OWASP 中国
Open Worldwide Application Security Project

# OWASP Non-Human Identities Top 10 2025

## 十大非人类身份风险

# 前言

欢迎来到 OWASP 十大非人类身份风险 2025。

该项目概述了应用程序开发人员面临的与非人类身份 (NHI) 相关的十大风险。随着 NHI 在开发流程中变得至关重要，了解这些风险至关重要。

该列表通过识别组织在 NHI 方面面临的主要风险并使用 **OWASP 风险评级方法**对其进行排名来编制。数据来源包括现实世界的违规行为、调查、CVE 数据库等。有关此流程的详细信息，请参阅《排名标准》和《方法和数据》。

先从《项目介绍》入手，深入了解《OWASP 十大非人类身份风险 2025》，全面认识相关风险。

欢迎各界贡献力量！查阅我们的贡献指南，参与其中，助力项目发展。

## 概述

“非人类身份（NHI）TOP 10”指的是与非人类身份在开发周期中相关的 10 个最关键挑战或安全风险列表。这些挑战基于可利用性、普遍性、可检测性和影响等因素进行排名。该列表旨在帮助安全专业人员了解非人类攻击面，并开发出更好的保护和管理方法。

非人类身份 NHI 通常被开发者用来促进应用程序的创建，但它们也可能带来重大的安全风险。NHI TOP 10 项目提供了对这些风险的全面概述，包括它们如何被利用的说明、它们的普遍性以及对组织可能产生的影响。此外，该项目还提供了可行的预防措施和事件响应手册，以帮助组织缓解这些风险。

OWASP Non-Human Identities Top 10 | OWASP Foundation

## 贡献者

| Name | Affiliation | Contact |
|---|---|---|
| Roni Lichtman | Torch Security | Twitter LinkedIn |
| Tal Skverer | Astrix Security | LinkedIn |
| Or Cohen | - | LinkedIn |
| Idan Basre | - | LinkedIn |
| Amir Benvenisti | - | LinkedIn |

| | | |
|---|---|---|
| Dor Dali | Cyolo | LinkedIn |
| Jack Schofield | Snyk | LinkedIn |
| Tomer Yahalom | Astrix Security | LinkedIn |
| Danielle Guetta | Astrix Security | LinkedIn |
| Bar Kaduri | Orca Security | LinkedIn |
| Yonatan Yosef | Orca Security | LinkedIn |

## 中文项目组成员

项目组成员：王文君、阮子禅、王厚奎、郭佩刚、程远冲、张坤

审核：王文君、张坤、王颉

# 目录

# 介绍

## 定义：什么是非人类身份

非人类身份（Non-Human Identity，NHI）用于为应用程序、API、机器人和自动化系统等软件实体提供访问受保护资源的授权。与人类身份不同，NHI 不受人类控制或直接拥有。它们的身份对象和身份验证通常与人类不同，常见的人类用户安全措施并不适用。

NHI 的示例包括：

- 后端系统用于连接多个子系统的**服务账户**。
- 与自动化服务相关的用于访问云资源的**角色**。
- 微服务用于访问数据库应用程序的 **API 密钥 或访问密钥**。
- 第三方用来执行任务和增强功能的**应用程序**。

NHI 支持多种凭证和身份验证方法，包括密码、证书、令牌、密钥等。

随着现代软件日益自动化和互联化，NHI 对于应用程序开发变得至关重要。

## 保护 NHIs 的重要性

NHI 管理不善会带来重大安全风险。

关键问题包括：

- **过度权限**：NHI 通常被授予非常广泛的资源访问权限，如果受到损害，会导致广泛的损害。
- **凭证管理错误**：NHI 凭证很容易被错误管理：在代码中留下硬编码的密钥、不良或没有轮换策略，以及使用过时的身份验证方法，会使 NHI 容易受到攻击。
- **缺乏监控**：NHI 的监控不足，导致有关于 NHI 的恶意活动被安全设备忽视。

上述这些主要问题意味着，一旦 NHI 遭到入侵，可能导致未经授权的访问、数据泄露或对基础设施的攻击。由于 NHI 在开发流程、云环境和 SaaS 生态系统中发挥着关键作用，因此保护它们至关重要。

## 风险及漏洞示例

近期随着 NHI 使用率的增加，涉及 NHI 被盗的现实世界事件呈指数级增长。下面介绍一些近期最引人注目的事件，以展示未受安全管控的 NHI 风险：

- 微软的午夜暴雪漏洞（2024 年 1 月）：午夜暴雪（俄罗斯黑客组织）对微软的租户发起了攻击。在获得非生产 Microsoft 365 测试租户的访问权限后，他们攻陷了一个遗留的 OAuth 应用程序（一个未受

管的非人类身份），该应用程序具有访问 Microsoft 生产环境的完全权限。这引发了未经授权访问公司电子邮件账户的事件，导致了敏感通信和文件泄露。事件分析（中文）

- Okta 的支持系统漏洞（2023 年 11 月）：Okta 遭遇了一起安全事件，涉及一个被盗用的服务账户。一名员工在 Okta 管理的设备上登录后，将此服务账户的凭证保存到了他的个人谷歌账户中。该员工个人谷歌账户的泄露使攻击者能够获得这些凭证，从而未经授权访问 Okta 的客户支持系统。攻击者访问了与 134 个客户相关的文件，包括包含会话令牌等敏感数据的 HTTP Archive（HAR）文件。
- Internet Archive 的 Zendesk 支撑平台被入侵（2024 年 10 月）：攻击者利用了与 Internet ArchiveZendesk 支撑平台绑定的未轮换访问令牌，导致未授权访问和潜在的数据泄露。这一事件凸显了定期轮换和保护非人类身份凭证以防止未授权访问的重要性。

## OWASP NHI Top 10 项目

OWASP 非人类身份（NHI）十大风险项目确定并排名了与 NHI 相关的最关键风险，为开发人员和组织提供了一个实践指南。

### *项目重要性：*

随着 NHIs 的普及，保护它们变得和保护人类用户账户一样重要。该项目旨在：

- 提高行业对 NHI 相关安全挑战的认知。
- 提供可行的预防手段来保护 NHIs 免受最危险风险。
- 帮助开发人员和组织按重要性考虑、排序组织内 NHI 相关风险并实施最佳实践。

### *我们如何建立此名单：*

我们通过真实事件、调查报告、CVE 数据库和行业投入来识别关键风险。我们利用收集的数据并基于 OWASP 风险评估方法将前十大风险进行了排名，同时提供了一个清晰的优先级列表。

### *开发人员应该做什么：*

开发人员可以使用此项目来：

- 了解其应用程序中与 NHIs 相关的风险。
- 应用推荐实践以保护 NHIs 并缓解威胁。
- 持续监控和改进他们的 NHI 安全性。

# 排名标准

OWASP NHI Top 10 2025 项目使用了 OWASP 的风险评估方法来定义和评估风险标准。

该项目的目标是识别、优先排序并对与非人类身份（NHIs）相关的风险进行排名。为了实现这一目标，我们仅关注固有风险因素，而不考虑特定威胁场景发生的可能性或组织管理 NHIs 的特殊性。

这种方法确保了采用一致的方法，既突出了最紧迫的风险，又可以适应各种场景。

## 标准和术语

下表总结了在排名过程中使用的标准和术语：

| 威胁代理：可利用性 | 安全脆弱性：普遍性 | 安全脆弱性：可检测性 | 影响：技术 |
|---|---|---|---|
| 容易：3 | 广泛：3 | 困难：3 | 严重：3 |
| 平均：2 | 常见：2 | 平均：2 | 中等：2 |
| 困难：1 | 不常见：1 | 容易：1 | 低：1 |

### *假设解释*

为保持一致性，在排名过程中应用了一些具体假设：

- **可利用性：**分数假定该组织对该风险易受攻击，并且威胁行为者有足够的知识尝试利用。
- **影响：**影响得分反映了“最坏情况”，考虑到可能从风险中产生的最严重的后果。
- **普遍性：**普遍性评级侧重于安全脆弱性在环境中是如何广泛存在的，而没有考虑缓解措施。
- **可检测性：**假设已经建立了常规检测机制，组织能识别安全脆弱性的难易程度。

通过将这些假设作为排名过程的基础，我们的目的是提供一个清晰可行的框架，用于理解和解决与非人类身份相关的问题。

# 发行说明

2024 年 12 月 9 日：2025 年 NHI Top 10 初始版本发布

2024 年 12 月 10 日：NHI Top 10 主网站已更新，以指向新的 2025 年 NHI Top 10 子网站

2024 年 12 月 16 日：NHI:5 - 上传特权过高的 NHI 页面

2024 年 12 月 19 日：NHI:3 - 上传易受攻击的第三方 NHI 页面

2024 年 12 月 19 日：NHI:1 - 上传了不当的下线页面

2024 年 12 月 19 日：NHI:4 - 上传不安全的身份验证页面

2024 年 12 月 19 日：NHI:9 - 上传 NHI 重用页面

2024 年 12 月 19 日：NHI:6 - 上传不安全的云部署配置页面

2024 年 12 月 20 日：NHI:10 - 上传 NHI 的人为使用页面

2024 年 12 月 20 日：NHI:8 - 上传环境隔离页面

2024 年 12 月 20 日：NHI:2 - 上传机密泄露页面

2024 年 12 月 23 日：NHI:7 - 上传长期秘密页面

2024 年 12 月 23 日：上传元页面

2024 年 12 月 24 日：OWASP NHI Top 10 2025 项目上线！

# 方法和数据

## 概述

在起草风险、示例和参考材料收集的过程中，我们从各种来源收集数据来支撑排名标准的生成。

### *数据源*

以下资源被收集并使用：

1. **近期的违规行为：** 过去三年中涉及非人类身份（NHI）滥用的一个或多个攻击阶段的高调违规事件汇编。
2. **CVE 分数：**利用 CVSS 严重性评分作为关键指标，分析来自 NVD（国家漏洞数据库）的公开可用漏洞。
3. **调查报告数据：**突显 NHI 领域紧迫问题的调查汇总，为所有 NHI 相关风险的标准化排名提供数据源。包括：
   - Datadog 的云安全状态报告（2022 年，2023 年和 2024 年）
   - CSA NHI 报告（2024 年）
   - Verizon 的数据泄露调查报告（DBIR，2024 年）

## 方法论

### *初步起草*

项目团队最初在非人类身份安全领域内起草了 12 个风险。这个过程中与下列信息来源共同协商起草：- 社区知名人士 - 漏洞数据库 - 公开可用的事故报告

事故报告仅考虑在过去三年内报告的事故。每个确定的风险都记录有描述、示例攻击和相关引用。最初的草案已公开分享供审查和反馈。

### *数据收集*

在第二阶段，团队开始收集和审查公开可用的数据。收集的数据与每个风险的具体数据源匹配，确保全面覆盖。

## 标准和排名方法

团队召开会议讨论并最终确定排名方法。选择了以下标准作为最相关的度量标准：

- 可利用性
- 可检测性
- 普遍性
- 技术影响

术语与这些标准保持一致（详见“排名标准”部分）。

## 风险排名

每位贡献者都会对特定的风险进行评估。团队根据收集到的数据源对每项风险进行排名，并将其与术语对齐。这项工作产生了 OWASP NHI Top-10 的初始草稿。

## 验证和定稿

排名草案被公开，团队举行了评审会议，以确保： - 验证跨风险的一致性 - 术语一致性 - 确认分数得到收集到的数据点的支撑。

在最后阶段，团队基于其对该风险影响严重性的判断给每个评判标准分配权重。将术语值转换成数字得分，并计算出每个风险的加权平均分作为最终的风险得分。

# OWASP 十大非人类身份风险 2025

| 风险 | 描述 |
| --- | --- |
| NHI1:2025 不当注销 | 不当注销是指在不再需要非人类身份（例如服务账户和访问密钥）时，未正确停用或删除它们。未受监控和已弃用的服务可能仍然易受攻击，攻击者可以利用其相关的非人类身份，未经授权的访问敏感系统和数据。 |
| NHI2:2025 密钥泄露 | 密钥泄露是指在整个软件开发生命周期中，敏感的非人类身份（例如 API 密钥、令牌、加密密钥和证书）泄露到未经授权的数据存储中。当密钥被泄露时（例如，硬编码到源代码中、以明文形式存储在配置文件里，或通过公共聊天应用程序发送出去），它们很容易被暴露。 |
| NHI3:2025 脆弱的第三方 NHI | 第三方非人类身份通过集成开发环境（IDEs）及其扩展插件，以及第三方软件即服务（SaaS）的使用，被广泛地融入开发工作流程中。无论是由于安全漏洞还是恶意更新，如果某个第三方扩展插件遭到攻击，攻击者就可以利用它来窃取这些凭证，或者滥用所授予的权限。 |
| NHI4:2025 不安全的身份验证 | 开发人员经常将内部和外部（第三方）服务集成到他们的应用程序中。这些服务需要访问系统内的资源，从而需要身份认证凭证。然而，一些身份认证方法已被弃用、易受已知攻击或因使用过时的安全实践而被认为脆弱。使用不安全或过时的认证机制可能使组织面临重大风险。 |
| NHI5:2025 权限过高的 NHI | 在应用程序开发和维护过程中，开发人员或管理员可能会为非人类身份分配超过其功能所需的权限。无论是通过应用程序中的漏洞、恶意软件还是其他安全漏洞，当一个权限过高的非人类身份被攻破，攻击者就可以利用这些过高权限。 |
| NHI6:2025 不安全的云部署配置 | 持续集成和持续部署（CI/CD）应用程序使开发人员能够自动化构建、测试和部署代码到生产环境的过程。这些集成通常需要与云服务进行认证，通常通过静态凭证或 OpenID Connect（OIDC）实现。静态凭证可能会通过代码库、日志或配置文件意外暴露。如果受到攻击，这些凭证可以为攻击者提供对生产环境的持久且潜在的特权访问权限。虽然 OIDC 提供了更安全的替代方案，但如果身份令牌未得到适当验证，或者对令牌声明没有严格条件，未经授权的用户可能会利用这些脆弱性获得访问权限。 |
| NHI7:2025 长期有效的密钥 | 长期有效的密钥指的是使用过期时间设置得过于遥远或永不过期的敏感非人类身份（如 API 密钥、令牌、加密密钥和证书）。如果一个被攻破的秘密长期有效，攻击者就可以不受任何时间限制地访问敏感服务。 |
| NHI8:2025 环境隔离 | 环境隔离是云应用程序部署中的基本安全实践，其中开发、测试、预发布和生产使用不同的环境。在部署过程中以及整个应用程序生命周期中，通常会使用非人类身份。然而，在多个环境中重用相同的非人类身份，尤其是在测试和生产之间，可能会引入重大的安全漏洞。 |
| NHI9:2025 NHI 重用 | 在不同应用程序、服务或组件之间重用相同的非人类身，即使它们一起部署，也会引入重大的安全风险。如果一个非人类身份在一点被攻破，攻击者就可以利用它未经授权地访问使用相同凭证的其他系统。 |
| NHI10:2025 人类使用 NHI | 在应用程序开发和维护期间，开发人员或管理员可能会滥用非人类身份执行手动任务，而这些任务本应使用具有适当权限的人类身份执行。这种做法带来了重大的安全风险，例如非人类身份的权限提升、由于人与自动化活动难以区分而导致缺乏审计和问责机制。 |

# NHI1:2025 不当注销

| 威胁代理&攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**容易** | 普遍性：**广泛** | 可检测性：**困难** | 技术：**严重** | 业务：**特定** |
| 利用不当注销的非人类身份（NHI）很大程度上取决于具体场景。如针对内部威胁场景，很容易确定利用不当注销的身份需要哪些必要凭证。 | 目前，诸如服务账户之类的 NHI 注销功能还不够完善，组织很少使用现有的功能来完成。因此，许多 NHI 在不再需要或原所有者离开后都没有得到妥善注销。<br>安全团队缺乏工具来检测不当注销的旧 NHI。检测此类 NHI 的现有技术依赖于需要耗费很长时间收集到的不完整信息。 | | 由于潜在内部威胁对组织有深入的了解，利用不当注销的 NHI 可能会导致关键系统受到威胁、敏感数据泄露以及使用高级持久性攻击手段的情况。 | |

## 风险描述

不当注销是指当不再需要非人类身份（NHI）（例如服务账户和访问密钥）时，未充分停用或删除它们。这种情况通常发生在应用程序弃用、服务下线或这些 NHI 的原始所有者或管理员离开组织时。不当注销 NHI 会带来重大安全风险。未受监控和弃用的服务可能仍然易受攻击，攻击者可以利用它们相关的 NHI 来未经授权访问敏感系统和数据。此外，孤立的 NHI 可能会保留着高权限，从而放大任何安全漏洞造成的潜在损害。

## 攻击场景示例

- **孤立的 Kubernetes 服务账户：**已停用服务的 Kubernetes 集群保留了仍然有效的服务账户。如果攻击者获得此不受监控集群的访问权限，他们可以利用这些服务账户与组织基础架构内的其他资源进行交互，从而可能导致数据泄露或进一步入侵。
- **离职员工利用未撤销的凭证：**管理自动化服务的员工离开了组织，但与这些自动化服务相关的 NHI 并未被停用或转移。离职员工可能会滥用仍然有效的凭证远程访问组织的系统，从而导致未经授权的数据访问、服务中断甚至蓄意破坏。
- **利用遗留应用程序进行权限提升和横向移动：**在测试环境中创建的应用程序用于测试工作负载，随后连接到敏感的生产环境以完成测试套件，并且工作负载转移到生产服务器中运行时该应用程序并未被停用。这样一来，进入安全性较低的测试环境的攻击者就可以利用这个应用程序在组织内进行横向移动。

## 如何预防

- 实施注销流程，审查与离职员工相关的所有 NHI。对于每个 NHI，确定是否仍然需要。如果不需要，则停用它；否则，将所有权转让给另一名员工，并轮换离职员工在创建期间可能访问过的所有凭证。
- 通过将人力资源系统与身份和访问管理（IAM）工具相集成，尽可能实现注销步骤的自动化。
- 定期审核活跃的 NHI，以识别正在进行的人类使用情况并阻止潜在的滥用行为。

## 参考文献

- CSA：淘汰孤立和陈旧的非人类身份
- 离职员工访问了前公司系统并删除资源
- 微软遭午夜暴雪组织入侵

## 支撑数据

- CSA NHI 报告
  - 31% 的受访者将 NHI 不当注销列为最令人担忧的三大 NHI 威胁之一。（5/10）
  - 32% 的 NHI 相关安全事件是由孤立身份导致的。（4/10）
  - 15% 的组织需要自动配置和取消配置作为 NHI 工具最重要的功能。（9/16）
  - 51% 的组织没有正式流程来注销或撤销长期 API 密钥。
- 近期违规事件：
  - 微软被攻击事件。

# NHI2:2025 密钥泄露

| 威胁代理&攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**容易** | 普遍性：**常见** | 可检测性：**困难** | 技术：**严重** | 业务：**特定** |
| 成功利用泄露的密钥极其容易，因为密钥使攻击者能够以合法应用程序的身份进行身份验证。 | 密钥是机器对机器通信的常用认证方法，包括 API 密钥、访问密钥和数据库凭证。在整个开发生命周期中，使用密钥来构建新功能并测试机器到机器集成（M2M intergrations），因此往往会扩散到组织内许多不同的数据存储中。<br>由于可能出现在各种各样的数据存储中，因此很难检测到密钥泄露。例如，密钥可能会泄漏到开发者端点、应用日志、配置文件、SaaS 提供商、云平台等常见场景。 | | 密钥通常包含高影响 NHIs（如 API 密钥和数据库连接字符串）凭证，因此在违规行为上的影响是严重的。 | |

## 风险描述

密钥泄漏是指敏感 NHIs（如 API 密钥、令牌、加密密钥和证书）在整个软件开发生命周期中泄漏到未经批准的数据存储的情况。开发人员在应用程序开发过程中经常利用这些密钥来使其能够与组织内的各种服务和资源进行身份验证并交互。然而，当密钥被泄漏时——例如，被硬编码到源代码中、存储在纯文本配置文件中或通过公共聊天应用程序被分享——它们就变得容易暴露。暴露的密钥可能导致重大的安全风险。如果一个密钥被泄漏了，无论是通过代码仓库、日志还是开发人员计算机上的恶意软件，恶意攻击者都可以利用它来获得未经授权的系统访问权限、窃取数据或在网络内提升特权。这会导致数据泄露、服务中断以及客户和利益相关者的信任丧失。

## 攻击场景示例

- **Azure SAS 令牌泄漏：**将 Azure SAS 令牌提交到公共 Github 存储库。攻击者使用该 SAS 令牌对关联的 Azure 订阅进行身份验证，并泄漏内部 Microsoft Teams 消息。
- **Delinea 管理员 API 密钥：**Delinea 管理员 API 密钥存储在一个员工公用共享文件的脚本中。具有公司网络有限权限的攻击者可以识别 API 密钥，从 PAM（特权访问管理）读取管理员凭证并在公司网络中升级其权限。

## 如何预防

- **尽可能使用临时凭证**
  - 用短存、按需生成的临时凭证替换静态密钥（例如，AWS STS、Azure 托管标识或 OAuth 令牌）。
  - 临时凭证降低了长期曝光的风险。
- **使用密钥管理工具**
  - 使用专用的密钥管理工具（例如，AWS Secrets Manager、Azure Key Vault 或 HashiCorp Vault），安全地存储密钥。
  - 确保密钥没有被硬编码在源代码、配置文件或脚本中。
- **自动检测密钥**
  - 将密钥扫描工具（例如，GitHub Secret Scanning、TruffleHog、Gitleaks）集成到 CI / CD 流水线中，以检测和防止密钥被提交到存储库。
- **限制密钥范围和权限**
  - 遵循最小特权原则，仅授予需要密钥的应用程序和服务访问权限。
  - 使用基于角色的访问控制（RBAC）强制实施细粒度权限。
- **定期轮换密钥**
  - 自动化密钥轮换过程，减少暴露凭证的影响。
  - 使用支持密钥版本化和自动更新的工具。

## 相关 OWASP 资源

- OWASP Secrets Management Cheat Sheet
- OWASP WrongSecrets project

## 参考文献


- 微软 AI 研究人员意外公开 38TB 数据
- 优步数据泄露的原因是什么？
- Microsoft Azure Key Vault：最佳实践
- HashiCorp Vault：什么是 Vault？
- GitHub：保护您的代码的最佳做法
- AWS Secrets Manager

## 支撑数据

- CSA NHI 报告
  - 31%的 NHI 相关安全事件是由不良密钥管理导致的。（6/10）
  - 21%的组织认为服务账户是最难管理的。（6/16）
  - 26%的组织认为密钥生命周期管理是 NHI 工具最重要的能力之一。（1/16）
  - 37%的组织上报密钥暴露在环境变量中或者被硬编码在应用程序代码中。
- Verizon DBIR
  - 21%的入侵是由被盗凭证引发的。（1/10）
- 最近的入侵
  - 微软 SAS 令牌入侵。
  - 优步入侵。
  - 互联网档案馆入侵。

# NHI3:2025 脆弱的第三方 NHI

| 威胁代理&攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**平均** | 普遍性：**常见** | 可检测性：**困难** | 技术：**严重** | 业务：**特定** |
| 找到存在漏洞的第三方应用程序并非易事，需要付出一些努力。但是，一旦被攻破，访问第三方客户/用户/客户端就很容易了。 | 第三方应用程序通常被开发人员使用，例如 VSCode 扩展插件及商业应用。在许多情况下，这些扩展插件的开发者是小型社区或个人，他们并未遵循最佳实践，因此存在安全隐患。第三方应用程序的安全漏洞事件很少会被公开披露，即便有公开的情况，相关细节也往往不完整，这使得对它们的监控变得困难重重。如果没有提前预警，几乎不可能检测到存在安全漏洞的第三方应用程序。 | | 第三方应用程序通常被赋予了对敏感资料（如源代码、秘密、文件等）的广泛访问权限。一旦开发者的生态系统遭到破坏，可能会引发供应链攻击、个人数据被盗取，甚至对关键系统造成影响。 | |

## 风险描述

第三方非人类身份（NHIs）通过集成开发环境（IDEs）及其扩展插件，以及第三方软件即服务（SaaS）的使用，被广泛地融入开发工作流程中。例如，Visual Studio Code（VSCode）有一个庞大的扩展插件市场，这些插件可以增强其功能。这些扩展插件通常需要大量访问开发者的计算机，包括读取代码、访问环境变量以及与其他系统资源进行交互的能力。为了实现其功能，第三方可能需要与外部服务进行集成，如版本控制系统、数据库、虚拟机和云环境。这种集成就需要向第三方提供敏感的非人类身份信息，如访问令牌、API 密钥或 SSH 密钥。如果第三方扩展插件受到攻击 —— 无论是由于安全漏洞还是恶意更新 —— 攻击者就可以利用它来窃取这些凭证，或者滥用所授予的权限。此外，第三方可能会接触到代码库中硬编码的凭证，或者包含敏感信息的环境变量。如果第三方能够访问这些元素，并且变得恶意，或者本身就是恶意的，它就可以将这些敏感信息泄露出去。因此，在没有进行适当审查和采取安全措施的情况下依赖第三方非人类身份，会给组织带来重大风险。

## 攻击场景示例

- **集成开发环境（IDE）与代码仓库的集成：**开发者常常会对其集成开发环境进行配置，以便直接与诸如 GitHub 或 GitLab 这样的代码仓库进行交互。这种配置需要向集成开发环境或其扩展插件提供个人访问令牌（PATs）或 SSH 密钥，从而实现代码同步、推送提交以及管理拉取请求等功能。如果某个扩展插件被攻破，这些凭证就会暴露，进而导致未经授权的人员能够访问开发者的代码仓库，甚至可能接触到组织的敏感代码。

- **扩展插件访问云资源：**那些用于在虚拟机或云服务上进行部署和测试的扩展插件，需要访问云环境。开发者会向这些扩展插件提供非人类身份，例如应用程序编程接口（API）密钥或访问令牌，以便它们能够与亚马逊网络服务 AWS、微软 Azure 或谷歌云平台等服务进行交互。一个恶意的扩展插件可能会利用这些凭证来访问、修改或删除云资源，从而引发数据泄露或服务中断。
- **第三方服务提供商：**开发者会集成像 Sisense 这样的第三方服务提供商来创建一个商业智能（BI）应用程序。在集成过程中，开发者会创建诸如数据库凭证之类的具有特权的非人类身份，并将其发送给服务提供商。如果该服务提供商遭到攻击，攻击者就可以利用这些非人类身份来访问开发者的环境。

## 如何预防

- **审查并限制第三方集成**
  - 在集成第三方工具或服务之前，务必进行安全审查，确保供应商遵循安全最佳实践。
  - 仅授予所需的最低权限（遵循最小特权原则），并定期对未使用的访问权限进行审核或撤销。
- **监控并检测第三方行为**
  - 持续通过日志记录、API 调用跟踪以及行为分析来监控第三方活动，以便及时发现异常情况。
  - 部署安全扫描工具，用于识别第三方软件中存在的漏洞或恶意行为。
- **使用临时且定期轮换的凭证**
  - 用短期有效的临时凭证（例如，亚马逊 AWS STS、微软 Azure 的托管标识）来替换长期有效的机密信息。
  - 实现凭证自动轮换机制，以降低因第三方集成被入侵而造成的影响。

## 参考文献


- 影响 GitHub 插件的 JetBrains 安全问题
- 恶意 VSCode 扩展窃取数据
- 深入研究 VSCode 扩展漏洞
- 解读 Sisense 黑客攻击事件

## 支撑数据

- Datadog 2024 年云安全状况
  - 与 AWS 进行的第三方集成中，有 10% 存在权限过高的情况。
  - 与 AWS 进行的第三方集成里，2% 容易受到 “委托混淆” 漏洞的影响。
  - 有通过恶意的第三方 OAuth 应用程序来初步获取对 Microsoft 365 的访问权限的情况。
- CSA NHI 报告
  - 38% 的受访者将供应链攻击列为最令人担忧的三大 NHI 威胁之一。（2/10）
  - 16% 的受访者将恶意供应商列为最令人担忧的三大 NHI 威胁之一。（9/10）
  - 29% 的受攻击的外部集成是导致 NHI 相关安全事件的原因。（7/10）
  - 21% 的组织认为管理第三方工具和服务请求是最难的。（10/16）
  - 26% 的组织需要了解第三方供应商，这是 NHI 工具最重要的功能。（1/16）
  - 38% 的组织报告称对第三方供应商的了解有限甚至完全没有。
- 近期违规行为
  - Sisense 漏洞 -链接。

# NHI4:2025 不安全的身份认证

| 威胁代理&攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**容易** | 普遍性：**广泛** | 可检测性：**容易** | 技术：**中等** | 业务：**特定** |
| 一旦攻击者发现使用不安全身份验证的 NHI，他们可以利用已知的技术和工具来滥用和破坏该 NHI。 | 遗留应用程序几乎存在于每个授权中，并且通常使用遗留/不安全的身份验证方法，如 OAuth 隐式流程（Implicit flow）或没有 多因子验证（MFA） 的服务账户。根据不安全身份验证的类型，检测能力可能在可用的简单发现功能之间有所不同，或者难以识别的具体不安全身份验证罪犯。 | | 不安全协议通常用于促进给予高访问权限的敏感流程。成功利用使用不安全身份验证的 NHI 可能会导致账户接管或特权提升。 | |

## 风险描述

开发人员经常将内部和外部（第三方）服务集成到他们的 SaaS 应用程序和云环境中以增强其体验或简化操作。这些服务需要对系统内的资源进行访问，因此需要身份验证凭证。各种平台提供了多种身份验证方法，然而，某些身份验证方法已被弃用、易受已知攻击或由于使用过时安全实践而被视为薄弱环节，因此开发人员必须谨慎选择最适合其特定用例的安全选项。使用不安全或过时的身份验证机制可能会使组织面临重大风险，包括未经授权的访问、数据泄露和合规违规。开发人员和组织必须评估所有可用的身份验证选项，遵循行业最佳实践并选择提供强大安全功能的方法，并遵守标准化协议，例如 OAuth 2.1 和 OpenID Connect (OIDC)。

## 攻击场景示例

- **弃用的 OAuth 流程：**某些早期 OAuth 版本（OAuth 1.0 和 OAuth 2.0）中的流因存在安全漏洞而被弃用。例如：
  - **隐式流程：**通常用于单页应用，现在不推荐使用，因为它会将访问令牌暴露在 URL 中，使其容易受到拦截和重放攻击。
  - **无 PKCE 的授权代码流：**易受拦截和跨站点请求伪造 (CSRF) 攻击。现代实现应使用证明密钥交换 (PKCE) 扩展来增强安全性。
- **非标准的 OAuth 实现：**一些平台偏离官方 OAuth 标准，通过实施自定义行为，例如将访问令牌转换为 cookie 或按需生成 JSON Web Token (JWT)。这些非标准做法可能会引入意想不到的漏洞，因为它们可能缺乏官方规范中概述的安全考虑因素，从而可能导致安全漏洞。
- **使用基于凭证的身份验证而不是无凭证方法：**云提供商提供无凭证的身份验证机制，例如使用实例简

档或 OIDC 联合的云内访问。依赖于静态、基于凭证的身份验证（如长期有效的 API 密钥或密码）是不可取的，因为这些凭证可以在泄漏、代码存储库或日志中公开。无凭证方法提供临时、范围限定的凭证，降低了凭证泄漏和误用的风险。

- **绕过多因子验证（MFA）的应用密码：**微软和谷歌等平台提供针对旧版应用程序的支持，而不支持现代身份验证协议的应用程序。这些密码绕过了 MFA，这意味着即使用户启用了 MFA，也可以使用应用密码访问账户而无需额外的验证。获得应用密码的攻击者可以利用这一点来进行未经授权的访问，有效地抵消了 MFA 的安全优势，并将用户账户转化为不安全的服务账户。
- **使用用户名和密码的遗留身份验证协议：**某些应用程序继续使用过时或专有的身份验证流程，这些流程依赖于直接传输用户名和密码，模仿 OAuth 类似的行为但不符合其安全标准。这些方法缺少官方 OAuth 流程提供的保护措施，因此容易遭受凭证截获、重放攻击和中间人攻击。

## 如何预防

- **采用现代身份验证标准：**使用 OAuth 2.1 和 OIDC 进行安全身份验证，避免使用像隐式流程或无 PKCE 的授权代码流这样的已弃用流。
- **利用无凭证方法：**通过实例简档或 OIDC 联合替换静态凭证，使用临时、范围限定的令牌。
- **标准化 OAuth 实施：**避免与 OAuth 标准相悖的自定义实践，以减少安全差距。
- **定期进行安全审计：**定期审查身份验证方法，以确定并消除过时或不安全的配置。

## 相关 OWASP 资源

- OWASP Authentication Cheat Sheet

## 参考文献

- Salesforce：禁用不安全的授权流
- OAuth 2.0 安全最佳当前实践 - 隐式授予
- Salesforce：用户名－密码 OAuth 流
- Auth0：带有表单提交的隐式流
- Microsoft 支持：使用应用程序密码与不支持两步验证的应用程序一起
- Google 账号帮助：使用应用程序密码登录

## 支撑数据

- CSA NHI 报告
  - 22% 的答案将过时的访问方式作为前三个最令人担忧的 NHI 威胁之一。（8/10）
- 最近的漏洞
  - MSFT SAS Token 漏洞
  - Uber 漏洞
  - CircleCI 漏洞
  - Cloudflare 漏洞
  - Snowflake 漏洞
  - env 文件漏洞

# NHI5:2025 权限过高的 NHI

| 威胁代理&攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**困难** | 普遍性：**广泛** | 可检测性：**平均** | 技术：**严重** | 业务：**特定** |
| 成功利用权限过高的非人类身份（NHI）需要威胁主体首先获得对目标环境的访问权限。因此，权限过高的 NHI 依赖于一个独立的初始访问向量。 | NHIs 通常被过度赋予特权，因为给 NHI 分配合适的权限是一项非常困难且耗时的任务。对于权限过高的非人类身份的检测，会因环境类型的不同而有所差异。尽管云环境提供了简化检测的工具，但本地环境缺乏类似的内置功能，使得此类身份的检测变得更加困难。 | | 权限过高的 NHI 由于其附带的高权限，可能导致严重影响。这类身份通常为管理员账户，其权限范围广泛，潜在风险较大。 | |

## 风险描述

非人类身份（NHI）（如服务账户、API 令牌和工作负载身份）旨在通过编程方式访问云资源和服务。它们使应用程序、服务和自动化流程能够在无需人工干预的情况下安全运行。然而，在应用程序开发与维护期间，开发人员或管理员可能会无意中为 NHI 分配超出其功能需求的权限，一旦这些身份遭到泄露，就会无端地扩大潜在的影响范围。当权限过高的 NHI 受到入侵时（无论是通过应用程序漏洞、恶意软件，还是其他安全漏洞），攻击者就能利用这些过高权限来完成以下操作：

- **访问敏感数据：**未经授权访问机密文件、数据库或用户信息。
- **提升权限：**获取系统内更高级别的访问权限，可能达到管理员或 root 级别。
- **在网络内横向移动：**访问组织网络中 NHI 所能触及的其他系统或服务。
- **安装恶意软件：**部署恶意软件、勒索软件或其他恶意工具，以进一步破坏系统安全。
- **完全接管云账户：**与云根账户或管理员相关的身份信息泄露，可能导致攻击者获得完全控制权和账户接管。

## 攻击场景示例

- **权限过高的 Web 服务器用户：** Web 服务器在 Linux 计算机上以本地用户账户运行，该账户同时具有对其他应用程序、系统文件或敏感数据目录的访问权限。如果 Web 服务器存在允许远程代码执行的漏洞，攻击者可以利用此漏洞控制 Web 服务器的进程。利用该用户账户过高的权限，攻击者可以访问或修改其他应用程序、窃取敏感数据或进行未经授权的系统更改操作。
- **权限过高的虚拟机(VM)：**运行 Jenkins 的亚马逊 EC2 实例被错误地赋予了“AWS Administrator Access ”托管策略，尽管它仅需针对 EKS 和 ECS 的权限。攻击者通过利用实例中的漏洞获得初始访问权限，利

用过高的权限浏览云环境，并从 S3 存储桶中窃取敏感数据。

- **权限过高的 OAuth 应用程序：**开发人员在生产环境的 Azure 账户中安装他们正在开发的 OAuth 应用程序，并赋予该应用程序 “AppRoleAssignment.ReadWriteAll”（应用角色分配的读写全部权限）权限，尽管该应用程序仅需对 Azure Blob 存储中特定目录进行读取访问。如果恶意实体控制了该应用程序，这将大大增加其可能造成的损害。
- **权限过高的数据库服务账户：**托管数据库服务使用的服务账户具有该账户的管理员操作权限。如果攻击者设法访问到该数据库，他们可以利用该服务账户的高级权限，对整个云账户进行访问和操作。
- **具备广泛网络访问权限的不受限应用程序用户：**数据库应用程序使用的服务账户具有服务器的管理员权限。若攻击者利用数据库软件中的漏洞，他们可以利用该服务账户的高级权限执行任意命令、安装恶意软件或创建新的用户账户，从而导致整个系统被入侵。

## 如何预防

- **实施最小权限原则：**为每个身份仅分配其特定任务所需的最基本权限，除非绝对必要，否则避免授予任何形式的管理员权限。
- **定期审计和审查权限：**持续评估已授予特定身份的权限，以确保其严格符合必要性。审计特权身份，以检测和解决潜在的滥用权限或权限过度分配问题。
- **建立预防性保护措施：**在组织层面实施拒绝策略，禁止设置过度宽松的配置，并强制执行严格的访问控制。
- **利用即时（JIT）访问机制：**利用支持临时、按需提升权限的工具，仅在需要时和在定义的时间范围内允许高级访问。

## 参考文献


- env 文件泄露事件（2024 年 8 月） - 链接 1，链接 2
- Microsoft “午夜暴雪” 数据泄露事件（2024 年 1 月） - 链接 1，链接 2
- Microsoft SAS 令牌泄露事件（2023 年 9 月） - 链接
- CircleCI 数据泄露事件（2023 年 1 月） - 链接
- Uber 数据泄露事件（2022 年 9 月） - 链接
- Verkada 数据泄露事件（2021 年 3 月） - 链接

## 支撑数据

- Datadog《2024 年云状态报告》
  - 17.6% 的情况存在过度的数据访问，例如能够列出并访问账户中所有亚马逊 S3 存储桶的数据。
  - 10% 的集群存在危险的节点角色，该角色拥有完全的管理员访问权限，可进行权限提升，具备过度宽松的数据访问权限（如对所有 S3 存储桶的访问权），或允许在账户内所有工作负载间进行横向移动。
  - 超过三分之一（33%）的谷歌云虚拟机对项目拥有敏感权限。
- 云安全联盟（CSA）非人类身份（NHI）报告
  - 33% 的受访者将权限过高的账户列为最令人担忧的三大 NHI 威胁之一。（3/10）
  - 37% 的 NHI 相关安全事件是由权限过高的身份导致的。（2/10）
  - 22% 的组织认为管理权限是 NHI 工具最重要的功能。（5/10）
  - 26% 的组织认为超过 50%的服务账户权限过高。
- Orca Security 《2022 年云安全状态报告》
  - 44% 的环境中至少有一个特权身份访问管理（IAM）角色。
  - 23% 的环境中至少有一个 EC2 实例具有管理员 IAM 角色。

# NHI6:2025 不安全的云部署配置

| 威胁代理&攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**平均** | 普遍性：**常见** | 可检测性：**容易** | 技术：**严重** | 业务：**特定** |
| 一般来说，发现配置错误的 CI/CD 流水线是很困难的，因为它们通常是在组织内部进行设置的。一旦威胁行为者获得简单的读取访问权限，他们就可以相对轻松地侦察组织内环境并定位易受攻击的配置。 | 管理 CI/CD 流水线的风险已经引起人们的关注，因此许多 CI/CD 提供商支持并鼓励用户使用身份认证授权。然而，许多组织仍然在使用硬编码凭证或不安全的身份认证授权，因此与安全的 CI/CD 流水线管理仍然有距离。<br>CI/CD 配置错误发生在组织的“主场”，因此很容易被搜索。此外，当前已知的错误配置都已被记录在案。 | | 由于大多数流水线都授予了高访问权限，因此成功利用 CI/CD 流水线的错误配置可能导致供应链攻击或对环境的恶意访问。 | |

## 风险描述

利用持续集成和持续部署（CI/CD）应用程序，开发人员能够自动化构建、测试、部署代码至生产环境的整个过程。此类集成通常要求 CI/CD 流水线与云服务进行身份验证，这通常是通过使用静态凭证的专用服务账户或 OpenID Connect (OIDC) 进行联合身份管理来实现的。

在 CI/CD 流水线中使用静态凭证验证的专用服务账户是不安全的。静态凭证可能会被暴露于代码存储库、日志或配置文件中。静态凭证可以令攻击者绕过多重身份验证 (MFA) 和其他安全措施，为攻击者提供持久且可能具有特权的生产环境访问权限。

OIDC 提供了一种更安全的选择，它允许 CI/CD 流水线获取用于身份验证的短生命周期、动态生成的令牌。但是，OIDC 中的错误配置同样会引入漏洞。如果未正确验证身份令牌或者令牌声明没有严格的条件——例如 sub（subject）声明——未经授权的用户可能会利用这些弱点获取云资源的访问权限。

正确的 CI/CD 集成配置和管理对于维护生产环境的安全至关重要。这包括强制执行最小特权原则，实施鲁棒的凭证管理，并确保 OIDC 令牌得到适当验证并只能为授权实体所用。

## 攻击场景示例

- **AWS IAM 角色与配置不当的 OIDC 信任关系：**AWS IAM 角色中配置错误的 OIDC 信任关系指的是在使用 `AssumeRoleWithWebIdentity` 允许 OIDC 访问时，未正确限制 `sub` 声明而信任公共 OIDC 提供者（如 GitHub

或 GitLab）的情况。如果没有这个限制，这些平台上的任何用户都可能获得该角色，并导致未经授权地访问 AWS 资源。

- **Azure 服务主体凭证硬编码：**硬编码 Azure 服务主体凭据是指在 GitHub Actions 中使用的 Azure 服务主体的凭据被硬编码到流水线配置文件中。如果这些文件存储在公开访问的仓库或以其他方式暴露出来，攻击者可以获取凭据并作为该服务主体进行身份验证，从而获得与之关联权限的 Azure 资源访问权。

## 如何预防

- **使用 OIDC 进行安全认证**
  - 使用 OIDC 生成短生命周期、动态生成的令牌认证替代静态凭证认证。
  - 严格验证令牌，包括颁发者、受众和声明。
- **强制执行最小特权原则**
  - 将 CI/CD 流水线权限限制为仅必要的权限。
  - 限制 IAM 角色和 OIDC 配置的信任关系。
- **避免硬编码凭证**
  - 使用工具如 AWS Secrets Manager、Azure Key Vault 或 HashiCorp Vault 安全地存储密钥。
  - 定期扫描仓库和配置以查找暴露的凭证。

## 参考文献

- 利用配置不当的 GitLab OIDC AWS IAM 角色。
- GitHub Actions 中的 AWS 访问安全性建议。

## 支撑数据

- CSA NHI 报告
  - 32% 的时间配置错误是 NHI 相关安全事件的原因。（4/10）。

# NHI7:2025 长期有效的密钥

| 威胁代理/攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**困难** | 普遍性：**广泛** | 可检测性：**容易** | 技术：**严重** | 业务：**特定** |
| 成功利用长期有效的密钥需要威胁代理首先获得对密钥值的访问权限。因此，长期有效的密钥攻击依赖于不同的初始访问载体。 | 在现代环境中，长期有效的密钥极为常见。这是由于密钥轮换存在诸多挑战，并且临时解决方案的可用性较低。鉴于大多数密钥管理工具能让用户查看自上次轮换以来所经过的时间，因此检测长期有效密钥并不困难。 | | 机密信息往往包含具有重要影响的非人类身份（NHI）的凭证（例如 API 密钥和数据库连接字符串） | |

## 风险描述

长期有效的密钥指的是使用过期时间设置得过于遥远或永不过期的敏感非人类身份（如 API 密钥、令牌、加密密钥和证书）。开发人员经常使用这些密钥来使应用程序能够认证并与组织内的各种服务和资源进行交互。通常，这些密钥可能会被攻破或泄露（见“密钥泄露”） 。如果一个被攻破的密钥长期有效，攻击者就可以不受任何时间限制地访问敏感服务。

## 攻击场景示例

- **通过过期的敏感访问令牌进行权限提升：**在企业网中拥有低级别权限的攻击者发现了一份一年前的数据转储。这份转储包含一个具有管理员权限的敏感访问令牌。攻击者利用这个敏感访问令牌在网络中提升自身权限。
- **通过窃取长期有效的 Cookie 进行会话劫持：**一个 Web 会话 Cookie 被设置为长期有效。一场信息窃取活动从企业网络中的一台浏览器中获取了 Cookie 信息。随后，信息窃取者将该 Cookie 出售给一名攻击者，攻击者利用这个会话 Cookie 侵入了企业网络。

## 如何预防

- **启用自动密钥轮换：**使用云原生工具或简单脚本实现应用程序编程接口（API）密钥或凭证的自动轮换，这既能减少人工操作量，又能确保凭证不会长期有效。
- **采用短期有效凭证：**许多云平台，如亚马逊 AWS 和微软 Azure，都提供了内置机制，可使用临时凭证。

这些临时凭证在完成其既定任务后会自动过期并刷新。

- **采用零信任原则：**对于访问敏感资源或执行高风险操作的非人类身份（NHIs），要求其重新进行身份验证。
- **实施最小权限原则：**仅授予非人类身份（NHI）执行其任务所需的最低权限，以此降低凭证遭泄露所带来的影响。

## 参考文献


- Rabbit Inc. 应用程序编程接口（API）密钥泄露事件（2024 年 6 月）— link
- Hugging Face Space 平台密钥泄露事件披露（2024 年 5 月）— link
- 围绕 Snowflake 公司近期（2024 年 5 月）被“黑客攻击”事件的风波— link
- 员工个人的 GitHub 仓库泄露 Azure 和红帽（Red Hat）的内部机密（2024 年 5 月）— link
- 微软共享访问签名（SAS）令牌泄露事件（2023 年 9 月）— link
- CircleCI 数据泄露事件（2023 年 1 月）— link
- 微软 Azure 站点恢复服务权限提升漏洞事件（2022 年 7 月）— link


## 支撑数据

- Datadog 2024 年云状况报告
  - 46%使用亚马逊（AWS）的组织中的用户使用长期有效的控制台凭证。
  - 60%的跨云提供商密钥使用期限超过 1 年。
- 云安全联盟（CSA）NHI 报告
  - 在 45%的情况下，未进行凭证轮换是导致与非人类身份（NHI）相关安全事件的原因（1/10）。
  - 26%的组织认为密钥生命周期管理是一项非人类身份（NHI）工具最重要的功能（1/16）。
  - 51%的组织没有正式的流程来停用或撤销长期有效的应用程序编程接口（API）密钥。
- Orca Security 2022 年云安全状况报告
  - 80%的组织禁用了密钥管理系统（KMS）的轮换功能。
  - 79%的组织至少有一个访问密钥的使用期限超过 90 天。

# NHI8:2025 环境隔离

| 威胁代理&攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**平均** | 普遍性：**不常见** | 可检测性：**困难** | 技术：**中等** | 业务：**特定** |
| 成功利用未隔离的 NHI 要求威胁代理首先获得对测试环境的访问权限。测试环境相较于生产环境通常缺少保护措施。 | 在非生产与生产环境中通常不会重复使用 NHIs。由于可以使用相同 NHI 的工作负载和环境排列组合变化很大，因此很难检测到未隔离的 NHI。 | | 隔离 NHI 的影响取决于关联 NHI 的特权。鉴于关联 NHI 自然存在于生产环境中，其影响是不可忽视的。 | |

## 风险描述

环境隔离是一种基本的安全实践，在云应用部署中使用不同的环境来开发、测试、研发和生产。这种分离确保一个问题在一个环境中不会影响其他环境，特别是生产环境中真实用户和敏感数据的存在。非人类身份（NHI），如服务账户、API 密钥和角色，经常在部署过程中和应用程序生命周期中使用。然而，在多个环境中重复使用相同的 NHI 可能会引入重大的安全漏洞。如果在不太安全的测试环境中使用的 NHI 具有访问生产资源的权限，则攻击者可以利用这个 NHI 渗透到生产环境中。为了减轻这些风险，基于它们操作的环境严格隔离 NHI 至关重要。这包括：

- 为每个环境分配单独的 NHI，以防止跨环境访问。
- 应用最小特权原则：限制 NHI 的权限仅限于其特定环境所需的权限。
- 实施环境特定的访问控制：确保非生产环境中的 NHI 无法访问生产资源。
- 定期审核和监控：持续监控 NHI 是否存在任何未经授权的访问尝试或异常。

通过隔离环境及相关 NHIs，可以显著减少组织暴露的攻击面，防止传播到各个环境的潜在入侵。

## 攻击场景示例

- **共享 AWS 访问密钥跨环境：**AWS 访问密钥在测试和生产环境中都可用于访问 Amazon S3 存储桶。虽然该密钥的设置初衷在于访问测试环境中的模拟数据，然而它也同时拥有访问敏感生产数据的权限。如果测试环境受到损害，攻击者可以使用共享访问密钥获取并操纵生产数据，进而导致数据泄露或服务中断。

- **系统分配的跨订阅共享托管 Azure 标识：**在 Azure 中，系统分配的身份验证是一种身份验证方式，它允许资源在测试订阅和生产订阅之间共享。这种配置使得测试环境中的进程可以访问生产环境中的资源。然而，如果攻击者获得了测试环境的访问权限，他们就可以利用该身份验证来获得对关键资源的未经授权访问，并可能破坏整个生产环境。

## 如何预防

- **严格环境隔离 NHI：**为每个环境（开发、测试、预发布、生产）分配唯一的 NHI，以确保一个环境中的访问凭证或身份不能在另一个环境中重用。
- **应用最小特权原则 (PoLP)：对** NHI 授予最小权限，以便在其指定环境中完成任务。这样即使 NHI 被攻破，也能将潜在损失降到最低。
- **实施环境特定的访问控制：**配置访问策略，使非生产环境（例如测试）中的 NHI 无法交互或访问生产环境中的资源。
- **隔离敏感资源的基础架构：**使用独立的资源组、订阅或账户来隔离生产与非生产环境。这样即使 NHI 被攻破，其影响范围也被限制在自身环境中。

## 参考文献

- AWS 关于工作负载隔离的建议

## 支撑数据

- CSA NHI 报告
  - 32% 的时间配置错误是 NHI 相关安全事件的原因。（4/10）

# NHI9:2025 NHI 重用

| 威胁代理&攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**困难** | 普遍性：**广泛** | 可检测性：**困难** | 技术：**低** | 业务：**特定** |
| 成功利用重用的 NHI 需要威胁代理首先获得环境的访问权限。因此，NHI 重用依赖于单独的初始访问向量。 | NHIs 被非常普遍地重用，因为为每个工作负载量身定制 NHI 是困难的。常见的案例包括使用单个 AWS IAM 角色用于多个工作负载，或使用单个 API 密钥用于多个工作负载。检测 NHI 的重用是困难的，因为可以使用 NHI 的工作负载具有高度可变性。 | | NHI 重用的影响取决于与之关联的 NHI 的权限。如果采用最小权限，则这种影响为低。 | |

## 风险描述

非人类身份（NHI），如服务账户、API 密钥和机器凭证，在使应用程序和服务能够进行验证和访问必要的资源方面发挥着至关重要的作用。然而，在不同的应用程序、服务或组件之间重用相同的 NHI——即使它们是部署在一起的——会引入显著的安全风险。如果 NHI 在一个区域被泄露，攻击者可以利用它来未经授权地访问使用相同凭证的系统其他部分。

除了这些风险外，NHIs 的重用还可能使违规缓解行动复杂化和影响审计。

为了将这些风险降至最低，为每个应用程序或服务分配唯一的 NHIs 是至关重要的。这种方法遵循最小特权原则，确保每个 NHI 仅拥有执行其特定功能所需的权限。通过隔离 NHIs，组织可以控制潜在的违规行为，并防止攻击者在环境中横向移动。

## 攻击场景示例

- **Kubernetes 服务账户重用：**在 Kubernetes 集群中，多个 Pod 可以共享同一个 Kubernetes 服务账户，包括负责编排任务的关键 Pod。如果一个 Pod 存在漏洞并被攻破，攻击者可以使用共享服务账户在集群中执行操作。这可能导致对敏感数据的未经授权访问、工作负载的操纵，甚至完全控制集群的资源。
- **应用程序之间共享的 API 密钥：**一个组织使用相同的 API 密钥为多个应用程序提供第三方服务的访问权限。如果一个应用程序被攻破且 API 密钥被泄露，攻击者可以使用它访问或操纵所有使用该共享密钥的应用程序中的数据，这有可能引发大规模的数据泄露。
- **重用云凭证：**组织内的不同服务使用相同的云凭证（例如 AWS IAM 角色、Azure 服务主体）来与云资源进行交互。如果攻击者从安全性较低的服务获取了这些凭证，他们可以绕过隔离机制，访问安全性较高的服务所使用的关键资源，进而扩大攻击范围。

## 如何预防

- **为每个应用程序或服务分配唯一的 NHIs**
  - 确保每个逻辑系统组件都被赋予一个其运行所依赖的唯一 NHI。
  - 确保只有相关的系统组件能够使用所分配的 NHI 进行身份验证。
- **在每个环境中分配唯一的 NHIs**
  - 确保每个逻辑环境在与其他系统或环境交互时使用不同的 NHIs。
- **执行最小权限原则**
  - 确保 NHI 仅被授予其功能所需的最低访问权限级别。
- **审计和审查 NHIs 的使用情况**
  - 审计所有 NHIs 及其访问权限和分配的系统组件。
  - 定期地审查，确保 NHIs 不被重复使用，并且它们继续遵循其功能的最小权限原则。

## 参考文献

- Kubernetes：管理服务账户
- OWASP：最小权限原则
- AWS 身份和访问管理最佳实践
- Azure 服务主体安全
- Google Cloud：管理服务账户的最佳实践
- Chronicle 跨客户桶访问

## 支撑数据

- **云漏洞数据库**
  - Chronicle 跨客户桶访问
- **CSA NHI 报告**
  - 14%的组织需要消费者身份识别作为 NHI 工具最重要的能力。（11/16）
- **近期的违规事件**
  - .env 文件泄露-链接

# NHI10:2025 人类使用 NHI

| 威胁代理&攻击向量 | 安全脆弱性 | | 影响 | |
|---|---|---|---|---|
| 可利用性：**困难** | 普遍性：**常见** | 可检测性：**困难** | 技术：**低** | 业务：**特定** |
| 人类使用 NHI 要求威胁代理首先获得对环境的访问权限。因此，人类使用 NHI 攻击依赖于单独的初始访问向量。 | 开发人员经常冒充服务账户来调试问题。<br>大多数NHI提供商不提供工具来区分假设为NHI的工作负载和人类。 | | 人类使用NHI的影响取决于关联NHI的特权。如果采用最小化特权，则此影响为“低”。 | |

## 风险描述

非人类身份(NHIs)是指诸如服务账户、API 令牌和工作负载身份等设计用于程序访问云资源和服务的身份。它们使应用程序、服务和自动化过程能够在无需人工干预的情况下安全运行。然而，在应用开发和维护过程中，开发者或管理员可能会将这些 NHIs 误用于应该使用具有适当权限的个人身份完成的手动任务。这种做法引入了重大的安全风险：

- **超出必要范围的特权：**由于 NHIs 的设计目的是为程序提供访问权限，因此将其用于手动任务可能导致超出实际需要的特权级别，从而增加了潜在的安全漏洞。
- **缺乏详细的审计和问责制：**当使用 NHIs 执行手动任务时，可能难以跟踪和审查活动。这使得难以确定谁负责特定操作，并且在发生问题时难以追究责任。
- **无法区分人类和自动化的活动：**由于 NHIs 可以模拟人类行为，攻击者可以利用这一点来混淆人类和自动化之间的区别。这使得更难检测到恶意活动并阻止攻击。
- **攻击者的混淆：**如果攻击者能够获得 NHIs 的访问权限，则他们可以伪装成合法用户或系统，并执行未经授权的操作。这可能导致数据泄露、信息窃取和其他严重的安全威胁。

## 攻击场景示例

- **管理员使用服务账户凭证：**管理员使用服务账户凭证登录云管理控制台。这种服务账户具有广泛的权限，旨在用于自动化部署任务。但是，管理员现在可以访问超出其角色要求之外的内容，并且他们执行的所有操作都会记录在服务账户下，这会模糊责任归属。
- **开发人员使用 NHIs 执行命令：**开发人员手动运行脚本或命令时使用 NHI（非人类身份验证）。如果开发人员犯了错误或进行了未经授权的更改，则很难将活动溯源到他们，因为日志会将这些错误违规行

为归因于 NHI。

- **团队成员共享 API 令牌：**为了快速访问某些资源，团队会共享与服务账户相关的 API 令牌。该令牌具有广泛的访问权限。一旦团队中任意一名成员的环境被攻击者入侵，攻击者都可以使用共享的令牌来访问敏感系统，识别漏洞的来源将变得非常困难。
- **绕过安全控制：**员工使用 NHI 访问受 MFA 要求或 IP 地址限制等策略限制在其用户账户下的资源。这破坏了组织的安全态势并可能导致未经授权的数据访问。
- **攻击者利用 NHIs 保持持久性：**在入侵环境中后，攻击者获取 NHI 凭证并使用它们来维持访问权限。由于 NHIs 不受定期密码更改或 MFA 约束，攻击者可以长期持续地访问该环境。

## 如何预防

- **使用专用的身份：**对于调试或维护任务，请使用具有相应角色和权限的专用人类身份。
- **审核和监控 NHI 活动：**使用支持 NHI 使用情况审核和跟踪的工具或平台，以便能够检测和追究人类使用的责任。
- **使用上下文感知访问控制：**使用条件访问策略，根据可疑模式检测和阻止人类访问 NHIs。
- **教育开发者和管理员：**提供有关手动使用 NHIs 的风险以及确保安全访问实践的替代方法的培训。

## 参考文献

- 微软 Azure：保护服务账户的最佳实践
- AWS 安全博客：管理 AWS 访问密钥的最佳实践
- 谷歌云：管理服务账号密钥的最佳实践

## 支撑数据

- Anetac 报告
  - 75%的企业滥用服务账户，导致关键安全风险
- CSA NHI 报告
  - 32%的企业认为服务账户是最难管理的。（1/16）
  - 26%的企业相信超过 50%的服务账户是过度授权的。