتكوين (تخطي للحفاظ على التكوين الحالي)

مولد خطة IP

الملف: `util_playbooks/ip_plan_generator.yml`

ينشئ وثائق الشبكة من الجرد، بما في ذلك:

- تعيينات IP للمضيف (NICs الأساسية والثانوية)
- نظرة عامة على مقطع الشبكة
- مخططات الاتصال بالواجهة (Diameter، GTP، PFCP، SIP، SS7)

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

ملفات الإخراج

الملف	التنسيق	الوصف
`/tmp/ip_plan_<customer>_<site>.md`	Markdown	وثائق نصية
`/tmp/ip_plan_<customer>_<site>.html`	HTML	مخطط تفاعلي مع طبقات قابلة للتصفية

تدقيق المنافذ

الملف: `util_playbooks/get_ports.yml`

يدقق جميع المنافذ المستمعة عبر النشر وينشئ وثائق.

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

ملفات الإخراج

الملف	الوصف
`/tmp/all_ports.csv`	CSV مع اسم المضيف، IP، الخدمة، المنفذ، البروتوكول
`./open_ports.rst`	جدول reStructuredText لوثائق Sphinx

البيانات المجمعة

الحقل	الوصف
اسم المضيف	اسم مضيف الجرد
IP	عنوان IP الخاص بـ `ansible_host`
إصدار IP	IPv4 أو IPv6
النقل	UDP أو TCP
المنفذ	رقم المنفذ المستمع
الخدمة	اسم العملية

استرجاع الالتقاط المحلي

الملف: `util_playbooks/getLocalCapture.yml`

يسترجع أحدث ملفي التقاط حزم من دليل `/etc/localcapture` لكل مضيف.

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

الإ��راج: `./localCapturePcaps/<hostname>/*.pcap`

إدارة المستخدمين

الملف: `util_playbooks/delete_local_user.yml`

يزيل حساب مستخدم محلي من جميع المضيفين في الجرد.

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

المطالبة: أدخل اسم المستخدم للحذف عند المطالبة.

مسح قاعدة بيانات Mnesia

الملف: `util_playbooks/clear_mnesia.yml`

يوقف الخدمة، يحذف دليل قاعدة بيانات Mnesia الخاصة بها، ويعيد تشغيلها. يعمل ضد مجموعة `ss7` (OmniSS7) ومجموعة `omnimessage` (OmniMessage).

> **تحذير:** هذا تدميري. يتم حذف قاعدة بيانات Mnesia بشكل دائم وإعادة بنائها فارغة عند إعادة التشغيل.

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

# تقييد إلى خدمة أو مضيف واحد
ansible-playbook -i hosts/customer/host_files/production.yml
util_playbooks/clear_mnesia.yml --limit ss7
```

حذف الترخيص

الملف: `util_playbooks/delete_license.yml`

يحذف `/etc/license_server/license.json` على مجموعة `license_server` ويعيد تشغيل خادم الترخيص.

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

تشغيل كتيبات الأدوات

الصياغة الأساسية

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

الخيارات الشائعة

الخيار	الوصف
`-i <inventory>`	تحديد ملف الجرد
`--limit <hosts>`	تقييد إلى مضيفين أو مجموعات معينة
`-v` / `-vv` / `-vvv`	زيادة مستوى التفاصيل
`--check`	تشغيل جاف (عرض ما سيتغير)
`--diff`	عرض اختلافات الملفات

الأمثلة

```
# تشغيل فحص الصحة على الإنتاج
ansible-playbook -i hosts/acme/host_files/production.yml
util_playbooks/health_check.yml

# استعادة HSS على مضيف معين
ansible-playbook -i hosts/acme/host_files/production.yml
util_playbooks/restore_hss.yml --limit hss01

# إنشاء خطة IP مع إخراج مفصل
ansible-playbook -i hosts/acme/host_files/production.yml
util_playbooks/ip_plan_generator.yml -v
```