欢迎学习模块 8：确保用户和应用程序访问安全。

模块概览

章节

1. 架构需求
2. 账户用户和 IAM
3. 组织用户
4. 用户联合身份管理
5. 多个账户

演示

- EC2 实例配置文件

活动

- **检查** IAM **策略**

实验

- 挑战实验：使用 IAM 控制 AWS 账户访问


知识考核

本模块包含以下章节：
1. 架构需求
2. 账户用户和 IAM
3. 组织用户
4. 用户联合身份管理
5. 多个账户

本模块还包括：
- 一个演示，向您展示常用功能。将一个 IAM 角色（该角色授予访问 Amazon Web Services (AWS) 其他服务的权限）附加到 Amazon Elastic Compute Cloud (Amazon EC2) 实例
- 一项活动，要求您分析 AWS Identity and Access Management (IAM) 策略文档，以确定策略允许或拒绝哪些操作
- 一个挑战实验，您将在实验中使用 IAM 配置适合咖啡馆使用案例的用户、组和访问策略

最后，您需要完成一个知识考核，以测试您对本模块中涵盖的关键概念的理解程度。

模块目标

学完本模块后，您应该能够：

- 说明 AWS Identity and Access Management (IAM) 用户、组和角色的用途
- 描述如何允许在架构中使用用户联合以提高安全性
- 了解 AWS Organizations 服务控制策略 (SCP) 如何提高架构内的安全性
- 描述如何管理多个 AWS 账户
- 配置 IAM 用户

学完本模块后，您应该能够：
- 说明 AWS Identity and Access Management (IAM) 用户、组和角色的用途
- 描述如何允许在架构中使用用户联合以提高安全性
- 了解 AWS Organizations 服务控制策略 (SCP) 如何提高架构内的安全性
- 描述如何管理多个 AWS 账户
- 配置 IAM 用户


第 1 节：架构需求
模块 8：确保用户和应用程序访问安全
aws
© 2023，Amazon Web Services, Inc. 或其联属公司。保留所有权利。

介绍第 1 节：架构需求

咖啡馆业务需求

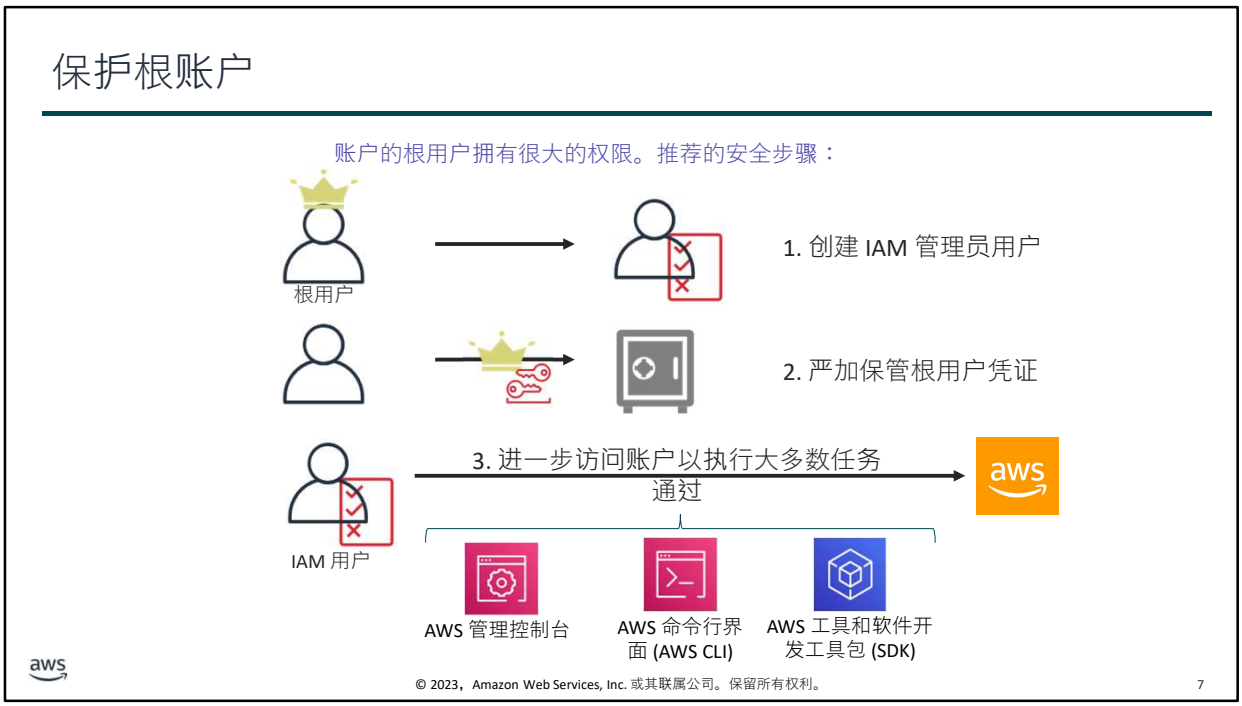
咖啡馆需要定义用户和系统在云资源中应具有的访问权限级别，然后在整个 AWS 账户中实施这些访问控制。

咖啡馆必须定义用户和系统在其云资源中应具有的访问权限级别。然后，他们必须在其整个 AWS 账户中实施这些访问控制。

现在，咖啡馆的规模扩张，有专门的团队成员负责在 AWS 上构建、维护或访问应用程序（例如开发人员或数据库管理员）。到目前为止，他们还没有开始根据每个用户的角色和责任明确定义他们应具有的访问权限级别。

在本模块中，您将了解 IAM，它可以提供您在满足这些新业务要求时所需的功能。

介绍第 2 节：账户用户和 IAM。



当您第一次创建 AWS 账户时，首先创建*根用户*。此用户可以使用用于创建账户的电子邮件地址登录 AWS 管理控制台。

AWS 账户根用户具有对账户中所有资源的完全访问权限，包括账单信息、用户资料中的个人数据以及在账户的任何 AWS 服务中创建的所有资源。您无法控制 AWS 账户根用户凭证的权限。

AWS 强烈建议您不要在与 AWS 的日常交互中使用根用户凭证，而是应创建一个或多个 IAM 用户。将根用户凭证保存在安全的位置。对于大多数正在进行的账户访问和管理任务，您可以使用 IAM 用户凭证。

AWS Identity and Access Management (IAM)

AWS Identity and Access Management (IAM)

- 安全地控制个人和组对 AWS 资源的访问权限
- 与其他 AWS 服务集成
- 联合身份管理
- 精细权限
- 支持多重身份验证

AWS Identity and Access Management 也称为 IAM。这项服务允许您配置对 AWS 资源的精细访问控制。IAM 允许您向用户和组授予唯一的安全凭证，从而实现安全最佳实践。这些凭证指定他们可以访问哪些 AWS 服务 Application Programming Interface (API) 和资源。IAM 默认已启用安全保护。用户无法访问 AWS 资源，除非被明确授予权限。

IAM 已集成到大多数 AWS 服务中。您可以在 AWS 管理控制台中集中定义访问控制，这些控制将在整个 AWS 环境中生效。

您可以使用现有身份系统通过 IAM 向您的员工和应用程序授予对 AWS 管理控制台和 AWS 服务 API 的访问权限。 AWS 支持从公司系统联合身份，如 Microsoft Active Directory 和标准的身份提供商。IAM 还支持多重身份验证 (MFA)。如果已启用 MFA 且 IAM 用户尝试登录，则系统将提示他们输入身份验证代码。身份验证代码将传送到 AWS MFA 设备。MFA 设备可以是硬件 MFA 设备，也可以是用户通过运行在用户智能手机上的应用程序访问的虚拟 MFA 设备，如 Google Authenticator。

您可以创建具有类似于 AWS 账户根用户的权限的账户。但是，最好创建管理账户，仅授予所需的账户权限。遵循最低权限原则。例如，思考您的数据库管理员 (DBA) 是否应该能够预置 EC2 实例。如果不能，则相应地预置账户。

IAM 组件：回顾

IAM 用户

在 AWS 账户中定义。使用凭证通过编程方式或 AWS 管理控制台进行身份验证。

IAM 组

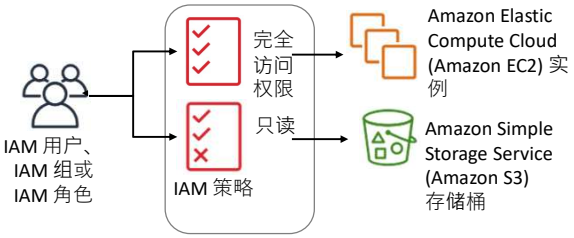
获得相同授权的 IAM 用户集合。

IAM 策略

定义可以访问哪些资源以及对每个资源的访问级别。

IAM 角色

授予临时访问权限以供发出 AWS 服务请求的机制。可由人员、应用程序或服务*代入*。



要了解如何使用 IAM 保护您的 AWS 账户，必须了解四个 IAM 组件的角色和功能。

*IAM 用户*是 AWS 账户中定义的人员或应用程序，它们必须对 AWS 产品进行 API 调用。每个用户在 AWS 账户内都必须具有唯一的名称（名称中不含空格）和一组不得与其他用户共享的安全凭证。这些凭证不同于 AWS 账户根用户的安全凭证。每个用户在一个且仅能在一个 AWS 账户中定义。

*IAM 组*是 IAM 用户的集合。您可使用 IAM 组简化为多个用户指定和管理权限的方式。

*IAM 策略*是一个定义权限的文档，可确定用户在 AWS 账户中可以执行和不可以执行的操作。

*IAM 角色*是一个工具，用于授予对 AWS 账户中特定 AWS 资源的临时访问权限。

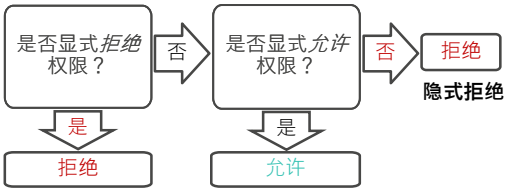
IAM 权限

IAM 策略

权限在 IAM 策略中指定：

- 文档采用 JavaScript 对象表示法 (JSON) 格式
- 它定义允许使用哪些资源和操作
- 最佳实践 – 遵循最低权限原则
- 两种策略类型 –
  - 基于身份：附加到 IAM 主体
  - 基于资源：附加到 AWS 资源

IAM 在收到请求时如何确定权限：



在 IAM 中，权限在 IAM 策略文档中定义。策略允许您对授予给主体的权限进行微调。示例主体包括 IAM 用户、IAM 角色或其他 AWS 服务。

在 IAM 确定是否允许权限时，它会先检查是否存在任何适用的*显式拒绝策略*。如果不存在显式拒绝策略，IAM 会继续检查是否存在任何适用的*显式允许策略*。如果不存在显式拒绝或显式允许策略，IAM 将恢复为默认设置并拒绝访问。这一过程称为*隐式拒绝*。仅当所请求的操作*不是*显式拒绝，而*是*显式允许时，才允许用户执行操作。

在制定 IAM 策略时，可能很难确定是否会向 IAM 实体授予对资源的访问权限。IAM 策略模拟器是一个非常有用的工具，可用于对 IAM 策略进行测试和故障排除。

策略存储为 JavaScript 对象表示法 (JSON) 文档。它们可以作为*基于身份的策略*附加到主体，也可以作为*基于资源的策略*附加到资源。

基于身份的策略与基于资源的策略

基于身份的策略

- 附加到用户、组或角色
- 策略类型
  - AWS 托管的策略
  - 客户托管的策略
  - 内联策略

基于资源的策略

- 附加到 AWS 资源
  - 示例：附加到 Amazon S3 存储桶
- 始终是内联策略

*基于身份的策略*是可以附加到主体（也称为“身份”，例如 IAM 用户、角色或组）的权限策略。这些策略可以*控制该身份在什么条件下可以针对哪些资源执行哪些操作*。

基于身份的策略可以进一步归类为 AWS 托管策略、客户托管策略或内联策略。*AWS 托管策略*由 AWS 创建和管理，可以附加到您的 AWS 账户中的多个用户、组和角色。如果您刚开始使用策略，建议您从使用 AWS 托管策略开始。*客户托管策略*是由您在 AWS 账户中创建和管理的策略。与 AWS 托管策略相比，客户托管策略可以更精确地控制您的策略。您可以通过可视化编辑器或直接创建 JSON 策略文档来创建和编辑 IAM 策略。*内联策略*是由您创建和管理的策略，直接嵌入在单个用户、组或角色中。

*基于资源的策略*是附加到资源（如 Amazon Simple Storage Service (Amazon S3) 存储桶）的 JSON 策略文档。这些策略可以*控制特定主体在什么条件下可以针对该资源执行哪些操作*。基于资源的策略是内联策略，没有基于资源的托管策略。

IAM 策略文档结构

```
{
   "Version": "2012-10-17",
   "Statement":[{
      "Effect": "effect",
      "Action": "action",
      "Resource": "arn",
      "Condition":{
         "condition":{
            "key": "value"
         }
      }
   }]
}
```

- Effect：效果可以是 *Allow*，也可以是 *Deny*
- Action：允许或拒绝的访问类型

```
"Action": "s3:GetObject"
```

- Resource：将对其执行操作的资源

```
"Resource": "arn:aws:sqs:us-west-2:123456789012:queue1"
```

- **Condition**：应用规则时必须满足的条件

```
"Condition" : {
  "StringEquals" : {
    "aws:username" : "johndoe"
  }
}
```

aws

IAM 策略在 AWS 中存储为 JSON 文档。基于身份的策略是附加到用户或角色的策略文档。基于资源的策略是附加到资源的策略文档。策略文档包括一个或多个单独语句。每个语句都包含有关单个权限的信息。如果策略包含多个语句，则 AWS 会在评估它们时跨语句应用逻辑“或”。

以下是 IAM 策略文档中的常见元素：

- **Version** – 指定要使用的策略语言的版本。作为最佳实践，请使用最新的 2012-10-17 版本。
- **Statement** – 使用此主要策略元素作为以下元素的容器。可以在一个策略中包含多个语句。
- **Effect** – 使用 Allow 或 Deny 来指示策略是允许还是拒绝访问。
- **Principal** – 如果创建基于资源的策略，您必须指示要允许或拒绝访问的账户、用户、角色或联合身份用户。如果您要创建 IAM 权限策略以附加到用户或角色，则不能包含此元素。主体隐式代表该用户或角色。
- **Action** – 包括策略允许或拒绝的操作列表。
- **Resource** – 如果创建 IAM 权限策略，您必须指定操作适用的资源列表。如果您创建基于资源的策略，则此元素是可选的。
- **Condition**（可选）– 指定策略在哪些情况下授予权限。

ARN 和通配符

- 使用 Amazon Resource Name (ARN) 格式标识资源
  - 语法 – `arn:partition:service:region:account:resource`
  - 示例 – `"Resource": "arn:aws:iam::123456789012:user/mmajor"`
- 您可以使用通配符 (*) 来授予针对特定 AWS 服务的所有操作
  - 示例 –
    - `"Action": "s3:*"`
    - `"Action": "iam:*AccessKey*"`

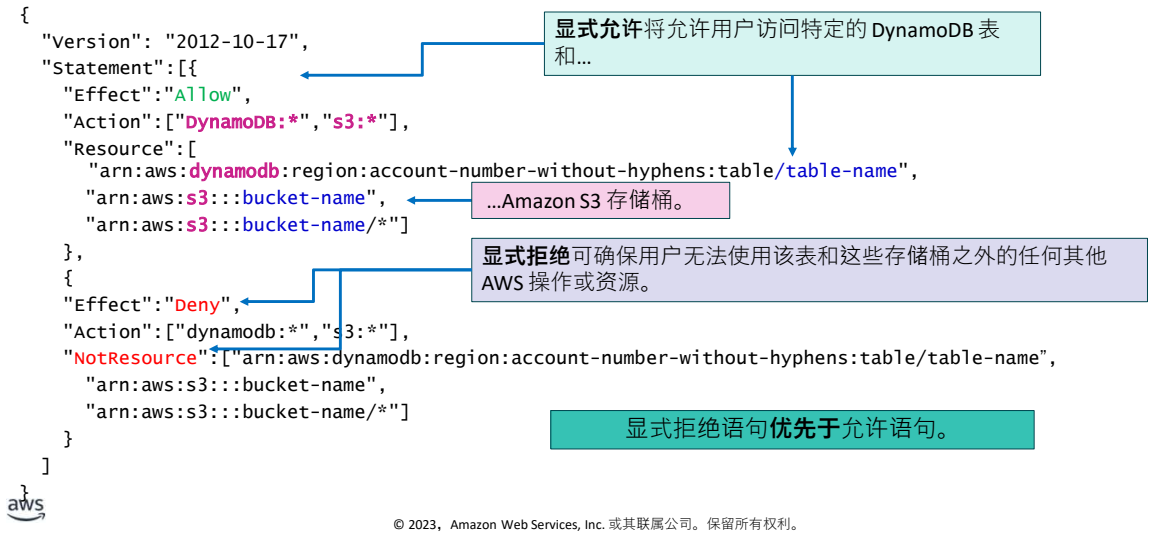
对于基于身份的（IAM 权限）策略，您必须指定操作适用的资源列表。*Resource* 元素指定语句涵盖的一个或多个对象。语句必须包含 Resource 或 NotResource 元素。

大多数资源都有易记名称（例如，名为 *Bob* 的用户或名为 *Developers* 的组）。不过，权限策略语言要求您使用以下 *Amazon Resource Name (ARN)* 格式指定资源。

每项服务都有自己的一套资源。虽然您总是使用 ARN 来指定资源，但资源的 ARN 细节取决于服务和资源。有关如何指定资源的信息，请根据编写的语句所涉及的资源，参阅相应资源所属服务的服务文档。

您还可以在 IAM 策略文档中使用通配符，例如在 ARN 或 Action 中。您可以使用通配符 (*)。星号 (*) 表示 0 个或多个字符的任意组合。例如，“*Action*”值“*s3:**”适用于所有 S3 操作。还可以使用通配符 (*) 作为操作名称的一部分。例如，“*Action*”值“*iam:*AccessKey**”适用于包含字符串 *AccessKey* 的所有 IAM 操作，包括 CreateAccessKey、DeleteAccessKey、ListAccessKeys、和 UpdateAccessKey。

IAM 策略示例



如前所述，IAM 策略文档以 JSON 格式编写。

此示例 IAM 策略将仅授予用户访问以下资源的权限：
- 名称用 *table-name* 表示的 Amazon DynamoDB 表。
- AWS 账户内名称用 *bucket-name* 表示的 S3 存储桶及其包含的所有对象。

该 IAM 策略还包含显式拒绝 ("Effect":"Deny") 元素。该 *NotResource* 元素有助于确保用户不能使用策略中指定的操作和资源以外的任何其他 DynamoDB 或 S3 操作或资源，即使在其他策略中已授予相关权限也不例外。显式拒绝语句优先于允许语句。

活动：检查 IAM 策略

照片由 Pixabay 提供（摘自 Pexels）。

在此讲师指导的活动中，会为您提供示例 IAM 策略。对于每项策略，您需要回答有关该策略是允许还是拒绝特定操作的问题。讲师引导对每个问题进行讨论，一次揭晓一个正确答案。

活动：IAM 策略分析 (1/3)

考虑此 IAM 策略，然后回答问题。

```
{
   "Version": "2012-10-17",
   "Statement": {
      "Effect": "Allow",
      "Action": [
         "iam:Get*",
         "iam:List*"
      ],
      "Resource": "*"
   }
}
```

1. 此策略授予您对哪种 AWS 服务的访问权限？
2. 它是否允许您创建 IAM 用户、组、策略或角色？
3. 转到 https://docs.aws.amazon.com/IAM/latest/UserGuide/，然后在左侧导航栏中展开 *Reference （参考）> Policy Reference （策略参考）> Actions, Resources, and Condition Keys （操作、资源和条件键）*。选择 *Identity and Access Management*。滚动到 Actions Defined by Identity And Access Management （Identity And Access Management 定义的操作）列表。

   至少指出 iam:Get* 操作允许的三个特定操作。

查看 IAM 策略文档示例。讲师现在将向您提出一系列问题，来评估您是否理解此策略将允许和拒绝哪些操作。

活动：IAM 策略分析 (1/3) – 答案

考虑此 IAM 策略，然后回答问题。

```
{
   "Version": "2012-10-17",
   "Statement": {
      "Effect": "Allow",
      "Action": [
         "iam:Get*",
         "iam:List*"
      ],
      "Resource": "*"
   }
}
```

1. 此策略授予您对哪种 AWS 服务的访问权限？
   - 答案：IAM 服务。
2. 它是否允许您创建 IAM 用户、组、策略或角色？
   - 答案：否。访问权限仅限于 *get* 和 *list* 请求。它实际上授予的是只读权限。
3. 转到 https://docs.aws.amazon.com/IAM/latest/UserGuide/，然后在左侧导航栏中展开 *Reference （参考）> Policy Reference （策略参考）> Actions, Resources, and Condition Keys （操作、资源和条件键）*。选择 *Identity and Access Management*。滚动到 Actions Defined by Identity And Access Management （Identity And Access Management 定义的操作）列表。

   至少指出 iam:Get* 操作允许的三个特定操作。
   - 答案：iam:Get* 允许许多特定的操作，包括 GetGroup、GetPolicy、GetRole 等。

答案揭晓。

活动：IAM 策略分析 (2/3)

考虑此 IAM 策略，然后回答问题。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
            },
            "Resource": ["*"]
        }
    ]
}
```

1. 该策略是否允许您随时无条件地终止任何 EC2 实例？
2. 您是否有权从任何地方进行终止实例调用？
3. 如果您从分配的 IP 地址为 *192.0.2.243* 的服务器进行调用，您能否终止实例？

分析第二个 IAM 策略文件示例。第一部分显示对资源的操作 ec2:TerminateInstance 实施“允许”效果。第二部分显示在 NotIpAddress aws:SourceIp 192.0.2.0/24 和 203.0.113.0/24 条件下，对资源的操作 ec2:TerminateInstances 实施“拒绝”效果。讲师现在将再次向您提出一系列问题，来评估您是否理解此策略将允许和拒绝哪些操作。

活动：IAM 策略分析 (2/3) – 答案

考虑此 IAM 策略，然后在显示问题时作答。

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ec2:TerminateInstances"],
            "Resource": ["*"]
        },
        {
            "Effect": "Deny",
            "Action": ["ec2:TerminateInstances"],
            "Condition": {
                "NotIpAddress": {
                    "aws:SourceIp": [
                        "192.0.2.0/24",
                        "203.0.113.0/24"
                    ]
                }
            },
            "Resource": ["*"]
        }
    ]
}
```

1. 该策略是否允许您随时无条件地终止任何 EC2 实例？
   - 答案：否。第一个语句对象允许。但是，第二个语句对象应用了一个条件。
2. 您是否有权从任何地方进行终止实例调用？
   - 答案：否。您只能从 *aws:SourceIp* 中指定的两个 IP 地址范围之一发出请求。
3. 如果您从分配的 IP 地址为 *192.0.2.243* 的服务器进行调用，您能否终止实例？
   - 答案：能，因为 192.0.2.0/24 无类域间路由 (CIDR) IP 地址范围包括从 192.0.2.0 到 192.0.2.255 的 IP 地址。CIDR 到 IP 范围工具等资源可用于计算 CIDR 块的范围。

aws

答案揭晓。

内容说明：JSON 格式的策略文档示例。显示一个包含两个部分的语句区域。第一部分显示对全部资源的操作 EC2:TerminateInstance 实施“允许”效果。第二部分显示在 NotIpAddress aws:SourceIp 192.0.2.0/24 和 203.0.113.0/24 条件下，对全部资源的操作 EC2:TerminateInstances 实施“拒绝”效果。**内容说明结束。**

活动：IAM 策略分析 (3/3)

考虑此 IAM 策略，然后回答问题。

```
{
  "Version": "2012-10-17",
  "Statement":[{
      "Condition": {
        "StringNotEquals": {
           "ec2:InstanceType": [
              "t2.micro",
              "t2.small“
           ]
        }
      },
      "Resource": "arn:aws:ec2:*:*:instance/*"
      "Action": [
         "ec2:RunInstances",
         "ec2:StartInstances“
      ],
     "Effect": "Deny“
   }
]
```

1. 该策略允许哪些操作？
2. 假设该策略包含一个额外的语句对象，如本例所示：

```
{
    "Effect": "Allow",
    "Action": "ec2:*",
    "Resource": "*"
}
```

   该策略将如何限制此额外语句授予您的访问权限？
3. 如果该策略同时包含左侧的语句和问题 2 中的语句，您是否能终止账户中存在的 m3.xlarge 实例？

观察第三个、也是最后一个 IAM 策略文档示例。讲师现在将再次向您提出一系列问题，来评估您是否理解此策略将允许和拒绝哪些操作。

活动：IAM 策略分析 (3/3)

考虑此 IAM 策略，然后回答问题。

```
{
  "Version": "2012-10-17",
  "Statement":[{
      "Condition": {
        "StringNotEquals": {
           "ec2:InstanceType": [
              "t2.micro",
              "t2.small"
           ]
        }
      },
      "Resource": "arn:aws:ec2:*:*:instance/*",
      "Action": [
         "ec2:RunInstances",
         "ec2:StartInstances"
      ],
     "Effect": "Deny"
   }
]
```

1. 该策略允许哪些操作？
   - 答案：它不允许您执行任何操作（效果是*拒绝*）。
2. 假设该策略包含一个额外的语句对象，如本例所示：

   ```
   {
        "Effect": "Allow",
        "Action": "ec2:*",
        "Resource": "*"
   }
   ```

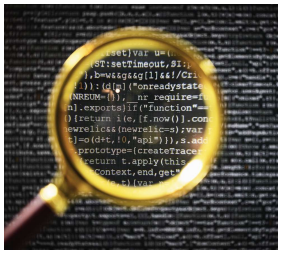
   该策略将如何限制此额外语句授予您的访问权限？
   - 答案：您将具有对 Amazon EC2 服务的完全访问权限。但是，您只能启动实例类型为 t2.micro 或 t2.small 的 EC2 实例。
3. 如果该策略同时包含左侧的语句和问题 2 中的语句，您是否能终止账户中存在的 m3.xlarge 实例？
   - 答案：是。

答案揭晓。

AWS CloudTrail

AWS CloudTrail

- 记录和监控用户活动
- 提供 AWS 账户的事件历史记录
  - 通过 AWS 管理控制台、 SDK、 AWS CLI 进行的操作
  - 提高用户和资源活动的可见性
  - 默认情况下免费提供 90 天的事件历史记录
- 可以确定以下信息
  - *谁*访问过您的账户
  - *何时*以及从*何处*访问的
  - 他们对 AWS 服务执行了*哪些*操作
- 用于以下用途的有用工具
  - 执行安全分析
  - 发现哪些调用被阻止（例如，被 IAM 策略阻止）



AWS CloudTrail 是一项可实现您的 AWS 账户的监管、合规性检查和审计的服务。借助 CloudTrail，您可以持续监控并保留与 AWS 基础设施中的操作相关的账户活动。它可提供账户活动的事件历史记录，包括通过 AWS 管理控制台、AWS SDK 和命令行工具执行的操作。事件历史记录可以简化安全性分析、资源更改跟踪和故障排除工作。

您可以通过捕捉特定时段内在您的 AWS 账户中所发生更改的全面历史记录，发现并解决安全性和运行问题。您可以识别调用 AWS 的用户和账户、发起调用的源 IP 地址以及执行调用的时间。借助 CloudTrail，您能够跟踪并自动应对威胁 AWS 资源安全性的账户活动。

通过与 Amazon EventBridge（以前称为 Amazon CloudWatch Events）的集成，您可以定义在检测到可能导致安全漏洞的事件时要运行的工作流。例如，您可以创建一个工作流，以在 CloudTrail 记录将导致某个 S3 存储桶被公开的 API 调用时将特定策略添加到该存储桶。

CloudTrail 可记录每个操作的重要信息，包括请求的发出方、使用的服务、执行的操作、操作的参数，以及 AWS 服务返回的响应元素。该服务还有助于企业满足必须遵守的合规性和审计要求。

第 2 节要点

- 避免使用账户根用户执行常见任务。相反，请创建和使用 IAM 用户凭证。
- 用于访问 AWS 账户资源的权限在一个或多个 IAM 策略文档中定义。
  - 将 IAM 策略附加到 IAM 用户、组或角色。
- 当 IAM 确定权限时，显式拒绝将始终覆盖任何允许语句。
- 在授予访问权限时，遵循最低权限原则是最佳实践。

本模块中这节内容的要点包括：

- 避免使用账户根用户执行常见任务。相反，请创建和使用 IAM 用户凭证。
- 用于访问 AWS 账户资源的权限在一个或多个 IAM 策略文档中定义。
  - 将 IAM 策略附加到 IAM 用户、组或角色。
- 当 IAM 确定权限时，显式拒绝将始终覆盖任何允许语句。
- 在授予访问权限时，遵循最低权限原则是最佳实践。


第 3 节：组织用户
模块 8：确保用户和应用程序访问安全
aws
© 2023，Amazon Web Services, Inc. 或其联属公司。保留所有权利。

介绍第 3 节：组织用户。

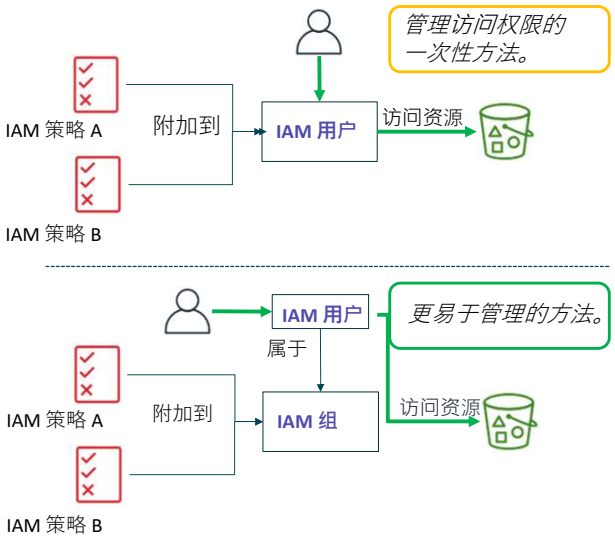
IAM 组

使用 IAM 组向多个用户授予相同的访问权限。

- 组中的所有用户将继承分配到该组的权限
  - 使多个用户的访问权限管理变得更加轻松

提示：结合多种方法可以精细地管理个人访问权限

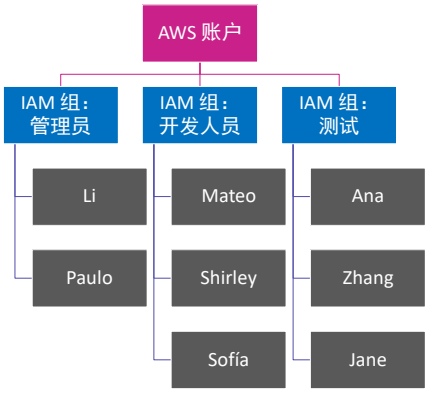
- 将用户添加到组，以根据工作职能应用标准访问权限
- 可选择将额外策略附加到需要例外情况的用户



IAM 组是 IAM 用户的集合。组是一种便利的方法，能让您更加轻松地管理用户集合的权限，而不必分别管理每个用户的权限。

将组成员作为一个简单列表来进行管理：
- 向组添加用户或将用户从组中删除。
- 一个用户可以属于多个组。
- 一个组无法从属于其他组。
- 可以通过使用访问控制策略授予组权限。
- 组没有安全凭证，也无法直接访问 Web 服务，它们存在的目的只是为了更轻松地管理用户权限。

IAM 组示例



提示：创建反映工作职能的组

- 在雇用新的开发人员之后，将他们添加到*开发人员*组
  - 立即继承已授予其他开发人员的相同访问权限
- 如果 Ana 担任新的开发人员角色 –
  - 将她从*测试*组中移除
  - 将她添加到*开发人员*组
- 用户可以属于多个组
  - 但是，将应用最严格的策略

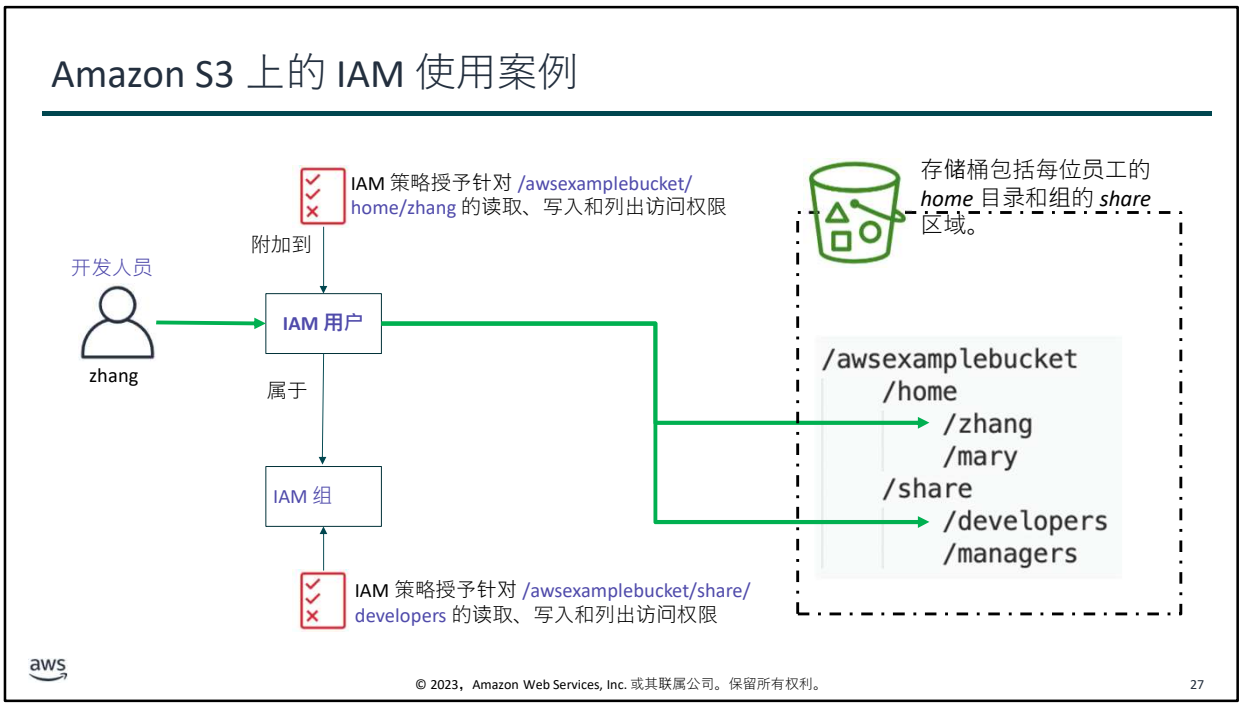
通常，您需要创建反映工作职能的组。例如，您可以为管理员创建一个组，为开发人员创建另一个组，为执行测试功能的团队再创建一个组。

然后，您可以将一个或多个策略文件附加到每个组，然后向组中添加用户。凭借其组成员资格，用户会拥有分配给他们所在的一个或多个组的访问权限。

在雇用新的开发人员之后，您可以将他们添加到现有开发人员组。他们将获得与其他开发人员相同的访问权限。

如果某个人员，例如 Ana（如示例中所示）在企业中担任了一个新的角色，则可以将其从*测试*组中移除，然后将其添加到*开发人员*组中。或者，如果 Ana 将执行这两项职能，您可以将她留在*测试*组中，然后将她添加到*开发人员*组中。

如果您发现开发人员需要访问账户中的其他资源，则可以更新*开发人员*组的策略或添加策略。该组的所有成员都将获得这种额外的访问权限级别。通过组功能，可以更加轻松地在团队之间维护一致的访问权限。



此示例演示如何在 S3 存储桶上配置 IAM 权限。

awsexamplebucket 有两个主要目录。*home* 目录为每个用户提供子目录，让他们可以在其中存储个人工作成果。在 *share* 目录包含的子目录中，不同团队都可以在其中存储内容。

如果新的团队成员 *zhang* 以开发人员身份加入企业，您可以采取三项操作来向其授予适当的访问权限。

首先，将 *zhang* 添加到面向开发人员的 IAM 组中。请注意，该组附加有一个 IAM 策略，用于授予针对 */awsexamplebucket/share/developers* 的访问权限。

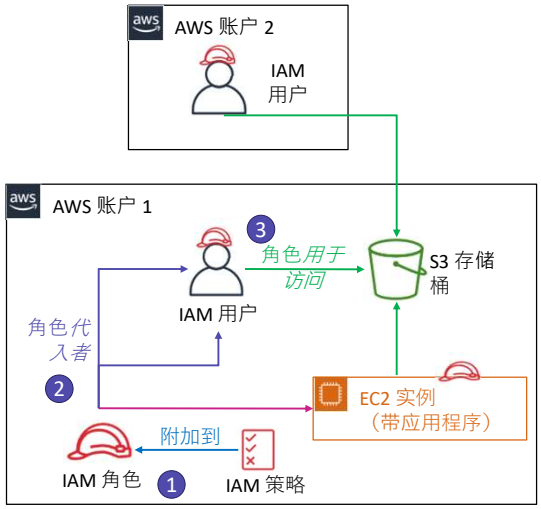
接下来，在 Amazon S3 中创建 */awsexamplebucket/home/zhang* 目录。

最后，附加一个 IAM 策略，用于将对 */awsexamplebucket/home/zhang* 目录的访问权限直接授予 *zhang* IAM 用户。Zhang 的访问权限将包括从该组授予的权限以及直接附加到 IAM 用户主体的权限。

介绍第 4 节：用户联合身份管理。

IAM 角色

- IAM 角色特性
  - 提供*临时*安全凭证
  - 不是唯一地与一个人相关联
  - 可由人员、应用程序或服务*代入*
  - 通常用于委派访问权限
- 使用案例
  - 为 AWS 资源提供对 AWS 服务的访问权限
  - 为经过外部身份验证的用户提供访问权限
  - 为第三方提供访问权限
  - 切换角色以访问以下位置的资源 –
    - 您的 AWS 账户
    - 任何其他 AWS 账户（跨账户访问）



IAM 角色使您能够定义一组权限，以便访问用户或服务所需的资源。但是，权限并不附加到 IAM 用户或组。而是附加到角色，而角色由用户或服务代入。

当用户代入角色时，系统会暂时忘记用户之前所具有的权限。AWS 返回临时安全凭证，随后用户或应用程序可以使用这些凭证以编程方式对 AWS 发出请求。

通过使用 IAM 角色，您不必与每个需要访问资源的实体共享长期安全凭证（例如创建 IAM 用户）。

对于像 Amazon EC2 这样的服务，应用程序或 AWS 服务可以在运行时以编程方式代入角色。

代入该角色的主体也可以是来自其他 AWS 账户的 IAM 用户、组或角色，包括不归您所有的账户。

通过创建用于外部账户访问的角色，您无需为第三方管理用户名和密码。如果您不再希望某人或某个系统具有访问权限，可以修改或删除该角色。因此，您不需要为企业之外的人员创建和管理账户。

演示：EC2 实例配置文件

现在，讲师可能会选择演示如何将 IAM 角色附加到 EC2 实例。此角色将 AWS 资源访问权限授予应用程序。

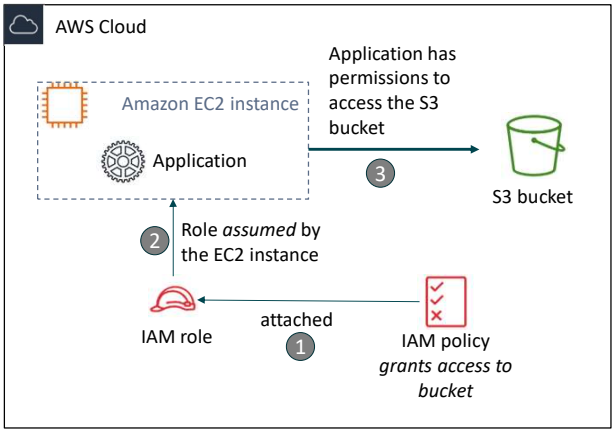
Summary: EC2 instance profile demonstration

Scenario:

- An application that runs on an EC2 instance needs access to an S3 bucket

Solution:

- Define an IAM policy that grants access to the S3 bucket
- Attach the policy to a role
- Allow the EC2 instance to assume the role



This diagram illustrates the educator-led demonstration.

- An application runs on an EC2 instance, and that application needs access to the S3 bucket.
- An administrator creates an *IAM role*.
- Then, they create an *IAM policy* that grants read-only access to the specified S3 bucket. The policy also includes a trust policy that allows the EC2 instance to assume the role and retrieve the temporary credentials.
- Next, they attach the IAM policy to the role.

When the application runs on the instance, it can assume the role and use the role's temporary credentials to access the bucket.

With this architecture, the administrator does not need to directly grant the application developer permission to access the bucket, and the developer never needs to share or manage credentials.

授予代入角色的权限

AWS Security Token Service (AWS STS)

- 要使 IAM 用户、应用程序或服务代入某个角色，您必须授予切换到该角色的权限
- AWS Security Token Service (AWS STS)
  - 允许您请求临时、有限权限凭证的 Web 服务
  - 凭证可供 IAM 用户使用，也可供您验证的用户（联合身份用户）使用
- 示例策略 – 允许 IAM 用户代入某个角色

```
{
   "Version": "2012-10-17",
   "Statement": {
  "Effect": "Allow",
  "Action": "sts:AssumeRole",
  "Resource": "arn:aws:iam::123456789012:role/Test*"
   }
}
```


AWS Security Token Service 也称为 *AWS STS*。它是一项 Web 服务，使 IAM 用户、联合身份用户或应用程序能够代入他们需要的 IAM 角色。

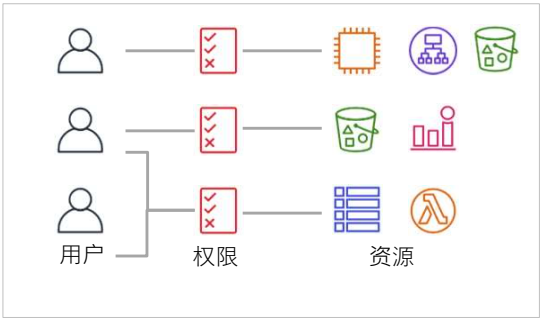
成功调用 AWS STS API 的 AssumeRole 操作后，Web 服务将返回 IAM 用户或经过联合身份验证的用户所请求的临时受限权限凭证。通常，AssumeRole 操作用于跨账户访问或联合身份访问。

本示例策略允许 IAM 用户代入在 AWS 账号 123456789012 中定义的任何角色，只要角色名称以 *Test* 开头。

基于*角色*的访问控制 (RBAC)

传统的访问控制方法：

- 根据工作职能授予用户特定权限（例如数据库管理员）
- 为每个权限组合创建不同的 IAM 角色
- 通过为每个新资源添加访问权限来更新权限（持续更新策略可能会非常耗时）



现在，您要考虑两种不同的访问控制方法：基于角色的访问控制 (RBAC) 和基于属性的访问控制 (ABAC)。您首先将了解 RBAC。

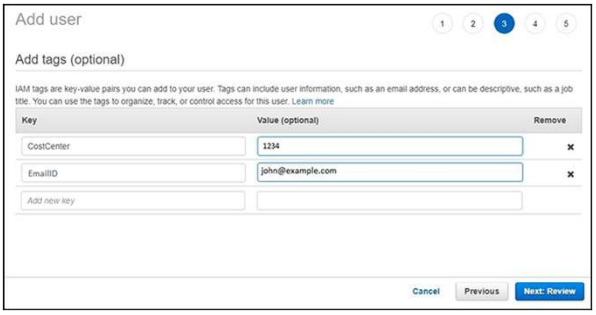
RBAC 一直在本地部署和云中使用。使用这个模型，您可以向用户授予对一组权限的显式访问权限。假设您拥有数据库管理员、网络管理员和开发人员。如果您的一个或多个网络管理员也是开发人员，则您不会创建新策略来授予这些权限。您会将这些用户添加到两个角色中。

这种方法很常见，而且具有许多优点。但是，在此模型中，维护权限的人员可能会发现，每次创建新资源时，他们必须不断更新权限文件来添加对特定角色的访问权限。例如，每当有人创建新资源并希望允许用户访问该资源时，他们都必须使用 ARN 更新策略。

最佳实践：标记

- 标签由名称和（可选）值组成
  - 可应用于您的 AWS 账户中的资源
  - 很多不同的 API 操作返回标签键及值
- 定义*自定义*标签
- 多种实际用途
  - 账单、筛选视图、访问控制等
- 应用于 EC2 实例的示例标签：
  - Name = web server
  - Project = unicorn
  - Stack = dev

- 标签还可以应用于 IAM 用户或 IAM 角色，例如 –



在考虑第二种权限控制方法之前，您应该了解 AWS 中的标记功能。

AWS 使客户能够以*标签*的形式将元数据分配给其 AWS 资源和身份。每个标签都是一个简单的标记，由一个客户定义的键和一个可选值组成。利用标签可以更轻松地管理、搜索和筛选资源。

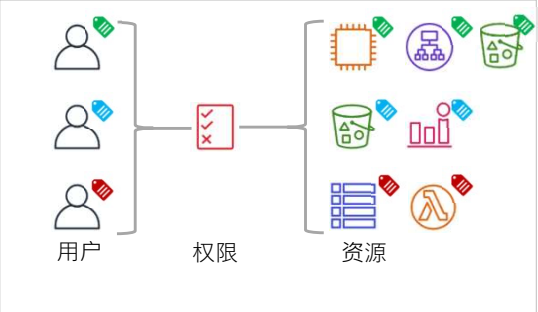
标签有许多实际用途。例如，您可以创建*技术标签*，来标识某项资源是 Web 服务器、是特定项目的一部分、是特定环境（测试、开发或生产）的一部分等。您还可以创建*业务标签*，用来标识应为此资源或此资源所属项目付费的部门或成本中心。最后，您还可以设置*安全标签*，例如用于资源支持的特定数据机密级别的标识符。

您最多可以为每个资源创建 个标签。对于每个资源，每个标签键都必须是唯一的，每个标签键只能有一个值。标签键和值区分大小写。

您还可以向 IAM 用户和 IAM 角色添加标签。标签是您接下来将学习的第二种访问控制方法的一个重要组成部分。

基于*属性*的访问控制 (ABAC)

- 高度可扩展的访问控制方法
  - 属性是键或键值对，例如标签
  - 示例属性 –
    - Team = Developers
    - Project = Unicorn
- 使用 ABAC 的权限（策略）规则比使用 RBAC 的权限（策略）规则更容易维护
- 益处
  - 根据属性自动应用权限
  - *无需*对每个新用户或资源进行权限更新，即可实现精细权限
  - 完全可审计



现在，您已经了解了标记功能，您将了解第二种访问控制方法：基于属性的访问控制 (ABAC)。

通过 ABAC，您可以使用属性创建随企业规模而扩展的常规权限规则。

在此模型中，IAM 用户具有您所创建和应用的属性，例如一个或多个标签。

资源还具有同样应用于资源的属性，例如匹配标签。

通过 RBAC 方法，编写权限变得相对简单。策略用来检查应用于 IAM 用户的属性是否也被应用于他们要访问的资源。在创建新的 IAM 用户和新的账户资源时，您要将正确的标签应用于用户和资源。

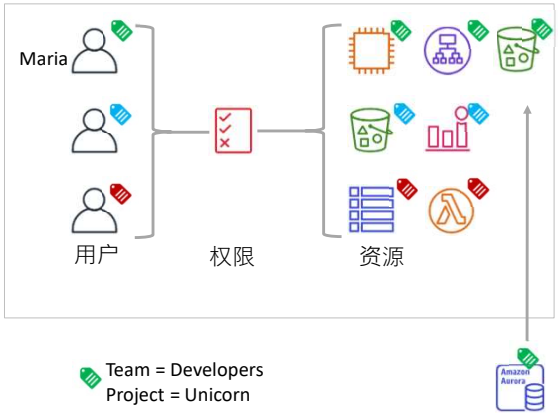
通过 ABAC 方法，您可以向开发人员授予对其项目资源的访问权限，但无需在策略文件中指定资源。

您可以想象一下 ABAC 访问管理方法的可扩展性如何。您无需修改权限设置。使用正确的标签创建资源或用户时，权限将自动应用。

将 ABAC 应用于您的企业

如何将 ABAC 应用于您的企业：

1. 设置身份的访问控制属性
2. 需要新资源的属性
3. 根据属性配置权限
4. 测试
   a) 创建新资源
   b) 验证权限是否自动应用



要将 ABAC 应用于您的企业，第一步是创建身份，例如 IAM 用户或 IAM 角色。这些身份必须具有将用于访问控制目的的属性。例如，您可以将 *Team = Developers* 和 *Project = Unicorn* 标签应用于 *Maria* 用户。

接下来，需要为新资源提供属性。您应该创建强制实施规则的策略。例如，您可以要求在创建任何资源时将 *Project* 属性和 *Team* 属性应用于资源。

第三步，根据属性配置访问权限。例如，假设 IAM 用户具有 *Project = Unicorn* 和 *Team = Developers* 标签。如果该用户尝试访问在这两个相同标签上具有匹配值的资源，该策略将允许访问。否则，该策略将拒绝访问。

第四步，测试配置。例如，您可以尝试创建 Amazon Aurora 数据库实例但不要包含必需的标签。创建操作应该会失败。再次尝试创建数据库实例但包含必需的标签。这次，您应该能够成功创建资源。最后，您可以尝试以 *Maria* 用户身份访问数据库实例。您应该可以成功访问。但是，如果您尝试以没有匹配标签的其他用户身份访问数据库实例，则您的访问应该会遭到拒绝。

外部验证的用户

联合身份

- 由 AWS 账户外的系统完成的用户身份验证
  - 示例：公司目录
- 这种方法允许通过现有身份进行访问而无需创建 IAM 用户

联合身份选项

1. AWS STS
   - 公有身份服务提供商 (IdP)
   - 自定义身份代理应用程序
2. 安全断言标记语言 (SAML)
3. Amazon Cognito

IdP 身份验证概览

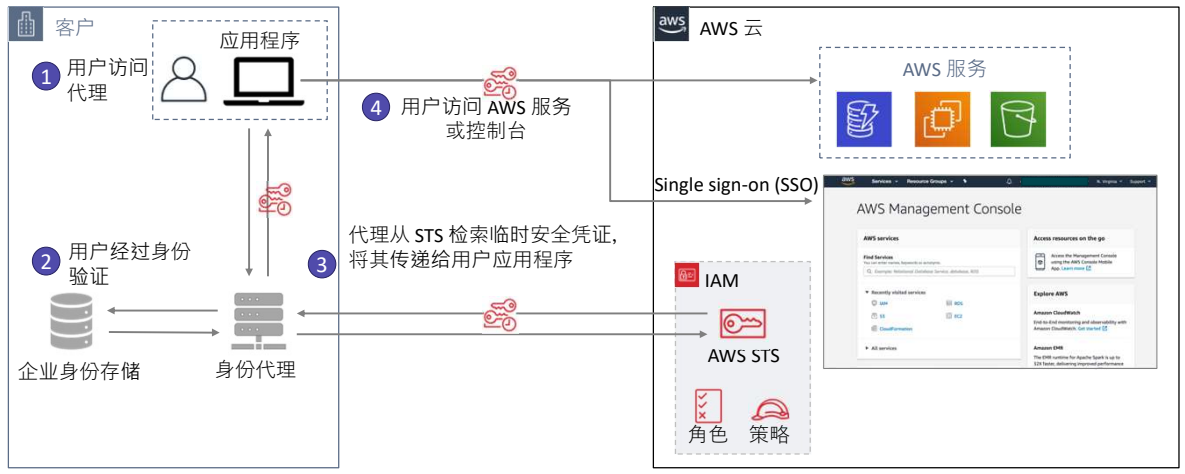
现在，您将了解一个新的主题：外部验证的用户。

IAM 支持联合身份，用于实现对 AWS 管理控制台或 AWS API 的委托访问。通过联合身份，外部身份可以安全访问您 AWS 账户中的资源，而*无需*创建 IAM 用户。

该图显示了使用*身份提供商 (IdP)* 为用户或应用程序创建临时凭证时发生的四个主要步骤。

联合身份可以通过三种方法来实现。第一种方法是使用公司 IdP（例如 Microsoft Active Directory）或自定义身份代理应用程序。每个选项都使用 AWS STS。第二种方法是创建使用安全断言标记语言 (SAML) 的集成。第三种方法是使用 Web 身份提供商，例如 Amazon Cognito。接下来几张幻灯片将讨论这三种方法。

使用身份代理的联合身份

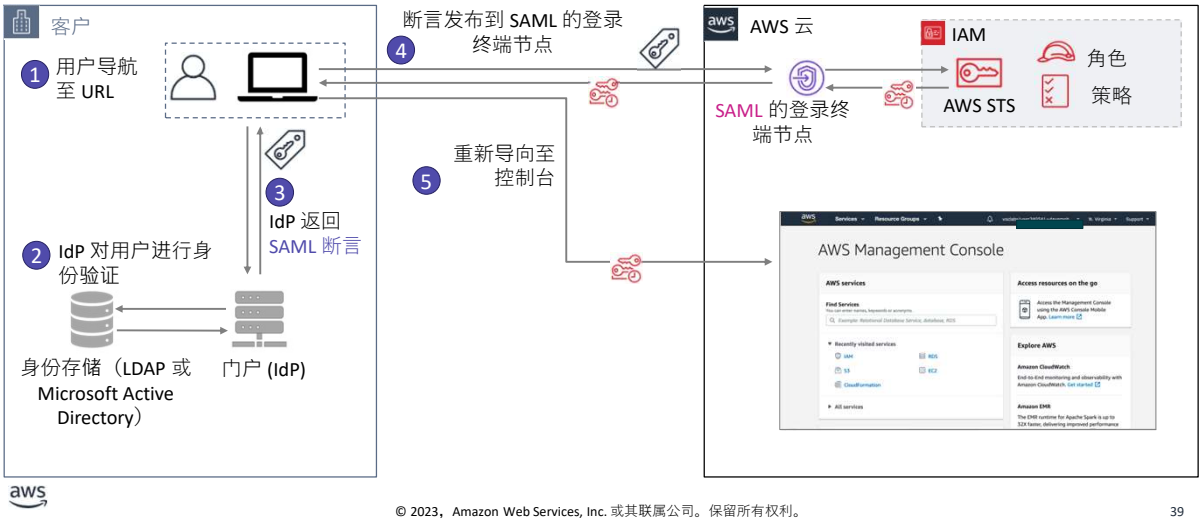


现在，您将学习如何使用身份代理来完成联合身份。

此过程包括以下步骤：

1. 用户访问应用程序。用户输入其用户 ID 和密码并提交
2. 身份代理接收身份验证请求。然后，它与公司身份存储（可能是 Microsoft Active Directory 或轻量级目录访问协议 (LDAP) 服务器）进行通信。
3. 如果身份验证请求成功，身份代理将向 AWS STS 发出请求。请求的内容是为用户应用程序检索临时 AWS 安全凭证。
4. 用户应用程序接收临时 AWS 安全凭证，并将用户重新导向到 AWS 管理控制台。用户无需使用另一组凭证集即可直接登录 AWS。此过程是 Single-Sign On (SSO) 实现的一个示例。如果 IAM 策略文档允许，用户应用程序还可以使用这些临时 AWS 安全凭证访问 AWS 服务。

使用 SAML 的联合身份



现在，您将了解实现联合身份验证的第二个选项。此方法使用 *SAML* 开放标准在 IdP 和服务提供商之间交换身份验证和授权数据。

此过程包括以下步骤：

1. 您企业内的用户导航至网络中的内部门户。该门户也充当 IdP，处理您的企业与 AWS 之间的 SAML 信任。
2. IdP 根据身份存储（可能是 LDAP 服务器或 Microsoft Active Directory）对用户的身份进行身份验证。
3. 门户从 IdP 接收身份验证响应作为 *SAML 断言*。
4. 客户端将 SAML 断言发布到 SAML 的 AWS 登录终端节点。该终端节点与 AWS STS 通信，并且它调用 AssumeRoleWithSAML 操作来请求临时安全凭证并构建登录 URL。
5. 客户端收到临时 AWS 安全凭证。客户端被重新导向到 AWS 管理控制台，并使用临时 AWS 安全凭证进行身份验证。

Amazon Cognito

Amazon Cognito

Amazon Cognito 是一项完全托管式的服务。

- 它为 Web 和移动应用程序提供身份验证、授权和用户管理
- Amazon Cognito 提供 Web 联合身份
  - 它们可以用作身份代理，支持与 OpenID Connect (OIDC) 兼容的 IdP
- 联合身份
  - 用户使用社交身份提供商（Amazon、Facebook、Google）或 SAML 登录
- 用户池
  - 您可以使用用户配置文件身份验证令牌维护目录

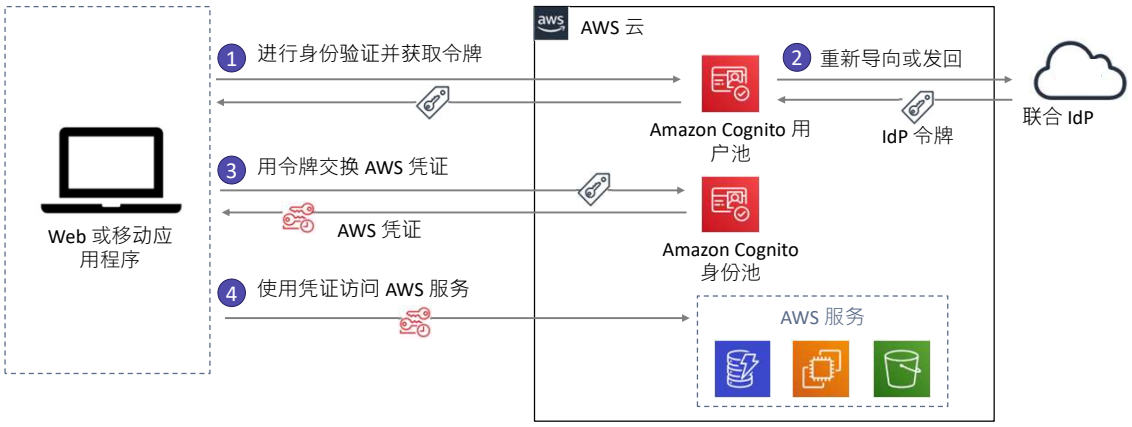
第三个也是最后一个联合身份验证选项是使用 Amazon Cognito。*Amazon Cognito* 是一项为 Web 和移动应用程序提供身份验证、授权和用户管理的完全托管式服务。用户可以使用用户名和密码直接登录，或通过 Facebook、Amazon 或 Google 等第三方登录。

Amazon Cognito 的两个主要组件是*用户池*和*身份池*。

*用户池*是 Amazon Cognito 中的用户目录。利用用户池，用户可以通过 Amazon Cognito 登录 Web 或移动应用程序。他们还可以通过第三方 IdP 进行联合。用户池的所有成员都有一个可通过 SDK 访问的目录配置文件。

借助*身份池*，您可以为用户创建唯一身份并分配权限。借助身份池，用户可以获取临时 AWS 凭证来访问 AWS 服务或资源。身份池可以通过 Facebook、Google、Login with Amazon 以及 OpenID Connect (OIDC) 提供商与 Amazon Cognito 用户池的社交登录进行通信。

Amazon Cognito 示例



在这种情况下，目标是使用 Amazon Cognito 对用户进行身份验证，然后向该用户授予其他 AWS 服务的访问权限。

- 在第 1 步中，应用程序用户通过 Amazon Cognito 用户池登录，在成功通过身份验证后，收到用户池令牌。
- 接下来，该应用程序通过 Amazon Cognito 身份池使用用户池令牌换取 AWS 凭证。
- 最后，应用程序用户使用这些 AWS 凭证访问其他 AWS 服务。

第 4 节要点

- IAM 角色提供可由人员、应用程序或服务代入的临时安全凭证
- 您可以通过 AWS Security Token Service (AWS STS) 请求临时 AWS 凭证
- 在联合身份中，用户身份验证是在 AWS 账户外部进行的
  - 通过使用 AWS STS、SAML 或 Amazon Cognito 完成

本模块中这节内容的要点包括：

- IAM 角色提供可由人员、应用程序或服务代入的临时安全凭证。
- 您可以通过 AWS Security Token Service (STS) 请求临时 AWS 凭证。
- 在联合身份验证中，用户身份验证是在 AWS 账户外部发生的。
  - 使用 STS、SAML 或 Amazon Cognito 完成。


第 5 节：多个账户
模块 8：确保用户和应用程序访问安全
aws
© 2023，Amazon Web Services, Inc. 或其联属公司。保留所有权利。

介绍第 5 节：多个账户。

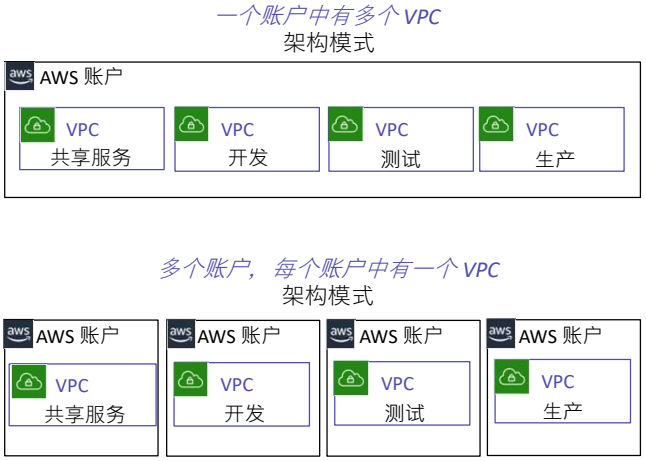
一个账户还是多个账户？

两种架构模式

- 大多数企业选择创建多个账户

多个账户的优势

- 隔离业务单元或部门
- 隔离开发、测试和生产环境
- 隔离审计数据、恢复数据
- 针对受监管的工作负载使用单独的账户
- 更容易针对每个业务单元的消耗触发成本警报



当您使用 AWS 支持企业中的不同团队和部门时，您可以在两种常规架构模式之间进行选择，以隔离和分隔每个团队使用的资源。

第一种模式是在单个 AWS 账户中定义多个 Virtual Private Cloud (VPC)。如果您倾向于开支最小的集中式信息安全管理方式，则可以选择使用单个 AWS 账户。

第二种模式是创建多个 AWS 账户并在每个账户中定义 VPC。实际上，大型和小型企业都倾向于为其企业创建多个账户。例如，他们可能会为各个业务单元创建各自的账户。他们还可以为其开发、测试和生产资源创建单独的账户。

当客户为开发和生产资源使用不同的 AWS 账户时（通常采用整合账单方式），可以明确区分不同类型的资源。同时也可提供一些安全优势。

或者，如果贵公司针对生产、开发和测试分别创建了不同的环境，那么您可以配置三个 AWS 账户，使每个环境都拥有独立账户。此外，如果您有多个自主部门，也可以为企业的每个自主部门创建不同的 AWS 账户。

当您使用多个账户时，更有效的策略是为通用项目资源创建一个 AWS 账户。通用资源可能包括域名系统 (DNS) 服务、Microsoft Active Directory 和内容管理系统 (CMS)。您还可以为自主项目或部门创建单独的账户。这种策略使您可以在每个部门或项目账户下分配权限和策略，并跨账户授予资源访问权限。

管理多个账户遇到的挑战

- 跨账户的安全管理
  - IAM 策略复制
- 创建新账户
  - 涉及许多手动流程
- 账单整合
- 需要集中监管以确保一致性

尽管大多数组织选择使用多个 AWS 账户，但这种选择会带来一些挑战。

首先，您必须确定如何有效管理所有账户的安全性。如果您复制所有账户中定义的 IAM 策略来确保一致性，则可能涉及自定义自动化、手动操作，或这两者都要涉及。

此外，您可能会不断地被要求创建更多账户。手动创建这些账户需要一些时间。也可能难以跟踪所有账户和每个账户的目的。

确定企业中的哪个成本中心应该为哪个账户中的哪些资源付费也是一项挑战。最后，您可能还希望实现确保一致性所需的集中管理。

使用 AWS Organizations 管理多个账户

跨多个 AWS 账户集中管理和强制实施策略

- 基于组的账户管理
- 对 AWS 服务的基于策略的访问
- 账户自动创建和管理
- 整合账单
- 基于 API

AWS 提供了一项服务，旨在解决这些管理难题。

AWS Organizations 是一项用于账户管理的托管服务。组织是您创建的用于整合、集中查看和管理所有 AWS 账户的实体。您可以通过启用的功能集来确定组织的功能。

Organizations 有助于您管理多个 AWS 账户的策略。您可以使用该服务创建账户组。然后，您可以将策略附加到组，以便在整个账户中应用正确的策略。

您可以创建 AWS 账户组，然后对每个组应用不同的策略。

Organizations API 能够以编程方式创建新账户，并将其添加到组中。附加到该组的策略将自动应用到新账户。

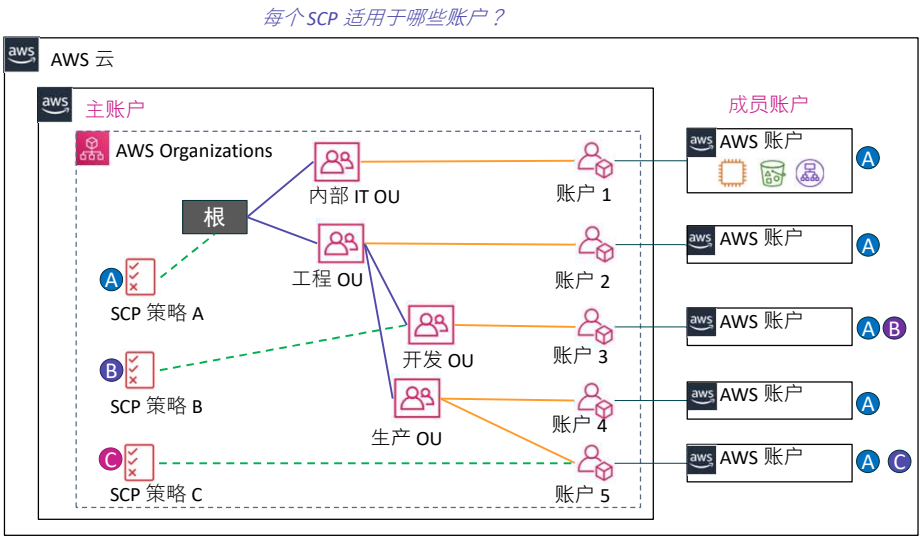
您也可以通过整合账单，为组织中的所有 AWS 账户设置单一付款方式。通过整合账单，您可以在一个综合视图中查看所有账户产生的费用。

最后，您可以在 API 级别管理 AWS 服务的使用。例如，您可以将策略应用到一组账户，仅允许这些账户中的 IAM 用户从 S3 存储桶中读取数据。

AWS Organizations：图解

在 AWS Organizations 主账户中：

1. 创建组织单位 (OU) 层次结构
2. 将账户作为成员账户分配给 OU
3. 定义将权限限制应用于特定成员账户的服务控制策略 (SCP)
4. 将 SCP 附加到根用户、OU 或账户



这是一个 AWS 组织的示例。它在常规 AWS 账户中定义，该账户就是幻灯片中的*主账户*，因为 AWS 组织是在该账户中定义的。

当您在主账户中创建*组织*时，组织会自动创建名为*根*的父容器。然后，您可以在组织中的每个根下定义*组织单位*，也称为 *OU*。每个 OU 都是*成员账户*的容器。一个 OU 还可以包含其他 OU，而这些 OU 可以包含更多账户。此功能使您能够创建树状层次结构。您可以把根和 OU 视为账户中伸出和结束的分支，就像枝叶一样。

为了跨账户配置访问控制，接下来您要定义*服务控制策略 (SCP)*。将每个策略附加到 OU 和账户层次结构中的适当位置。该策略从根中流出，影响其下的所有 OU 和账户。因此，如果您将 SCP 应用于根（如示例中的 *SCP 策略 A*），它将适用于组织中的所有 OU 和账户。您可以将 SCP 附加到根、任何 OU 或单个账户。

请记住，与 IAM 策略一样，SCP 只有在被明确允许且未被应用于用户的任何其他 SCP 或 IAM 策略明确拒绝的情况下才会授予访问权限。例如，假设应用于组织根的 SCP 策略 A 对特定服务或资源集设置的限制比 SCP 策略 C 更多。那么，账户 5 中的用户需要遵守策略 A 设置的更严格的权限。类似的，如果单个账户级别的任何 IAM 策略明确拒绝用户的任何操作，则这些 IAM 策略将覆盖 SCP 中授予该账户的任何权限。

SCP 的示例用途

- 服务控制策略 (SCP) 的特性
  - 它们使您能够控制成员账户中的 IAM 用户可以访问哪些服务
  - SCP 不能被本地管理员覆盖
  - 在单个账户中定义的 IAM 策略仍然适用

- SCP 的示例用途
  - 创建*阻止*服务访问或特定操作的策略
    - 示例：拒绝用户在所有成员账户中禁用 AWS CloudTrail
  - 创建*允许*完全访问特定服务的策略
    - 示例：允许对 Amazon EC2 和 CloudWatch 的完全访问权限
  - 创建*强制实施资源标记*的策略

服务控制策略 (SCP) 使您能够控制成员账户中的 IAM 用户可以访问哪些服务。假设您有要跨多个账户应用的特定策略。相较于将这些权限设置复制到每个账户的 IAM 策略文档，在 SCP 中定义这些策略会更容易。

SCP 应与在每个账户中定义的 IAM 策略一起使用。您可以认为 SCP 能够提供有关服务的一般边界，以及应当允许或拒绝用户访问的一般权限。然后，您可以使用 IAM 策略设置特定于单个账户的更精细的访问控制。

您可以创建旨在阻止（或拒绝）访问某些服务的 SCP。您还可以定义旨在允许访问某些服务的 SCP。最后，您可能会决定创建一个 SCP 来执行资源标记。这样，当在您的账户中创建新资源时，您的访问控制或成本分配标记策略可以保持有效。

第 5 节要点

- 您可以使用多个 AWS 账户隔离业务单元、开发和测试环境、受监管的工作负载以及审计数据
- AWS Organizations 使您能够配置自动账户创建和整合账单
- 您可以使用服务控制策略 (SCP) 配置跨账户的访问控制

本模块中这节内容的要点包括:

- 您可以使用多个 AWS 账户隔离业务单元、开发和测试环境、受监管的工作负载以及审计数据
- AWS Organizations 使您能够配置自动账户创建和整合账单
- 您可以使用服务控制策略 (SCP) 配置跨账户的访问控制

模块 8 – 挑战实验：使用 IAM 控制 AWS 账户访问权限

现在，您将完成模块 8 – 挑战实验：使用 IAM 控制 AWS 账户访问。

业务需求：用户访问控制

咖啡馆必须定义用户在云资源中应具有的访问权限级别。然后，他们必须在整个 AWS 账户中实施这些访问控制。

Mateo 最近造访咖啡馆时向 Sofía 介绍了 IAM 服务的功能。她打算使用 IAM 来实现自己的目标。

在与 Mateo 讨论过咖啡馆的 AWS 基础设施后，Sofía 意识到，她必须解决一些关于咖啡馆员工使用 AWS 账户的基本安全问题。

咖啡馆现在已经小有规模，团队成员各自分工（如开发人员或数据库管理员），在 AWS 上构建、维护或访问应用程序。不过，到目前为止，团队还没有根据用户的角色和职责来明确定义应拥有的访问级别。

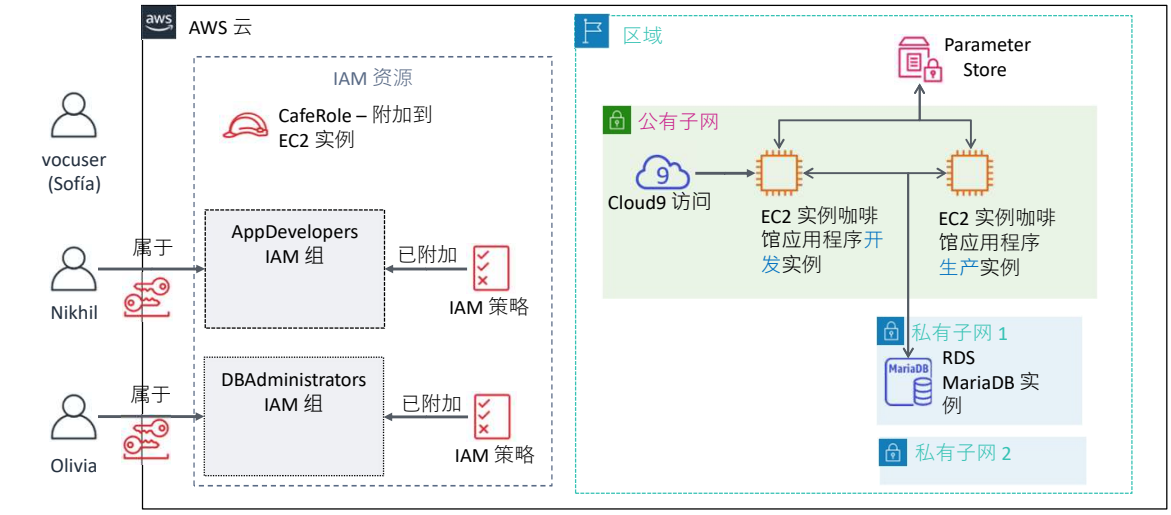
挑战实验：任务

1. 使用策略和 IAM 用户配置 IAM 组
2. 以 Nikhil 的身份登录并测试访问
3. 配置有关数据库管理员用户访问权限的 IAM
4. 以数据库管理员的身份登录并解决数据库连接问题
5. 使用 IAM 策略模拟器并通过可视化编辑器创建自定义 IAM 策略

在本挑战实验中，您将完成以下任务：

1. 使用策略和 IAM 用户配置 IAM 组
2. 以 Nikhil 的身份登录并测试访问
3. 配置有关数据库管理员用户访问权限的 IAM
4. 以数据库管理员的身份登录并解决数据库连接问题
5. 使用 IAM 策略模拟器并通过可视化编辑器创建自定义 IAM 策略

挑战实验：最终产品



该图总结了您完成实验后将构建的内容。

大约 80 分钟

开始模块 8 – 挑战实验：使用 IAM 控制 AWS 账户访问

现在可以开始挑战实验了。

挑战实验总结：要点

完成这个挑战实验之后，您的讲师现在可能会带您讨论此挑战实验的要点。

现在来回顾下本模块，并对知识考核以及对实践认证考试问题的讨论进行总结。

模块总结

总的来说，在本模块中，您学习了如何：

- 说明 AWS Identity and Access Management (IAM) 用户、组和角色的用途
- 描述如何允许在架构中使用用户联合以提高安全性
- 了解 AWS Organizations 服务控制策略 (SCP) 如何提高架构内的安全性
- 描述如何管理多个 AWS 账户
- 配置 IAM 用户

总的来说，在本模块中，您学习了如何：
- 说明 AWS Identity and Access Management (IAM) 用户、组和角色的用途
- 描述如何允许在架构中使用用户联合以提高安全性
- 了解 AWS Organizations 服务控制策略 (SCP) 如何提高架构内的安全性
- 描述如何管理多个 AWS 账户
- 配置 IAM 用户

完成知识考核

现在该完成本模块的知识考核了。

考试样题

公司将访问密钥（访问密钥 ID 和秘密访问密钥）存储在自定义 AMI 上的文本文件中。该公司使用访问密钥访问从 AMI 创建的实例中的 DynamoDB 表。安全团队要求采用更安全的解决方案。

哪种解决方案能满足安全团队的要求？

选项	答案
A	将访问密钥放入 S3 存储桶中，然后在启动时从实例中检索访问密钥。
B	将访问密钥通过实例用户数据传递到实例。
C	从在私有子网中启动的密钥服务器获取访问密钥。
D	创建具有访问表权限的 IAM 角色，然后使用新角色启用所有实例。

思考答案选项，并根据关键词排除错误选项。

考试样题答案

公司将访问密钥（访问密钥 ID 和秘密访问密钥）存储在自定义 AMI 上的文本文件中。该公司使用访问密钥访问从 AMI 创建的实例中的 DynamoDB 表。安全团队要求采用更安全的解决方案。

哪种解决方案能满足安全团队的要求？

正确答案是 D。

该问题中的关键词是“存储访问密钥”、“来自实例的 DynamoDB 表”、“自定义 AMI”和“最安全的解决方案”。

以下是要识别的关键词：**“存储访问密钥”、“实例中的 DynamoDB 表”、“自定义 AMI”和“最安全的解决方案”。**

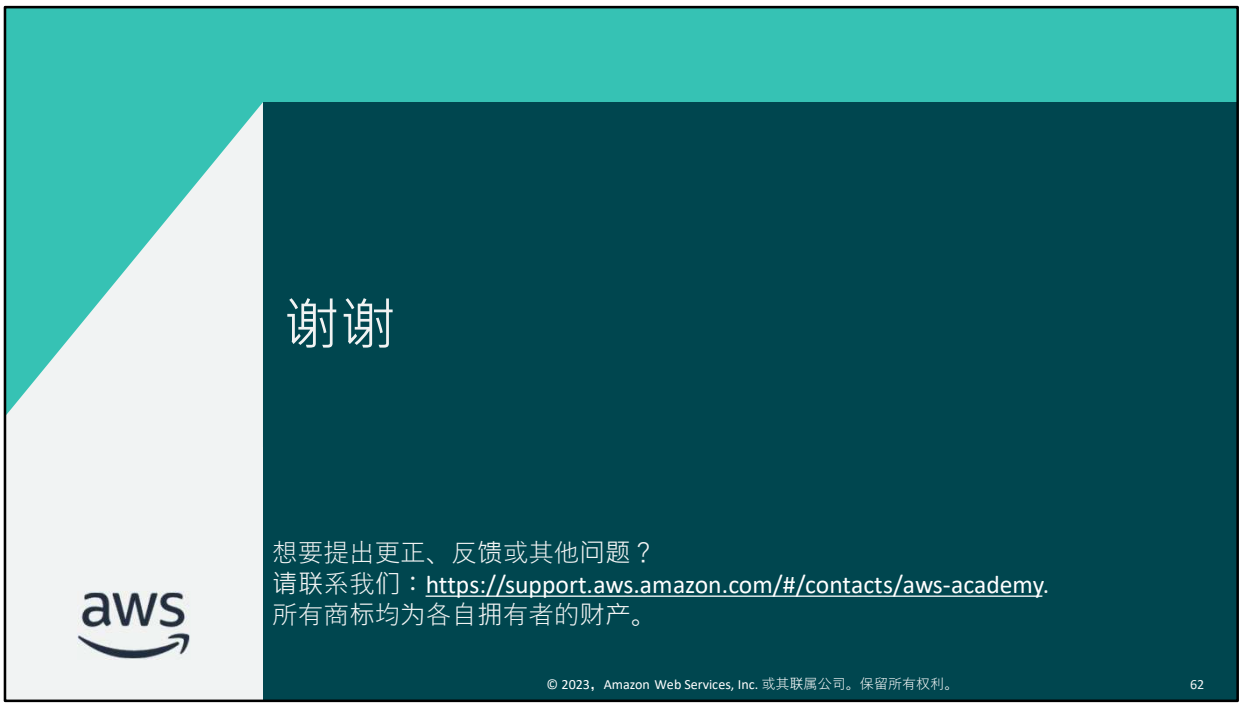
正确答案是 D。EC2 实例的 IAM 角色允许运行在实例上的应用程序访问 AWS 资源，而无需创建和存储任何访问密钥。任何需要创建访问密钥的解决方案都会引入管理该密钥的复杂性。

其他资源

- AWS Well-Architected Framework – 安全性支柱
- IAM 常见问题
- 创建 IAM 策略视频
- 不同层的身份视频
- 身份提供商和联合身份

如果您想进一步了解本模块中涵盖的主题，以下额外资源可能会对您有所帮助：

- AWS Well-Architected Framework – 安全性支柱
- IAM 常见问题
- 创建 IAM 策略视频
- 不同层的身份视频
- 身份提供商和联合身份



感谢您完成本模块的学习。