AWS Ground Station 代理使用者指南

AWS Ground Station

AWS Ground Station: AWS Ground Station 代理使用者指南

Table of Contents

概觀

什麼是代 AWS Ground Station 理？

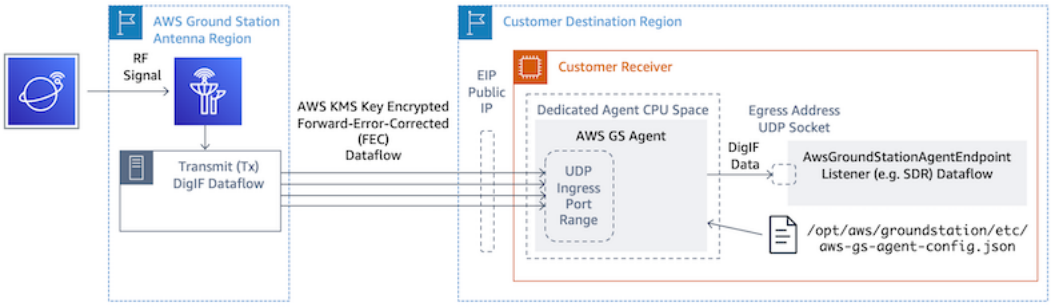
G AWS round Station 代理程式提供RPM，可讓您在接觸 Ground Station 站時接收 (下行連結) 同步寬頻數位中頻 (DigIF) 資料流。AWS您可以選取兩個資料傳送選項：

1. 資料傳送至EC2執行個體-將資料傳送至您擁有的EC2執行個體。您管理 AWS Ground Station 代理程式。如果您需要接近實時的數據處理，此選項可能最適合您。如需資料交付的相關資訊，請參閱 Amazon 彈性運算雲端的資EC2料傳遞指南。
2. 資料交付至 S3 儲存貯體 — 透過 Ground Station 受管服務將資料交付至您擁有的 AWS S3 儲存貯體。如需 S3 資料交付的相關資訊，請參閱入門 AWS Ground Station指南。

這兩種資料傳遞模式都需要您建立一組AWS資源。強烈建議您使用 CloudFormation 來建立AWS資源，以確保可靠性、準確性和支援性。每個聯絡人只能將資料傳送到 EC2 S3，但不能同時傳送給兩者。

> **Note**
>
> 由於 S3 資料交付是 Ground Station 受管服務，因此本指南著重於將資料交付到您的EC2執行個體。

下圖顯示使用軟體定義無線電 (SDR) 或類似接聽程式，從 AWS Ground Station 天線區域到EC2執行個體的 DigIF If 資料流。



AWS Ground Station 代理程式的功能

AWS Ground Station 代理程式會接收數位中頻 (DigIF) 下行連結資料，並輸出解密的資料，以啟用下列功能：

- 從 40 MHz 到 400 頻寬MHz的下行鏈路能力。
- 高速率、低抖動 DigIF 資料傳遞至網路上的任何公有 IP (AWS彈性 IP)。AWS
- 使用前向糾錯（FEC）可靠的數據傳輸。
- 使用客戶管理的 AWS KMS 金鑰進行加密，確保資料傳遞安全。

代理需求

> **Note**
>
> 此 AWS Ground Station 代理程式指南假設您已使用入門指南登AWS Ground Station 入 Ground Station 台。

AWS Ground Station 代理程式接收器EC2執行個體需要一組相依AWS資源，才能可靠且安全地將 DigIF 資料傳遞至您的端點。

1. 一個VPC在其中啟動接EC2收器。
2. 用於數據加密/解密的密AWSKMS鑰。
3. 為SSM工作階段管理員設定的SSH金鑰或EC2執行個體設定檔
4. 網路/安全性群組規則允許下列作業：
    1. UDP來自資料流端 AWS Ground Station 點群組中指定之連接埠的流量。代理程式會保留一系列連續通訊埠，用於將資料傳送至輸入資料流程端點。
    2. SSH存取您的執行個體 (注意：您也可以使用AWS工作階段管理員存取您的EC2執行個體)。
    3. 讀取可公開存取 S3 儲存貯體的存取權以進行代理程式管理
    4. SSL連接埠 443 上允許代理程式與 AWS Ground Station 服務通訊的流量。
    5. 來自 AWS Ground Station 受管理前置詞清單的流量`com.amazonaws.global.groundstation`。

此外，還需要包含公用子網路的VPC組態。如需子網路組態的背景資訊，請參閱VPC使用者指南。

兼容配置：

1. 在公有子網路中與EC2執行個體相關聯的彈性 IP。
2. 與公用子網路ENI中的彈性 IP 相關聯，連接至您的EC2執行個體 (位於與公有子網路位於相同可用區域的任何子網路中)。

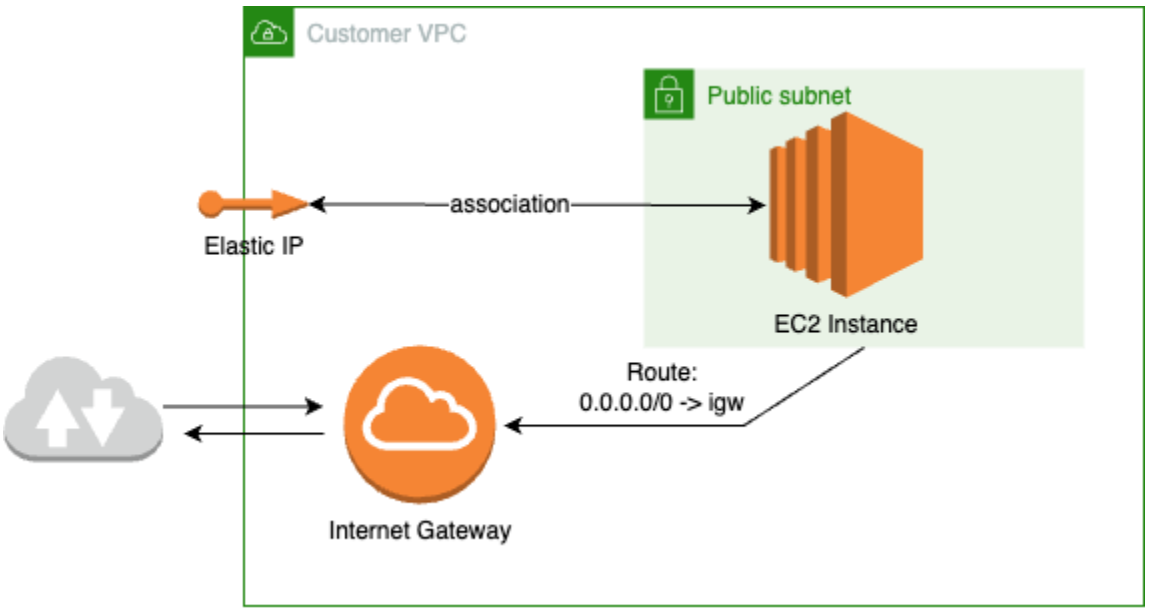
您可以使用與您的EC2執行個體相同的安全群組，也可以指定至少包含下列規則組成的最少一組規則：

- UDP來自資料流端 AWS Ground Station 點群組中指定之連接埠的流量。

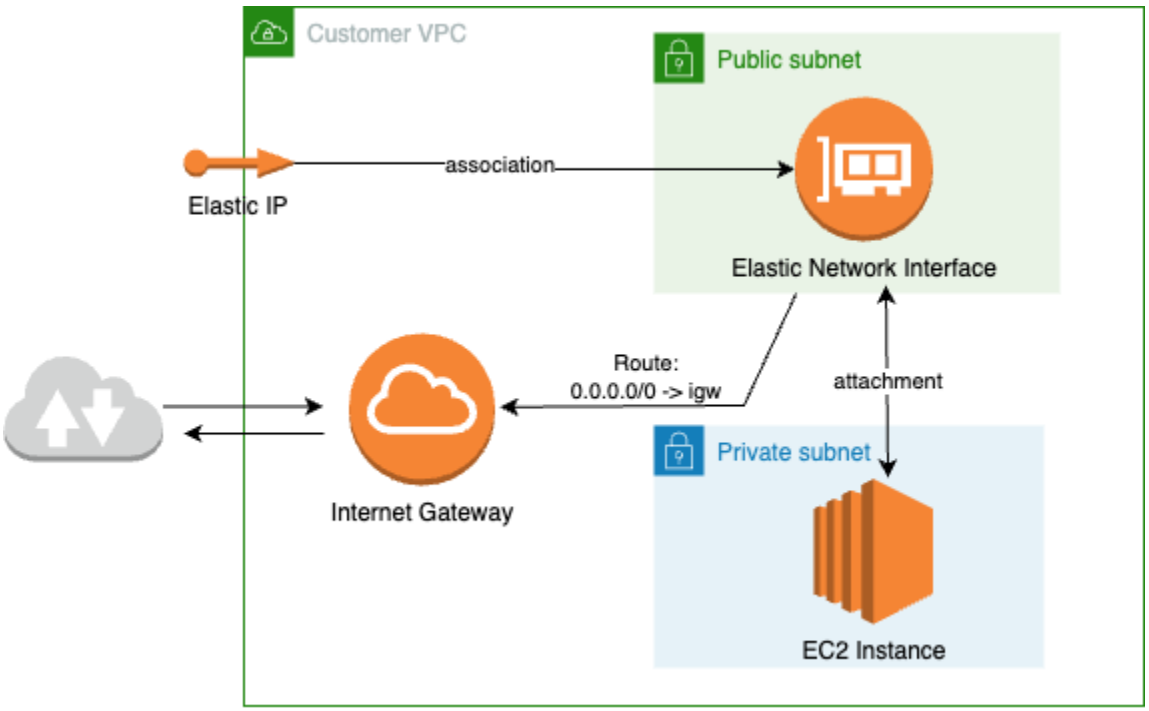
例如，預先設定了這些資源的資 AWS CloudFormation EC2料傳遞範本，請參閱使用 AWS Ground Station 代理程式 (寬頻) 的公共廣播衛星伺服器。

VPC圖表

圖表：在公有子網路中與執行個EC2體相關聯的彈性 IP



圖表：與公用子網路ENI中的彈性 IP 相關聯，連接至私有子網路中的EC2執行個體



支援的作業系統

Amazon Linux 2 與 5.10 以上的內核。

支援的執行個體類型列於 Amazon EC2 執行個體選擇和CPU規劃

透過 AWS Ground Station 代理商傳送資料

下圖概述了 AWS Ground Station 在寬頻數位中頻 (DigIf) 接觸期間，資料如何流經。

AWS Ground Station 代理將處理協調聯繫人的數據通道組件。在排定聯絡人之前，必須正確設定、啟動代理程式，並且必須與註冊 (代理程式啟動時會自動註冊) AWS Ground Station。此外，資料接收軟體 (例如軟體定義的無線電) 必須執行並設定為在 AwsGroundStationAgentEndpointegressAddress.

在幕後， AWS Ground Station 代理程式會接收來自 AWS Ground Station 傳輸中套用的 AWS KMS 加密工作，然後再將其轉送至您的軟體定義 Radio (SDR) 正 egressAddress 在偵聽的目的地端點。 AWS Ground Station 代理程式及其基礎元件會遵守組態檔中設定的CPU界限，以確保它不會影響執行個體上執行的其他應用程式的效能。

您必須讓 AWS Ground Station Agent 在與聯繫人相關的接收器實例上運行。如果您希望在單一接收器執行個體上接收所有資料流，單一 AWS Ground Station 代理程式可以協調多個資料流，如下所示。

多個數據流，單接收器

示例場景：

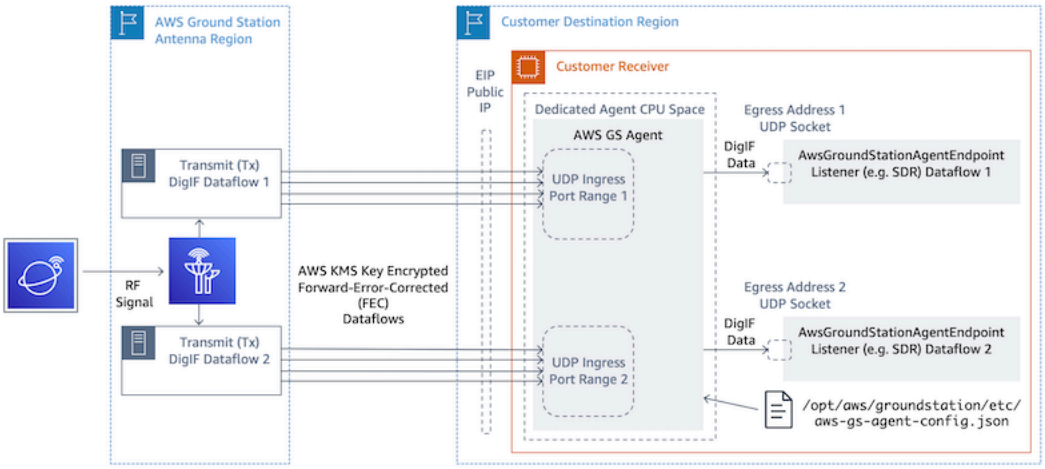
您希望在同一個接收機實例上接收兩個天線下行鏈路作為 DigIF If 數據流。EC2這兩個下行鏈接將是 200 MHz 和 100 MHz。

AwsGroundStationAgentEndpoints:

將有兩個`AwsGroundStationAgentEndpoint`資源，每個數據流一個。兩個端點將具有相同的公用 IP 位址 (`ingressAddress.socketAddress.name`)。輸入`portRange`不應重疊，因為資料流正在同一個執行個體接收。EC2兩者`egressAddress.socketAddress.port`都必須是唯一的。

CPU規劃：

- 1 個核心 (2 vCPU)，用於在執行個體上執行單一 AWS Ground Station 代理程式。
- 6 個核心 (12 vCPU) 以接收 DigIF 料流程 1 (表格中CPU核心規劃的 200 個MHz查閱)。
- 4 個核心 (8 vCPU) 以接收 DigIF 資料流程 2 (表格中CPU核心規劃的 100 個MHz查閱)。
- 專用代理程式CPU空間總計 = 相同插槽上的 11 個核心 (22 vCPU)。



多個數據流，多個接收器

示例場景：

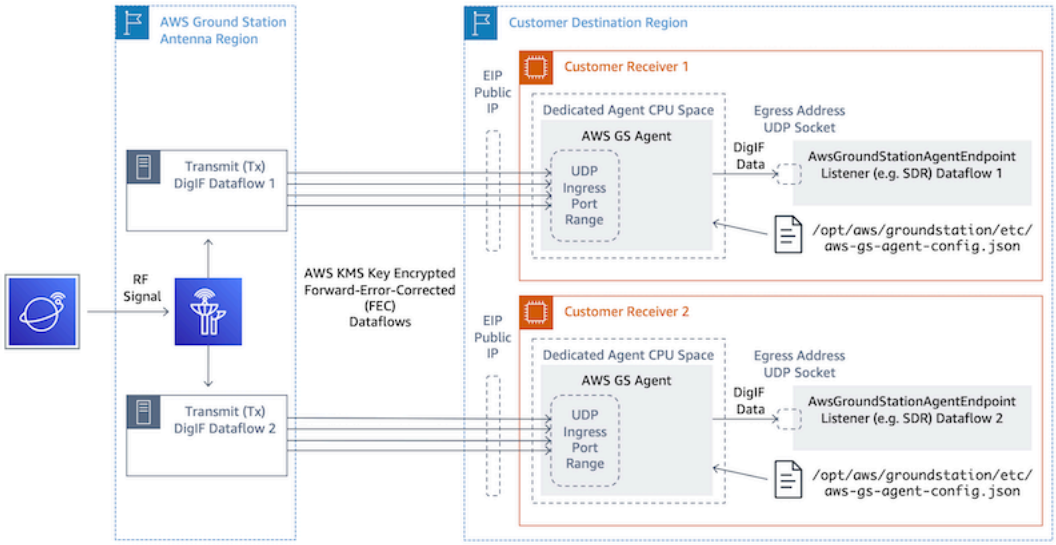
您希望在不同的接收器實例中接收兩個天線下行鏈路作為 DigIF 數據流。EC2這兩個下行鏈接都將是 400 MHz。

AwsGroundStationAgentEndpoints:

將有兩個`AwsGroundStationAgentEndpoint`資源，每個數據流一個。端點將具有不同的公用 IP 位址 (`ingressAddress.socketAddress.name`)。在不同的基礎架構上接收到資料流時`egressAddress`，對連接埠值沒有任`ingressAddress`何限制，也不會互相衝突。

CPU規劃：

- 接收者實例 1
  - 1 個核心 (2 vCPU)，用於在執行個體上執行單一 AWS Ground Station 代理程式。
  - 9 個核心 (18 vCPU) 以接收 DigIF 料流程 1 (表格中CPU核心規劃的 400 個MHz查閱)。
  - 專用代理程式CPU空間總計 = 相同插槽上的 10 個核心 (20 vCPU)。
- 接收器實例 2
  - 1 個核心 (2 vCPU)，用於在執行個體上執行單一 AWS Ground Station 代理程式。
  - 9 個核心 (18 vCPU) 以接收 DigIF 料流程 2 (表格中CPU核心規劃的 400 個MHz查閱)。
  - 專用代理程式CPU空間總計 = 相同插槽上的 10 個核心 (20 vCPU)。


AWS Ground Station
Antenna Region
Transmit (Tx)
DigIF Dataflow 1
RF
Signal
Transmit (Tx)
DigIF Dataflow 2
AWS KMS Key Encrypted
Forward-Error-Corrected
(FEC)
Dataflows
Customer Destination Region
EIP
Public
IP
Customer Receiver 1
Dedicated Agent CPU Space
AWS GS Agent
UDP
Ingress
Port
Range
Egress Address
UDP Socket
DigIF
Data
AwsGroundStationAgentEndpoint
Listener (e.g. SDR) Dataflow 1
/opt/aws/groundstation/etc/
aws-gs-agent-config.json
EIP
Public
IP
Customer Receiver 2
Dedicated Agent CPU Space
AWS GS Agent
UDP
Ingress
Port
Range
Egress Address
UDP Socket
DigIF
Data
AwsGroundStationAgentEndpoint
Listener (e.g. SDR) Dataflow 2
/opt/aws/groundstation/etc/
aws-gs-agent-config.json

Amazon EC2 執行個體選擇和CPU規劃

支援 Amazon EC2 執行個體

由於運算密集型資料傳遞工作流程，AWS Ground Station 代理程式需要專用CPU核心才能運作。我們支援下列執行個體類型。請參CPU核心規劃閱決定哪種執行個體類型最適合您的使用案例。

執行個體類型	預設 vCPUs	預設CPU核心
c5.12xlarge	48	24
c5.18xlarge	72	36
c5.24xlarge	96	48
c5n.18xlarge	72	36
c5n.metal	72	36
c6i.32xlarge	128	64
g4dn.12xlarge	48	24
g4dn.16xlarge	64	32
g4dn.metal	96	48
m4.16xlarge	64	32
m5.12xlarge	48	24
m5.24xlarge	96	48
m6i.32xlarge	128	64
p3dn.24xlarge	96	48
p4d.24xlarge	96	48
r5.24xlarge	96	48

執行個體類型	預設 vCPUs	預設CPU核心
r5.metal	96	48
r5n.24xlarge	96	48
r5n.metal	96	48
r6i.32xlarge	128	64

CPU核心規劃

AWS Ground Station 代理程式需要專用的處理器核心，共用每個資料流程的 L3 快取記憶體。此代理程式的設計目的是利用超執行緒 (HT) CPU 配對，且需要保留 HT 配對以供其使用。超執行緒配對是包含在單一核心內的一對虛擬 CPUs (vCPU)。下表提供資料流資料速率對應至保留給單一資料流之代理程式所需的核心數目。此表格假設串聯湖泊或更新版本，CPUs且對任何支援的執行個體類型都有效。如果您的頻寬介於表格中的項目之間，請選取下一個最高頻寬。

代理程式需要額外的預留核心來進行管理和協調，因此所需的核心總數將是每個資料流所需的核心總和 (從下圖所示)，加上一個額外的核心 (2 vCPUs)。

AntennaDownlink 頻寬 (MHz)	預期 VITA -49.2 分 DigIF 料傳輸率 (MB/ 秒)	核心數量 (HT CPU 配對)	總計 v CPU
50	1000	3	6
100	2000	4	8
150	3000	5	10
200	4000	6	12
250	5000	6	12
300	6000	7	14
350	7000	8	16

AntennaDownlink 頻寬 (MHz)	預期 VITA -49.2 分 DigIF 料傳輸率 (MB/ 秒)	核心數量 (HT CPU 配對)	總計 v CPU
400	8000	9	18

收集架構資訊

`lscpu`提供有關系統架構的資訊。基本輸出會顯示哪些節點 vCPUs (標示為 "CPU「) 屬於哪些NUMA節點 (以及每個NUMA節點共用一個 L3 快取)。下面我們檢查一個`c5.24xlarge`實例，以便收集必要的信息來配置 AWS Ground Station 代理。這包括有用的信息，例如數量vCPUs，內核和 v CPU 到節點關聯。

```
> lscpu
Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit
Byte Order: Little Endian
CPU(s): 96
On-line CPU(s) list: 0-95
Thread(s) per core: 2             <------
Core(s) per socket: 24
Socket(s): 2
NUMA node(s): 2
Vendor ID: GenuineIntel
CPU family: 6
Model: 85
Model name: Intel(R) Xeon(R) Platinum 8275CL CPU @ 3.00GHz
Stepping: 7
CPU MHz: 3601.704
BogoMIPS: 6000.01
Hypervisor vendor: KVM
Virtualization type: full
L1d cache: 32K
L1i cache: 32K
L2 cache: 1024K
L3 cache: 36608K
NUMA node0 CPU(s): 0-23,48-71     <------
NUMA node1 CPU(s): 24-47,72-95    <------
```

AWS Ground Station 代理程式專用的核心應包含每個指派核心 vCPUs 的兩個核心。資料流程的所有核心都必須存在於同NUMA一個節點上。該lscpu命令的-p選項為我們提供了配置代理程序所需的CPU關聯核心。相關欄位包括 CPU (我們稱之為 vCPU)、Core 和 L3 (表示該核心共用哪個 L3 快取記憶體)。請注意，在大多數的 Intel 處理器上，NUMA節點等於 L3 快取記憶體。

考慮lscpu -p輸出的以下子集c5.24xlarge（為了清晰起見，縮寫和格式化）。

```
CPU,Core,Socket,Node,,L1d,L1i,L2,L3
0   0    0      0     0   0   0  0
1   1    0      0     1   1   1  0
2   2    0      0     2   2   2  0
3   3    0      0     3   3   3  0
...
16  0    0      0     0   0   0  0
17  1    0      0     1   1   1  0
18  2    0      0     2   2   2  0
19  3    0      0     3   3   3  0
```

從輸出中，我們可以看到核心 vCPUs 0 包括 0 和 16，核心 vCPUs 1 包括 1 和 17，核心 2 包括 vCPUs 2 和 18。換句話說，超線程對是：0 和 16，1 和 17，2 和 18。

CPU分配示例

舉例來說，我們將使用 350 的雙極性寬頻下行連結c5.24xlarge執行個體。MHz從表中CPU核心規劃我們知道，一個 350 MHz 下行鏈路需要 8 個內核（16vCPUs）的單個數據流。這表示使用兩個資料流的雙極性設定總共需要 16 個核心 (32vCPUs)，再加上代理程式的一個核心 (2vCPUs)。

我們知道lscpu輸出的c5.24xlarge包括NUMA node0 CPU(s): 0-23,48-71和NUMA node1 CPU(s): 24-47,72-95. 由於 NUMA node0 比我們需要的更多，因此我們只會從內核中分配：0-23 和 48-71。

首先，我們將為共用 L3 快取記憶體或節點的每個資料流程選取 8 個核心。NUMA然後，我們將在中的lscpu -p輸出中查找相應的 vCPUs（標記為「CPU」）附錄：lscpu -p輸出（完整）為大。範例核心選取程序可能如下所示：

- 為作業系統保留 0-1 的核心。
- 流程 1：選取對應至 2-9 和 50-57 的鐵芯 vCPUs 2-9。
- 流程 2：選取對 vCPUs 應至 10-17 和 58-65 的核心。

- 代理核心：選擇映射到 vCPUs 18 和 66 的核心 18。

這會導致 vCPUs 2-18 和 50-66，因此提供代理程式的清單為。[2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66]您應該確保您自己的處理序未CPUs如中所述在這些程序上執行與 AWS Ground Station 代理程式一起執行服務和程序。

請注意，在此範例中選取的特定鐵芯有點任意。其他核心集合只要滿足所有共用每個資料流的 L3 快取記憶體的需求，就可以運作。

附錄：lscpu -p輸出（完整）為大

```
> lscpu -p
# The following is the parsable format, which can be fed to other
# programs. Each different item in every column has an unique ID
# starting from zero.
# CPU,Core,Socket,Node,,L1d,L1i,L2,L3
0,0,0,0,,0,0,0,0
1,1,0,0,,1,1,1,0
2,2,0,0,,2,2,2,0
3,3,0,0,,3,3,3,0
4,4,0,0,,4,4,4,0
5,5,0,0,,5,5,5,0
6,6,0,0,,6,6,6,0
7,7,0,0,,7,7,7,0
8,8,0,0,,8,8,8,0
9,9,0,0,,9,9,9,0
10,10,0,0,,10,10,10,0
11,11,0,0,,11,11,11,0
12,12,0,0,,12,12,12,0
13,13,0,0,,13,13,13,0
14,14,0,0,,14,14,14,0
15,15,0,0,,15,15,15,0
16,16,0,0,,16,16,16,0
17,17,0,0,,17,17,17,0
18,18,0,0,,18,18,18,0
19,19,0,0,,19,19,19,0
20,20,0,0,,20,20,20,0
21,21,0,0,,21,21,21,0
22,22,0,0,,22,22,22,0
```

```
23,23,0,0,,23,23,23,0
24,24,1,1,,24,24,24,1
25,25,1,1,,25,25,25,1
26,26,1,1,,26,26,26,1
27,27,1,1,,27,27,27,1
28,28,1,1,,28,28,28,1
29,29,1,1,,29,29,29,1
30,30,1,1,,30,30,30,1
31,31,1,1,,31,31,31,1
32,32,1,1,,32,32,32,1
33,33,1,1,,33,33,33,1
34,34,1,1,,34,34,34,1
35,35,1,1,,35,35,35,1
36,36,1,1,,36,36,36,1
37,37,1,1,,37,37,37,1
38,38,1,1,,38,38,38,1
39,39,1,1,,39,39,39,1
40,40,1,1,,40,40,40,1
41,41,1,1,,41,41,41,1
42,42,1,1,,42,42,42,1
43,43,1,1,,43,43,43,1
44,44,1,1,,44,44,44,1
45,45,1,1,,45,45,45,1
46,46,1,1,,46,46,46,1
47,47,1,1,,47,47,47,1
48,0,0,0,,0,0,0,0
49,1,0,0,,1,1,1,0
50,2,0,0,,2,2,2,0
51,3,0,0,,3,3,3,0
52,4,0,0,,4,4,4,0
53,5,0,0,,5,5,5,0
54,6,0,0,,6,6,6,0
55,7,0,0,,7,7,7,0
56,8,0,0,,8,8,8,0
57,9,0,0,,9,9,9,0
58,10,0,0,,10,10,10,0
59,11,0,0,,11,11,11,0
60,12,0,0,,12,12,12,0
61,13,0,0,,13,13,13,0
62,14,0,0,,14,14,14,0
63,15,0,0,,15,15,15,0
64,16,0,0,,16,16,16,0
65,17,0,0,,17,17,17,0
66,18,0,0,,18,18,18,0
```

```
67,19,0,0,,19,19,19,0
68,20,0,0,,20,20,20,0
69,21,0,0,,21,21,21,0
70,22,0,0,,22,22,22,0
71,23,0,0,,23,23,23,0
72,24,1,1,,24,24,24,1
73,25,1,1,,25,25,25,1
74,26,1,1,,26,26,26,1
75,27,1,1,,27,27,27,1
76,28,1,1,,28,28,28,1
77,29,1,1,,29,29,29,1
78,30,1,1,,30,30,30,1
79,31,1,1,,31,31,31,1
80,32,1,1,,32,32,32,1
81,33,1,1,,33,33,33,1
82,34,1,1,,34,34,34,1
83,35,1,1,,35,35,35,1
84,36,1,1,,36,36,36,1
85,37,1,1,,37,37,37,1
86,38,1,1,,38,38,38,1
87,39,1,1,,39,39,39,1
88,40,1,1,,40,40,40,1
89,41,1,1,,41,41,41,1
90,42,1,1,,42,42,42,1
91,43,1,1,,43,43,43,1
92,44,1,1,,44,44,44,1
93,45,1,1,,45,45,45,1
94,46,1,1,,46,46,46,1
95,47,1,1,,47,47,47,1
```

安裝代理程式

您可以使用下列方式安裝 AWS Ground Station 代理程式：

1. AWS CloudFormation 模板（推薦）。
2. 在 Amazon 上手動安裝EC2。

使用 AWS CloudFormation 範本

資EC2料傳遞 AWS CloudFormation 範本會建立必要的AWS資源，將資料傳送至EC2執行個體。此 AWS CloudFormation 範本使用已預先安裝 AWS Ground Station 代 AWS Ground Station 理AMI程式的受管理。然後，建立的EC2執行個體的開機指令碼會填入代理程式組態檔，並套用必要的效能調整(EC2實例性能調整)。

步驟 1：建立 AWS 資源

利用 AWS Ground Station 代理程式 (寬頻) 使用範本公共廣播衛星來建立AWS資源堆疊。

步驟 2：檢查代理程式狀態

依預設，代理程式已設定且處於作用中狀態 (已啟動)。為了檢查代理狀態，您可以連接到EC2實例（SSH或SSM會話管理器）並查看AWS Ground Station 代理狀態。

手動安裝 EC2

雖然 Ground Station 建議使用 CloudFormation 範本來佈建您的AWS資源，但有些使用案例可能無法使用標準範本。對於這種情況，我們建議您自定義模板以滿足您的需求。如果仍不符合您的需求，您可以手動建立資AWS源並安裝代理程式。

步驟 1：建立AWS資源

有關手動設置聯繫人所需AWS資源的說明，請參見示例任務配置文件配置。

AwsGroundStationAgentEndpoint資源定義了透過 AWS Ground Station 代理程式接收 DigIF 資料流程的端點，對於成功聯絡至關重要。雖然API文件位於「API參考」中，但本節將簡要討論與 AWS Ground Station 代理程式相關的概念。

端點ingressAddress是 AWS Ground Station 代理程式將從天線接收 AWS KMS 加密UDP流量的位置。socketAddressname是EC2執行個體的公用 IP (來自附加的EIP)。portRange應該至少有 300 個連續通訊埠，範圍內已保留任何其他用途。如需說明，請參閱 保留輸入連接埠-影響網路。必須將這些連接埠設定為允許執行接收器執行個體所VPC在安全群組上的UDP輸入流量。

端點egressAddress是代理程式會將 DigIF 資料流傳送給您的位置。您應該有一個應用程序（例如 SDR）UDP通過此位置的套接字接收數據。

步驟 2：建立EC2執行個體

支援以下 AMIs：

1. AWS Ground Station AMI-groundstation-al2-gs-agent-ami-* 其中 * 是建立日期-隨附已安裝代理程式 (建議)。AMI
2. amzn2-ami-kernel-5.10-hvm-x86_64-gp2.

步驟 3：下載並安裝代理程式

> **Note**
>
> 如果您未在上一個步驟中選擇「 AWS Ground Station 代理程式」，則必須完成本節AMI中的步驟。

下載代理

AWS Ground Station 代理程式可從區域特定的 S3 儲存貯體取得，並且可以使用AWS命令列 (CLI) 將代理程式下載到支援EC2執行個體，s3://groundstation-wb-digif-software-${AWS::Region}/aws-groundstation-agent/latest/amazon_linux_2_x86_64/aws-groundstation-agent.rpm其中 ${AWS:: Region} 指的是其中一個支援的 G AWSround Station 主控台和資料傳遞區域。

範例：從本機AWS區域 us-east-2 下載最新的 rpm 版本至 /tmp 資料夾。

```
aws s3 --region us-east-2 cp s3://groundstation-wb-digif-software-us-east-2/aws-groundstation-agent/latest/amazon_linux_2_x86_64/aws-groundstation-agent.rpm /tmp
```

如果您需要下載特定版本的 AWS Ground Station 代理程式，可以從 S3 儲存貯體中的版本特定資料夾下載該代理程式。

範例：將轉速的 1.0.2716.0 版本從本 us-east-2 AWS 域下載至 /tmp 資料夾。

```
aws s3 --region us-east-2 cp s3://groundstation-wb-digif-software-us-east-2/aws-groundstation-agent/1.0.2716.0/amazon_linux_2_x86_64/aws-groundstation-agent.rpm /tmp
```

> **Note**
>
> 如果您要確認RPM您下載的是由出售 AWS Ground Station，請按照的說明進行操作RPM安裝驗證。

安裝代理

```
sudo yum install ${MY_RPM_FILE_PATH}

Example: Assumes agent is in the "/tmp" directory
sudo yum install /tmp/aws-groundstation-agent.rpm
```

步驟 4：設定代理程式

安裝代理程式之後，您必須更新代理程式組態檔。請參閱 設定代理程式。

步驟 5：套用效能調整

AWS Ground Station 代理程式 AMI：如果您在上一個步驟AMI中選擇「 AWS Ground Station 代理程式」，請套用下列效能調整。

- 調整硬件中斷和接收隊列-影響CPU和網絡
- 保留輸入連接埠-影響網路
- 重新開機

其他 AMIs：如果您在上一步AMI中選擇了其他任何內容，請應用下面列出的所有調整EC2實例性能調整並重新啟動實例。

步驟 6：管理代理程式

若要啟動、停止和檢查代理程式狀態，請參閱管理代理程式。

管理代理程式

AWS Ground Station 代理程式提供下列功能，可使用內建 Linux 命令工具來設定、啟動、停止、升級、降級和解除安裝代理程式。

主題

AWS Ground Station 代理配置

導航到/opt/aws/groundstation/etc，其中應包含名為 aws-gs-agent-config .json 的單個文件。請參閱代理程式設定檔

AWS Ground Station 代理啟動

```
#start
sudo systemctl start aws-groundstation-agent

#check status
systemctl status aws-groundstation-agent
```

應該產生輸出顯示代理處於活動狀態。

```
aws-groundstation-agent.service - aws-groundstation-agent
```

```
Loaded: loaded (/usr/lib/systemd/system/aws-groundstation-agent.service; enabled;
 vendor preset: disabled)
Active: active (running) since Tue 2023-03-14 00:39:08 UTC; 1 day 13h ago
Docs: https://aws.amazon.com/ground-station/
Main PID: 8811 (aws-gs-agent)
CGroup: /system.slice/aws-groundstation-agent.service
##8811 /opt/aws/groundstation/bin/aws-gs-agent production
```

AWS Ground Station 代理停止

```
#stop
sudo systemctl stop aws-groundstation-agent

#check status
systemctl status aws-groundstation-agent
```

應該產生顯示代理程式處於非作用中狀態 (已停止)

```
aws-groundstation-agent.service - aws-groundstation-agent
Loaded: loaded (/usr/lib/systemd/system/aws-groundstation-agent.service; enabled;
 vendor preset: disabled)
Active: inactive (dead) since Thu 2023-03-09 15:35:08 UTC; 6min ago
Docs: https://aws.amazon.com/ground-station/
Process: 84182 ExecStart=/opt/aws/groundstation/bin/launch-aws-gs-agent (code=exited,
 status=0/SUCCESS)
Main PID: 84182 (code=exited, status=0/SUCCESS)
```

AWS Ground Station 代理升級

1. 下載最新版本的代理程式。請參閱 下載代理。
2. 停止 代理程式。

```
#stop
sudo systemctl stop aws-groundstation-agent
```

```
#confirm inactive (stopped) state
systemctl status aws-groundstation-agent
```

3. 更新代理程式。

```
sudo yum update ${MY_RPM_FILE_PATH}

# check the new version has been installed correctly by comparing the agent version
 with the starting agent version
yum info aws-groundstation-agent

# reload the systemd configuration
sudo systemctl daemon-reload

# restart the agent
sudo systemctl restart aws-groundstation-agent

# check agent status
systemctl status aws-groundstation-agent
```

AWS Ground Station 代理程式降級

1. 下載您需要的代理程式版本。請參閱 下載代理。
2. 降級代理程式。

```
# get the starting agent version
yum info aws-groundstation-agent

# stop the agent service
sudo systemctl stop aws-groundstation-agent

# downgrade the rpm
sudo yum downgrade ${MY_RPM_FILE_PATH}

# check the new version has been installed correctly by comparing the agent version
 with the starting agent version
```

```
yum info aws-groundstation-agent

# reload the systemd configuration
sudo systemctl daemon-reload

# restart the agent
sudo systemctl restart aws-groundstation-agent

# check agent status
systemctl status aws-groundstation-agent
```

AWS Ground Station 代理程式解

解除安裝代理程式會將 /opt/aws/地站/等/.json 重新命名為 /選項/aws/地表站/等/Json.rpm儲存。aws-gs-agent-config aws-gs-agent-config再次在相同的執行個體上安裝代理程式將會寫入 aws-gs-agent-config .json 的預設值，並且必須使用與AWS資源對應的正確值進行更新。請參閱 代理程式設定檔。

```
sudo yum remove aws-groundstation-agent
```

AWS Ground Station 代理狀態

代理程式狀態為作用中（代理程式正在執行中）或離線（代理程式已停止）。

```
systemctl status aws-groundstation-agent
```

輸出範例顯示代理程式已安裝、非作用中狀態 (已停止) 和已啟用 (開機時啟動服務)。

```
aws-groundstation-agent.service - aws-groundstation-agent
Loaded: loaded (/usr/lib/systemd/system/aws-groundstation-agent.service; enabled;
 vendor preset: disabled)
Active: inactive (dead) since Thu 2023-03-09 15:35:08 UTC; 6min ago
Docs: https://aws.amazon.com/ground-station/
Process: 84182 ExecStart=/opt/aws/groundstation/bin/launch-aws-gs-agent (code=exited,
 status=0/SUCCESS)
```

```
Main PID: 84182 (code=exited, status=0/SUCCESS)
```

AWS Ground Station 代理RPM信息

```
yum info aws-groundstation-agent
```

其輸出如下：

> **Note**
>
> 根據最新的代理程式發佈版本，「版本」可能會有所不同。

```
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Installed Packages
Name        : aws-groundstation-agent
Arch        : x86_64
Version     : 1.0.2677.0
Release     : 1
Size        : 51 M
Repo        : installed
Summary     : Client software for AWS Ground Station
URL         : https://aws.amazon.com/ground-station/
License     : Proprietary
Description : This package provides client applications for use with AWS Ground Station
```

設定代理程式

安裝代理程式之後，您必須在更新代理程式組態檔/opt/aws/groundstation/etc/aws-gs-agent-config.json。

代理程式設定檔

範例

```
{
  "capabilities": [
    "arn:aws:groundstation:eu-central-1:123456789012:dataflow-endpoint-group/
bb6c19ea-1517-47d3-99fa-3760f078f100"
  ],
  "device": {
  "privateIps": [
    "127.0.0.1"
  ],
  "publicIps": [
    "1.2.3.4"
  ],
  "agentCpuCores":
 [ 24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,72,73,74,75,76,77,78,79,80,81
}
```

欄位分解

功能

功能指定為資料流端點群組 Amazon 資源名稱。

必要：是

格式：字符串數組

- 值：能力 ARNs → 字符串

範例：

```
"capabilities": [
    "arn:aws:groundstation:${AWS::Region}:${AWS::AccountId}:dataflow-endpoint-group/
${DataflowEndpointGroupId}"
]
```

裝置

此欄位包含列舉目前 EC2「裝置」所需的其他欄位。

必要：是

格式：物件

成員：

- privateIps
- publicIps
- agentCpuCores
- networkAdapters

privateIps

此欄位目前未使用，但會包含在 future 的使用案例中使用。如果沒有包含任何值，它將默認為 [「127.0.0.1」]

必要：False

格式：字符串數組

- 值：IP 位址 → 字串

範例：

```
"privateIps": [
    "127.0.0.1"
],
```

publicIps

每個資料流端點群組的彈性 IP (EIP)。

必要：是

格式：字符串數組

- 值：IP 位址 → 字串

範例：

```
"publicIps": [
    "9.8.7.6"
],
```

agentCPUCores

這會指定為 aws-gs-agent 處理序保留哪些虛擬核心。如需適當設定此值CPU核心規劃的需求，請參閱。

必要：是

格式：詮釋數組

- 價值觀：核心編號 → 整型

範例：

```
"agentCpuCores": [

 24,25,26,27,28,29,30,31,32,33,34,35,36,37,38,39,40,41,42,43,44,72,73,74,75,76,77,78,79,80,81,8
]
```

networkAdapters

這對應於將接收資料的乙太網路介面卡或連接到ENIs的介面卡。

必要：False

格式：字符串數組

- 值：以太網適配器名稱（可以通過運行找到它們`ifconfig`）

範例：

```
"networkAdapters": [
    "eth0"
]
```

EC2實例性能調整

> **Note**
>
> 如果您使用 CloudFormation 範本佈建AWS資源，則會自動套用這些調整。如果您使用AMI或手動建立EC2執行個體，則必須套用這些效能調整，才能達到最可靠的效能。

請記得在套用任何調整後重新啟動執行個體。

主題

調整硬件中斷和接收隊列-影響CPU和網絡

本節設定 systemd、SMPIRQs接收封包轉向 (RPS) 和接收流量轉向 () 的CPU核心用途。RFS如附錄：中斷RPS/調諧的推薦參數需根據您使用的執行個體類型建議的一組建議設定，請參閱。

1. 將系統程序釘選到遠離代理程式CPU核心。
2. 遠離代理程式CPU核心的硬體中斷要求。
3. 設定RPS以防止單一網路介面卡的硬體佇列成為網路流量瓶頸。
4. 設定RFS以提高CPU快取命中率，進而減少網路延遲。

由提供的`set_irq_affinity.sh`腳本為您RPM配置上述所有內容。添加到 crontab，所以它在每次啟動時應用：

```
echo "@reboot sudo /opt/aws/groundstation/bin/set_irq_affinity.sh
 '${interrupt_core_list}' '${rps_core_mask}' >> /var/log/user-data.log 2>&1" >>/var/
spool/cron/root
```

- 替換`interrupt_core_list`為保留給內核和操作系統的內核-通常是第一個和第二個以及超線程核心對。這不應與上述選取的核心重疊。（例如：對於超線程，96-實例為 '0,1,48,49'）。CPU
- `rps_core_mask`是十六進位元遮罩，指定CPUs要處理傳入封包的位元遮罩，每個數字都代表 4 CPUs。它也必須以逗號分隔，每 8 個字符從右邊開始。建議允許所有CPUs並讓緩存處理平衡。
  - 若要查看每個執行個體類型的建議參數清單，請參閱附錄：中斷RPS/調諧的推薦參數。
- 96-CPU 實例的示例：

```
echo "@reboot sudo /opt/aws/groundstation/bin/set_irq_affinity.sh '0,1,48,49'
 'ffffffff,ffffffff,ffffffff' >> /var/log/user-data.log 2>&1" >>/var/spool/cron/root
```

調整 Rx 中斷合併-影響網絡

中斷聯合有助於防止主機系統發生過多中斷，並有助於提高網路輸送量。使用此配置時，會收集封包，每 128 微秒產生一個中斷。添加到 crontab，所以它在每次啟動時應用：

```
echo "@reboot sudo ethtool -C ${interface} rx-usecs 128 tx-usecs 128 >>/var/log/user-
data.log 2>&1" >>/var/spool/cron/root
```

- 更換`interface`為設定為接收資料的網路介面卡 (乙太網路介面卡)。一般而言，這是`eth0`因為這是指派給EC2執行個體的預設網路介面。

調整 Rx 環形緩衝區-影響網絡

增加 Rx 環形緩衝區的響鈴項目數，以防止在突發連線期間封包丟失或溢位。添加到 crontab 中，以便在每次啟動時正確設置：

```
echo "@reboot sudo ethtool -G ${interface} rx 16384 >>/var/log/user-data.log 2>&1" >>/
var/spool/cron/root
```

- 更換interface為設定為接收資料的網路介面卡 (乙太網路介面卡)。一般而言，這是eth0因為這是指派給EC2執行個體的預設網路介面。
- 如果設定 c6i.32xlarge 實體，則需要修改指令以將環形緩衝區設定為，而不是。8192 16384

調 CPU C 狀態-影響 CPU

設定 CPU C 狀態以防止閒置，這可能會在接觸開始期間造成封包遺失。需要重新啟動實例。

```
echo "GRUB_CMDLINE_LINUX_DEFAULT=\"console=tty0 console=ttyS0,115200n8
 net.ifnames=0 biosdevname=0 nvme_core.io_timeout=4294967295 intel_idle.max_cstate=1
 processor.max_cstate=1 max_cstate=1\"" >/etc/default/grub
echo "GRUB_TIMEOUT=0" >>/etc/default/grub
grub2-mkconfig -o /boot/grub2/grub.cfg
```

保留輸入連接埠-影響網路

保留輸入位址連接埠範圍內AwsGroundStationAgentEndpoint的所有連接埠，以避免與核心使用衝突。連接埠使用衝突會導致連絡人和資料傳送失敗。

```
echo "net.ipv4.ip_local_reserved_ports=${port_range_min}-${port_range_max}" >> /etc/
sysctl.conf
```

- 範例：echo "net.ipv4.ip_local_reserved_ports=42000-43500" >> /etc/sysctl.conf。

重新開機

成功套用所有調整之後，請重新啟動執行個體，讓調整生效。

```
sudo reboot
```

附錄：中斷RPS/調諧的推薦參數

此段落決定用於調整段落「調整硬體中斷和接收佇列-影響CPU」和「網路」的建議參數值。

系列	執行個體類型	${中斷列表}	${克雷 _ 掩碼}
c6i	• c6i.32xlarge	• 0,1,64,65	• FFFFFF，FFFFFF，FFFFFF，FFFFFF
C5	• c5.24xlarge • c5.18xlarge • c5.12xlarge	• 0,1,48,49 • 0,1,36,37 • 0,1,24,25	• FFFFFF，FFFFFF，FFFFFF • FF, FFFFFF, FFFFFF • FFFF, FFFF
C5n	• c5n.metal • c5n.18xlarge	• 0,1,36,37 • 0,1,36,37	• FF, FFFFFF, FFFFFF • FF, FFFFFF, FFFFFF
m5	• m5.24xlarge • m5.12xlarge	• 0,1,48,49 • 0,1,24,25	• FFFFFF，FFFFFF，FFFFFF • FFFF, FFFF
r5	• r5.metal • r5.24xlarge	• 0,1,48,49 • 0,1,48,49	• FFFFFF，FFFFFF，FFFFFF

系列	執行個體類型	${中斷列表}	${克雷_掩碼}
			• FFFFFF，FFFFFF，FFFFFF
r5n	• r5n.metal • r5n.24xlarge	• 0,1,48,49 • 0,1,48,49	• FFFFFF，FFFFFF，FFFFFF • FFFFFF，FFFFFF，FFFFFF
g4dn	• g4dn.metal • g4dn.16xlarge • g4dn.12xlarge	• 0,1,48,49 • 0,1,32,33 • 0,1,24,25	• FFFFFF，FFFFFF，FFFFFF • FFFFFF，FFFFFF • FFFF，FFFF
p4d	• p4d.24xlarge	• 0,1,48,49	• FFFFFF，FFFFFF，FFFFFF
p3dn	• p3dn.24xlarge	• 0,1,48,49	• FFFFFF，FFFFFF，FFFFFF

準備接受 DigIF 連絡人

1. 檢閱CPU核心規劃以取得所需的資料流程，並提供代理程式可使用的核心清單。請參閱 CPU核心規劃。
2. 檢閱代 AWS Ground Station 理程式組態檔。請參閱 AWS Ground Station 代理配置。
3. 確認已套用必要的效能調整。請參閱 EC2實例性能調整。
4. 確認您正在遵循所有被稱為的最佳實踐。請參閱 最佳實務。
5. 透過下列方式，確認 AWS Ground Station 代理程式在排定的聯絡人開始時間之前啟動：

```
systemctl status aws-groundstation-agent
```

6. 在排定的聯絡人開始時間之前，透過下列方式確認 AWS Ground Station 代理程式狀況良好：

```
aws groundstation get-dataflow-endpoint-group --dataflow-endpoint-group-id
 ${DATAFLOW-ENDPOINT-GROUP-ID} --region ${REGION}
```

驗證您`agentStatus`的`awsGroundStationAgentEndpoint`是ACTIVE和`auditResults`是HEALTHY。

最佳實務

Amazon EC2 最佳實踐

遵循目前的EC2最佳做法，並確保足夠的資料儲存可用性

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-best-practices.html

排程器

如果相應的進程沒有固定到特定的核心，Linux 調度程序可以重新排序UDP套接字上的數據包。在數據傳輸期間，任何發送或接收UDP數據的線程都應將自己固定到特定的核心。

AWS Ground Station 管理前綴列表

建議在指定網路規則時使用 `com.amazonaws.global.groundstation` AWS-managed 前置詞清單，以允許來自 Antenna 的通訊。如需有關AWS受管理字首清單的詳細資訊，請參閱使用AWS受管理的前置詞

單一連絡人限制

AWSGround Station 代理程式支援每個聯絡人多個串流，但一次只支援單一聯絡人。為了避免排程問題，請勿在多個資料流程端點群組之間共用執行個體。如果單一代理程式組態與多個不同的組態相關聯DFEGARNs，將無法註冊。

與 AWS Ground Station 代理程式一起執行服務和程序

在與代理程式相同的EC2執行個體上啟動服務和處 AWS Ground Station 理程序時，請務必將它們繫結到 AWS Ground Station 代理程式和 Linux 核心 vCPUs 不使用，因為這可能會造成瓶頸，甚至在連絡人期間遺失資料。綁定到特定的這種概念 vCPUs 被稱為親和力。

要避免的核心：

- `agentCpuCores`從 代理程式設定檔
- 來自 調整硬件中斷和接收隊列-影響CPU和網絡 的 `interrupt_core_list`。
  - 默認值可以從中找到 附錄：中斷RPS/調諧的推薦參數

做為使用執行個c5.24xlarge體的範例

如果您指定

"agentCpuCores": [24,25,26,27,72,73,74,75]"

並跑

```
echo "@reboot sudo /opt/aws/groundstation/bin/set_irq_affinity.sh
'0,1,48,49' 'ffffffff,ffffffff,ffffffff' >> /var/log/user-data.log 2>&1"
>>/var/spool/cron/root
```

然後避免使用以下內核：

0,1,24,25,26,27,48,49,72,73,74,75

親和服務（系統）

新推出的服務將自動與interrupt_core_list前面提到的服務產生關係。如果您啟動的服務使用案例需要額外的核心，或是需要較少擁擠的核心，請遵循本節。

使用命令檢查您的服務當前配置為什麼親和性：

```
systemctl show --property CPUAffinity <service name>
```

如果你看到一個空值CPUAffinity=，這意味著它可能會使用上面的命令中的默認內核 ...bin/set_irq_affinity.sh <using the cores here> ...

若要覆寫並設定特定相似性，請執行下列步驟尋找服務檔案的位置：

```
systemctl show -p FragmentPath <service name>
```

打開並修改文件（使用 vinano，等等），然後將其放CPUAffinity=<core list>在以下[Service]部分中：

```
[Unit]
...

[Service]
...
CPUAffinity=2,3

[Install]
...
```

儲存檔案並重新啟動服務以套用相似性：

```
systemctl daemon-reload
systemctl restart <service name>

# Additionally confirm by re-running
systemctl show --property CPUAffinity <service name>
```

如需詳細資訊，請造訪 RHEL 8-管理、監控及更新核心-第 27 章。使用 systemd 設定CPU相似性和NUMA原則。

親和化程序 (指令碼)

強烈建議您對新啟動的腳本和進程進行手動關聯，因為默認的 Linux 行為將允許它們使用機器上的任何核心。

為了避免任何正在運行的進程（如 python，bash 腳本等）的核心衝突，請使用以下命令啟動該進程：

```
taskset -c <core list> <command>
# Example: taskset -c 8 ./bashScript.sh
```

如果處理序已在執行中，請使用pidoftop、或等指令ps來尋找特定處理程序的處理程序 ID (PID)。使用，PID您可以看到目前的相似性：

```
taskset -p <pid>
```

並可以使用以下命令修改它：

```
taskset -p <core mask> <pid>
# Example: taskset -p c 32392 (which sets it to cores 0xc -> 0b1100 -> cores 2,3)
```

如需有關工作集的詳細資訊，請參閱工作集-Linux 線上手冊

故障診斷

代理程式無法啟動

AWS Ground Station 代理程式可能因多種原因而無法啟動，但最常見的案例可能是代理程式組態檔設定錯誤。啟動代理程式之後 (請參閱AWS Ground Station 代理啟動)，您可能會看到下列狀態：

```
#agent is automatically retrying a restart
aws-groundstation-agent.service - aws-groundstation-agent
Loaded: loaded (/usr/lib/systemd/system/aws-groundstation-agent.service; enabled;
 vendor preset: disabled)
Active: activating (auto-restart) (Result: exit-code) since Fri 2023-03-10 01:48:14
 UTC; 23s ago
Docs: https://aws.amazon.com/ground-station/
Process: 43038 ExecStart=/opt/aws/groundstation/bin/launch-aws-gs-agent (code=exited,
 status=101)
Main PID: 43038 (code=exited, status=101)

#agent has failed to start
aws-groundstation-agent.service - aws-groundstation-agent
Loaded: loaded (/usr/lib/systemd/system/aws-groundstation-agent.service; enabled;
 vendor preset: disabled)
Active: failed (Result: start-limit) since Fri 2023-03-10 01:50:15 UTC; 13s ago
Docs: https://aws.amazon.com/ground-station/
Process: 43095 ExecStart=/opt/aws/groundstation/bin/launch-aws-gs-agent (code=exited,
 status=101)
Main PID: 43095 (code=exited, status=101)
```

故障診斷

```
sudo journalctl -u aws-groundstation-agent | grep -i -B 3 -A 3 'Loading Config' | tail
 -6
```

可能會導致以下輸出：

```
launch-aws-gs-agent[43095]: Running with options Production(ProductionOptions
 { endpoint: None, region: None })
launch-aws-gs-agent[43095]: Loading Config
launch-aws-gs-agent[43095]: System has 96 logical cores
systemd[1]: aws-groundstation-agent.service: main process exited, code=exited,
 status=101/n/a
systemd[1]: Unit aws-groundstation-agent.service entered failed state.
```

無法在「載入組 Config」之後啟動代理程式，表示代理程式組態發生問題。請參閱代理程式設定檔以驗證您的代理程式組態。

AWS Ground Station 用戶端記錄

AWS Ground Station 代理程式會將聯絡人執行、錯誤和健全狀態的相關資訊寫入執行代理程式的執行個體上的記錄檔。您可以透過手動連線至執行個體來檢視記錄檔。

您可以在下列位置檢視代理程式記錄檔。

```
/var/log/aws/groundstation
```

無可用的聯絡人

安排聯繫人需要一個健康的 AWS Ground Station 代理。請 AWS Ground Station API通過以下方式查詢以下方式確認您的 AWS Ground Station 代理已啟動並且它是否正常 get-dataflow-endpoint-group：

```
aws groundstation get-dataflow-endpoint-group --dataflow-endpoint-group-id ${DATAFLOW-
ENDPOINT-GROUP-ID} --region ${REGION}
```

驗證您agentStatus的awsGroundStationAgentEndpoint是ACTIVE和auditResults是HEALTHY。

取得支援

透過支援與 Ground Station 團隊聯 S AWS upport。

1. 提`contact_id`供任何受影響的聯繫人。如果沒有此資訊，AWS Ground Station 小組就無法調查特定的連絡人。
2. 提供已採取的所有疑難排解步驟的詳細資料
3. 在我們的疑難排解指南中提供執行命令時發現的任何錯誤訊息。

代理程式版本注

最新的代理版本

版本

發行日期:3 月 27 日

Support 結束日期：二零二零二年八月三十一日

RPM總和檢查碼：

- SHA256: `108f3aceb00e5af549839cd766c56149397e448a6e1e1429c89a9eebb6bc0fc1`
- MD5: `65b72fa507fb0af32651adbb18d2e30f`

變更：

- 在任務啟動期間為選取的可執行檔版本新增代理程式
- 添加配置文件支持，以在其他版本可用時避免特定的可執行版本。
- 新增網路和路由診斷。
- 其他安全功能。
- 修正部分量度報告錯誤寫入 stdout/journal 而非記錄檔的問題。
- 妥善處理網路無法到達的通訊端錯誤。
- 測量來源和目標代理程式之間的封包遺失和延遲。
- 發行 2.0 aws-gs-datapipe 版以支援新的通訊協定功能，以及透明地將聯絡人升級至新通訊協定的能力。

已淘汰的代理程

1.0.2942.0 版本

發行日期:6 月 26 日

Support 結束日期：二零二四年五月三十一日

RPM總和檢查碼：

- SHA256: 7d94b642577504308a58bab28f938507f2591d4e1b2c7ea170b77bea97b5a9b6
- MD5: 661ff2b8f11aba5d657a6586b56e0d8f

變更：

- 已新增在磁碟上更新代理程式RPM且需要重新啟動代理程式以使變更生效的錯誤記錄。
- 新增網路調整驗證，以確保已遵循並正確套用 Agent 使用者指南調整步驟。
- 修正在代理程式記錄檔中造成有關記錄檔的錯誤警告的錯誤。
- 改善封包遺失偵測。
- 已更新代理程式安裝，以防止安裝或升級代理程式 (RPM如果代理程式已在執行)。

版本：

發行日期：3 月 15 日

Support 結束日期：二零二四年五月三十一日

RPM總和檢查碼：

- SHA256: cb05b6a77dfcd5c66d81c0072ac550affbcefefc372cc5562ee52fb220844929
- MD5: 65266490c4013b433ec39ee50008116c

變更：

- 當代理程式在任務期間發生失敗時啟用上傳記錄
- 修復提供的網絡調整腳本中的 linux 兼容性錯誤。

版本

發行日期：二零二三年二月十五日

Support 結束日期：二零二四年五月三十一日

RPM總和檢查碼：

- SHA256: 77cfe94acb00af7ca637264b17c9b21bd7afdc85b99dffdd627aec9e99397489
- MD5: b8533be7644bb4d12ab84de21341adac

變更：

- 首次正式推出代理程式版本。

RPM安裝驗證

最新RPM版本，MD5哈希驗證RPM，並使用 sha256sum SHA256 哈希如下所示。結合這些值可用於驗證用於地面站代理程式的RPM版本。

最新的代理版本

版本

發行日期:3 月 27 日

Support 結束日期：二零二零二年八月三十一日

RPM總和檢查碼：

- SHA256: `108f3aceb00e5af549839cd766c56149397e448a6e1e1429c89a9eebb6bc0fc1`
- MD5: `65b72fa507fb0af32651adbb18d2e30f`

變更：

- 在任務啟動期間為選取的可執行檔版本新增代理程式
- 添加配置文件支持，以在其他版本可用時避免特定的可執行版本。
- 新增網路和路由診斷。
- 其他安全功能。
- 修正部分量度報告錯誤寫入 stdout/journal 而非記錄檔的問題。
- 妥善處理網路無法到達的通訊端錯誤。
- 測量來源和目標代理程式之間的封包遺失和延遲。
- 發行 2.0 aws-gs-datapipe 版以支援新的通訊協定功能，以及透明地將聯絡人升級至新通訊協定的能力。

驗證 RPM

您將需要能夠驗證此RPM安裝的工具包括：

- 沙 256

- 每分鐘

這兩種工具都默認在 Amazon Linux 2 上提供。這些工具將有助於驗證RPM您使用的是正確的版本。首先RPM從 S3 儲存貯體下載最新版本 (下載代理請參閱下載指示RPM)。下載此文件後，將有一些事情要檢查：

- 計算文件的總和。RPM從您正在使用的計算執行個體的命令列執行下列動作：

```
sha256sum aws-groundstation-agent.rpm
```

  拿這個值並將其與上表進行比較。這表明下載的RPM文件是一個有效的文件，用於使用該 AWS Ground Station 已經向客戶出售。如果雜湊不相符，請勿安裝，然後RPM將其從運算執行個體中刪除。
- 也請檢查檔案的MD5雜湊值，以確保RPM未遭到入侵。若要這麼做，請執行下列RPM命令來使用命令列工具：

```
rpm -Kv ./aws-groundstation-agent.rpm
```

  驗證此處列出的MD5哈MD5希值與上表中的版本的哈希相同。一旦這兩個雜湊都已根據AWS文件中列出的表格進行驗證後，客戶就可以確保下載和安裝的檔案是安全且毫不妥協的. RPM RPM

AWS Ground Station 代理程式使用指南的文件歷史記錄

下表說明《 AWS Ground Station 代理程式使用者指南》每個發行版本中的重要變更。

變更	描述	日期
文件更新	在代理程式需求中新增關於將子網路和 Amazon EC2 執行個體保留在相同可用區域的註解。	2024年7月18日
文件更新	將 AWS Ground Station 代理程式分割為自己的使用者指南。有關以前的更改，請參閱：AWSGround Station 用戶指南的文檔歷史記錄。	2024年7月18日

本文為英文版的機器翻譯版本，如內容有任何歧義或不一致之處，概以英文版為準。