



Red Hat Enterprise Linux 9

在身份管理中使用外部的红帽实用程序

在 IdM 中集成服务和红帽产品

Red Hat Enterprise Linux 9 在身份管理中使用外部的红帽实用程序

在 IdM 中集成服务和红帽产品

法律通告

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux[®] is the registered trademark of Linus Torvalds in the United States and other countries.

Java[®] is a registered trademark of Oracle and/or its affiliates.

XFS[®] is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL[®] is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js[®] is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack[®] Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

摘要

管理员可以在红帽身份管理(IdM)域中集成服务和红帽产品。这包括 Samba、Ansible 和自动挂载等服务, 以及 OpenShift Container Platform、OpenStack 和 Satellite 等产品。然后, IdM 用户可以访问这些服务和产品。

目录

对红帽文档提供反馈	3
第 1 章 IDM 与红帽产品集成	4
第 2 章 使用外部身份提供程序向 IDM 进行身份验证	5
2.1. 将 IDM 连接到外部 IDP 的好处	5
2.2. IDM 如何通过外部 IDP 融合登录	5
2.3. 创建对外部身份提供程序的引用	6
2.4. IDM 中不同外部 IDP 的引用示例	7
2.5. 在 IDM 中管理外部身份提供程序的 IPA IDP114 命令的选项	8
2.6. 管理对外部 IDP 的引用	9
2.7. 启用 IDM 用户通过外部 IDP 进行身份验证	10
2.8. 以外部 IDP 用户身份检索 IDM TICKET-GRANTING TICKET	11
2.9. 以外部 IDP 用户身份通过 SSH 登录到 IDM 客户端	12
2.10. IPA IDP114 命令中的 --PROVIDER 选项	13
第 3 章 在 IDM 域成员中设置 SAMBA	17
3.1. 准备 IDM 域以便在域成员中安装 SAMBA	17
3.2. 在 IDM 客户端中安装和配置 SAMBA 服务器	18
3.3. 如果 IDM 信任新域，请手动添加 ID 映射配置	20
3.4. 其他资源	21
第 4 章 从 NIS 迁移到身份管理	22
4.1. 在 IDM 中启用 NIS	22
4.2. 将用户条目从 NIS 迁移到 IDM	22
4.3. 将用户组从 NIS 迁移到 IDM	24
4.4. 将主机条目从 NIS 迁移到 IDM	24
4.5. 将网络组条目从 NIS 迁移到 IDM	26
4.6. 将自动挂载映射从 NIS 迁移到 IDM	27
第 5 章 在 IDM 中使用自动挂载	29
5.1. IDM 中的 AUTOFS 和自动挂载	29
5.2. 在 RED HAT IDENTITY MANAGEMENT 域中使用 KERBEROS 建立一个 NFS 服务器	30
5.3. 使用 IDM CLI 在 IDM 中配置自动挂载位置和映射	31
5.4. 在 IDM 客户端上配置自动挂载	32
5.5. 验证 IDM 用户能否访问 IDM 客户端上的 NFS 共享	32
第 6 章 使用 ANSIBLE 为 IDM 用户自动挂载 NFS 共享	35
6.1. IDM 中的 AUTOFS 和自动挂载	35
6.2. 在 RED HAT IDENTITY MANAGEMENT 域中使用 KERBEROS 建立一个 NFS 服务器	36
6.3. 使用 ANSIBLE 在 IDM 中配置自动挂载位置、映射和密钥	37
6.4. 使用 ANSIBLE 将 IDM 用户添加到拥有 NFS 共享的组中	39
6.5. 在 IDM 客户端上配置自动挂载	41
6.6. 验证 IDM 用户能否访问 IDM 客户端上的 NFS 共享	41

对红帽文档提供反馈

我们感谢您对我们文档的反馈。帮助我们如何进行改进。

通过 Jira 提交反馈（需要帐户）

1. 登录到 [Jira](#) 网站。
2. 在顶部导航栏中点 **Create**
3. 在 **Summary** 字段中输入描述性标题。
4. 在 **Description** 字段中输入您的建议以改进。包括文档相关部分的链接。
5. 点对话框底部的 **Create**。

第 1 章 IDM 与红帽产品集成

查找与 IdM 集成的其他红帽产品的文档。您可以配置这些产品，以允许 IdM 用户可以访问它们的服务。

- [Ansible Automation Platform](#)
- [OpenShift Container Platform](#)
- [Red Hat OpenStack Platform](#)
- [Red Hat Satellite](#)
- [红帽单点登录](#)
- [Red Hat Virtualization](#)

第 2 章 使用外部身份提供程序向 IDM 进行身份验证

您可以将用户与支持 OAuth 2 设备授权流的外部身份提供者(IdP)关联。当这些用户使用 RHEL 9.1 或更高版本中提供的 SSSD 版本进行身份验证时，它们会在外部 IdP 执行身份验证和授权后获得带有 Kerberos 票据的 RHEL 身份管理(IdM)单点登录功能。

主要特性包括：

- 使用 **ipa idp-*** 命令添加、修改和删除对外部 IdP 的引用。
- 使用 **ipa user-mod --user-auth-type=idp** 命令为用户启用 IdP 身份验证。

2.1. 将 IDM 连接到外部 IDP 的好处

作为管理员，您可能想要允许存储在外部身份源（如云服务供应商）中的用户访问连接到 Identity Management (IdM) 环境的 RHEL 系统。要达到此目的，您可以将这些用户的 Kerberos 票据的身份验证和授权过程委托给该外部实体。

您可以使用此功能扩展 IdM 的功能，并允许存储在外部身份提供程序 (IdP) 中的用户访问由 IdM 管理的 Linux 系统。

2.2. IDM 如何通过外部 IDP 融合登录

SSSD 2.7.0 包含 **sssd-idp** 软件包，该软件包可实施 **idp** Kerberos pre-authentication 方法。这个验证方法遵循 OAuth 2.0 设备授权流，将授权决策委派给外部 IdP：

1. IdM 客户端用户启动 OAuth 2.0 设备授权流，例如，通过在命令行中使用 **kinit** 实用程序检索 Kerberos TGT。
2. 一个特殊的代码和网站链接从授权服务器发送到 IdM KDC 后端。
3. IdM 客户端显示用户的链接和代码。在本例中，IdM 客户端会在命令行上输出链接和代码。
4. 用户在浏览器中打开网站链接，可以在另一个主机上、移动电话等：
 - a. 用户输入特殊代码。
 - b. 如有必要，用户登录到基于 OAuth 2.0 的 IdP。
 - c. 系统将提示用户授权客户端访问信息。
5. 用户在原始设备提示符处确认访问。在这个示例中，用户在命令行中点 **Enter** 键。
6. IdM KDC 后端轮询 OAuth 2.0 授权服务器以访问用户信息。

支持什么：

- 启用了 **键盘互动** 验证方法通过 SSH 远程登录，它允许调用可插拔式身份验证模块 (PAM) 库。
- 通过 **logind** 服务，使用控制台本地登录。
- 使用 **kinit** 实用程序检索 Kerberos ticket-granting ticket (TGT)。

当前不支持什么：

- 直接登录到 IdM WebUI。要登录到 IdM WebUI，您必须首先获取一个 Kerberos ticket。
- 直接登录 Cockpit WebUI。要登录 Cockpit Web UI，您必须首先获取一个 Kerberos ticket。

其他资源

- [对外部身份提供程序进行身份验证](#)
- [RFC 8628 : OAuth 2.0 设备授权](#)

2.3. 创建对外部身份提供程序的引用

要将外部身份提供程序(IdP)连接到您的身份管理(IdM)环境，请在 IdM 中创建 IdP 参考。完成此流程，根据 Keycloak 模板创建一个名为 **my-keycloak-idp** 的引用。如需了解更多引用模板，请参阅 [IdM 中对不同外部 IdP 的引用](#)。

先决条件

- 您已将 IdM 作为 OAuth 应用程序注册到外部 IdP，并获取了客户端 ID。
- 您可以作为 IdM admin 帐户进行身份验证。
- 您的 IdM 服务器使用 RHEL 9.1 或更高版本。
- 您的 IdM 服务器使用 SSSD 2.7.0 或更高版本。

流程

1. 在 IdM 服务器中作为 IdM 管理员进行身份验证。

```
[root@server ~]# kinit admin
```

2. 根据 Keycloak 模板，创建一个名为 **my-keycloak-idp** 的引用，其中 **--base-url** 选项指定 Keycloak 服务器的 URL，格式为 **server-name.\$DOMAIN:\$PORT/prefix**。

```
[root@server ~]# ipa idp-add my-keycloak-idp \
    --provider keycloak --organization main \
    --base-url keycloak.idm.example.com:8443/auth \
    --client-id id13778

-----
Added Identity Provider reference "my-keycloak-idp"
-----

Identity Provider reference name: my-keycloak-idp
Authorization URI:
https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-connect/auth
Device authorization URI:
https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-
connect/auth/device
Token URI: https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-
connect/token
User info URI: https://keycloak.idm.example.com:8443/auth/realms/main/protocol/openid-
connect/userinfo
Client identifier: ipa_oidc_client
Scope: openid email
External IdP user identifier attribute: email
```

验证

- 验证 `ipa idp-show` 命令的输出显示您创建的 IdP 引用。

```
[root@server ~]# ipa idp-show my-keycloak-idp
```


其他资源

- [IdM 中不同外部 IdP 的引用示例](#)
- [在 IdM 中管理外部身份提供程序的 ipa idp114 命令的选项](#)
- [ipa idp114 命令中的 --provider 选项](#)
- `ipa help idp-add`

2.4. IDM 中不同外部 IDP 的引用示例

下表列出了 `ipa idp-add` 命令示例，用于在 IdM 中创建对不同 IdP 的引用。

身份供应商	重要选项	命令示例
Microsoft Identity Platform, Azure AD	--provider microsoft --organization	<pre># ipa idp-add my-azure-idp \ --provider microsoft \ --organization main \ --client-id <azure_client_id></pre>
Google	--provider google	<pre># ipa idp-add my-google-idp \ --provider google \ --client-id <google_client_id></pre>
GitHub	--provider github	<pre># ipa idp-add my-github-idp \ --provider github \ --client-id <github_client_id></pre>

身份供应商	重要选项	命令示例
Keycloak, Red Hat Single Sign-On	--provider keycloak --organization --base-url	<pre># ipa idp-add my-keycloak-idp \ --provider keycloak \ --organization main \ --base-url keycloak.idm.example.com:8443/auth \ --client-id <keycloak_client_id></pre> <div>注意 Keycloak 17 及更新版本的 Quarkus 版本已删除 URI 的 /auth/ 部分。如果您在部署中使用 Keycloak 的非 Quarkus 分发，请在 --base-url 选项中包含 /auth/。</div>
Okta	--provider okta	<pre># ipa idp-add my-okta-idp \ --provider okta --base-url dev-12345.okta.com \ --client-id <okta_client_id></pre>

其他资源

- [创建对外部身份提供程序的引用](#)
- [在 IdM 中管理外部身份提供程序的 ipa idp114 命令的选项](#)
- [ipa idp114 命令中的 --provider 选项](#)

2.5. 在 IDM 中管理外部身份提供程序的 IPA IDP114 命令的选项

以下示例演示了如何根据不同的 IdP 模板配置对外部 IdP 的引用。使用以下选项指定设置：

--provider

其中一个已知的身份提供程序的预定义模板

--client-id

IdP 在应用程序注册期间发布的 OAuth 2.0 客户端标识符。当应用程序注册步骤特定于每个 IdP 时，请参考它们的文档了解详情。如果外部 IdP 是红帽单点登录(SSO)，请参阅 [创建 OpenID Connect 客户端](#)。

--base-url

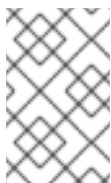
Keycloak 和 Okta 所需的 IdP 模板的基本 URL

--organization

Microsoft Azure 需要的 IdP 中的域或机构 ID

--secret

(可选) 如果您已将外部 IdP 配置为需要来自机密 OAuth 2.0 客户端的 secret，请使用此选项。如果您在创建 IdP 引用时使用这个选项，则会以交互方式会提示您输入 secret。将客户端 secret 作为密码保护。



注意

RHEL 9.1 中的 SSSD 只支持不使用客户端 secret 的非机密 OAuth 2.0 客户端。如果要使用需要机密客户端 secret 的外部 IdP，您必须在 RHEL 9.2 及之后的版本中使用 SSSD。

其他资源

- [创建对外部身份提供程序的引用](#)
- [IdM 中不同外部 IdP 的引用示例](#)
- [ipa idp114 命令中的 --provider 选项](#)

2.6. 管理对外部 IDP 的引用

创建对外部身份提供程序 (IdP) 的引用后，您可以找到、显示、修改和删除该引用。本例演示了如何管理对名为 **keycloak-server1** 的外部 IdP 的引用。

先决条件

- 您可以作为 IdM admin 帐户进行身份验证。
- 您的 IdM 服务器使用 RHEL 9.1 或更高版本。
- 您的 IdM 服务器使用 SSSD 2.7.0 或更高版本。
- 您已在 IdM 中创建了对外部 IdP 的引用。请参阅[创建对外部身份提供程序的引用](#)。

步骤

1. 在 IdM 服务器中作为 IdM 管理员进行身份验证。

```
[root@server ~]# kinit admin
```

2. 管理 IdP 参考。

- 查找 IdP 参考，其条目包括字符串 **keycloak**：

```
[root@server ~]# ipa idp-find keycloak
```

- 显示名为 **my-keycloak-idp** 的 IdP 参考：

```
[root@server ~]# ipa idp-show my-keycloak-idp
```

- 要修改 IdP 参考，请使用 **ipa idp-mod** 命令。例如，要更改名为 **my-keycloak-idp** 的 IdP 参考的 secret，请指定要提示输入 secret 的 **--secret** 选项：

```
[root@server ~]# ipa idp-mod my-keycloak-idp --secret
```

- 删除名为 **my-keycloak-idp** 的 IdP 参考：

```
[root@server ~]# ipa idp-del my-keycloak-idp
```

2.7. 启用 IDM 用户通过外部 IDP 进行身份验证

要启用 IdM 用户通过外部身份提供程序 (IdP)，将之前创建的外部 IdP 引用与用户帐户关联。这个示例将外部 IdP 参考 **keycloak-server1** 与用户 **idm-user-with-external-idp** 关联。

先决条件

- 您的 IdM 客户端和服务端使用 RHEL 9.1 或更高版本。
- 您的 IdM 客户端和服务端使用 SSSD 2.7.0 或更高版本。
- 您已在 IdM 中创建了对外部 IdP 的引用。请参阅[创建对外部身份提供程序的引用](#)。

步骤

- 修改 IdM 用户条目，将 IdP 引用与用户帐户关联：

```
[root@server ~]# ipa user-mod idm-user-with-external-idp \
    --idp my-keycloak-idp \
    --idp-user-id idm-user-with-external-idp@idm.example.com \
    --user-auth-type=idp
-----
Modified user "idm-user-with-external-idp"
-----
User login: idm-user-with-external-idp
First name: Test
Last name: User1
Home directory: /home/idm-user-with-external-idp
Login shell: /bin/sh
Principal name: idm-user-with-external-idp@idm.example.com
Principal alias: idm-user-with-external-idp@idm.example.com
Email address: idm-user-with-external-idp@idm.example.com
UID: 35000003
GID: 35000003
User authentication types: idp
External IdP configuration: keycloak
External IdP user identifier: idm-user-with-external-idp@idm.example.com
Account disabled: False
Password: False
Member of groups: ipausers
Kerberos keys available: False
```

验证

- 验证该用户的 **ipa user-show** 命令的输出是否显示对 IdP 的引用：

```
[root@server ~]# ipa user-show idm-user-with-external-idp
User login: idm-user-with-external-idp
First name: Test
Last name: User1
Home directory: /home/idm-user-with-external-idp
Login shell: /bin/sh
Principal name: idm-user-with-external-idp@idm.example.com
Principal alias: idm-user-with-external-idp@idm.example.com
Email address: idm-user-with-external-idp@idm.example.com
ID: 35000003
GID: 35000003
User authentication types: idp
External IdP configuration: keycloak
External IdP user identifier: idm-user-with-external-idp@idm.example.com
Account disabled: False
Password: False
Member of groups: ipausers
Kerberos keys available: False
```

2.8. 以外部 IDP 用户身份检索 IDM TICKET-GRANTING TICKET

如果您已将身份管理(IdM)用户的身份验证委派给外部身份提供程序(IdP)，IdM 用户可以通过向外部 IdP 进行身份验证来请求 Kerberos 票据授予票据(TGT)。

完成这个流程以：

1. 在本地检索和存储匿名 Kerberos 票据。
2. 使用带有 **-T** 选项的 **kinit** 和 Secure Tunneling (FAST)频道在 **idm-user-with-external-idp** 用户请求 TGT，以便在 Kerberos 客户端和 Kerberos 分发中心(KDC)之间提供灵活的身份验证。

先决条件

- 您的 IdM 客户端和服务端使用 RHEL 9.1 或更高版本。
- 您的 IdM 客户端和服务端使用 SSSD 2.7.0 或更高版本。
- 您已在 IdM 中创建了对外部 IdP 的引用。请参阅[创建对外部身份提供程序的引用](#)。
- 您已与用户帐户关联了一个外部 IdP 参考。请参阅[启用 IdM 用户以通过外部 IdP 进行身份验证](#)。
- 您最初以身份登录的用户对本地文件系统中的目录具有写入权限。

步骤

1. 使用 Anonymous PKINIT 获取 Kerberos 票据，并将其存储在名为 **./fast.ccache** 的文件中。

```
$ kinit -n -c ./fast.ccache
```

2. [可选] 查看检索到的票据：

```
$ *klist -c fast.ccache *
Ticket cache: FILE:fast.ccache
Default principal: WELLKNOWN/ANONYMOUS@WELLKNOWN:ANONYMOUS
```

```
Valid starting    Expires          Service principal
03/03/2024 13:36:37 03/04/2024 13:14:28
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
```

3. 开始以 IdM 用户身份进行身份验证，使用 **-T** 选项启用 FAST 通信频道。

```
[root@client ~]# kinit -T ./fast.ccache idm-user-with-external-idp
Authenticate at https://oauth2.idp.com:8443/auth/realms/master/device?user_code=YHMQ-
XKTL and press ENTER.:
```

4. 在浏览器中，以命令输出中提供的网站的用户身份进行身份验证。
5. 在命令行中，按 **Enter** 键来完成身份验证过程。

验证

- 显示您的 Kerberos ticket 信息，并确认对于带有外部 IdP 的预身份验证的行 **config: pa_type** 显示 **152**。

```
[root@client ~]# klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting    Expires          Service principal
05/09/22 07:48:23 05/10/22 07:03:07 krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45 08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

pa_type = 152 表示外部 IdP 身份验证。

2.9. 以外部 IDP 用户身份通过 SSH 登录到 IDM 客户端

要通过 SSH 作为外部身份提供程序 (IdP) 用户身份登录 IdM 客户端，请在命令行中开始登录过程。出现提示时，在与 IdP 关联的网站上执行身份验证过程，并在 Identity Management (IdM) 客户端上完成该过程。

先决条件

- 您的 IdM 客户端和服务器使用 RHEL 9.1 或更高版本。
- 您的 IdM 客户端和服务器使用 SSSD 2.7.0 或更高版本。
- 您已在 IdM 中创建了对外部 IdP 的引用。请参阅[创建对外部身份提供程序的引用](#)。
- 您已与用户帐户关联了一个外部 IdP 参考。请参阅[启用 IdM 用户以通过外部 IdP 进行身份验证](#)。

步骤

1. 尝试通过 SSH 登录到 IdM 客户端。

```
[user@client ~]$ ssh idm-user-with-external-idp@client.idm.example.com
```


(idm-user-with-external-idp@client.idm.example.com) Authenticate at https://oauth2.idp.com:8443/auth/realms/main/device?user_code=XYFL-ROYR and press ENTER.

- 2. 在浏览器中，以命令输出中提供的网站的用户身份进行身份验证。
- 3. 在命令行中，按 **Enter** 键来完成身份验证过程。

验证

- 显示您的 Kerberos ticket 信息，并确认对于带有外部 IdP 的预身份验证的行 **config: pa_type** 显示 **152**。

```
[idm-user-with-external-idp@client ~]$ klist -C
Ticket cache: KCM:0:58420
Default principal: idm-user-with-external-idp@IDM.EXAMPLE.COM

Valid starting    Expires          Service principal
05/09/22 07:48:23 05/10/22 07:03:07 krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: fast_avail(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = yes
08/17/2022 20:22:45 08/18/2022 20:22:43
krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM
config: pa_type(krbtgt/IDM.EXAMPLE.COM@IDM.EXAMPLE.COM) = 152
```

2.10. IPA IDP114 命令中的 --PROVIDER 选项

以下身份提供程序 (IdP) 支持 OAuth 2.0 设备授权流：

- Microsoft Identity Platform，包括 Azure AD
- Google
- GitHub
- Keycloak，包括 Red Hat Single Sign-On (SSO)
- Okta

当使用 **ipa idp-add** 命令创建对其中一个外部 IdP 的引用时，您可以使用 **--provider** 选项指定 IdP 类型，它扩展至额外的选项，如下所述：

--provider=microsoft

Microsoft Azure IdP 允许基于 Azure 租户 ID 进行半虚拟化 ID，您可以使用 **--organization** 选项指定 **ipa idp-add** 命令。如果您需要对 live.com IdP 的支持，请指定 **--organization common** 的选项。选择 **--provider=microsoft** 扩展以使用以下选项：**--organization** 选项的值替换了表中的字符串 **\${ipaidporg}**。

选项	值
--auth-uri=URI	https://login.microsoftonline.com/\${ipaidporg}/oauth2/v2.0/authorize

选项	值
--dev-auth-uri=URI	https://login.microsoftonline.com/\${ipaidporg}/oauth2/v2.0/devicecode
--token-uri=URI	https://login.microsoftonline.com/\${ipaidporg}/oauth2/v2.0/token
--userinfo-uri=URI	https://graph.microsoft.com/oidc/userinfo
--keys-uri=URI	https://login.microsoftonline.com/common/discovery/v2.0/keys
--scope=STR	openid email
--idp-user-id=STR	email

--provider=google

选择 **--provider=google** 扩展以使用以下选项：

选项	值
--auth-uri=URI	https://accounts.google.com/o/oauth2/auth
--dev-auth-uri=URI	https://oauth2.googleapis.com/device/code
--token-uri=URI	https://oauth2.googleapis.com/token
--userinfo-uri=URI	https://openidconnect.googleapis.com/v1/userinfo
--keys-uri=URI	https://www.googleapis.com/oauth2/v3/certs
--scope=STR	openid email
--idp-user-id=STR	email

--provider=github

选择 **--provider=github** 展开以使用以下选项：

选项	值
--auth-uri=URI	https://github.com/login/oauth/authorize
--dev-auth-uri=URI	https://github.com/login/device/code
--token-uri=URI	https://github.com/login/oauth/access_token

选项	值
--userinfo-uri=URI	https://openidconnect.googleapis.com/v1/userinfo
--keys-uri=URI	https://api.github.com/user
--scope=STR	user
--idp-user-id=STR	login

--provider=keycloak

使用 Keycloak 时，您可以定义多个域或机构。由于它是自定义部署的一部分，基本 URL 和域 ID 都是必需的，因此您可以使用 **--base-url** 和 **--organization** 选项指定它们到 **ipa idp-add** 命令：

```
[root@client ~]# ipa idp-add MySSO --provider keycloak \
--org main --base-url keycloak.domain.com:8443/auth \
--client-id <your-client-id>
```

选择 **--provider=keycloak** 扩展以使用以下选项：您在 **--base-url** 选项中指定的值替换表中的字符串 **\${ipaidpbaseurl}**，而您为 **--organization** 指定的选项替换字符串 **`\${ipaidporg}**。

选项	值
--auth-uri=URI	https://\${ipaidpbaseurl}/realms/\${ipaidporg}/protocol/openid-connect/auth
--dev-auth-uri=URI	https://\${ipaidpbaseurl}/realms/\${ipaidporg}/protocol/openid-connect/auth/device
--token-uri=URI	https://\${ipaidpbaseurl}/realms/\${ipaidporg}/protocol/openid-connect/token
--userinfo-uri=URI	https://\${ipaidpbaseurl}/realms/\${ipaidporg}/protocol/openid-connect/userinfo
--scope=STR	openid email
--idp-user-id=STR	email

--provider=okta

在注册一个 Okta 中的新机构后，会关联一个新的基本 URL。您可以使用 **ipa idp-add** 命令的 **--base-url** 选项指定这个基本 URL：

```
[root@client ~]# ipa idp-add MyOkta --provider okta --base-url dev-12345.okta.com --client-id
<your-client-id>
```

选择 **--provider=okta** 扩展以使用以下选项：您为 **--base-url** 选项指定的值替换了表中字符串 **\${ipaidpbaseurl}**。

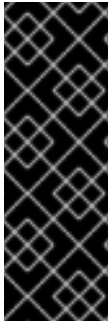
选项	值
--auth-uri=URI	https://\${ipaidpbaseurl}/oauth2/v1/authorize
--dev-auth-uri=URI	https://\${ipaidpbaseurl}/oauth2/v1/device/authorize
--token-uri=URI	https://\${ipaidpbaseurl}/oauth2/v1/token
--userinfo-uri=URI	https://\${ipaidpbaseurl}/oauth2/v1/userinfo
--scope=STR	openid email
--idp-user-id=STR	email

其他资源

- [预填充的 IdP 模板](#)

第 3 章 在 IDM 域成员中设置 SAMBA

您可以在加入到 Red Hat Identity Management (IdM)域的主机上设置 Samba。来自 IdM 的用户，以及来自受信任的 Active Directory(AD)域的用户(如果有的话)可以访问 Samba 提供的共享和打印机服务。



重要

对 IdM 域成员使用 Samba 是一种不受支持的技术预览特性，且包含了某些限制。例如，IdM 信任控制器不支持 Active Directory 全局目录服务，它们不支持使用分布式计算环境/远程过程调用 (DCE/RPC) 协议解析 IdM 组。因此，AD 用户只能在登录到其他 IdM 客户端时访问托管在 IdM 客户端中的 Samba 共享和打印机；登录到 Windows 机器的 AD 用户无法访问托管在 IdM 域成员中的 Samba 共享。

我们鼓励在 IdM 域成员中部署 Samba 的用户向红帽提供反馈意见。

如果 AD 域中的用户需要访问 Samba 提供的共享和打印机服务，请确保在 AD 中启用了 AES 加密类型。如需更多信息，请参阅使用 [GPO 在活动目录中启用 AES 加密类型](#)。

先决条件

- 主机作为 IdM 域的客户端加入。
- IdM 服务器和客户端必须在 RHEL 9.0 或更高版本中运行。

3.1. 准备 IDM 域以便在域成员中安装 SAMBA

在 IdM 客户端上设置 Samba 之前，必须在 IdM 服务器上使用 **ipa-adtrust-install** 工具来准备 IdM 域。



注意

运行 **ipa-adtrust-install** 命令的任何系统都会自动成为 AD 信任控制器。但是，您必须在 IdM 服务器上只运行一次 **ipa-adtrust-install**。

先决条件

- IdM 服务器已安装。
- 您需要 root 权限才能安装软件包并重新启动 IdM 服务。

步骤

1. 安装所需的软件包：

```
[root@ipaserver ~]# dnf install ipa-server-trust-ad samba-client
```

2. 以 IdM 管理用户身份进行身份验证：

```
[root@ipaserver ~]# kinit admin
```

3. 运行 **ipa-adtrust-install** 工具：

```
[root@ipaserver ~]# ipa-adtrust-install
```

如果 IdM 安装了集成的 DNS 服务器，则会自动创建 DNS 服务记录。

如果您在没有集成 DNS 服务器的情况下安装了 IdM，**ipa-adtrust-install** 会打印一个服务记录列表，您必须手动将它们添加到 DNS，然后才能继续操作。

4. 该脚本提示您 **/etc/samba/smb.conf** 已存在，并将被重写：

```
WARNING: The smb.conf already exists. Running ipa-adtrust-install will break your existing Samba configuration.
```

```
Do you wish to continue? [no]: yes
```

5. 该脚本提示您配置 **slapi-nis** 插件，这是一个兼容插件，允许旧的 Linux 客户端与受信任的用户一起工作：

```
Do you want to enable support for trusted domains in Schema Compatibility plugin?
This will allow clients older than SSSD 1.9 and non-Linux clients to work with trusted users.
```

```
Enable trusted domains support in slapi-nis? [no]: yes
```

6. 系统会提示您运行 SID 生成任务，以便为任何现有用户创建 SID：

```
Do you want to run the ipa-sidgen task? [no]: yes
```

这是一个资源密集型任务，因此如果您有大量的用户，您可以在其他时间运行此操作。

7. **(可选)** 默认情况下，对于 Windows Server 2008 及更高版本，动态 RPC 端口范围定义为 **49152-65535**。如果需要为您的环境定义一个不同的动态 RPC 端口范围，请将 Samba 配置为使用不同的端口，并在防火墙设置中开放这些端口。以下示例将端口范围设置为 **55000-65000**。

```
[root@ipaserver ~]# net conf setparm global 'rpc server dynamic port range' 55000-65000
```

```
[root@ipaserver ~]# firewall-cmd --add-port=55000-65000/tcp
```

```
[root@ipaserver ~]# firewall-cmd --runtime-to-permanent
```

8. 重启 **ipa** 服务：

```
[root@ipaserver ~]# ipactl restart
```

9. 使用 **smbclient** 工具来验证 Samba 是否响应 IdM 端的 Kerberos 身份验证：

```
[root@ipaserver ~]# smbclient -L ipaserver.idm.example.com -U user_name --use-kerberos=required
```

```
lp_load_ex: changing to config backend registry
```

```
Sharename      Type      Comment
```

```
-----
```

```
IPC$           IPC       IPC Service (Samba 4.15.2)
```

```
...
```

3.2. 在 IDM 客户端中安装和配置 SAMBA 服务器

您可以在 IdM 域注册的客户端上安装和配置 Samba。

先决条件

- IdM 服务器和客户端必须在 RHEL 9.0 或更高版本中运行。
- 已准备好 IdM 域，如 [为在域成员上安装 Samba 准备 IdM 域](#) 中所述。
- 如果 IdM 具有配置了 AD 的信任，请为 Kerberos 启用 AES 加密类型。例如，使用组策略对象 (GPO) 来启用 AES 加密类型。详情请参阅使用 [GPO 在活动目录中启用 AES 加密](#)。

流程

1. 安装 **ipa-client-samba** 软件包：

```
[root@idm_client]# dnf install ipa-client-samba
```

2. 使用 **ipa-client-samba** 工具准备客户端并创建初始 Samba 配置：

```
[root@idm_client]# ipa-client-samba
Searching for IPA server...
IPA server: DNS discovery
Chosen IPA master: idm_server.idm.example.com
SMB principal to be created: cifs/idm_client.idm.example.com@IDM.EXAMPLE.COM
NetBIOS name to be used: IDM_CLIENT
Discovered domains to use:

Domain name: idm.example.com
NetBIOS name: IDM
    SID: S-1-5-21-525930803-952335037-206501584
    ID range: 212000000 - 212199999

Domain name: ad.example.com
NetBIOS name: AD
    SID: None
    ID range: 1918400000 - 1918599999

Continue to configure the system with these values? [no]: yes
Samba domain member is configured. Please check configuration at /etc/samba/smb.conf
and start smb and winbind services
```

3. 默认情况下，**ipa-client-samba** 会自动将 **[homes]** 部分添加到 **/etc/samba/smb.conf** 文件中，该文件在用户连接时动态共享用户的主目录。如果用户在这个服务器上没有主目录，或者您不想共享主目录，请从 **/etc/samba/smb.conf** 中删除以下行：

```
[homes]
    read only = no
```

4. 共享目录和打印机。详情请查看以下部分：

- [设置使用 POSIX ACL 的 Samba 文件共享](#)
- [设置使用 Windows ACL 的共享](#)
- [将 Samba 设置为打印服务器](#)

5. 在本地防火墙中打开 Samba 客户端所需的端口：

```
[root@idm_client]# firewall-cmd --permanent --add-service=samba-client
[root@idm_client]# firewall-cmd --reload
```

6. 启用并启动**smb**和**winbind**服务：

```
[root@idm_client]# systemctl enable --now smb winbind
```

验证步骤

在安装了 **samba-client** 软件包的不同的 IdM 域成员上运行以下验证步骤：

- 使用 Kerberos 身份验证列出 Samba 服务器中的共享：

```
$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry

Sharename      Type      Comment
-----
example        Disk
IPC$           IPC       IPC Service (Samba 4.15.2)
...
```

其他资源

- **ipa-client-samba(1)** man page

3.3. 如果 IDM 信任新域，请手动添加 ID 映射配置

Samba 需要一个 ID 映射配置，用户可从该域访问资源。在 IdM 客户端上运行的现有 Samba 服务器上，在管理员向 Active Directory(AD)域添加了新的信任后，您必须手动添加 ID 映射配置。

先决条件

- 您在 IdM 客户端中配置了 Samba。之后，IdM 增加了一个新的信任。
- 在可信 AD 域中必须禁用 Kerberos 的 DES 和 RC4 加密类型。为了安全起见，RHEL 9 不支持这些弱加密类型。

步骤

1. 使用主机的 keytab 进行身份验证：

```
[root@idm_client]# kinit -k
```

2. 使用**ipa idrange-find**命令来显示新域的基本 ID 和 ID 范围大小。例如，以下命令显示了**ad.example.com**域的值：

```
[root@idm_client]# ipa idrange-find --name="AD.EXAMPLE.COM_id_range" --raw
-----
1 range matched
-----
cn: AD.EXAMPLE.COM_id_range
ipabaseid: 1918400000
```



```

ipairangesize: 200000
ipabaserid: 0
ipanttrusteddomainsid: S-1-5-21-968346183-862388825-1738313271
iparangetype: ipa-ad-trust
-----

```

```

Number of entries returned 1
-----

```

在后续步骤中，您需要 **ipabaserid** 和 **ipairangesize** 属性的值。

3. 要计算可用最高的 ID，请使用以下公式：

```

maximum_range = ipabaserid + ipairangesize - 1

```

使用上一步中的值，**ad.example.com** 域的最大可用 ID 是 **1918599999** (1918400000 + 200000 - 1)。

4. 编辑 **/etc/samba/smb.conf** 文件，并将域的 ID 映射配置添加到 **[global]** 部分：

```

idmap config AD : range = 1918400000 - 1918599999
idmap config AD : backend = sss

```

将 **ipabaserid** 属性的值指定为最小值，将上一步中的计算值指定为该范围的最大值。

5. 重启 **smb** 和 **winbind** 服务：

```

[root@idm_client]# systemctl restart smb winbind

```

验证步骤

- 使用 Kerberos 身份验证列出 Samba 服务器中的共享：

```

$ smbclient -L idm_client.idm.example.com -U user_name --use-kerberos=required
lp_load_ex: changing to config backend registry

```

Sharename	Type	Comment
example	Disk	
IPC\$	IPC	IPC Service (Samba 4.15.2)
...		

3.4. 其他资源

- [安装身份管理客户端](#)

第 4 章 从 NIS 迁移到身份管理

网络信息服务 (NIS) 服务器可以包含关于用户、组、主机、网络用户组和自动挂载映射的信息。作为系统管理员，您可以将这些条目类型、验证和授权从 NIS 服务器迁移到 Identity Management (IdM) 服务器，以便在 IdM 服务器中执行所有用户管理操作。从 NIS 迁移到 IdM 还可让您使用更安全的协议，比如 Kerberos。

4.1. 在 IDM 中启用 NIS

要允许 NIS 和 Identity Management (IdM) 服务器间的通信，您必须在 IdM 服务器中启用 NIS 兼容性选项。

先决条件

- 在 IdM 服务器中具有 root 访问权限。

步骤

- 在 IdM 服务器中启用 NIS 侦听程序和兼容性插件：

```
[root@ipaserver ~]# ipa-nis-manage enable
[root@ipaserver ~]# ipa-compat-manage enable
```

- 可选：**对于更严格的防火墙配置，请设置一个固定端口。
例如，将端口设置为未使用的端口 **514**：

```
[root@ipaserver ~]# ldapmodify -x -D 'cn=directory manager' -W
dn: cn=NIS Server,cn=plugins,cn=config
changetype: modify
add: nsslapd-pluginarg0
nsslapd-pluginarg0: 514
```



警告

为了避免与其他服务冲突，不要使用任何超过 1024 的端口号。

- 启用并启动端口 mapper 服务：

```
[root@ipaserver ~]# systemctl enable rpcbind.service
[root@ipaserver ~]# systemctl start rpcbind.service
```

- 重启 Directory 服务器：

```
[root@ipaserver ~]# systemctl restart dirsrv.target
```

4.2. 将用户条目从 NIS 迁移到 IDM

NIS **passwd** map 包含关于用户的信息，如名称、UID、主组、GECOS、shell 和主目录。使用这个数据将 NIS 用户帐户迁移到 Identity Management (IdM)：

先决条件

- 在 NIS 服务器中具有 root 访问权限。
- [IdM 中启用了 NIS](#)。
- NIS 服务器被注册到 IdM。

步骤

1. 安装 **yp-tools** 软件包：

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. 在 NIS 服务器中，使用以下内容创建 **/root/nis-users.sh** 脚本：

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -d $1 -h $2 passwd > /dev/shm/nis-map.passwd 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.passwd) ; do
  IFS=' '
  username=$(echo $line | cut -f1 -d:)
  # Not collecting encrypted password because we need cleartext password
  # to create kerberos key
  uid=$(echo $line | cut -f3 -d:)
  gid=$(echo $line | cut -f4 -d:)
  gecos=$(echo $line | cut -f5 -d:)
  homedir=$(echo $line | cut -f6 -d:)
  shell=$(echo $line | cut -f7 -d:)

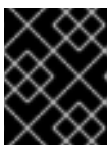
  # Now create this entry
  echo passw0rd1 | ipa user-add $username --first=NIS --last=USER \
    --password --gidnumber=$gid --uid=$uid --gecos="$gecos" --homedir=$homedir \
    --shell=$shell
  ipa user-show $username
done
```

3. 以 IdM **admin** 用户身份进行身份验证：

```
[root@nis-server ~]# kinit admin
```

4. 运行脚本。例如：

```
[root@nis-server ~]# sh /root/nis-users.sh nisdomain nis-server.example.com
```



重要

此脚本在名字、姓氏中使用硬编码的值，并将密码设置为 **passw0rd1**。用户必须在下一次登录时更改临时密码。

4.3. 将用户组从 NIS 迁移到 IDM

NIS **group** 映射包含有关组的信息，如组名称、GID 或组成员。使用此数据将 NIS 组迁移到 Identity Management (IdM)：

先决条件

- 在 NIS 服务器中具有 root 访问权限。
- [IdM 中启用了 NIS。](#)
- NIS 服务器被注册到 IdM。

步骤

1. 安装 **yp-tools** 软件包：

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. 在 NIS 服务器中使用以下内容创建 **/root/nis-groups.sh** 脚本：

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -d $1 -h $2 group > /dev/shm/nis-map.group 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.group); do
    IFS=' '
    groupname=$(echo $line | cut -f1 -d:)
    # Not collecting encrypted password because we need cleartext password
    # to create kerberos key
    gid=$(echo $line | cut -f3 -d:)
    members=$(echo $line | cut -f4 -d:)

    # Now create this entry
    ipa group-add $groupname --desc=NIS_GROUP_$groupname --gid=$gid
    if [ -n "$members" ]; then
        ipa group-add-member $groupname --users={$members}
    fi
    ipa group-show $groupname
done
```

3. 以 IdM **admin** 用户身份进行身份验证：

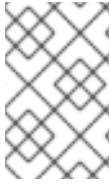
```
[root@nis-server ~]# kinit admin
```

4. 运行脚本。例如：

```
[root@nis-server ~]# sh /root/nis-groups.sh nisdomain nis-server.example.com
```

4.4. 将主机条目从 NIS 迁移到 IDM

NIS **hosts** 映射包含有关主机的信息，如主机名和 IP 地址。使用这个数据将 NIS 主机条目迁移到 Identity Management (IdM)：



注意

当您在 IdM 中创建主机组时，会自动创建对应的 shadow NIS 组。不要在这些影子 NIS 组中使用 **ipa netgroup-*** 命令。使用 **ipa netgroup-*** 命令仅管理通过 **netgroup-add** 命令创建的原生 netgroups。

先决条件

- 在 NIS 服务器中具有 root 访问权限。
- [IdM 中启用了 NIS](#)。
- NIS 服务器被注册到 IdM。

步骤

1. 安装 **yp-tools** 软件包：

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. 在 NIS 服务器中使用以下内容创建 **/root/nis-hosts.sh** 脚本：

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -d $1 -h $2 hosts | egrep -v "localhost|127.0.0.1" > /dev/shm/nis-map.hosts 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.hosts); do
    IFS=' '
    ipaddress=$(echo $line | awk '{print $1}')
    hostname=$(echo $line | awk '{print $2}')
    primary=$(ipa env xmlrpc_uri | tr -d '[:space:]' | cut -f3 -d: | cut -f3 -d/)
    domain=$(ipa env domain | tr -d '[:space:]' | cut -f2 -d:)
    if [ $(echo $hostname | grep "\." | wc -l) -eq 0 ] ; then
        hostname=$(echo $hostname.$domain)
    fi
    zone=$(echo $hostname | cut -f2- -d.)
    if [ $(ipa dnszone-show $zone 2>/dev/null | wc -l) -eq 0 ] ; then
        ipa dnszone-add --name-server=$primary --admin-email=root.$primary
    fi
    ptrzone=$(echo $ipaddress | awk -F. '{print $3 "." $2 "." $1 ".in-addr.arpa."}')
    if [ $(ipa dnszone-show $ptrzone 2>/dev/null | wc -l) -eq 0 ] ; then
        ipa dnszone-add $ptrzone --name-server=$primary --admin-email=root.$primary
    fi
    # Now create this entry
    ipa host-add $hostname --ip-address=$ipaddress
    ipa host-show $hostname
done
```

3. 以 IdM **admin** 用户身份进行身份验证：

```
[root@nis-server ~]# kinit admin
```

4. 运行脚本。例如：

```
[root@nis-server ~]# sh /root/nis-hosts.sh nisdomain nis-server.example.com
```



注意

此脚本不会迁移特殊主机配置，如别名。

4.5. 将网络组条目从 NIS 迁移到 IDM

NIS **netgroup** 映射包含有关网络用户组的信息。使用这个数据将 NIS netgroups 迁移到 Identity Management (IdM)：

先决条件

- 在 NIS 服务器中具有 root 访问权限。
- [IdM 中启用了 NIS](#)。
- NIS 服务器被注册到 IdM。

步骤

1. 安装 **yp-tools** 软件包：

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. 在 NIS 服务器中使用以下内容创建 **/root/nis-netgroups.sh** 脚本：

```
#!/bin/sh
# $1 is the NIS domain, $2 is the primary NIS server
ypcat -k -d $1 -h $2 netgroup > /dev/shm/nis-map.netgroup 2>&1

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.netgroup); do
  IFS=' '
  netgroupname=$(echo $line | awk '{print $1}')
  triples=$(echo $line | sed "s/^$netgroupname //" )
  echo "ipa netgroup-add $netgroupname --desc=NIS_NG_$netgroupname"
  if [ $(echo $line | grep "(," | wc -l) -gt 0 ]; then
    echo "ipa netgroup-mod $netgroupname --hostcat=all"
  fi
  if [ $(echo $line | grep ",," | wc -l) -gt 0 ]; then
    echo "ipa netgroup-mod $netgroupname --usercat=all"
  fi

  for triple in $triples; do
    triple=$(echo $triple | sed -e 's/-//g' -e 's/(//' -e 's/)//')
    if [ $(echo $triple | grep ",.*," | wc -l) -gt 0 ]; then
      hostname=$(echo $triple | cut -f1 -d,)
      username=$(echo $triple | cut -f2 -d,)
      domain=$(echo $triple | cut -f3 -d,)
      hosts=""; users=""; doms="";
```

```

[ -n "$hostname" ] && hosts="--hosts=$hostname"
[ -n "$username" ] && users="--users=$username"
[ -n "$domain" ] && doms="--nisdomain=$domain"
echo "ipa netgroup-add-member $netgroup $hosts $users $doms"
else
netgroup=$triple
echo "ipa netgroup-add $netgroup --desc=<NIS_NG>_$netgroup"
fi
done
done

```

3. 以 IdM **admin** 用户身份进行身份验证：

```
[root@nis-server ~]# kinit admin
```

4. 运行脚本。例如：

```
[root@nis-server ~]# sh /root/nis-netgroups.sh nisdomain nis-server.example.com
```

4.6. 将自动挂载映射从 NIS 迁移到 IDM

自动挂载映射是一系列嵌套和相互相关的条目，它们定义位置（父条目）、关联的键和映射。将 NIS 自动挂载映射迁移到 Identity Management (IdM)：

先决条件

- 在 NIS 服务器中具有 root 访问权限。
- IdM 中启用了 NIS。
- NIS 服务器被注册到 IdM。

步骤

1. 安装 **yp-tools** 软件包：

```
[root@nis-server ~]# dnf install yp-tools -y
```

2. 在 NIS 服务器中使用以下内容创建 **/root/nis-automounts.sh** 脚本：

```

#!/bin/sh
# $1 is for the automount entry in ipa

ipa automountlocation-add $1

# $2 is the NIS domain, $3 is the primary NIS server, $4 is the map name

ypcat -k -d $2 -h $3 $4 > /dev/shm/nis-map.$4 2>&1

ipa automountmap-add $1 $4

basedn=$(ipa env basedn | tr -d '[:space:]' | cut -f2 -d:)
cat > /tmp/amap.ldif <<EOF
dn: nis-domain=$2+nis-map=$4,cn=NIS Server,cn=plugins,cn=config

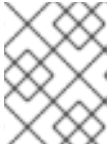
```

```

objectClass: extensibleObject
nis-domain: $2
nis-map: $4
nis-base: automountmapname=$4,cn=$1,cn=automount,$basedn
nis-filter: (objectclass=*)
nis-key-format: %{automountKey}
nis-value-format: %{automountInformation}
EOF
ldapadd -x -h $3 -D "cn=Directory Manager" -W -f /tmp/imap.ldif

IFS=$'\n'
for line in $(cat /dev/shm/nis-map.$4); do
  IFS=" "
  key=$(echo "$line" | awk '{print $1}')
  info=$(echo "$line" | sed -e "s^$key[ \t]*")
  ipa automountkey-add nis $4 --key="$key" --info="$info"
done

```



注意

该脚本导出 NIS 自动挂载信息，为自动挂载位置和关联映射生成 LDAP 数据交换格式 (LDIF)，并将 LDIF 文件导入到 IdM Directory Server。

3. 以 IdM **admin** 用户身份进行身份验证：

```
[root@nis-server ~]# kinit admin
```

4. 运行脚本。例如：

```
[root@nis-server ~]# sh /root/nis-automounts.sh location nisdomain
nis-server.example.com map_name
```


第 5 章 在 IDM 中使用自动挂载

自动挂载是在多个系统间管理、组织和访问目录的一种方式。每当请求访问一个目录时，Automount 会自动挂载该目录。这在身份管理(IdM)域中工作良好，因为它允许您在域中的客户端上轻松共享目录。

这个示例使用以下场景：

- `nfs-server.idm.example.com` 是网络文件系统(NFS)服务器的完全限定域名(FQDN)。
- 为了简单起见，`nfs-server.idm.example.com` 是一个 IdM 客户端，为 `raleigh` 自动挂载位置提供映射。



注意

自动挂载位置是一组唯一的 NFS 映射。理想情况下，这些映射都位于同一地理位置，例如：客户端可以从快速连接中受益，但这不是强制要求。

- NFS 服务器以读写形式导出 `/exports/project` 目录。
- 属于 `developers` 组的任何 IdM 用户都可以在使用 `raleigh` 自动挂载位置的任何 IdM 客户端中以 `/devel/project/` 来访问导出的目录。
- `idm-client.idm.example.com` 是位于 `raleigh` 自动挂载位置的 IdM 客户端。



重要

如果要使用 Samba 服务器而不是 NFS 服务器来为 IdM 客户端提供共享，请参阅 [如何在 IPA 环境中使用 Autofs 配置进行过Kerberos 的 CIFS 挂载？KCS 解决方案](#)。

5.1. IDM 中的 AUTOFS 和自动挂载

autofs 服务可根据需要自动化目录的挂载，方法是在目录被访问时，将 **automount** 守护进程定向到挂载目录。此外，在不活动一段时间后，**autofs** 将 **automount** 定向到未卸载的自动挂载的目录。与静态挂载不同，按需挂载可节省系统资源。

自动挂载映射

在使用 **autofs** 的系统上，**automount** 配置存储在几个不同的文件中。主要的 **automount** 配置文件是 `/etc/auto.master`，其中包含系统上 **automount** 的主映射以及相关的资源。此映射称为 *自动挂载映射*。

`/etc/auto.master` 配置文件包含 *主映射*。它可以包含对其他映射的引用。这些映射可以是直接的，也可以是间接的。直接映射使用挂载点的绝对路径名，而间接映射则使用相对路径名。

IdM 中的自动挂载配置

虽然 **automount** 通常从本地 `/etc/auto.master` 和相关文件检索其映射数据，但它也可以从其他源检索映射数据。一个通用源是 LDAP 服务器。在身份管理(IdM)环境中，这是一个 389 目录服务器。如果使用 **autofs** 的系统是 IdM 域中的一个客户端，则 **automount** 配置不会存储在本地配置文件中。相反，**autofs** 配置（如映射、位置和密钥）作为 LDAP 条目存储在 IdM 目录中。例如，对于 `idm.example.com` IdM 域，默认的主映射存储如下：

```
dn:
automountmapname=auto.master,cn=default,cn=automount,dc=idm,dc=example,dc=com
objectClass: automountMap
```

```
objectClass: top
automountMapName: auto.master
```

其他资源

- [根据需要挂载文件系统](#)

5.2. 在 RED HAT IDENTITY MANAGEMENT 域中使用 KERBEROS 建立一个 NFS 服务器

如果您使用 Red Hat Identity Management (IdM)，您可以将 NFS 服务器加入到 IdM 域中。这可让您集中管理用户和组，并使用 Kerberos 进行身份验证、完整性保护和流量加密。

先决条件

- NFS 服务器在 Red Hat Identity Management (IdM)域中 [已注册](#)。
- NFS 服务器正在运行并已配置。

流程

1. 以 IdM 管理员身份获取 kerberos 票据：

```
# kinit admin
```

2. 创建一个 **nfs/<FQDN>** 服务主体：

```
# ipa service-add nfs/nfs_server.idm.example.com
```

3. 从 IdM 检索 **nfs** 服务主体，并将其存储在 **/etc/krb5.keytab** 文件中：

```
# ipa-getkeytab -s idm_server.idm.example.com -p nfs/nfs_server.idm.example.com -k /etc/krb5.keytab
```

4. 可选：显示 **/etc/krb5.keytab** 文件中的主体：

```
# klist -k /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
-----
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
```

默认情况下，当您将主机加入到 IdM 域时，IdM 客户端会将主机主体添加到 **/etc/krb5.keytab** 文件中。如果缺少主机主体，请使用 **ipa-getkeytab -s idm_server.idm.example.com -p host/nfs_server.idm.example.com -k /etc/krb5.keytab** 命令添加它。

5. 使用 **ipa-client-automount** 工具配置 IdM ID 的映射：

```
# ipa-client-automount
Searching for IPA server...
IPA server: DNS discovery
Location: default
Continue to configure the system with these values? [no]: yes
Configured /etc/ldapd.conf
Restarting sssd, waiting for it to become available.
Started autofs
```

6. 更新 **/etc/exports** 文件，并将 Kerberos 安全方法添加到客户端选项中。例如：

```
/nfs/projects/ 192.0.2.0/24(rw,sec=krb5i)
```

如果您希望客户端可以从多个安全方法中选择，请使用冒号分割它们：

```
/nfs/projects/ 192.0.2.0/24(rw,sec=krb5:krb5i:krb5p)
```

7. 重新载入导出的文件系统：

```
# exportfs -r
```

5.3. 使用 IDM CLI 在 IDM 中配置自动挂载位置和映射

位置是一组映射，全部存储在 **auto.master** 中。一个位置可以存储多个映射。位置条目仅充当映射条目的容器；它本身并不是一个自动挂载配置。

作为身份管理(IdM)中的系统管理员，您可以在 IdM 中配置自动挂载位置和映射，以便指定位置中的 IdM 用户可以通过导航到其主机上的特定挂载点来访问 NFS 服务器导出的共享。导出的 NFS 服务器目录和挂载点都在映射中指定。这个示例描述了如何配置 **raleigh** 位置和映射，其将 **nfs-server.idm.example.com:/exports/project** 共享作为读写目录，挂载到 IdM 客户端上的 **/devel/** 挂载点。

先决条件

- 您以 IdM 管理员的身份登录到任何注册了 IdM 的主机上。

流程

1. 创建 **raleigh** 自动挂载位置：

```
$ ipa automountlocation-add raleigh
-----
Added automount location "raleigh"
-----
Location: raleigh
```

2. 在 **raleigh** 位置创建一个 **auto.devel** 自动挂载映射：

```
$ ipa automountmap-add raleigh auto.devel
-----
Added automount map "auto.devel"
```

```
-----
Map: auto.devel
```

3. 添加 **exports/** 共享的密钥和挂载信息：

a. 为 **auto.devel** 映射添加密钥和挂载信息：

```
$ ipa automountkey-add raleigh auto.devel --key='*' --info='-sec=krb5p,vers=4 nfs-
server.idm.example.com:/exports/&'
-----
Added automount key "*"
-----
Key: *
Mount information: -sec=krb5p,vers=4 nfs-server.idm.example.com:/exports/&
```

b. 为 **auto.master** 映射添加密钥和挂载信息：

```
$ ipa automountkey-add raleigh auto.master --key=/devel --info=auto.devel
-----
Added automount key "/devel"
-----
Key: /devel
Mount information: auto.devel
```

5.4. 在 IDM 客户端上配置自动挂载

作为身份管理(IdM)系统管理员，您可以在 IdM 客户端上配置自动挂载服务，以便在用户登录客户端时 IdM 用户可以自动访问为已添加客户端的位置配置的 NFS 共享。这个示例描述了如何配置 IdM 客户端，以使用 **raleigh** 位置中可用的 automount 服务。

先决条件

- 您有访问 IdM 客户端的 **root** 权限。
- 以 IdM 管理员身份登录。
- 自动挂载位置存在。示例位置为 **raleigh**。

流程

1. 在 IdM 客户端上，输入 **ipa-client-automount** 命令并指定位置。使用 **-U** 选项以无人值守方式运行脚本：

```
# ipa-client-automount --location raleigh -U
```

2. 停止 **autofs** 服务，清除 **SSSD** 缓存，然后启动 **autofs** 服务来加载新的配置设置：

```
# systemctl stop autofs ; sss_cache -E ; systemctl start autofs
```

5.5. 验证 IDM 用户能否访问 IDM 客户端上的 NFS 共享

作为身份管理(IdM)系统管理员，您可以在登录到特定的 IdM 客户端时测试作为特定组一员的 IdM 用户是否可以访问 NFS 共享。

在示例中，测试了以下场景：

- 属于 **developers** 组的名为 **idm_user** 的 IdM 用户可以读写自动挂载在 **idm-client.idm.example.com**（一个位于 **raleigh** 自动挂载位置的 IdM 客户端）上的 **/devel/project** 目录中的内容。

先决条件

- 您已 [在 IdM 主机上建立了一个带有 Kerberos 的 NFS 服务器](#)。
- 您已 [在 IdM 中配置了自动挂载位置、映射和挂载点](#)，您在其中配置了 IdM 用户如何访问 NFS 共享。
- 您已 [在 IdM 客户端上配置了自动挂载](#)。

流程

1. 验证 IdM 用户能否可以访问 **读-写** 目录：

- a. 以 IdM 用户身份连接到 IdM 客户端：

```
$ ssh idm_user@idm-client.idm.example.com
Password:
```

- b. 获取 IdM 用户的票据授权票据(TGT)：

```
$ kinit idm_user
```

- c. [可选] 查看 IdM 用户的组成员身份：

```
$ ipa user-show idm_user
User login: idm_user
[...]
Member of groups: developers, ipausers
```

- d. 进入到 **/devel/project** 目录：

```
$ cd /devel/project
```

- e. 列出目录内容：

```
$ ls
rw_file
```

- f. 对目录中的文件添加一行来测试 **写** 权限：

```
$ echo "idm_user can write into the file" > rw_file
```

- g. [可选] 查看更新的文件内容：

```
$ cat rw_file
this is a read-write file
idm_user can write into the file
```

输出确认 **idm_user** 可以对该文件进行写入。

第 6 章 使用 ANSIBLE 为 IDM 用户自动挂载 NFS 共享

自动挂载是在多个系统间管理、组织和访问目录的一种方式。每当请求访问一个目录时，Automount 会自动挂载该目录。这在身份管理(IdM)域中工作良好，因为它允许您在域中的客户端上轻松共享目录。

您可以使用 Ansible 配置 NFS 共享，以使其可以被 IdM 位置中登录到 IdM 客户端的 IdM 用户自动挂载。

本章中的示例使用以下场景：

- `nfs-server.idm.example.com` 是网络文件系统(NFS)服务器的完全限定域名(FQDN)。
- `nfs-server.idm.example.com` 是位于 `raleigh` 自动挂载位置的 IdM 客户端。
- NFS 服务器以读写形式导出 `/exports/project` 目录。
- 属于 `developers` 组的任何 IdM 用户都可以访问导出的目录的内容，因为 IdM 客户端上的 `/devel/project/` 位于与 NFS 服务器相同的 `raleigh` 自动挂载位置。
- `idm-client.idm.example.com` 是位于 `raleigh` 自动挂载位置的 IdM 客户端。



重要

如果要使用 Samba 服务器而不是 NFS 服务器来为 IdM 客户端提供共享，请参阅 [如何在 IPA 环境中使用 Autofs 配置进行过Kerberos 的 CIFS 挂载？KCS 解决方案](#)。

本章包含以下部分：

1. [IdM 中的 autofs 和自动挂载](#)
2. [在 IdM 中建立一个具有 Kerberos 的 NFS 服务器](#)
3. [使用 Ansible 在 IdM 中配置自动挂载位置、映射和密钥](#)
4. [使用 Ansible 将 IdM 用户添加到拥有 NFS 共享的组中](#)
5. [在 IdM 客户端上配置自动挂载](#)
6. [验证 IdM 用户能否访问 IdM 客户端上的 NFS 共享](#)

6.1. IDM 中的 AUTOFS 和自动挂载

autofs 服务可根据需要自动化目录的挂载，方法是在目录被访问时，将 **automount** 守护进程定向到挂载目录。此外，在不活动一段时间后，**autofs** 将 **automount** 定向到未卸载的自动挂载的目录。与静态挂载不同，按需挂载可节省系统资源。

自动挂载映射

在使用 **autofs** 的系统上，**automount** 配置存储在几个不同的文件中。主要的 **automount** 配置文件是 `/etc/auto.master`，其中包含系统上 **automount** 的主映射以及相关的资源。此映射称为 *自动挂载映射*。

`/etc/auto.master` 配置文件包含 *主映射*。它可以包含对其他映射的引用。这些映射可以是直接的，也可以是间接的。直接映射使用挂载点的绝对路径名，而间接映射则使用相对路径名。

IdM 中的自动挂载配置

虽然 **automount** 通常从本地 **/etc/auto.master** 和相关文件检索其映射数据，但它也可以从其他源检索映射数据。一个通用源是 LDAP 服务器。在身份管理(IdM)环境中，这是一个 389 目录服务器。如果使用 **autofs** 的系统是 IdM 域中的一个客户端，则 **automount** 配置不会存储在本地配置文件中。相反，**autofs** 配置（如映射、位置和密钥）作为 LDAP 条目存储在 IdM 目录中。例如，对于 **idm.example.com** IdM 域，默认的主映射存储如下：

```
dn:
automountmapname=auto.master,cn=default,cn=automount,dc=idm,dc=example,dc=com
objectClass: automountMap
objectClass: top
automountMapName: auto.master
```

其他资源

- [根据需要挂载文件系统](#)

6.2. 在 RED HAT IDENTITY MANAGEMENT 域中使用 KERBEROS 建立一个 NFS 服务器

如果您使用 Red Hat Identity Management (IdM)，您可以将 NFS 服务器加入到 IdM 域中。这可让您集中管理用户和组，并使用 Kerberos 进行身份验证、完整性保护和流量加密。

先决条件

- NFS 服务器在 Red Hat Identity Management (IdM)域中 [已注册](#)。
- NFS 服务器正在运行并已配置。

流程

1. 以 IdM 管理员身份获取 kerberos 票据：

```
# kinit admin
```

2. 创建一个 **nfs/<FQDN>** 服务主体：

```
# ipa service-add nfs/nfs_server.idm.example.com
```

3. 从 IdM 检索 **nfs** 服务主体，并将其存储在 **/etc/krb5.keytab** 文件中：

```
# ipa-getkeytab -s idm_server.idm.example.com -p nfs/nfs_server.idm.example.com -k /etc/krb5.keytab
```

4. 可选：显示 **/etc/krb5.keytab** 文件中的主体：

```
# klist -k /etc/krb5.keytab
Keytab name: FILE:/etc/krb5.keytab
KVNO Principal
-----
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
```



```
1 nfs/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
7 host/nfs_server.idm.example.com@IDM.EXAMPLE.COM
```

默认情况下，当您将主机加入到 IdM 域时，IdM 客户端会将主机主体添加到 `/etc/krb5.keytab` 文件中。如果缺少主机主体，请使用 `ipa-getkeytab -s idm_server.idm.example.com -p host/nfs_server.idm.example.com -k /etc/krb5.keytab` 命令添加它。

5. 使用 `ipa-client-automount` 工具配置 IdM ID 的映射：

```
# ipa-client-automount
Searching for IPA server...
IPA server: DNS discovery
Location: default
Continue to configure the system with these values? [no]: yes
Configured /etc/idmapd.conf
Restarting sssd, waiting for it to become available.
Started autofs
```

6. 更新 `/etc/exports` 文件，并将 Kerberos 安全方法添加到客户端选项中。例如：

```
/nfs/projects/ 192.0.2.0/24(rw,sec=krb5i)
```

如果您希望客户端可以从多个安全方法中选择，请使用冒号分割它们：

```
/nfs/projects/ 192.0.2.0/24(rw,sec=krb5:krb5i:krb5p)
```

7. 重新载入导出的文件系统：

```
# exportfs -r
```

6.3. 使用 ANSIBLE 在 IDM 中配置自动挂载位置、映射和密钥

作为身份管理(IdM)系统管理员，您可以在 IdM 中配置自动挂载位置和映射，以便指定位置中的 IdM 用户可以通过导航到其主机上的特定挂载点来访问 NFS 服务器导出的共享。导出的 NFS 服务器目录和挂载点都在映射中指定。在 LDAP 术语中，位置是此类映射条目的一个容器。

这个示例描述了如何使用 Ansible 来配置 `raleigh` 位置和映射，其将 `nfs-server.idm.example.com:/exports/project` 共享作为读写目录挂载到 IdM 客户端上的 `/devel/project` 挂载点。

先决条件

- 您需要知道 IdM **admin** 密码。
- 您已配置了 Ansible 控制节点以满足以下要求：
 - 您使用 Ansible 版本 2.14 或更高版本。
 - 您已在 Ansible 控制器上安装了 **ansible-freeipa** 软件包。

- 示例假定在 `~/MyPlaybooks/` 目录中，您已创建了带有 IdM 服务器的完全限定域名(FQDN)的 [Ansible 清单文件](#)。
- 示例假定 `secret.yml` Ansible vault 存储了 `ipaadmin_password`。
- 目标节点（这是执行 `ansible-freeipa` 模块的节点）是 IdM 域的一部分，来作为 IdM 客户端、服务器或副本。

流程

1. 在 Ansible 控制节点上，导航到 `~/MyPlaybooks/` 目录：

```
$ cd ~/MyPlaybooks/
```

2. 复制位于 `/usr/share/doc/ansible-freeipa/playbooks/automount/` 目录中的 `automount-location-present.yml` Ansible playbook 文件：

```
$ cp /usr/share/doc/ansible-freeipa/playbooks/automount/automount-location-present.yml automount-location-map-and-key-present.yml
```

3. 打开 `automount-location-map-and-key-present.yml` 文件进行编辑。
4. 通过在 `ipaautomountlocation` 任务部分设置以下变量来调整文件：

- 将 `ipaadmin_password` 变量设为 IdM `admin` 的密码。
 - 将 `name` 变量设为 `raleigh`。
 - 确保 `state` 变量设置为 `present`。
- 这是当前示例修改的 Ansible playbook 文件：

```
---
- name: Automount location present example
  hosts: ipaserver
  vars_files:
    - /home/user_name/MyPlaybooks/secret.yml
  tasks:
    - name: Ensure automount location is present
      ipaautomountlocation:
        ipaadmin_password: "{{ ipaadmin_password }}"
        name: raleigh
        state: present
```

5. 继续编辑 `automount-location-map-and-key-present.yml` 文件：

- a. 在 `tasks` 部分中，添加一个任务来确保存在一个自动挂载映射：

```
[...]
vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
tasks:
[...]
```

```
- name: ensure map named auto.devel in location raleigh is created
  ipaautomountmap:
    ipaadmin_password: "{{ ipaadmin_password }}"
```

```

name: auto.devel
location: raleigh
state: present

```

- b. 添加另一个任务，来将挂载点和 NFS 服务器信息添加到映射中：

```

[...]  
vars_files:  
- /home/user_name/MyPlaybooks/secret.yml  
tasks:  
[...]  
- name: ensure automount key /devel/project is present  
  ipaautomountkey:  
    ipaadmin_password: "{{ ipaadmin_password }}"  
    location: raleigh  
    mapname: auto.devel  
    key: /devel/project  
    info: nfs-server.idm.example.com:/exports/project  
    state: present

```

- c. 添加另一个任务来确保 auto.devel 连接到 auto.master：

```

[...]  
vars_files:  
- /home/user_name/MyPlaybooks/secret.yml  
tasks:  
[...]  
- name: Ensure auto.devel is connected in auto.master:  
  ipaautomountkey:  
    ipaadmin_password: "{{ ipaadmin_password }}"  
    location: raleigh  
    mapname: auto.map  
    key: /devel  
    info: auto.devel  
    state: present

```

6. 保存该文件。
7. 运行 Ansible playbook，并指定 playbook 和清单文件：

```

$ ansible-playbook --vault-password-file=password_file -v -i inventory automount-  
location-map-and-key-present.yml

```

6.4. 使用 ANSIBLE 将 IDM 用户添加到拥有 NFS 共享的组中

作为身份管理(IdM)系统管理员，您可以使用 Ansible 来创建可以访问 NFS 共享的用户组，并将 IdM 用户添加到此组中。

本例描述了如何使用 Ansible playbook 来确保 idm_user 帐户属于 developers 组，以便 idm_user 可以访问 /exports/project NFS 共享。

先决条件

- 您有访问 `nfs-server.idm.example.com` NFS 服务器的 **root** 权限，该服务器是一个位于 `raleigh` 自动挂载位置的 IdM 客户端。
- 您需要知道 IdM **admin** 密码。
- 您已配置了 Ansible 控制节点以满足以下要求：
 - 您使用 Ansible 版本 2.14 或更高版本。
 - 您已在 Ansible 控制器上安装了 **ansible-freeipa** 软件包。
 - 示例假定在 `~/MyPlaybooks/` 目录中，您已创建了带有 IdM 服务器的完全限定域名(FQDN)的 **Ansible 清单文件**。
 - 示例假定 `secret.yml` Ansible vault 存储了 `ipaadmin_password`。
- 目标节点（这是执行 **ansible-freeipa** 模块的节点）是 IdM 域的一部分，来作为 IdM 客户端、服务器或副本。
 - 在 `~/MyPlaybooks/` 中，您已创建了 **automount-location-map-and-key-present.yml** 文件，该文件已包含 **使用 Ansible 在 IdM 中配置自动挂载位置、映射和密钥** 中的任务。

流程

1. 在 Ansible 控制节点上，进到 `~/MyPlaybooks/` 目录：

```
$ cd ~/MyPlaybooks/
```

2. 打开 **automount-location-map-and-key-present.yml** 文件进行编辑。
3. 在 **tasks** 部分，添加一个任务来确保 IdM **developers** 组存在，并且 `idm_user` 已添加到此组中：

```
[...]
vars_files:
- /home/user_name/MyPlaybooks/secret.yml
tasks:
[...]
- ipagroup:
  ipaadmin_password: "{{ ipaadmin_password }}"
  name: developers
  user:
  - idm_user
  state: present
```

4. 保存该文件。
5. 运行 Ansible playbook，并指定 playbook 和清单文件：

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory automount-
location-map-and-key-present.yml
```

6. 在 NFS 服务器上，将 `/exports/project` 目录的组所有权更改为 **developers**，以便组中的每个 IdM 用户都可以访问该目录：

```
# chgrp developers /exports/project
```

6.5. 在 IDM 客户端上配置自动挂载

作为身份管理(IdM)系统管理员，您可以在 IdM 客户端上配置自动挂载服务，以便在用户登录客户端时 IdM 用户可以自动访问为已添加客户端的位置配置的 NFS 共享。这个示例描述了如何配置 IdM 客户端，以使用 **raleigh** 位置中可用的 automount 服务。

先决条件

- 您有访问 IdM 客户端的 **root** 权限。
- 以 IdM 管理员身份登录。
- 自动挂载位置存在。示例位置为 **raleigh**。

流程

1. 在 IdM 客户端上，输入 **ipa-client-automount** 命令并指定位置。使用 **-U** 选项以无人值守方式运行脚本：

```
# ipa-client-automount --location raleigh -U
```

2. 停止 autofs 服务，清除 SSSD 缓存，然后启动 autofs 服务来加载新的配置设置：

```
# systemctl stop autofs ; sss_cache -E ; systemctl start autofs
```

6.6. 验证 IDM 用户能否访问 IDM 客户端上的 NFS 共享

作为身份管理(IdM)系统管理员，您可以在登录到特定的 IdM 客户端时测试作为特定组一员的 IdM 用户是否可以访问 NFS 共享。

在示例中，测试了以下场景：

- 属于 **developers** 组的名为 **idm_user** 的 IdM 用户可以读写自动挂载在 **idm-client.idm.example.com**（一个位于 **raleigh** 自动挂载位置的 IdM 客户端）上的 **/devel/project** 目录中的内容。

先决条件

- 您已 [在 IdM 主机上建立了一个具有 Kerberos 的 NFS 服务器](#)。
- 您已 [在 IdM 中配置了自动挂载位置、映射和挂载点](#)，您已在其中配置了 IdM 用户如何访问 NFS 共享。
- 您已 [使用 Ansible 将 IdM 用户添加到拥有 NFS 共享的 developers 组中](#)。
- 您已 [在 IdM 客户端上配置了自动挂载](#)。

流程

1. 验证 IdM 用户能否可以访问 **读-写** 目录：
 - a. 以 IdM 用户身份连接到 IdM 客户端：

```
$ ssh idm_user@idm-client.idm.example.com
Password:
```

- b. 获取 IdM 用户的票据授权票据(TGT)：

```
$ kinit idm_user
```

- c. [可选] 查看 IdM 用户的组成员身份：

```
$ ipa user-show idm_user
User login: idm_user
[...]
Member of groups: developers, ipausers
```

- d. 进入到 `/devel/project` 目录：

```
$ cd /devel/project
```

- e. 列出目录内容：

```
$ ls
rw_file
```

- f. 对目录中的文件添加一行来测试 **写** 权限：

```
$ echo "idm_user can write into the file" > rw_file
```

- g. [可选] 查看更新的文件内容：

```
$ cat rw_file
this is a read-write file
idm_user can write into the file
```

输出确认 `idm_user` 可以对该文件进行写入。