SYNOPSYS®

使用 Kubernetes 安装 Black Duck

Black Duck 2023.1.0

内容

前言

Black Duck 文档

Black Duck 的文档包括在线帮助和以下文档：

标题	文件	说明
发行说明	release_notes.pdf	包含与当前版本和先前版本中的新功能和改进功能、已解决问题和已知问题有关的信息。
使用 Docker Swarm 安装 Black Duck	install_swarm.pdf	包含有关使用 Docker Swarm 安装和升级 Black Duck 的信息。
入门	getting_started.pdf	为初次使用的用户提供了有关使用 Black Duck 的信息。
扫描最佳做法	scanning_best_practices.pdf	提供扫描的最佳做法。
SDK 入门	getting_started_sdk.pdf	包含概述信息和样本使用案例。
报告数据库	report_db.pdf	包含有关使用报告数据库的信息。
用户指南	user_guide.pdf	包含有关使用 Black Duck 的 UI 的信息。

在 Kubernetes 或 OpenShift 环境中安装 Black Duck 软件的安装方法是 Synopsysctl 和 Helm。单击以下链接查看文档。

- Helm 是 Kubernetes 的软件包管理器，可用于安装 Black Duck。Black Duck 支持 Helm3，Kubernetes 的最低版本为 1.13。
- Synopsysctl 是一款云原生管理命令行工具，用于在 Kubernetes 和 Red Hat OpenShift 中部署 Black Duck 软件。

Black Duck 集成文档可在 Confluence 上找到。

客户支持

如果您在软件或文档方面遇到任何问题，请联系 Synopsys 客户支持。

您可以通过以下几种方式联系 Synopsys 支持：

- 在线：https://www.synopsys.com/software-integrity/support.html
- 电话：请参阅我们的支持页面底部的“联系我们”部分以查找您当地的电话号码。

要打开支持案例，请登录 Synopsys Software Integrity 社区网站，网址为：https://community.synopsys.com/s/contactsupport。

另一个可随时使用的方便资源是在线客户门户。

Synopsys Software Integrity 社区

Synopsys Software Integrity 社区是我们提供客户支持、解决方案和信息的主要在线资源。该社区允许用户快速轻松地打开支持案例，监控进度，了解重要产品信息，搜索知识库，以及从其他 Software Integrity Group (SIG) 客户那里获得见解。社区中包含的许多功能侧重于以下协作操作：

- 连接 - 打开支持案例并监控其进度，以及监控需要工程或产品管理部门协助的问题
- 学习 - 其他 SIG 产品用户的见解和最佳做法，使您能够从各种行业领先的公司那里汲取宝贵的经验教训。此外，客户中心还允许您轻松访问 Synopsys 的所有最新产品新闻和动态，帮助您更好地利用我们的产品和服务，最大限度地提高开源组件在您的组织中的价值。
- 解决方案 - 通过访问 SIG 专家和我们的知识库提供的丰富内容和产品知识，快速轻松地获得您正在寻求的答案。
- 分享 - 与 Software Integrity Group 员工和其他客户协作并进行沟通，以众包解决方案，并分享您对产品方向的想法。

访问客户成功社区。如果您没有帐户或在访问系统时遇到问题，请单击此处开始，或发送电子邮件至 community.manager@synopsys.com。

培训

Synopsys Software Integrity 的客户教育 (SIG Edu) 板块是满足您的所有 Black Duck 教育需求的一站式资源。它使您可以全天候访问在线培训课程和操作方法视频。

每月都会添加新视频和课程。

在 Synopsys Software Integrity 的客户教育 (SIG Edu) 板块，您可以：

- 按照自己的节奏学习。
- 按照您希望的频率回顾课程。
- 进行评估以测试您的技能。
- 打印完成证书以展示您的成就。

要了解更多信息，请访问 https://community.synopsys.com/s/education，或者，要获取 Black Duck 的帮助信息，请选择 Black Duck 教程（位于 Black Duck UI 的“帮助”菜单（  ）中）。

Synopsys 关于包容性和多样性的声明

Synopsys 致力于打造一个包容性的环境，让每位员工、客户和合作伙伴都感到宾至如归。我们正在审查并移除产品中的排他性语言以及支持面向客户的宣传材料。我们的举措还包括通过内部计划从我们的工程和工作环境中移除偏见语言（包括嵌入我们软件和 IP 中的术语）。同时，我们正在努力确保我们的 Web 内容和软件应用程序可供不同能力的人使用。由于我们的 IP 实施了行业标准规范，目前正在审查这些规范以移除排他性语言，因此您可能仍在我们的软件或文档中找到非包容性语言的示例。

1. 使用 Synopsysctl 安装

Kubernetes 是一种编排工具，用于通过容器管理云工作负载。

使用 synopsysctl 安装

使用 synopsysctl 在 Kubernetes 或 OpenShift 上安装 Black Duck。

Synopsysctl 是一个命令行工具，可帮助在 Kubernetes 和 OpenShift 群集中部署和管理 Synopsys 软件。安装 synopsysctl 后，您可以利用它轻松部署和管理 Synopsys 软件。

- 单击此处了解 synopsysctl 的概述。
- 单击此处了解有关安装和使用 synopsysctl 的文档。

注: 有关可扩展性调整原则，请参见《Black Duck 发行说明》的“容器可扩展性”一节。请勿删除 Black Duck 数据库 (bds_hub) 中的数据，除非 Synopsys 技术支持代表指示这样做。确保遵循适当的备份程序。删除数据会导致 UI 问题、Black Duck 完全无法启动等问题。Synopsys 技术支持无法重新创建已删除的数据。如果没有可用的备份，Synopsys 将尽力提供支持。

2. 硬件要求

以下性能数据是使用 Black Duck 2022.10.0（缩减了特征扫描持久性）（默认值）和 Synopsys Detect 8.0.0 收集的。SPH 值是使用特征扫描、软件包管理器探测器扫描和快速扫描的组合计算的。平均扫描大小因客户而异，因此确切的 SPH 吞吐量极度取决于客户。这些指标是从 Google Cloud Platform 收集的，该平台为不同配置提供不同的数据库读/写 IOPS。

10sph	每小时扫描数：50 SPH 百分比增加： 400% 每小时 API 数：2500 项目版本：10000	IOPS: • 读取：15000 • 写入：9000 Black Duck 服务： • CPU：12 核 • 内存：30 GB PostgreSQL： • CPU：2 核 • 内存：8 GB	总数： • CPU：14 核 • 内存：38 GB
120sph	每小时扫描数：120 SPH 百分比增加：0% 每小时 API 数：3000 项目版本：13000	IOPS: • 读取：15000 • 写入：15000 Black Duck 服务： • CPU：13 核 • 内存：46 GB PostgreSQL： • CPU：4 核 • 内存：16 GB	总数： • CPU：17 核 • 内存：62 GB
250sph	每小时扫描数：300 SPH 百分比增加：20% 每小时 API 数：7500 项目版本：15000	IOPS: • 读取：15000 • 写入：15000 Black Duck 服务： • CPU：17 核 • 内存：118 GB PostgreSQL： • CPU：6 核 • 内存：24 GB	总数： • CPU：23 核 • 内存：142 GB
500sph	每小时扫描数：650 SPH 百分比增加：30% 每小时 API 数：18000 项目版本：18000	IOPS: • 读取：15000 • 写入：15000 Black Duck 服务： • CPU：28 核	总数： • CPU：38 核 • 内存：250 GB

		• 内存：210 GB PostgreSQL： • CPU：10 核 • 内存：40 GB	
1000sph	每小时扫描数：1400 SPH 百分比增加：40% 每小时 API 数：26000 项目版本：25000	IOPS: • 读取：25000 • 写入：25000 Black Duck 服务： • CPU：47 核 • 内存：411 GB PostgreSQL： • CPU：18 核 • 内存：72 GB	总数： • CPU：65 核 • 内存：483 GB
1500sph	每小时扫描数：1600 SPH 百分比增加：6% 每小时 API 数：41000 项目版本：28000	IOPS: • 读取：25000 • 写入：25000 Black Duck 服务： • CPU：60 核 • 内存：597 GB PostgreSQL： • CPU：26 核 • 内存：104 GB	总数： • CPU：92 核 • 内存：701 GB
2000sph	每小时扫描数：2300 SPH 百分比增加：15% 每小时 API 数：50000 项目版本：35000	IOPS: • 读取：60000 • 写入：25000 Black Duck 服务： • CPU：66 核 • 内存：597 GB PostgreSQL： • CPU：34 核 • 内存：136 GB	总数： • CPU：100 核 • 内存：733 GB

这个新指南基于当前的 Black Duck 2022.10.0 架构。此指南可能会针对后续版本进一步完善。如果您有任何疑问或疑虑，请联系产品管理部门。

注: 所需的磁盘空间量取决于要管理的项目的数量，因此各个要求可能有所不同。考虑每个项目大约需要 200 MB。

Black Duck Software 建议监视 Black Duck 服务器上的磁盘利用率，以防止磁盘达到可能导致 Black Duck 出现问题的容量。

调整 binaryscanner 副本的数量，以及根据每小时将执行的二进制扫描的预期数量增加 PostgreSQL 资源，从而完成 BDBA 扩展。对于每小时每 15 次二进制扫描，添加以下资源：

- 一个 binaryscanner 副本
- 一个用于 PostgreSQL 的 CPU
- 用于 PostgreSQL 的 4GB 内存

如果您的预期扫描速率不是 15 的倍数，则向上舍入。例如，每小时 24 次二进制扫描将需要以下资源：

- 两个 binaryscanner 副本、
- 两个用于 PostgreSQL 的额外 CPU，以及
- 用于 PostgreSQL 的 8GB 额外内存。

当二进制扫描为总扫描量（按扫描计数）的 20% 或更少时，此指南有效。

二进制扫描

如果您获得了二进制扫描的许可，则可能需要增加 uploadcache 容器/Pod 内存，因为这是二进制扫描程序提取和处理二进制文件的位置。默认情况下，内存设置为 512MB，这不足以进行大型扫描。扫描大二进制文件时，建议将 uploadcache 容器/Pod 的内存至少增加到 4 GB。为此，请查找覆盖 yaml 并将内存限制更新为 4096MB。

对于 Swarm 安装：

```
uploadcache:
  deploy:
    resources:
      limits:
        cpus: ".200"
        memory: "4096M"
      reservations:
        cpus: ".100"
        memory: "4096M"
      replicas: 1
```

对于 Kubernetes 安装：

```
uploadcache:
  replicas: 1
  resources:
    limits:
      cpu: "200m"
      memory: "4096Mi"
    requests:
      cpu: "100m"
      memory: "4096Mi"
```

注：安装 Black Duck Alert 需要 1 GB 额外内存。

3. PostgreSQL 版本

Black Duck 2022.10.0 支持新的 PostgreSQL 特性和功能，以提高 Black Duck 服务的性能和可靠性。从 Black Duck 2022.10.0 开始，PostgreSQL 容器 13 是内部 PostgreSQL 容器当前支持的 PostgreSQL 版本。

从较旧版本的 Black Duck（早于 2022.10.0）升级的客户将需要迁移到 PostgreSQL 13。Black Duck 2022.10.0 更新将内部 Black Duck PostgreSQL 数据库容器迁移到 PostgreSQL 版本 13。如果您使用数据库容器并在 OpenShift 上部署，则需要运行一次性迁移作业，如 Black Duck 发行说明和安装指南中所述。

注：有关 PostgreSQL 调整指南，请参阅 Black Duck 硬件扩展指南。

如果您选择运行自己的外部 PostgreSQL 实例，Synopsys 建议为新安装使用 PostgreSQL 14。由于 PostgreSQL 14.0 到 14.3 中的索引损坏错误，支持的最低 PostgreSQL 14 版本是 14.4。

警告：不要在 PostgreSQL 数据目录上运行防病毒扫描。防病毒软件会打开大量文件，锁定文件，等等。这些操作会干扰 PostgreSQL 的运行。具体错误因产品而异，但通常会导致 PostgreSQL 无法访问其数据文件。一个例子是，PostgreSQL 失败，并显示“系统中打开的文件太多”。

常规迁移过程

此处的指南适用于从任何基于 PG 9.6 的 Hub（早于 2022.2.0 的版本）升级到 2022.10.0 或更高版本。

1. 迁移由 blackduck-postgres-upgrader 容器执行。
2. 如果从基于 PostgreSQL 9.6 的 Black Duck 版本升级：
   - PostgreSQL 数据卷的文件夹布局经过重新排列，使未来的 PostgreSQL 版本升级更加简单。
   - 数据卷所有者的 UID 已更改。新的默认 UID 为 1001，但请参见特定于部署的说明。
3. 运行 pg_upgrade 脚本以将数据库迁移到 PostgreSQL 13。
4. 在 PostgreSQL 13 数据库上运行普通“分析”以初始化查询计划程序统计信息。
5. blackduck-postgres-upgrader 退出。

4. 使用 Helm 安装 Black Duck

Helm 图表说明了一组 Kubernetes 资源，Helm 部署 Black Duck 需要用到这些资源。Black Duck 支持 Helm3，Kubernetes 的最低版本为 1.13。

您可以在以下网址获取 Helm 图表：https://sig-repo.synopsys.com/artifactory/sig-cloudnative

单击此处了解有关使用 Helm 安装 Black Duck 的说明。Helm 图表引导在 Kubernetes 群集上使用 Helm 软件包管理器部署 Black Duck。

使用 Helm 在 Kubernetes 上进行迁移

如果您从基于 PostgreSQL 9.6 的 Black Duck 版本升级，此迁移将用 Synopsys 提供的容器替换 CentOS PostgreSQL 容器。此外，synopsys-init 容器将替换为 blackduck-postgres-waiter 容器。

在普通 Kubernetes 上，升级作业的容器将以 root 身份运行（除非覆盖）。但是，唯一的要求是作业与 PostgreSQL 数据卷的所有者以相同的 UID 运行（默认为 UID=26）。

在 OpenShift 上，升级作业假定它将使用与 PostgreSQL 数据卷所有者相同的 UID 运行。

5. 管理任务

在 Kubernetes 中配置密钥加密

Black Duck 支持对系统中的关键数据进行静态加密。此加密基于编排环境（Docker Swarm 或 Kubernetes）调配给 Black Duck 安装的密钥。创建和管理此密钥、创建备份密钥以及根据您所在组织的安全策略轮换密钥的过程如下所述。

要加密的关键数据如下：

- SCM 集成 Oauth 令牌
- SCM 集成提供商 OAuth 应用程序客户端密钥
- LDAP 凭据
- SAML 私有签名证书

注: 一旦启用了密钥加密，就永远不能禁用它。

什么是加密密钥？

加密密钥是一个随机序列，用于生成内部加密密钥以解锁系统内的资源。Black Duck 中的密钥加密由 3 个对称密钥（根密钥、备份密钥和以前的密钥）控制。这三个密钥通过传递到 Black Duck 的种子，作为 Kubernetes 和 Docker Swarm 密钥生成。这三个密钥被命名：

- crypto-root-seed
- crypto-backup-seed
- crypto-prev-seed

在正常情况下，并非全部三个种子都会被激活使用。除非正在执行轮换操作，否则唯一活动的种子将是根种子。

保护根种子

必须保护根种子。拥有您的根种子以及系统数据副本的用户可以解锁并读取系统的受保护内容。某些 Docker Swarm 或 Kubernetes 系统默认情况下不静态加密密钥。强烈建议将这些编排系统配置为在内部加密，以便以后在系统中创建的密钥能够保持安全。

根种子是从备份重新创建系统状态（作为灾难恢复计划的一部分）所必需的。根种子文件的副本应存储在独立于编排系统的秘密位置，以便种子与备份的组合可以重新创建系统。不建议将根种子存储在与备份文件相同的位置。如果一组文件被泄露或被盗 - 两种情况都会出现，因此，建议为备份数据和种子备份设置单独的位置。

在 Kubernetes 中启用密钥加密

要在 Kubernetes 中启用密钥加密，必须在 values.yaml 编排文件中将 enableApplicationLevelEncryption 的值更改为 true：

```
# if true, enables application level encryption
enableApplicationLevelEncryption: true
```

密钥种子管理脚本

您可以在 Black Duck GitHub 公共存储库中找到示例管理脚本：

https://github.com/blackducksoftware/secrets-encryption-scripts

这些脚本不是用来管理 Black Duck 密钥加密，而是用来说明此处所述的低级 Docker 和 Kubernetes 命令的用法。有两组脚本，每组都在其自己的子目录中，对应于在 Kubernetes 和 Docker Swarm 平台上使用。对于 Kubernetes 和 Docker Swarm，各个脚本之间存在一对一对应关系（如果适用）。例如，两组脚本都包含一个具有如下名称的脚本：

createInitialSeeds.sh

在 Kubernetes 中生成种子

在 OpenSSL 中生成种子

可以使用任何机制（生成至少 1024 字节长度的安全随机内容）生成种子的内容。一旦创建种子并将其保存在密钥中，就应将其从文件系统中移除并保存在一个私密的安全位置。

OpenSSL 命令如下所示：

```
openssl rand -hex 1024 > root_seed
```

在 Kubernetes 中生成种子

有许多 Kubernetes 命令行将创建密钥。下面列出的命令可以更好地跟踪密钥及其是否更改，并确保能够使用联机系统操纵密钥。在 Black Duck 激活运行时，密钥可以在 Kubernetes 中创建和删除。

```
kubectl create secret generic crypto-root-seed -n $NAMESPACE --save-config --dry-run=client --
from-file=crypto-root-seed=./root_seed -o yaml | kubectl apply -f -
```

要删除 Kubernetes 中之前的密钥：

```
kubectl delete secret crypto-prev-seed -n $NAMESPACE
```

配置备份种子

建议备份根种子，以确保系统可以在灾难恢复场景中恢复。备份根种子是一个备用根种子，可用于恢复系统。因此，它必须以与根种子相同的方式安全地存储。

备份根种子具有一些特殊特性，即，一旦它与系统关联，即使在根种子轮换期间，它也仍然可行。一旦系统处理了备份种子，应将其从密钥中移除，以限制其受到攻击和泄漏的可能性。备份根种子可能有不同的（频率较低的）轮换计划，因为系统中的密钥不应在任何时候都处于“活动”状态。

当您需要或想要轮换根种子时，首先需要将当前根种子定义为上一个根种子。然后，您可以生成一个新的根种子并将其放置到位。

当系统处理这些种子时，以前的根密钥将用于轮换资源，以使用新的根种子。完成此处理后，应从密钥中移除之前的根种子，以完成轮换并清理旧资源。

创建备份根种子

初始创建后，备份种子/密钥将 TDEK（租户解密、加密密钥）低级密钥打包。示例脚本 createInitialSeeds.sh 将创建根种子和备份种子。一旦 Black Duck 运行，它使用两个密钥来打包 TDEK。

该操作完成并且根种子和备份种子都安全地存储在其他位置后，应删除备份种子密钥；请参阅示例脚本cleanupBackupSeed.sh。

如果根密钥丢失或泄漏，备份密钥可用于替换根密钥；请参阅示例脚本useRootSeed.sh。

轮换备份种子

与根密钥类似，备份种子应定期轮换。与根种子不同（旧的根种子存储为以前的种子密钥，而新的根种子密钥提供给系统），备份种子只是通过创建新的备份种子来进行轮换。请参阅示例脚本rotateBackupSeed.sh。

轮换完成后，新的备份种子应安全存储并从 Black Duck 主机文件系统中移除。

在 Kubernetes 中管理密钥轮换

根据组织的安全策略定期轮换正在使用的根种子是一种好做法。要执行此操作，还需要一个额外的密钥来执行轮换。要轮换根种子，将当前根种子配置为“上一个根种子”，并生成新生成的根种子并将其配置为根种子。一旦系统处理此配置（具体细节如下），密钥将被轮换。

此时，新旧种子都能够解锁系统内容。默认情况下，将使用新的根种子，允许您测试并确保系统按预期工作。一旦所有内容都得到验证，您就可以通过移除“以前的根种子”来完成轮换。

从系统中移除之前的根种子后，就不能再将其用于解锁系统内容，并且可以将其丢弃。新的根种子现在是正确的根种子，应适当地备份和保护。

根密钥用于打包实际加密和解密 Black Duck 密钥的低级 TDEK（租户解密、加密密钥）。应该在方便 Black Duck 管理员并符合用户组织规则时，定期轮换根密钥。

轮换根密钥的过程是使用当前根种子的内容创建以前的种子密钥。然后，应创建一个新的根种子并将其存储在根种子密钥中。

Kubernetes 中的密钥轮换

对于 Kubernetes 来说，这三个操作都可以在运行 Black Duck 的情况下完成。Kubernetes 示例脚本 rotateRootSeed.sh 将把根种子提取到 prev_root 中，创建一个新的根种子，然后重新创建以前的种子和根种子。

轮换完成后，应移除上一个种子密钥；请参阅示例脚本cleanupPreviousSeed.sh。同样，可以对正在运行的 Kubernetes Black Duck 实例执行此清理。

在用户界面中，转到“管理”>“系统信息”>“加密”，查看“加密诊断”选项卡即可跟踪轮换状态。