aws training and certification

AWS Certified DevOps Engineer – Professional (DOP-C01) 考试指南

简介

AWS Certified DevOps Engineer – Professional (DOP-C01) 考试适用于担任 DevOps 工程师角色的个人。本考试考查考生在 AWS 平台上预置、运行和管理分布式应用程序系统的技术专业知识。

本考试还考查考生能否完成以下任务：

- 在 AWS 上实施和管理持续交付系统和方法
- 实施和自动完成安全控制、监管流程和合规性验证
- 在 AWS 上定义和部署监控、指标和日志记录系统
- 在 AWS 平台上实施高度可用、可扩展且自行修复的系统
- 设计、管理和维护用于自动完成运行流程的工具

目标考生描述

目标考生应具有 2 年或更长时间的 AWS 环境预置、运行和管理经验。目标考生还必须具有使用至少一种高级编程语言开发代码的经验。

建议掌握的 AWS 知识

目标考生应具备以下知识：

- 构建高度自动化的基础架构的经验
- 管理操作系统的经验
- 了解最新的开发和运行流程和方法

哪些内容被视为超出目标考生的范围？

以下列出的是不要求目标考生能够完成的相关工作任务（非详尽列表）。以下内容被视为超出考试范围：

- 高级网络（高级路由算法、故障转移方法）
- 为开发人员提供深层次的安全建议
- 数据库查询和性能优化
- 全栈应用程序代码开发

- 规范化数据架构

有关本考试中可能涉及的特定工具和技术的详细列表以及列入考试范围的 AWS 服务列表，请参阅附录。

考试内容

答案类型

本考试具有两种类型的试题：

- **单选题：**具有一个正确答案和三个错误答案（干扰项）
- **多选题：**在 5 个或更多答案选项中具有两个或更多正确答案

选择一个或多个最准确描述或回答试题的答案。干扰项或错误答案是知识或技能不全面的考生可能会选择的答案选项。干扰项通常是与内容领域相符的看似合理的答案。

未回答的试题将计为回答错误；猜答案不会扣分。本考试包括
65 道计分试题，这些试题将影响您的分数。

不计分内容

考试包括 10 道不计分试题，这些试题不影响您的分数。AWS 收集这些不计分试题的考生答题情况以进行评估，以便将来将这些试题作为计分试题。不会在考试中标明这些不计分试题。

考试成绩

AWS Certified DevOps Engineer – Professional (DOP-C01) 考试成绩分为及格和不及格。本考试按照 AWS 专业人员根据认证行业最佳实践和准则制定的最低标准进行评分。

您的考试成绩换算分数为 100 – 1000 分。最低及格分数为 750 分。您的分数表明您的总体考试答题情况以及是否通过考试。换算评分模型有助于在难度水平可能略有不同的多种考试形式中换算分数。

您的成绩单可能包含一个分类表，其中列出您在每个部分的考试成绩。此信息旨在提供有关您的考试成绩的一般反馈。本考试采用补偿评分模型，这意味着您无需在每个部分都达到及格分数。您只需通过整个考试。

考试的每个部分具有特定的权重，因此，某些部分的试题比其他部分多。该表包含常规信息以重点说明您的强项和弱项。在解读各个部分的反馈时，请务必小心谨慎。通过考试的考生将不会收到此额外信息。

内容大纲

本考试指南包括考试的权重、测试领域和目标，并未列出考试的全部内容。不过，为每个目标提供了额外的背景信息，以帮助指导您为考试做好准备。下表列出了主要内容领域及其权重。该表位于完整考试内容大纲之前，其中包括额外的背景信息。每个领域中的百分比仅代表计分内容。

领域	在考试中所占的百分比
领域 1：SDLC 自动化	22%
领域 2：配置管理和基础架构即代码	19%
领域 3：监控和日志记录	15%
领域 4：策略和标准自动化	10%
领域 5：事件响应	18%
领域 6：高可用性、容错和灾难恢复	16%
总计	**100%**

领域 1：SDLC 自动化

1.1 应用自动完成 CI/CD 管道所需的概念

- 设置存储库
- 设置构建服务
- 集成自动化的测试（例如单元测试、完整性测试）
- 设置部署产品/服务
- 编排多个管道阶段

1.2 确定源代码控制策略以及如何实施它们

- 确定集成来自多个贡献者的代码更改的工作流
- 评估安全要求并建议代码存储库访问设计
- 将运行的应用程序版本与存储库版本（标签）保持一致
- 区分不同的源代码控制类型

1.3 应用自动完成和集成测试所需的概念

- 运行集成测试以作为代码合并过程的一部分
- 批量运行负载/压力测试和基准测试应用程序
- 根据应用程序退出代码测量应用程序运行状况（可靠的运行状况检查）
- 自动完成单元测试以检查通过/失败、代码覆盖
    - CodePipeline、CodeBuild 等
- 将测试与管道集成在一起

1.4 应用安全地构建和管理构件所需的概念

- 根据构件安全分类区分存储选项
- 将应用程序要求转换为操作系统和包配置（构建规范）
- 确定代码/环境依赖项和所需的资源
    - 示例：CodeDeploy 应用程序规范、CodeBuild 构建规范
- 运行代码构建过程

1.5 确定部署/交付策略（例如 A/B、蓝/绿、金丝雀、红/黑）以及如何使用 AWS 服务实施这些策略

- 根据业务需求确定正确的交付策略
- 评估现有的部署策略并提出改进建议
- 根据业务连续性目标建议 DNS/路由策略（例如 Route 53、ELB、ALB、负载均衡器）
- 验证部署成功/失败并自动回滚

领域 2：配置管理和基础架构即代码

2.1 根据部署需求确定部署服务

- 表明了解部署模型的流程
- 根据特定的部署模型，对相关 AWS 服务进行分类和实施以满足要求
    - 根据具有 DynamoDB 的要求，选择 CloudFormation 而不是 OpsWorks
    - 确定如何处理滚动更新

2.2 根据业务需求确定应用程序和基础架构部署模型

- 根据业务要求，平衡不同的考虑因素（成本、可用性、恢复时间）以选择最佳的部署模型
- 根据特定的 AWS 服务确定部署模型
- 分析与部署模型有关的风险以及相关的修复措施

2.3 在资源预置自动化中应用安全概念

- 根据要求选择最佳的自动化工具
- 表明了解资源预置的安全最佳实践（例如，加密数据包、动态生成凭证）

- 检查 IAM 策略，并评估是否为部署的所有生命周期阶段（例如，创建、更新、提升）授予了足够但最少的权限
- 检查凭证管理解决方案（例如，EC2 参数存储、第三方）
- 构建自动化
    - CloudFormation 模板、Chef 配方、说明书、代码管道等

2.4 确定如何在部署上实施生命周期挂钩

- 确定相应的集成方法以满足项目要求
- 在 Auto Scaling 组中选择相应的挂钩解决方案（例如，在节点发生故障后实施主节点选择方案）
- 评估挂钩实施的失败影响（如果远程调用失败，或者相关服务（例如 Amazon S3）暂时不可用），并提供恢复能力改进建议
- 评估部署过程的失败影响并评估回滚/恢复过程

2.5 应用使用 AWS 配置管理工具和服务管理系统所需的概念

- 确定 AWS 配置管理工具的优缺点
- 表明了解配置管理组件
- 表明可以在没有帮助的情况下端到端地运行配置管理服务，同时遵循行业最佳实践

领域 3：监控和日志记录

3.1 确定如何设置日志和指标聚合、存储和分析

- 实施和配置分布式日志收集和处理（例如，代理、syslog、flumed、CW 代理）
- 聚合日志（例如，Amazon S3、CW 日志、中间系统 (EMR)、Kinesis FH - 转换、ELK/BI）
- 实施自定义 CW 指标、日志订阅筛选条件
- 管理日志存储生命周期（例如，CW 到 S3、S3 生命周期、S3 事件）

3.2 应用自动完成环境监控和事件管理所需的概念

- 分析日志（例如，Amazon S3 数据事件/事件日志/ELB/ALB/CF 访问日志），与其他警报/事件（例如，发送到 AWS Lambda 的 CW 事件）相关联并采取相应的措施
- 使用 CloudTrail/VPC Flow Logs 进行检测控制（例如 CT、CW 日志筛选条件、Athena、NACL 或 WAF 规则），并根据错误处理逻辑（状态机）采取相关的措施（AWS 步骤）
- 使用 ESM (SSM)、Inspector、CodeDeploy、OpsWorks 和 CW 代理配置和实施补丁/清单/状态管理
    - EC2 停用/维护
- 处理扩展/故障转移事件（例如，ASG、DB HA、路由表/DNS 更新、应用程序配置、自动恢复、PH 控制面板、TA）

- 确定如何自动创建监控

3.3 应用审核、记录和监控操作系统、基础架构和应用程序所需的概念

- 使用可用的 AWS 工具（具有 EB 和 Lambda 的 X-Ray）监控端到端服务指标 (DDB/S3)
- 通过审核 (Inspector)、配置规则、CloudTrail（流程和操作）和 AWS API 验证环境/操作系统状态
- 启用、配置和分析自定义指标（例如，应用程序指标、内存、KCL/KPL）并采取措施
- 确保容器监控（例如，任务状态、放置、日志记录、端口映射、LB）
- 区分启用服务级别或操作系统级别监控的服务
    - 示例：使用操作系统代理的 AWS 服务（例如，Inspector、SSM）

3.4 确定如何实施标记和其他元数据策略

- 根据标记（生命周期阶段 - 开发/生产）和条件上下文键分离权限
- 利用 Amazon S3 系统/用户定义的元数据进行分类和自动化
- 使用 CodeDeploy 设计和实施基于标签的部署组
- 使用标记进行成本分配/优化的最佳实践

领域 4：策略和标准自动化

4.1 应用实施日志记录、指标、监控、测试和安全标准所需的概念

- 检测、报告和响应监管和安全违规情况
- 在应用程序、操作系统和基础架构中应用日志记录标准
- 应用上下文特定的应用程序运行状况和性能监控
- 简要说明日志和指标（例如，JSON、XML、数据规范化）的交付模型标准

4.2 确定如何通过自动化优化成本

- 优先考虑自动化工作以降低人工成本
- 根据指标实施正确的工作负载大小调整
- 评估通过自动化流程编排和可重复任务缩短产品上市时间的方法
- 诊断异常值以确定是否符合使用案例要求
    - 示例：配置偏差
- 通过事件测量和自动完成成本优化
    - 示例：Trusted Advisor

4.3 应用实施监管策略所需的概念

- 在 CI/CD 管道中推广监管标准
- 简要说明并测量是否符合监管策略的实时状态
- 有关符合监管策略的报告

- 部署与自助服务功能相关的监管策略
    - 示例：服务目录、CFN Nag

领域 5：事件响应

5.1 解决问题并确定如何恢复运行

- 根据给定的问题，评估如何尽快缩小非正常运行组件的范围
- 鉴于负载增加，确定采取哪些措施以减轻影响
- 确定失败原因和影响
    - 示例：部署、运行
- 确定发生故障后恢复运行的最佳方法
- 调查记录的事件，并将其与应用程序组件相关联
    - 示例：应用程序源代码

5.2 确定如何自动化事件管理和发出警报

- 设置在发生灾难性故障时通过备份自动还原
- 设置方法以提供适用于不同类型的事件的警报和通知
- 评估警报的质量/可操作性
- 配置适用于应用程序 SLA 的指标
- 主动更新限制

5.3 应用实施自动修复所需的概念

- 设置正确的扩展策略，以便在发生故障时启用自动修复（例如，使用 Auto Scaling 策略）
- 使用正确的回滚策略以避免部署失败的影响
- 配置 Route 53 以确保跨区域故障转移
- 检测和响应维护或 Spot 终止事件

5.4 应用设置事件驱动的自动化操作所需的概念

- 配置 Lambda 函数或 CloudWatch 操作以实施自动化操作
- 设置 CloudWatch 事件规则和/或配置规则和目标
- 使用 AWS Systems Manager 或 Step Functions 协调组件（例如，Lambda，使用维护时段）
- 配置构建/部署流程以自动响应关键软件更新

领域 6：高可用性、容错和灾难恢复

6.1 确定正确使用多可用区与多区域架构

- 根据 HA/DR 要求确定部署策略

- 根据成本和持久性要求确定数据复制策略
- 根据 HA/DR 要求确定基础架构、平台和服务
- 基于服务可用性的 HA/FT/DR 设计（即全球/区域/单一可用区）

6.2 确定如何实施高可用性、可扩展性和容错

- 设计部署策略以支持 HA/FT/可扩展性
- 评估应用程序基础架构组件的有状态性
- 使用负载均衡在多个可用区/ASGS/实例类型（Spot/M4 与 C4）/目标之间分配流量
- 使用相应的缓存解决方案以提高可用性和性能

6.3 根据业务需求（例如，RTO/RPO、成本）确定正确的服务

- 为您的应用程序确定经济高效的存储解决方案
    - 示例：分层、存档、EBS 类型、热/冷
- 选择满足业务要求的数据库平台和配置
- 根据业务要求选择经济高效的计算平台
    - 示例：Spot
- 根据业务要求选择部署服务/模型
    - 示例：Code Deploy、蓝/绿部署
- 确定何时使用托管服务与自行管理的基础架构（EC2 上的 Docker 与 ECS）

6.4 确定如何设计和自动完成灾难恢复策略

- 自动完成故障检测
- 自动完成组件/环境恢复
- 选择相应的部署策略以进行环境恢复
- 设计自动化以在混合环境中支持故障转移

6.5 评估部署的故障点

- 确定相应的部署特定运行状况检查
- 在部署期间实施故障检测
- 实施故障事件处理/响应
- 确保具有资源/组件/流程以应对部署期间发生的故障
- 在每个部署事件上查找退出代码
- 将错误映射到不同的部署点

附录

本考试可能涵盖哪些关键的工具、技术和概念？

以下是考试中可能出现的工具和技术列表（非详尽列表）。该列表可能会有更改，用于帮助您了解考试涵盖的服务、功能或技术的一般范围。该列表中的一般工具和技术未按特定顺序显示。AWS 服务根据其主要功能进行分组。尽管在本考试中考查的一些技术可能比其他技术多，但这些技术在该列表中的顺序和位置并不表示其相对权重或重要性：

- 应用程序部署
- 应用程序集成
- 应用程序管道
- 自动化
- 代码存储库最佳实践
- 成本优化
- 部署要求
- 混合部署
- IAM 策略
- 指标、监控、警报和日志记录
- 网络 ACL 和安全组设计和实施
- 运行最佳实践
- 回滚过程

AWS 服务和功能

分析：

- Amazon Athena
- Amazon EMR
- Amazon Kinesis Data Firehose
- Amazon Kinesis Data Streams
- Amazon QuickSight

计算：

- Amazon EC2
- Amazon EC2 Auto Scaling

容器：

- AWS App Runner
- Amazon Elastic Container Registry (Amazon ECR)
- Amazon Elastic Container Service (Amazon ECS)
- Amazon Elastic Kubernetes Service（Amazon EKS）
- AWS Fargate

数据库：

- Amazon DynamoDB
- Amazon RDS
- Amazon Redshift

开发人员工具：

- AWS Cloud Development Kit (AWS CDK)
- AWS CloudShell
- AWS CodeArtifact
- AWS CodeBuild
- AWS CodeCommit
- AWS CodeDeploy
- Amazon CodeGuru
- AWS CodePipeline
- AWS CodeStar
- AWS 命令行界面 (CLI)
- AWS X-Ray

管理和监管：

- AWS CloudFormation
- AWS CloudTrail
- Amazon CloudWatch
- AWS Config
- AWS OpsWorks
- AWS Organizations
- AWS Systems Manager
- AWS Trusted Advisor

联网和内容分发：

- Amazon API Gateway
- AWS 客户端 VPN

- Amazon CloudFront
- Amazon Route 53
- AWS 站点到站点 VPN
- AWS Transit Gateway
- Amazon VPC
- Elastic Load Balancing

安全性、身份和合规性：

- Amazon GuardDuty
- AWS Identity and Access Management (IAM)
- Amazon Inspector
- AWS Key Management Service (AWS KMS)
- AWS Secrets Manager
- AWS Single Sign-On
- AWS WAF

无服务器：

- Amazon EventBridge (Amazon CloudWatch Events)
- AWS Lambda
- AWS Serverless Application Model (AWS SAM)
- Amazon Simple Notification Service (Amazon SNS)
- Amazon Simple Queue Service (Amazon SQS)
- AWS Step Functions

存储：

- Amazon Elastic Block Store (Amazon EBS)
- Amazon Elastic File System（Amazon EFS）
- Amazon S3
- AWS Storage Gateway