aws

開發人員指南

Amazon Managed Blockchain Query

Amazon Managed Blockchain Query: 開發人員指南

Table of Contents

什麼是 Amazon Managed Blockchain (AMB) 查詢？

Amazon Managed Blockchain (AMB) 是一項全受管服務，旨在協助您在公有和私有區塊鏈上建置彈性的 Web3 應用程式。使用 AMB Access 來即時無伺服器存取多個區塊鏈。建置可供 Web3-ready的應用程式，無需部署專門的區塊鏈基礎設施，並保持它們與區塊鏈網路的連線。透過 AMB Query，您可以使用開發人員易用的 API 操作，從多個區塊鏈存取即時和歷史資料。標準化區塊鏈資料可以與 AWS 服務整合，而不需要專門的區塊鏈基礎設施或 ETL （擷取、轉換和載入）。所有 AMB 功能都會針對機構級和主流消費者應用程式建置安全地擴展。

Amazon Managed Blockchain (AMB) Query 透過開發人員易用的 API 操作，提供標準化、多區塊鏈資料集的無伺服器存取。您可以使用 AMB Query 快速運送需要來自一或多個公有區塊鏈資料的應用程式，而不需要額外負荷來剖析區塊鏈資料、追蹤合約，以及維護專門的索引基礎設施。無論您是分析易記字符或非易記字符 (NFTs) 的歷史字符餘額、檢視指定錢包地址的交易歷史記錄，還是對 Ether 等原生加密貨幣的分佈執行資料分析，AMB Query 都可讓您存取區塊鏈資料。

您是第一次使用 AMB 查詢嗎？

如果您是第一次使用 AMB Query，建議您先閱讀以下章節：

- 關鍵概念：Amazon Managed Blockchain (AMB) 查詢
- 設定 Amazon Managed Blockchain (AMB) 查詢
- Amazon Managed Blockchain (AMB) 查詢入門
- Amazon Managed Blockchain (AMB) 查詢的使用案例

關鍵概念：Amazon Managed Blockchain (AMB) 查詢

> **Note**
>
> 本指南假設您熟悉基本的區塊鏈概念。這些概念包括分散、權杖、合約、交易、proof-of-work、錢包、公有和私有金鑰、任務、採礦、分片等。

Amazon Managed Blockchain (AMB) Query 可讓您方便存取多區塊鏈網路資料，讓您更輕鬆地擷取與區塊鏈活動相關的內容資料。您可以使用 AMB Query 從公有區塊鏈網路讀取資料，例如 Bitcoin Mainnet 和 Ethereum Mainnet。您也可以取得資訊，例如地址的目前和歷史餘額，也可以取得指定期間內的區塊鏈交易清單。此外，您可以取得指定交易的詳細資訊，例如交易事件，您可以進一步分析或用於應用程式的商業邏輯。

使用 Amazon Managed Blockchain (AMB) 查詢的考量和限制

當您使用 AMB 查詢時，請考慮下列事項：

- 可用區域

  美國東部（維吉尼亞北部）`us-east-1`區域支援 AMB 查詢。

- 服務端點

  您可以使用下列端點存取 AMB 查詢：

  `https://managedblockchain-query.us-east-1.amazonaws.com`.

- 支援的區塊鏈網路

  AMB Query 支援下列公有區塊鏈網路：

  - Bitcoin Mainnet — 公有 Bitcoin 區塊鏈網路，受到proof-of-work共識保護，並在其中發行和交易 Bitcoin (BTC) 加密貨幣。Mainnet 上的交易具有實際值（即會產生實際成本），並記錄在公有區塊鏈上。
  - Bitcoin Testnet — Bitcoin Mainnet 的 testnet。此網路上的比特幣 (BTC) 與 Mainnet BTC 不同，通常沒有任何值。

- Ethereum Mainnet — 公有 Ethereum 區塊鏈的proof-of-stake主網路。Mainnet 上的交易具有實際值（即會產生實際成本），並記錄在分散式總帳上。
- Sepolia Testnet — Ethereum Mainnet 的測試網路。此網路上的 Ether (ETH) 與 Mainnet ETH 不同，且通常沒有任何值。

- 支援的區塊鏈字符和合約

  AMB Query 支援下列原生和標準 Ethereum 合約字符。

  - 公有區塊鏈原生字符
    - 比特幣 (BTC) — 這是比特幣相關區塊鏈的原生字符。
    - Ether (ETH) — 這是 Ethereum 相關區塊鏈的原生字符。

  - Ethereum 合約標準
    - ERC-20 權杖標準 — ERC-20 是易讀權杖的標準。它有一個 屬性，可讓每個 ERC-20 字符與另一個 ERC-20 字符完全相同（類型和值），這表示一個字符是且永遠等於所有其他字符。如需詳細資訊，請參閱 https：//Ethereum.org 上的 ERC-20 Token Standard。
    - ERC-721 不可分割權杖標準 — ERC-721 是不可分割權杖 (NFTs) 的標準。這種類型的字符是唯一的，可以具有與相同合約中另一個字符不同的值，可能是因為其存留期、罕見性或其他屬性。如需詳細資訊，請參閱 https：//Ethereum.org 上的 ERC-721 Token Standard。

      ERC-1155 Multi-token Standard — ERC-1155 是建立合約界面的標準，可代表和控制任意數量的易記和非易記字符類型。透過這種方式，ERC-1155 權杖可以與 ERC-20 和 ERC-721 權杖運作相同，甚至可以同時與兩者運作。ERC-1155 權杖可改善 ERC-20 和 ERC-721 標準的功能，使其更有效率，同時更正明顯的實作錯誤。如需詳細資訊，請參閱 https：//Ethereum.org 上的 ERC-1155 Token Standard。

- 最終性

  在區塊鏈中，最終性表示不太可能反轉有效交易。針對 Bitcoin Mainnet，AMB Query 會將交易視為 6 個區塊之後的最終交易。對於比特幣 Testnet，它會在 6 個區塊或 60 分鐘後將交易視為最終交易，以先到者為準。對於支援的 Ethereum 網路，AMB Query 會將交易視為 64 個區塊之後的最終交易。

  AMB Query 的權杖平衡和合約 API 操作只會傳回已達到最終結果的資料。不過，AMB Query 的交易和交易事件 API 操作可以傳回在區塊鏈網路上確認的交易資料，即使它們尚未達到最終。

- 不支援 NULL 地址

AMB 查詢不支援 NULL(0x0000000000000000000000000000000000000000) 地址。

- 簽章第 4 版簽署 API 呼叫

  呼叫 AMB 查詢 APIs 時，您可以透過使用 Signature 第 4 版簽署程序驗證的 HTTPS 連線來執行此操作。這表示只有帳戶中的授權 IAM AWS 主體才能進行 AMB Query API 呼叫。若要這樣做，必須使用 呼叫提供 AWS 憑證 （存取金鑰 ID 和私密存取金鑰）。

  > **Important**
  >
  > 請勿在面向使用者的應用程式中嵌入用戶端登入資料。

- AMB 查詢支援比特幣交易識別符和交易雜湊

  對於比特幣網路，AMB Query API 操作支援交易識別符 (`transactionId`) 和交易雜湊 ()`transactionHash`。`transactionId` 是交易的雙 SHA 雜湊，不包含見證資料。`transactionHash` 是交易的雙 SHA 雜湊，包括見證資料 （也稱為見證交易 ID)。

  叫用 Bitcoin 網路的 `GetTransaction`或 `ListTransactionEvents` API 操作時，您可以指定 `transactionId`或 `transactionHash`。此外，在 Bitcoin 網路上傳回 `transactionId`或 的所有 AMB 查詢操作`transactionHash`，都會包含這兩個值做為回應的一部分。

設定 Amazon Managed Blockchain (AMB) 查詢

首次使用 Amazon Managed Blockchain (AMB) 查詢之前，請遵循本節中的步驟來建立 AWS 帳戶。下一節討論如何開始使用 AMB Query。

先決條件和考量事項

首次使用 Amazon Web Services 之前，您必須擁有 AWS 帳戶。

註冊 AWS

當您註冊 Amazon Web Services (AWS) 時， AWS 您的帳戶會自動註冊所有 AWS 服務，包括 Amazon Managed Blockchain (AMB) 查詢。您只需針對所使用的服務付費。

如果您已有 AWS 帳戶 ，請前往下一個步驟。如果您還沒有 AWS 帳戶，請使用下列程序建立新帳戶。

建立 AWS 帳戶

1. 開啟 https://portal.aws.amazon.com/billing/signup。
2. 請遵循線上指示進行。

   部分註冊程序需接收來電或簡訊，並在電話鍵盤輸入驗證碼。

   當您註冊 時 AWS 帳戶，AWS 帳戶根使用者會建立 。根使用者有權存取該帳戶中的所有 AWS 服務 和資源。作為安全最佳實務，請將管理存取權指派給使用者，並且僅使用根使用者來執行需要根使用者存取權的任務。

建立具有適當許可的 IAM 使用者

若要建立和使用 AMB Query，您必須建立具有許可的 AWS Identity and Access Management (IAM) 委託人（使用者或群組），以允許必要的受管區塊鏈動作。

只有 IAM 主體可以提出 AMB Query API 請求。呼叫 AMB 查詢 APIs 時，您可以透過使用 Signature 第 4 版簽署程序驗證的 HTTPS 連線來執行此操作。這表示只有帳戶中的授權 IAM AWS 主體才能進行 AMB Query API 呼叫。若要這樣做，必須使用 呼叫提供 AWS 憑證 （存取金鑰 ID 和私密存取金鑰）。

如需如何建立 IAM 使用者的資訊，請參閱在 AWS 帳戶中建立 IAM 使用者。如需如何將許可政策連接至使用者的詳細資訊，請參閱變更 IAM 使用者的許可。如需可用於授予使用者使用 AMB 查詢許可的許可政策範例，請參閱 Amazon Managed Blockchain (AMB) 查詢的身分型政策範例。

安裝和設定 AWS Command Line Interface

如果您尚未這麼做，請安裝最新的 AWS Command-Line Interface (CLI) 以使用來自終端機 AWS 的資源。如需詳細資訊，請參閱安裝或更新最新版本的 AWS CLI。

> **Note**
>
> 對於 CLI 存取，您需要存取金鑰 ID 和私密存取金鑰。盡可能使用臨時憑證，而不是長期存取金鑰。臨時憑證包含存取金鑰 ID、私密存取金鑰，以及指出憑證何時到期的安全符記。如需詳細資訊，請參閱《IAM 使用者指南》中的將臨時登入資料與 AWS 資源搭配使用。

使用 AWS 管理主控台 查詢使用 Amazon Managed Blockchain (AMB) 查詢的區塊鏈

您可以使用 存取 Amazon Managed Blockchain (AMB) 查詢，並在支援的區塊鏈網路上進行查詢 AWS 管理主控台。下列步驟顯示如何執行此操作：

1. 開啟位於 https：//https://console.aws.amazon.com/managedblockchain/ 的 Amazon Managed Blockchain 主控台。
2. 從查詢區段中選擇查詢編輯器。
3. 從其中一個支援的區塊鏈網路中選擇。
4. 選擇您要執行的查詢類型。
5. 輸入您選取的查詢類型和執行查詢的相關參數。

AMB Query 將執行您的查詢，您會在查詢結果視窗中看到結果。

Amazon Managed Blockchain (AMB) 查詢入門

使用本節中的step-by-step教學課程，了解如何使用 Amazon Managed Blockchain (AMB) 查詢執行任務。這些程序需要一些先決條件。如果您是初次使用 AMB 查詢，您可以檢閱本指南的設定一節。如需詳細資訊，請參閱設定 Amazon Managed Blockchain (AMB) 查詢。

> **Note**
>
> 這些範例中的某些變數已刻意混淆。在執行這些範例之前，將它們替換為您自己的有效範例。

主題

建立 IAM 政策以存取 AMB Query API 操作

若要提出 AMB Query API 請求，您必須使用具有 Amazon Managed Blockchain (AWS_ACCESS_KEY_ID 和 AWS_SECRET_ACCESS_KEY)。在 AWS CLI 已安裝 的終端機中，執行下列命令來建立 IAM 政策以存取 AMB Query API 操作：

```
cat <<EOT > ~/amb-query-access-policy.json
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid" : "AMBQueryAccessPolicy",
            "Effect": "Allow",
            "Action": [
                "managedblockchain-query:*"
            ],
            "Resource": "*"
        }
```

```
    ]
}
EOT
aws iam create-policy --policy-name AmazonManagedBlockchainQueryAccess --policy-
document file://$HOME/amb-query-access-policy.json
```

建立政策後，將該政策連接至 IAM 使用者的角色，讓政策生效。在 中 AWS 管理主控台，導覽至 IAM 服務，並將政策連接至指派給將使用服務的 IAM 使用者AmazonManagedBlockchainQueryAccess的角色。如需詳細資訊，請參閱建立角色並指派給 IAM 使用者。

> **Note**
>
> AWS 建議您授予特定 API 操作的存取權，而不是使用萬用字元 *。如需詳細資訊，請參閱存取特定 Amazon Managed Blockchain (AMB) 查詢 API 動作。

使用 Go 提出 Amazon Managed Blockchain (AMB) 查詢 API 請求

透過 Amazon Managed Blockchain (AMB) 查詢，您可以在區塊鏈上確認區塊鏈資料後，建立依賴即時存取的應用程式，即使尚未達到最終版本。AMB Query 啟用數個使用案例，例如填入錢包的交易歷史記錄、根據交易雜湊提供交易的相關內容資訊，或取得原生字符以及 ERC-721, ERC-1155和 ERC-20 字符的平衡。

下列範例是以 Go 語言建立，並使用 AMB Query API 操作。如需 Go 的詳細資訊，請參閱 Go 文件。如需 AMB 查詢 API 的詳細資訊，請參閱 Amazon Managed Blockchain (AMB) 查詢 API 參考文件。

下列範例使用 ListTransactions和 GetTransaction API 動作，先取得 Ethereum Mainnet 上指定外部擁有地址 (EOA) 的所有交易清單，接著下一個範例會從清單中擷取單一交易的交易詳細資訊。

Example— 使用 Go 進行 ListTransactions API 動作

將下列程式碼複製到 ListTransactions listTransactions.go 目錄中名為 的檔案。

```
package main

import (
    "fmt"
    "github.com/aws/aws-sdk-go/aws"
    "github.com/aws/aws-sdk-go/aws/session"
    "github.com/aws/aws-sdk-go/service/managedblockchainquery"
```

```
    "time"
)

func main() {

    // Set up a session
    ambQuerySession := session.Must(session.NewSessionWithOptions(session.Options{
        Config: aws.Config{
            Region: aws.String("us-east-1"),
        },
    }))
    client := managedblockchainquery.New(ambQuerySession)

    // Inputs for ListTransactions API
    ownerAddress := "0x00000bf26964af9d7eed9e03e53415d********"
    network := managedblockchainquery.QueryNetworkEthereumMainnet
    sortOrder := managedblockchainquery.SortOrderAscending
    fromTime := time.Date(1971, 1, 1, 1, 1, 1, 1, time.UTC)
    toTime := time.Now()
    nonFinal := "NONFINAL"
    // Call ListTransactions API. Transactions that have reached finality are always
 returned
    listTransactionRequest, listTransactionResponse :=
 client.ListTransactionsRequest(&managedblockchainquery.ListTransactionsInput{
        Address: &ownerAddress,
        Network: &network,
        Sort: &managedblockchainquery.ListTransactionsSort{
            SortOrder: &sortOrder,
        },
        FromBlockchainInstant: &managedblockchainquery.BlockchainInstant{
            Time: &fromTime,
        },
        ToBlockchainInstant: &managedblockchainquery.BlockchainInstant{
            Time: &toTime,
        },

        ConfirmationStatusFilter: &managedblockchainquery.ConfirmationStatusFilter{
            Include: []*string{&nonFinal},
          },
    })
    errors := listTransactionRequest.Send()

    if errors == nil {
        // handle API response
```

```
        fmt.Println(listTransactionResponse)
    } else {
        // handle API errors
        fmt.Println(errors)
    }
}
```

儲存檔案之後，請在 ListTransactions 目錄中使用下列命令來執行程式碼：go run listTransactions.go。

後面的輸出類似以下內容：

```
{
  Transactions: [
    {
      ConfirmationStatus: "FINAL",
      Network: "ETHEREUM_MAINNET",
      TransactionHash:
 "0x12345ea404b45323c0cf458ac755ecc45985fbf2b18e2996af3c8e8693354321",
      TransactionTimestamp: 2020-06-01 01:59:11 +0000 UTC
    },
    {
      ConfirmationStatus: "FINAL",
      Network: "ETHEREUM_MAINNET",
      TransactionHash:
 "0x1234547c65675d867ebd2935bb7ebe0996e9ec8e432a579a4516c7113bf54321",
      TransactionTimestamp: 2021-09-01 20:06:59 +0000 UTC
    },
     {
      ConfirmationStatus: "NONFINAL",
      Network: "ETHEREUM_MAINNET",
      TransactionHash:
 "0x123459df7c1cd42336cd1c444cae0eb660ccf13ef3a159f05061232a24954321",
      TransactionTimestamp: 2024-01-23 17:10:11 +0000 UTC
    }
  ]
}
```

Example— 使用 Go 進行 GetTransaction API 動作

此範例使用上一個輸出的交易雜湊。將下列程式碼複製到 GetTransaction GetTransaction.go 目錄中名為 的檔案。

```
package main

import (
    "fmt"
    "github.com/aws/aws-sdk-go/aws"
    "github.com/aws/aws-sdk-go/aws/session"
    "github.com/aws/aws-sdk-go/service/managedblockchainquery"
)

func main() {

    // Set up a session
    ambQuerySession := session.Must(session.NewSessionWithOptions(session.Options{
        Config: aws.Config{
            Region: aws.String("us-east-1"),
        },
    }))
    client := managedblockchainquery.New(ambQuerySession)

    // inputs for GetTransaction API
    transactionHash :=
 "0x123452695a82868950d9db8f64dfb2f6f0ad79284a6c461d115ede8930754321"
    network := managedblockchainquery.QueryNetworkEthereumMainnet

    // Call GetTransaction API. This operation will return transaction details for all
    // transactions that are con#rmed on the blockchain, even if they have not
    // reached #nality.
    getTransactionRequest, getTransactionResponse :=
 client.GetTransactionRequest(&managedblockchainquery.GetTransactionInput{
        Network:         &network,
        TransactionHash: &transactionHash,
    })

    errors := getTransactionRequest.Send()
    if errors == nil {
        // handle API response
        fmt.Println(getTransactionResponse)
    } else {
        // handle API errors
        fmt.Println(errors)
    }
}
```

儲存檔案之後，請在 GetTransaction 目錄中使用下列命令來執行程式碼：`go run GetTransaction.go`。

後面的輸出類似以下內容：

```
{
  Transaction: {
    BlockHash: "0x000005c6a71d1afbc005a652b6ceca71cd516d97b0fc514c2a1d0f2ca3912345",
    BlockNumber: "11111111",
    CumulativeGasUsed: "5555555",
    EffectiveGasPrice: "44444444444",
    From: "0x9157f4de39ab4c657ad22b9f19997536********",
    GasUsed: "22222",
    Network: "ETHEREUM_MAINNET",
    NumberOfTransactions: 111,
    SignatureR: "0x99999894fd2df2d039b3555dab80df66753f84be475069dfaf6c6103********",
    SignatureS: "0x77777a101e7f37dd2dd0bf878b39080d5ecf3bf082c9bd4f40de783e********",
    SignatureV: 0,
    ConfirmationStatus: "FINAL",
    ExecutionStatus: "SUCCEEDED",
    To: "0x5555564f282bf135d62168c1e513280d********",
    TransactionHash:
 "0x123452695a82868950d9db8f64dfb2f6f0ad79284a6c461d115ede8930754321",
    TransactionIndex: 11,
    TransactionTimestamp: 2022-02-02 01:01:59 +0000 UTC
  }
}
```

GetTokenBalance API 可讓您取得原生字符 (ETH 和 BTC) 的餘額，可用於在某個時間點取得外部擁有帳戶 (EOA) 的目前餘額。

Example— 使用 GetTokenBalance API 動作來取得 Go 中原生權杖的平衡

在下列範例中，您可以使用 GetTokenBalance API 在 Ethereum Mainnet 上取得地址 Ether (ETH) 餘額。將下列程式碼複製到 GetTokenBalance `GetTokenBalanceEth.go` 目錄中名為 的檔案。

```
package main

import (
    "fmt"
    "github.com/aws/aws-sdk-go/aws"
    "github.com/aws/aws-sdk-go/aws/session"
```

```
    "github.com/aws/aws-sdk-go/service/managedblockchainquery"
)

func main() {
    // Set up a session
    ambQuerySession := session.Must(session.NewSessionWithOptions(session.Options{
        Config: aws.Config{
            Region: aws.String("us-east-1"),
        },
    }))
    client := managedblockchainquery.New(ambQuerySession)

    // inputs for GetTokenBalance API
    ownerAddress := "0xBeE510AF9804F3B459C0419826b6f225********"
    network := managedblockchainquery.QueryNetworkEthereumMainnet
    nativeTokenId  := "eth" //Ether on Ethereum mainnet

    // call GetTokenBalance API
    getTokenBalanceRequest, getTokenBalanceResponse :=
 client.GetTokenBalanceRequest(&managedblockchainquery.GetTokenBalanceInput{
        TokenIdentifier: &managedblockchainquery.TokenIdentifier{
            Network:         &network,
            TokenId: &nativeTokenId,
        },
        OwnerIdentifier: &managedblockchainquery.OwnerIdentifier{
            Address: &ownerAddress,
        },
    })
    errors := getTokenBalanceRequest.Send()

    if errors == nil {
        // process API response
        fmt.Println(getTokenBalanceResponse)
    } else {
        // process API errors
        fmt.Println(errors)
    }
}
```

儲存檔案之後，請在 GetTokenBalance 目錄中使用下列命令來執行程式碼：`go run GetTokenBalanceEth.go`。

後面的輸出類似以下內容：

```
{
  AtBlockchainInstant: {
    Time: 2020-12-05 11:51:01 +0000 UTC
  },
  Balance: "4343260710",
  LastTransactionHash:
 "0x00000ce94398e56641888f94a7d586d51664eb9271bf2b3c48297a50a0711111",
  LastTransactionTime: 2023-03-14 18:33:59 +0000 UTC,
  OwnerIdentifier: {
    Address: "0x12345d31750D727E6A3a7B534255BADd********"
  },
  TokenIdentifier: {
    Network: "ETHEREUM_MAINNET",
    TokenId: "eth"
  }
}
```

使用 Node.js 提出 Amazon Managed Blockchain (AMB) 查詢 API 請求

若要執行這些節點範例，適用下列先決條件：

1. 您必須在機器上安裝節點版本管理員 (nvm) 和 Node.js。您可以在此處找到作業系統的安裝指示。
2. 使用 `node --version`命令並確認您正在使用 Node 第 14 版或更新版本。如果需要，您可以使用 `nvm install 14`命令，接著使用 `nvm use 14`命令來安裝 第 14 版。
3. 環境變數 `AWS_ACCESS_KEY_ID`和 `AWS_SECRET_ACCESS_KEY`必須包含與帳戶相關聯的登入資料。

   使用下列命令，將這些變數匯出為用戶端上的字串。使用來自 IAM 使用者帳戶的適當值取代下列反白顯示的值。

   ```
   export AWS_ACCESS_KEY_ID="AKIAIOSFODNN7EXAMPLE"
   export AWS_SECRET_ACCESS_KEY="wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY"
   ```

> **Note**
>
> - 完成所有先決條件後，您可以透過 HTTPS 提交已簽署的請求，以存取 Amazon Managed Blockchain (AMB) 查詢 API 操作，並使用 Node.js 中的原生 https 模組提出請求，或者您可以使用第三方程式庫，例如 AXIOS 並從 AMB Query 擷取資料。
> - 這些範例使用 Node.js 的第三方 HTTP 用戶端，但您也可以使用 AWS JavaScript 開發套件向 AMB 查詢提出請求。
> - 下列範例說明如何使用 Axios 和 AWS SigV4 的 SDK 模組提出 AMB Query API 請求。

將下列package.json檔案複製到您本機環境的工作目錄：

```
{
  "name": "amb-query-examples",
  "version": "1.0.0",
  "description": "",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "",
  "license": "ISC",
  "dependencies": {
    "@aws-crypto/sha256-js": "^4.0.0",
    "@aws-sdk/credential-provider-node": "^3.360.0",
    "@aws-sdk/protocol-http": "^3.357.0",
    "@aws-sdk/signature-v4": "^3.357.0",
    "axios": "^1.4.0"
  }
}
```

Example— 使用 AMB Query GetTokenBalance API 從特定外部擁有的地址 (EOA) 擷取歷史字符餘額

您可以使用 GetTokenBalance API 來取得各種權杖（例如 ERC20, ERC721和 ERC1155) 和原生金幣（例如 ETH 和 BTC) 的平衡，您可以使用它根據歷史 timestamp(Unix 時間戳記 - 秒）來取得外部擁有帳戶 (EOA) 的目前平衡。在此範例中，您可以使用 GetTokenBalance API 在 Ethereum Mainnet 上取得 ERC20 權杖 USDC 的地址餘額。

若要測試 GetTokenBalance API，請將下列程式碼複製到名為 的檔案token-balance.js，並將檔案儲存至相同的工作目錄：

```
const axios = require('axios').default;
const SHA256 = require('@aws-crypto/sha256-js').Sha256
const defaultProvider = require('@aws-sdk/credential-provider-node').defaultProvider
const HttpRequest = require('@aws-sdk/protocol-http').HttpRequest
const SignatureV4 = require('@aws-sdk/signature-v4').SignatureV4

// define a signer object with AWS service name, credentials, and region
const signer = new SignatureV4({
  credentials: defaultProvider(),
  service: 'managedblockchain-query',
  region: 'us-east-1',
  sha256: SHA256,
});

const queryRequest = async (path, data) => {
  //query endpoint
  let queryEndpoint = `https://managedblockchain-query.us-east-1.amazonaws.com/
${path}`;

  // parse the URL into its component parts (e.g. host, path)
  const url = new URL(queryEndpoint);

  // create an HTTP Request object
  const req = new HttpRequest({
    hostname: url.hostname.toString(),
    path: url.pathname.toString(),
    body: JSON.stringify(data),
    method: 'POST',
    headers: {
      'Content-Type': 'application/json',
      'Accept-Encoding': 'gzip',
      host: url.hostname,
    }
  });


  // use AWS SignatureV4 utility to sign the request, extract headers and body
  const signedRequest = await signer.sign(req, { signingDate: new Date() });

  try {
    //make the request using axios
    const response = await axios({...signedRequest, url: queryEndpoint, data: data})
```

```
    console.log(response.data)
  } catch (error) {
    console.error('Something went wrong: ', error)
    throw error
  }


}


let methodArg = 'get-token-balance';

let dataArg = {
  " atBlockchainInstant": {
    "time": 1688071493
  },
  "ownerIdentifier": {
      "address": "0xf3B0073E3a7F747C7A38B36B805247B2********" // externally owned
 address
  },
  "tokenIdentifier": {
      "contractAddress":"0xA0b86991c6218b36c1d19D4a2e9Eb0cE********", //USDC contract
 address
      "network":"ETHEREUM_MAINNET"
  }
}

//Run the query request.
queryRequest(methodArg, dataArg);
```

若要執行程式碼，請在與 檔案相同的目錄中開啟終端機，然後執行下列命令：

```
npm i
node token-balance.js
```

此命令會執行指令碼，傳入程式碼中定義的引數，以請求 Ethereum Mainnet 上所列 EOA 的 ERC20 USDC 平衡。回應類似如下：

```
 {
  atBlockchainInstant: { time: 1688076218 },
  balance: '140386693440144',
  lastUpdatedTime: { time: 1688074727 },
  ownerIdentifier: { address: '0xf3b0073e3a7f747c7a38b36b805247b2********' },
```

```
  tokenIdentifier: {
    contractAddress: '0xa0b86991c6218b36c1d19d4a2e9eb0ce********',
    network: 'ETHEREUM_MAINNET'
  }
```

使用 Python 提出 Amazon Managed Blockchain (AMB) 查詢 API 請求

若要執行這些 Python 範例，適用下列先決條件：

1. 您必須在機器上安裝 Python。您可以在此處找到作業系統的安裝指示。
2. 安裝適用於 Python (Boto3) 的 AWS 開發套件。
3. 安裝 AWS Command Line Interface 並執行 命令`aws configure`來設定 `Access Key ID`、`Secret Access Key`和 的變數`Region`。

完成所有先決條件之後，您可以透過 HTTPS 使用適用於 Python 的 AWS SDK 提出 Amazon Managed Blockchain (AMB) 查詢 API 請求。

下列 Python 範例使用來自 boto3 的模組，將附加必要 SigV4 標頭的請求傳送至 AMB Query `ListTransactionEvents` API 操作。此範例會擷取 Ethereum Mainnet 上指定交易發出的事件清單。

將下列`list-transaction-events.py`檔案複製到您本機環境的工作目錄：

```
import json
from botocore.auth import SigV4Auth
from botocore.awsrequest import AWSRequest
from botocore.session import Session
from botocore.httpsession import URLLib3Session

def signed_request(url, method,  params, service, region):

    session = Session()
    sigv4 = SigV4Auth(session.get_credentials(), service, region)
    data = json.dumps(params)
    request = AWSRequest(method, url, data=data)
    sigv4.add_auth(request)
    http_session = URLLib3Session()
    response = http_session.send(request.prepare())
```

```
    return(response)

url = 'https://managedblockchain-query.us-east-1.amazonaws.com/list-transaction-events'
method = 'POST'
params = {
'network': 'ETHEREUM_MAINNET',
'transactionHash': '0x125714bb4db48757007fff2671b37637bbfd6d47b3a4757ebbd0c5222984f905'
}
service = 'managedblockchain-query'
region = 'us-east-1'

# Call the listTransactionEvents operation. This operation will return transaction
 details for
# all transactions that are con#rmed on the blockchain, even if they have not reached
# #nality.
listTransactionEvents = signed_request(url, method, params, service, region)

print(json.loads(listTransactionEvents.content.decode('utf-8')))
```

若要執行範本程式碼至 ListTransactionEvents，請將 檔案儲存在您的工作目錄中，然後執行命令 python3 list-transaction-events.py。此命令會執行指令碼，傳入程式碼中定義的引數，以請求與 Ethereum Mainnet 上指定交易雜湊相關聯的事件。回應類似如下：

```
{
 'events':
 [
  {
      'contractAddress': '0x95ad61b0a150d79219dcf64e1e6cc01f********',
      'eventType': 'ERC20_TRANSFER',
      'from': '0xab5801a7d398351b8be11c439e05c5b3********',
      'network': 'ETHEREUM_MAINNET',
      'to': '0xdead000000000000000420694206942********',
      'transactionHash':
 '0x125714bb4db48757007fff2671b37637bbfd6d47b3a4757ebbd0c522********',
      'value': '4102419967718718947718261747554 64'
  }
 ]
}
```

在 上使用 Amazon Managed Blockchain (AMB) 查詢 AWS 管理主控台 來執行 GetTokenBalance 操作

下列範例示範如何在 Ethereum Mainnet 上使用 上的 Amazon Managed Blockchain (AMB) 查詢取得權杖的餘額 AWS 管理主控台

Example

1. 開啟位於 https：//https://console.aws.amazon.com/managedblockchain/ 的 Amazon Managed Blockchain 主控台。
2. 從查詢區段中選擇查詢編輯器。
3. 選擇 ETHEREUM_MAINNET 做為區塊鏈網路。
4. 選擇 GetTokenBalance 做為查詢類型。
5. 輸入字符的區塊鏈地址。
6. 輸入字符的合約地址。
7. 輸入權杖的選用權杖 ID。
8. 選擇權杖餘額的日期。
9. 輸入權杖餘額的選用時間點。
10. 選擇 Run query (執行查詢)。

AMB Query 將執行您的查詢，您會在查詢結果視窗中看到結果。

Amazon Managed Blockchain (AMB) 查詢的使用案例

本主題提供清單 AMB 查詢使用案例。

主題

查詢目前和歷史權杖餘額

`GetTokenBalance` API 會使用外部擁有的帳戶 (EOAs) 的通用時間戳記 (Unix 時間戳記，以秒為單位），取得支援字符 (ERC20ERC721, ERC1155) 和原生金幣 (ETH、BTC) 的餘額，以取得目前或歷史餘額。例如，您可以使用 `GetTokenBalance` API 操作，在 Ethereum Mainnet 上取得 ERC20 權杖 USDC 的地址餘額。您也可以使用 `BatchGetTokenBalance` API 操作批次擷取權杖和原生金幣的平衡。

如需詳細資訊，請參閱《Amazon Managed Blockchain (AMB) 查詢參考指南》。

擷取歷史交易資料

使用 Amazon Managed Blockchain (AMB) 查詢，您可以從 Ethereum 和 Bitcoin 等公有區塊鏈擷取歷史資料。此功能可啟用數個使用案例，例如擷取區塊鏈錢包上的交易歷史記錄，或根據交易雜湊提供交易的相關內容資訊。您可以使用 `ListTransactions` API 操作在 Ethereum Mainnet 上取得指定外部擁有地址 (EOA) 的交易清單，然後使用 `GetTransaction` API 操作從清單中擷取單一交易的交易詳細資訊。

如需詳細資訊，請參閱《Amazon Managed Blockchain (AMB) 查詢參考指南》。

取得指定地址的所有字符餘額

您可以使用 `ListTokenBalances` API 操作來取得錢包、使用者介面、Web3 公用程式等項目的平衡。此 API 操作會使用單一 API 操作，傳回指定公有區塊鏈上跨字符 (ERC20、ERC721, ERC1155

和原生金幣 (ETH、BTC) 之地址的所有餘額清單。例如，您可以提供外部擁有的地址 (EOA) 和網路 (Ethereum Mainnet)，而且您可以在回應中收到字符和原生金幣餘額的清單。

如需詳細資訊，請參閱《Amazon Managed Blockchain (AMB) 查詢參考指南》。

列出為交易發出的事件

您可以使用 `ListTransactionEvents` API 操作來擷取因指定交易而發出的合約事件清單，並以其雜湊（交易識別符）識別。例如，您可以使用 `ListTransactionEvents`來擷取呼叫 Ethereum Blockchain 上 ERC20 字符合約函數的交易產生事件，例如 Transfer 事件或 ERC20 合約的撤回事件。

如需詳細資訊，請參閱《Amazon Managed Blockchain (AMB) 查詢參考指南》。

取得合約開採的所有字符

您可以使用 `ListTokenBalances` API 操作，傳回在傳遞合約地址做為輸入時，由合約開採的所有支援字符 (ERC20、ERC721, ERC1155清單。例如，您可以使用 `ListTokenBalances` API 操作，擷取由 Ethereum 區塊鏈上 ERC721 合約標準開採的非可信任權杖 (NFTs) 相關資訊。

如需詳細資訊，請參閱《Amazon Managed Blockchain (AMB) 查詢參考指南》。

列出合約並取得合約資訊

您可以使用 `ListAssetContracts` API 操作來列出由指定地址部署ERC-721, ERC-1155 或 ERC-20 合約。此外，如果您有合約地址，您可以使用 `GetAssetContract` API 操作來擷取合約的屬性，例如合約類型部署程式地址和相關的字符中繼資料。

如需詳細資訊，請參閱《Amazon Managed Blockchain (AMB) 查詢參考指南》。

Amazon Managed Blockchain (AMB) 查詢 API 參考

Amazon Managed Blockchain (AMB) Query 提供 API 操作來查詢支援的區塊鏈。這包括用於查詢字符、交易和合約APIs。如需詳細資訊，請參閱 AMB 查詢 API 參考 。

Amazon Managed Blockchain (AMB) 查詢中的安全性

的雲端安全 AWS 是最高優先順序。身為 AWS 客戶，您可以受益於資料中心和網路架構，這些架構專為滿足最安全敏感組織的需求而建置。

安全性是 AWS 與您之間的共同責任。共同責任模型將此描述為雲端安全性和雲端安全性：

- 雲端的安全性 – AWS 負責保護在 中執行 AWS 服務的基礎設施 AWS 雲端。 AWS 也為您提供可安全使用的服務。在 AWS 合規計劃中，第三方稽核員會定期測試並驗證我們的安全功效。若要了解適用於 Amazon Managed Blockchain (AMB) 查詢的合規計劃，請參閱AWS 合規計劃範圍內的服務。
- 雲端的安全性 – 您的責任取決於您使用 AWS 的服務。您也必須對其他因素負責，包括資料的機密性、您公司的要求和適用法律和法規。

為了提供資料保護、身分驗證和存取控制，Amazon Managed Blockchain 使用 AWS Managed Blockchain 中執行的開放原始碼架構的功能。

本文件可協助您了解如何在使用 AMB 查詢時套用共同責任模型。下列主題說明如何設定 AMB 查詢以符合您的安全與合規目標。您也可以了解如何使用其他 AWS 服務來協助您監控和保護 AMB Query 資源。

主題

資料加密

資料加密有助於防止未經授權的使用者從區塊鏈網路和相關聯的資料儲存系統讀取資料。這包括在網路移動時可能攔截的資料，稱為傳輸中的資料。

傳輸中加密

根據預設，受管區塊鏈會使用 HTTPS/TLS 連線來加密從 AWS CLI 用戶端傳輸到 AWS 服務端點的所有資料。

Amazon Managed Blockchain (AMB) 查詢的身分和存取管理

AWS Identity and Access Management (IAM) 是一種 AWS 服務 ，可協助管理員安全地控制對 AWS 資源的存取。IAM 管理員可控制誰可以進行身分驗證 （ 登入 ） 和授權 （ 具有許可 ），以使用 AMB 查詢資源。IAM 是您可以免費使用 AWS 服務 的 。

主題

目標對象

使用方式 AWS Identity and Access Management (IAM) 會根據您的角色而有所不同：

- 服務使用者 — 若無法存取某些功能，請向管理員申請所需許可 (請參閱 對 Amazon Managed Blockchain (AMB) 查詢身分和存取進行故障診斷)
- 服務管理員 — 負責設定使用者存取權並提交相關許可請求 (請參閱 Amazon Managed Blockchain (AMB) 查詢如何與 IAM 搭配使用)
- IAM 管理員 — 撰寫政策以管理存取控制 (請參閱 Amazon Managed Blockchain (AMB) 查詢的身分型政策範例)

使用身分驗證

身分驗證是您 AWS 使用身分憑證登入 的方式。您必須以 AWS 帳戶根使用者、IAM 使用者或擔任 IAM 角色身分進行身分驗證。

您可以使用身分來源的登入資料，例如 AWS IAM Identity Center (IAM Identity Center)、單一登入身分驗證或 Google/Facebook 登入資料，以聯合身分的形式登入。如需有關登入的詳細資訊，請參閱《AWS 登入 使用者指南》中的如何登入您的 AWS 帳戶。

對於程式設計存取， AWS 提供 SDK 和 CLI 以密碼編譯方式簽署請求。如需詳細資訊，請參閱《IAM 使用者指南》中的 API 請求的AWS 第 4 版簽署程序。

AWS 帳戶 根使用者

當您建立 時 AWS 帳戶，您會從一個名為 AWS 帳戶 theroot 使用者的登入身分開始，該身分可完整存取所有 AWS 服務 和資源。強烈建議不要使用根使用者來執行日常任務。有關需要根使用者憑證的任務，請參閱《IAM 使用者指南》中的需要根使用者憑證的任務。

聯合身分

最佳實務是要求人類使用者使用聯合身分提供者，以 AWS 服務 使用臨時憑證存取 。

聯合身分是您企業目錄、Web 身分提供者的使用者，或使用身分來源的 AWS 服務 憑證存取 Directory Service 。聯合身分會擔任角色，而該角色會提供臨時憑證。

若需集中化管理存取權限，建議使用 AWS IAM Identity Center。如需詳細資訊，請參閱 AWS IAM Identity Center 使用者指南中的什麼是 IAM Identity Center？。

IAM 使用者和群組

IAM 使用者https://docs.aws.amazon.com/IAM/latest/UserGuide/id_users.html是一種身分具備單人或應用程式的特定許可權。建議以臨時憑證取代具備長期憑證的 IAM 使用者。如需詳細資訊，請參閱《IAM 使用者指南》中的要求人類使用者使用聯合身分提供者來 AWS 使用臨時憑證存取 。

IAM 群組會指定 IAM 使用者集合，使管理大量使用者的許可權更加輕鬆。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 使用者的使用案例。

IAM 角色

IAM 角色https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles.html的身分具有特定許可權，其可以提供臨時憑證。您可以透過從使用者切換到 IAM 角色（主控台）或呼叫 AWS CLI 或 AWS API 操作來擔任角色。如需詳細資訊，請參閱《IAM 使用者指南》中的擔任角色的方法。

IAM 角色適用於聯合身分使用者存取、臨時 IAM 使用者許可、跨帳戶存取權與跨服務存取，以及在 Amazon EC2 執行的應用程式。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的快帳戶資源存取。

使用政策管理存取權

您可以透過建立政策並將其連接到身分或資源 AWS 來控制 AWS 中的存取。政策定義與身分或資源相關聯的許可。當委託人提出請求時 AWS ， 會評估這些政策。大多數政策會以 JSON 文件 AWS 的形式存放在 中。如需進一步了解 JSON 政策文件，請參閱《IAM 使用者指南》中的 JSON 政策概觀。

管理員會使用政策，透過定義哪些主體可在哪些條件下對哪些資源執行動作，以指定可存取的範圍。

預設情況下，使用者和角色沒有許可。IAM 管理員會建立 IAM 政策並將其新增至角色，供使用者後續擔任。IAM 政策定義動作的許可，無論採用何種方式執行。

身分型政策

身分型政策是附加至身分 (使用者、使用者群組或角色) 的 JSON 許可政策文件。這類政策控制身分可對哪些資源執行哪些動作，以及適用的條件。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的透過客戶管理政策定義自訂 IAM 許可。

身分型政策可分為內嵌政策 (直接內嵌於單一身分) 與受管政策 (可附加至多個身分的獨立政策)。如需了解如何在受管政策及內嵌政策之間做選擇，請參閱《IAM 使用者指南》中的在受管政策與內嵌政策之間選擇。

資源型政策

資源型政策是附加到資源的 JSON 政策文件。範例包括 IAM 角色信任政策與 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。您必須在資源型政策中指定主體。

資源型政策是位於該服務中的內嵌政策。您無法在資源型政策中使用來自 IAM 的 AWS 受管政策。

其他政策類型

AWS 支援其他政策類型，可設定更多常見政策類型授予的最大許可：

- 許可界限 — 設定身分型政策可授與 IAM 實體的最大許可。如需詳細資訊，請參閱《 IAM 使用者指南》中的 IAM 實體許可界限。
- 服務控制政策 (SCP) — 為 AWS Organizations中的組織或組織單位指定最大許可。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的服務控制政策。
- 資源控制政策 (RCP) — 設定您帳戶中資源可用許可的上限。如需詳細資訊，請參閱《AWS Organizations 使用者指南》中的資源控制政策 (RCP)。
- 工作階段政策 — 在以程式設計方式為角色或聯合身分使用者建立臨時工作階段時，以參數形式傳遞的進階政策。如需詳細資訊，請參《IAM 使用者指南》中的工作階段政策。

多種政策類型

當多種類型的政策適用於請求時，產生的許可會更複雜而無法理解。若要了解如何 AWS 決定是否在涉及多個政策類型時允許請求，請參閱《IAM 使用者指南》中的政策評估邏輯。

Amazon Managed Blockchain (AMB) 查詢如何與 IAM 搭配使用

在您使用 IAM 管理 AMB 查詢的存取權之前，請先了解哪些 IAM 功能可與 AMB 查詢搭配使用。

您可以搭配 Amazon Managed Blockchain (AMB) 查詢使用的 IAM 功能

IAM 功能	AMB 查詢支援
身分型政策	是
資源型政策	否
政策動作	是
政策資源	否
政策條件索引鍵	否
ACL	否
ABAC(政策中的標籤)	否
臨時憑證	是
主體許可	是
服務角色	否
服務連結角色	否

若要全面了解 AMB Query 和其他 AWS 服務如何與大多數 IAM 功能搭配使用，請參閱《IAM 使用者指南》中的與 IAM AWS 搭配使用的 服務。

AMB 查詢的身分型政策

支援身分型政策：是

身分型政策是可以附加到身分 (例如 IAM 使用者、使用者群組或角色) 的 JSON 許可政策文件。這些政策可控制身分在何種條件下能對哪些資源執行哪些動作。如需了解如何建立身分型政策，請參閱《IAM 使用者指南》中的透過客戶管理政策定義自訂 IAM 許可。

使用 IAM 身分型政策，您可以指定允許或拒絕的動作和資源，以及在何種條件下允許或拒絕動作。如要了解您在 JSON 政策中使用的所有元素，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素參考。

AMB 查詢的身分型政策範例

若要檢視 AMB 查詢身分型政策的範例，請參閱 Amazon Managed Blockchain (AMB) 查詢的身分型政策範例。

AMB 查詢中的資源型政策

支援資源型政策：否

資源型政策是附加到資源的 JSON 政策文件。資源型政策的最常見範例是 IAM 角色信任政策和 Amazon S3 儲存貯體政策。在支援資源型政策的服務中，服務管理員可以使用它們來控制對特定資源的存取權限。對於附加政策的資源，政策會定義指定的主體可以對該資源執行的動作以及在何種條件下執行的動作。您必須在資源型政策中指定主體。委託人可以包含帳戶、使用者、角色、聯合身分使用者或 AWS 服務。

如需啟用跨帳戶存取權，您可以在其他帳戶內指定所有帳戶或 IAM 實體作為資源型政策的主體。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的快帳戶資源存取。

AMB 查詢的政策動作

支援政策動作：是

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

JSON 政策的 `Action` 元素描述您可以用來允許或拒絕政策中存取的動作。政策會使用動作來授予執行相關聯動作的許可。

若要查看 AMB 查詢動作清單，請參閱服務授權參考中的 Amazon Managed Blockchain (AMB) 查詢定義的動作。

AMB Query 中的政策動作在動作之前使用以下字首：

```
managedblockchain-query:
```

若要在單一陳述式中指定多個動作，請用逗號分隔。

```
"Action": [
      "managedblockchain-query::ListTransaction",
      "managedblockchain-query::GetTransaction"
         ]
```

若要檢視 AMB 查詢身分型政策的範例，請參閱 Amazon Managed Blockchain (AMB) 查詢的身分型政策範例。

AMB 查詢的政策資源

支援政策資源：否

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

`Resource` JSON 政策元素可指定要套用動作的物件。最佳實務是使用其 Amazon Resource Name (ARN) 來指定資源。若動作不支援資源層級許可，使用萬用字元 (*) 表示該陳述式適用於所有資源。

```
"Resource": "*"
```

若要查看 AMB 查詢資源類型及其 ARNs，請參閱《服務授權參考》中的 Amazon Managed Blockchain (AMB) 查詢定義的資源。若要了解您可以使用哪些動作指定每個資源的 ARN，請參閱 Amazon Managed Blockchain (AMB) 定義的動作查詢 。

若要檢視 AMB 查詢身分型政策的範例，請參閱 Amazon Managed Blockchain (AMB) 查詢的身分型政策範例。

AMB 查詢的政策條件索引鍵

支援服務特定政策條件金鑰：否

管理員可以使用 AWS JSON 政策來指定誰可以存取內容。也就是說，哪個主體在什麼條件下可以對什麼資源執行哪些動作。

`Condition` 元素會根據定義的條件，指定陳述式的執行時機。您可以建立使用條件運算子的條件運算式 (例如等於或小於)，來比對政策中的條件和請求中的值。若要查看所有 AWS 全域條件索引鍵，請參閱《IAM 使用者指南》中的AWS 全域條件內容索引鍵。

若要查看 AMB 查詢條件索引鍵的清單，請參閱《服務授權參考》中的 Amazon Managed Blockchain (AMB) 查詢的條件索引鍵。若要了解您可以使用條件金鑰的動作和資源，請參閱 Amazon Managed Blockchain (AMB) 定義的動作查詢 。

若要檢視 AMB 查詢身分型政策的範例，請參閱 Amazon Managed Blockchain (AMB) 查詢的身分型政策範例。

AMB 查詢中的 ACLs

支援 ACL：否

存取控制清單 (ACL) 可控制哪些主體 (帳戶成員、使用者或角色) 擁有存取某資源的許可。ACL 類似於資源型政策，但它們不使用 JSON 政策文件格式。

ABAC 搭配 AMB 查詢

支援 ABAC (政策中的標籤)：否

屬性型存取控制 (ABAC) 是一種授權策略，依據稱為標籤的屬性來定義許可。您可以將標籤連接至 IAM 實體 AWS 和資源，然後設計 ABAC 政策，以便在委託人的標籤符合資源上的標籤時允許操作。

如需根據標籤控制存取，請使用 `aws:ResourceTag/key-name`、`aws:RequestTag/key-name` 或 `aws:TagKeys` 條件索引鍵，在政策的條件元素中，提供標籤資訊。

如果服務支援每個資源類型的全部三個條件金鑰，則對該服務而言，值為 Yes。如果服務僅支援某些資源類型的全部三個條件金鑰，則值為 Partial。

如需 ABAC 的詳細資訊，請參閱《IAM 使用者指南》中的使用 ABAC 授權定義許可。如要查看含有設定 ABAC 步驟的教學課程，請參閱《IAM 使用者指南》中的使用屬性型存取控制 (ABAC)。

搭配 AMB Query 使用臨時登入資料

支援臨時憑證：是

臨時登入資料提供 AWS 資源的短期存取權，當您使用聯合或切換角色時，會自動建立。 AWS 建議您動態產生臨時登入資料，而不是使用長期存取金鑰。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM 中的臨時安全憑證與可與 IAM 搭配運作的AWS 服務。

AMB 查詢的跨服務主體許可

支援轉寄存取工作階段 (FAS)：是

轉送存取工作階段 (FAS) 使用呼叫 的委託人許可 AWS 服務，並結合 AWS 服務 請求向下游服務提出請求。如需提出 FAS 請求時的政策詳細資訊，請參閱轉發存取工作階段。

AMB 查詢的服務角色

支援服務角色：否

服務角色是服務擔任的 IAM 角色，可代您執行動作。IAM 管理員可以從 IAM 內建立、修改和刪除服務角色。如需詳細資訊，請參閱《IAM 使用者指南》中的建立角色以委派許可給 AWS 服務。

> **Warning**
>
> 變更服務角色的許可可能會中斷 AMB 查詢功能。只有在 AMB Query 提供指引時，才能編輯服務角色。

AMB 查詢的服務連結角色

支援服務連結角色：否

服務連結角色是連結至 的一種服務角色 AWS 服務。服務可以擔任代表您執行動作的角色。服務連結角色會出現在您的 中 AWS 帳戶 ，並由服務擁有。IAM 管理員可以檢視，但不能編輯服務連結角色的許可。

如需建立或管理服務連結角色的詳細資訊，請參閱可搭配 IAM 運作的AWS 服務。在資料表中尋找服務，其中包含服務連結角色欄中的 Yes。選擇是連結，以檢視該服務的服務連結角色文件。

Amazon Managed Blockchain (AMB) 查詢的身分型政策範例

根據預設，使用者和角色沒有建立或修改 AMB 查詢資源的許可。若要授予使用者對其所需資源執行動作的許可，IAM 管理員可以建立 IAM 政策。

如需了解如何使用這些範例 JSON 政策文件建立 IAM 身分型政策，請參閱《IAM 使用者指南》中的建立 IAM 政策 (主控台)。

如需 AMB 查詢所定義之動作和資源類型的詳細資訊，包括每種資源類型的 ARNs 格式，請參閱《服務授權參考》中的 Amazon Managed Blockchain (AMB) 查詢的動作、資源和條件金鑰。

主題

- 政策最佳實務

- 允許使用者檢視他們自己的許可
- 存取特定 Amazon Managed Blockchain (AMB) 查詢 API 動作

政策最佳實務

身分型政策會判斷您帳戶中的某個人員是否可以建立、存取或刪除 AMB 查詢資源。這些動作可能會讓您的 AWS 帳戶產生費用。當您建立或編輯身分型政策時，請遵循下列準則及建議事項：

- 開始使用 AWS 受管政策並邁向最低權限許可 – 若要開始將許可授予您的使用者和工作負載，請使用將許可授予許多常見使用案例的 AWS 受管政策。它們可在您的 中使用 AWS 帳戶。我們建議您定義特定於使用案例 AWS 的客戶受管政策，以進一步減少許可。如需更多資訊，請參閱《IAM 使用者指南》中的 AWS 受管政策或任務職能的AWS 受管政策。
- 套用最低權限許可 – 設定 IAM 政策的許可時，請僅授予執行任務所需的許可。為實現此目的，您可以定義在特定條件下可以對特定資源採取的動作，這也稱為最低權限許可。如需使用 IAM 套用許可的更多相關資訊，請參閱《IAM 使用者指南》中的 IAM 中的政策和許可。
- 使用 IAM 政策中的條件進一步限制存取權 – 您可以將條件新增至政策，以限制動作和資源的存取。例如，您可以撰寫政策條件，指定必須使用 SSL 傳送所有請求。如果透過特定 例如 使用服務動作 AWS 服務，您也可以使用條件來授予其存取權 CloudFormation。如需詳細資訊，請參閱《IAM 使用者指南》中的 IAM JSON 政策元素：條件。
- 使用 IAM Access Analyzer 驗證 IAM 政策，確保許可安全且可正常運作 – IAM Access Analyzer 驗證新政策和現有政策，確保這些政策遵從 IAM 政策語言 (JSON) 和 IAM 最佳實務。IAM Access Analyzer 提供 100 多項政策檢查及切實可行的建議，可協助您撰寫安全且實用的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的使用 IAM Access Analyzer 驗證政策。
- 需要多重要素驗證 (MFA) – 如果您的案例需要 IAM 使用者或 中的根使用者 AWS 帳戶，請開啟 MFA 以提高安全性。如需在呼叫 API 操作時請求 MFA，請將 MFA 條件新增至您的政策。如需詳細資訊，請參閱《IAM 使用者指南》中的透過 MFA 的安全 API 存取。

如需 IAM 中最佳實務的相關資訊，請參閱《IAM 使用者指南》中的 IAM 安全最佳實務。

允許使用者檢視他們自己的許可

此範例會示範如何建立政策，允許 IAM 使用者檢視附加到他們使用者身分的內嵌及受管政策。此政策包含在主控台或使用 或 AWS CLI AWS API 以程式設計方式完成此動作的許可。

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

存取特定 Amazon Managed Blockchain (AMB) 查詢 API 動作

> **Note**
>
> 為了存取 AMB 查詢以進行 API 呼叫，您需要具有 AMB 查詢適當 IAM 許可的使用者登入資料 (AWS_ACCESS_KEY_ID 和 AWS_SECRET_ACCESS_KEY)。

Example存取所有 Amazon Managed Blockchain (AMB) 查詢 APIs IAM 政策

此範例會授予您 AWS 帳戶 存取所有 AMB Query APIs的 IAM 使用者。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessAllAMBQueryAPIs",
            "Effect": "Allow",
            "Action": [
                "managedblockchain-query:*"
            ],
            "Resource": "*"
        }
    ]
}
```

Example存取 Amazon Managed Blockchain (AMB) 查詢ListTransactions和 GetTransaction APIs IAM 政策

此範例授予您 AWS 帳戶 存取 AMB Query `ListTransaction`和 `GetTransaction` APIs的 IAM 使用者

> **Note**
>
> 您可以使用其他 APIs取代或新增範例中APIs，以授予其他或多個 APIs存取權。如需 AMB 查詢 APIs 的清單，請參閱《Amazon Managed Blockchain (AMB) 查詢 API 參考指南》。

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AccessAMBQueryAPIs",
            "Effect": "Allow",
            "Action": [
                "managedblockchain-query:ListTransactions",
                "managedblockchain-query:GetTransaction"
```

```
            ],
            "Resource": "*"
        }
    ]
}
```

對 Amazon Managed Blockchain (AMB) 查詢身分和存取進行故障診斷

使用以下資訊來協助您診斷和修正使用 AMB Query 和 IAM 時可能遇到的常見問題。

主題

- 我無權在 AMB 查詢中執行動作

我無權在 AMB 查詢中執行動作

如果您收到錯誤，告知您未獲授權執行動作，您的政策必須更新，允許您執行動作。

下列範例錯誤會在mateojackson IAM 使用者嘗試使用主控台檢視一個虛構 *my-example-widget* 資源的詳細資訊，但卻無虛構 managedblockchain-query::*GetWidget* 許可時發生。

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 managedblockchain-query::GetWidget on resource: my-example-widget
```

在此情況下，必須更新 mateojackson 使用者的政策，允許使用 managedblockchain-query::*GetWidget* 動作存取 *my-example-widget* 資源。

如果您需要協助，請聯絡您的 AWS 管理員。您的管理員提供您的簽署憑證。

Amazon CloudWatch 上的 Amazon Managed Blockchain (AMB) 查詢 API 用量指標

Amazon CloudWatch 上的 API 用量指標

發佈至 CloudWatch 的 API 用量指標對應至 Amazon Managed Blockchain (AMB) 查詢服務配額。您可以設定警示，在用量接近服務配額時提醒您。如需 CloudWatch 與服務配額整合的詳細資訊，請參閱《Amazon CloudWatch 使用者指南》中的 AWS 用量指標。

AMB Query 會使用`Amazon Managed Blockchain Query`服務名稱，在 `AWS/Usage` 命名空間中發佈下列 API 指標。

指標	Description
`CallCount`	AMB 查詢中對 API 進行的呼叫總數。SUM 代表指定期間內對 API 的呼叫總數。

Amazon Managed Blockchain (AMB) 查詢會以下列維度將用量指標發佈至`AWS/Usage`命名空間。

維度	描述
Service (服務)	包含 資源 AWS 的服務名稱。 `Amazon Managed Blockchain Query` 一律是此維度的值。
Type	要報告的實體類型。 `API` 一律是此維度的值。
資源	報告的資源類型。使用的 AMB Query API 操作名稱將是此維度的值。
類別	正在報告的資源類別。 `None` 一律是此維度的值。

AMB 查詢使用者指南的文件歷史記錄

下表說明 AMB 查詢的文件版本。

變更	描述	日期
AMB 查詢支援比特幣交易識別符和交易雜湊	對於比特幣網路，AMB Query API 操作支援交易識別符 (`transactionId`) 和交易雜湊 ()`transactionHash` 。	2024 年 3 月 21 日
支援 Amazon CloudWatch 上的 API 用量指標	AMB 查詢新增了對 CloudWatch 上 API 用量指標的支援。這些用量指標對應至 AMB Query 服務配額。	2024 年 2 月 8 日
支援尚未完成的交易	AMB 查詢已新增對尚未達到最終交易的支援。它也會從 `GetTransaction` 操作的回應中移除對 `status` 屬性的支援。反之，您將使用 `confirmationStatus` 和 `executionStatus` 屬性來判斷交易的狀態。	2024 年 2 月 1 日
交易資料類型中 `status` 屬性的棄用	Amazon Managed Blockchain (AMB) 查詢已棄用交易資料類型中的 `status` 屬性。您必須使用 `confirmationStatus` 和 `executionStatus` 欄位來判斷交易`status`的 是 `FINAL`還是 `FAILED`。	2023 年 12 月 20 日
支援 Sepolia Testnet	Amazon Managed Blockchain (AMB) 查詢現在支援 Ethereum Sepolia Testnet 上的查詢。	2023 年 10 月 19 日

支援資產合約	您可以使用 ListAssetContracts API 操作來列出由指定地址部署的 。此外，如果您有合約地址，您可以使用 GetAssetContract API 操作來擷取合約的詳細資訊。	2023 年 10 月 16 日
支援比特幣 Testnet	Amazon Managed Blockchain (AMB) Query 現在支援 Bitcoin Testnet 上的查詢。	2023 年 10 月 16 日
初始版本	AMB Query 服務的初始版本。	2023 年 7 月 27 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。