Red Hat Enterprise Linux 8

ID 관리에서 자격 증명 모음 작업

IdM에 민감한 데이터 저장 및 관리

Last Updated: 2023-09-19

Red Hat Enterprise Linux 8 ID 관리에서 자격 증명 모음 작업

IdM에 민감한 데이터 저장 및 관리

법적 공지

Copyright © 2023 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.

초록

자격 증명 모음은 서비스에 대한 인증 자격 증명과 같은 중요한 데이터를 저장, 검색 및 공유할 수 있는 Red Hat IdM(Identity Management)의 안전한 위치입니다. 명령줄 또는 Ansible 플레이북을 사용하여 자격 증명 모음을 관리할 수 있습니다.

차례

보다 포괄적 수용을 위한 오픈 소스 용어 교체

Red Hat은 코드, 문서 및 웹 속성에서 문제가 있는 언어를 교체하기 위해 최선을 다하고 있습니다. 먼저 마스터(master), 슬레이브(slave), 블랙리스트(blacklist), 화이트리스트(whitelist) 등 네 가지 용어를 교체하고 있습니다. 이러한 변경 작업은 작업 범위가 크므로 향후 여러 릴리스에 걸쳐 점차 구현할 예정입니다. 자세한 내용은 CTO Chris Wright의 메시지 를 참조하십시오.

ID 관리에서 계획된 용어 교체는 다음과 같습니다.

- ***차단 목록*** *블랙리스트*대체
- ***허용 목록*** 대체 *허용 목록*
- ***슬레이브*** 교체
- *마스터* 라는 단어는 컨텍스트에 따라 보다 정확한 언어로 교체되고 있습니다.
  - ***IdM 서버가*** *IdM 마스터*교체
  - ***CA 갱신 마스터를 대체하는CA 갱신서버***
  - ***CRL 게시자 서버는*** *CRL 마스터*교체
  - ***다중*** *마스터*교체

RED HAT 문서에 관한 피드백 제공

문서에 대한 피드백에 감사드립니다. 어떻게 개선할 수 있는지 알려주십시오.

특정 문구에 대한 의견 제출

1. **Multi-page HTML** 형식으로 설명서를 보고 페이지가 완전히 로드된 후 오른쪽 상단 모서리에 **피드백** 버튼이 표시되는지 확인합니다.
2. 커서를 사용하여 주석 처리할 텍스트 부분을 강조 표시합니다.
3. 강조 표시된 텍스트 옆에 표시되는 **피드백 추가** 버튼을 클릭합니다.
4. 의견을 추가하고 **제출** 을 클릭합니다.

Jira를 통해 피드백 제출 (등록 필요)

1. Jira 웹 사이트에 로그인합니다.
2. 상단 탐색 모음에서 **생성** 을 클릭합니다.
3. **Summary** (요약) 필드에 설명 제목을 입력합니다.
4. **Description** (설명) 필드에 개선을 위한 제안을 입력합니다. 문서의 관련 부분에 대한 링크를 포함합니다.
5. 대화 상자 하단에서 **생성** 을 클릭합니다.

1장. IDM의 자격 증명 모음

이 장에서는 IdM(Identity Management)의 자격 증명 모음에 대해 설명합니다. 다음과 같은 주제를 소개합니다.

- 자격 증명 모음의 개념입니다.
- 자격 증명 모음과 관련된 다양한 역할.
- 보안 및 액세스 제어 수준에 따라 IdM에서 사용할 수 있는 다양한 유형의 자격 증명 모음 .
- 소유권에 따라 IdM에서 사용할 수 있는 다양한 유형의 자격 증명 모음입니다.
- 자격 증명 모음 컨테이너의 개념.
- IdM에서 자격 증명 모음을 관리하기 위한 기본 명령 .
- IdM에서 자격 증명 모음을 사용하기 위한 사전 요구 사항인 KRA(키 복구 기관)를 설치합니다.

1.1. 자격 증명 모음 및 이점

자격 증명 모음은 모든 중요한 데이터를 안전하게 저장하지만 한 곳에 편리하게 저장하려는 IdM(Identity Management) 사용자에게 유용한 기능입니다. 다양한 유형의 자격 증명 모음이 있으며 요구 사항에 따라 사용할 자격 증명을 선택해야 합니다.

자격 증명 모음은 시크릿을 저장, 검색, 공유 및 복구하기 위한 (IdM)의 안전한 위치입니다. 시크릿은 제한된 사용자 또는 엔티티 그룹만 액세스할 수 있는 보안 민감한 데이터(일반적으로 인증 자격 증명)입니다. 예를 들어 시크릿은 다음과 같습니다.

- 암호
- 핀
- 개인 SSH 키

자격 증명 모음은 암호 관리자와 비교됩니다. 암호 관리자와 마찬가지로 자격 증명 모음에는 일반적으로 사용자가 자격 증명 모음에 저장된 정보를 잠금 해제하고 액세스하기 위해 하나의 기본 암호를 생성하고 기억해야 합니다. 그러나 사용자는 표준 자격 증명 모음을 선택할 수도 있습니다. 표준 자격 증명 모음에서는 사용자가 자격 증명 모음에 저장된 시크릿에 액세스하기 위해 암호를 입력할 필요가 없습니다.

참고

IdM에서 자격 증명 모음의 목적은 IdM이 아닌 외부 서비스에 인증할 수 있는 인증 자격 증명을 저장하는 것입니다.

IdM 자격 증명 모음의 기타 중요한 특성은 다음과 같습니다.

- 자격 증명 모음 소유자는 vault 소유자와 자격 증명 모음 소유자가 자격 증명 모음 멤버로 선택하는 IdM 사용자만 액세스할 수 있습니다. 또한 IdM 관리자는 자격 증명 모음에 액세스할 수 있습니다.
- 사용자가 자격 증명 모음을 생성할 수 있는 권한이 없는 경우 IdM 관리자는 자격 증명 모음을 생성하고 사용자를 소유자로 설정할 수 있습니다.

- 사용자 및 서비스는 IdM 도메인에 등록된 시스템에서 자격 증명 모음에 저장된 시크릿에 액세스할 수 있습니다.
- 하나의 자격 증명 모음에는 하나의 시크릿만 포함할 수 있습니다(예: 하나의 파일). 그러나 파일 자체는 암호, 키탭 또는 인증서와 같은 여러 시크릿을 포함할 수 있습니다.

참고

Vault는 IdM 웹 UI가 아닌 IdM 명령행(CLI)에서만 사용할 수 있습니다.

1.2. VAULT 소유자, 멤버 및 관리자

IdM(Identity Management)은 다음과 같은 자격 증명 모음 사용자 유형을 구분합니다.

Vault 소유자

자격 증명 모음 소유자는 자격 증명 모음에 대한 기본 관리 권한이 있는 사용자 또는 서비스입니다. 예를 들어 자격 증명 모음 소유자는 자격 증명 모음의 속성을 수정하거나 새 자격 증명 모음 멤버를 추가할 수 있습니다.
각 자격 증명 모음에는 하나 이상의 소유자가 있어야 합니다. 자격 증명 모음에는 소유자가 여러 개 있을 수도 있습니다.

Vault 멤버

자격 증명 모음 멤버는 다른 사용자 또는 서비스에서 생성한 자격 증명 모음에 액세스할 수 있는 사용자 또는 서비스입니다.

Vault 관리자

Vault 관리자는 모든 자격 증명 모음에 대한 무제한 액세스 권한이 있으며 모든 자격 증명 모음 작업을 수행할 수 있습니다.

참고

대칭 및 symmetric 자격 증명 모음은 암호 또는 키로 보호되며 특수 액세스 제어 규칙을 적용합니다(Vault 유형 참조). 관리자는 다음 규칙을 충족해야 합니다.

- 대칭 및 symmetric 자격 증명 모음의 시크릿에 액세스합니다.
- vault 암호 또는 키를 변경하거나 재설정합니다.

자격 증명 모음 관리자는 **Vault 관리자** 권한이 있는 모든 사용자입니다. IdM의 RBAC(역할 기반 액세스 제어) 컨텍스트에서는 역할에 적용할 수 있는 권한 그룹입니다.

Vault 사용자

자격 증명 모음 사용자는 자격 증명 모음에 있는 컨테이너를 나타냅니다. **Vault 사용자** 정보는 **ipa vault-show** 와 같은 특정 명령의 출력에 표시됩니다.

```
$ ipa vault-show my_vault
  Vault name: my_vault
  Type: standard
  Owner users: user
  Vault user: user
```

자격 증명 모음 컨테이너 및 사용자 자격 증명 모음에 대한 자세한 내용은 Vault 컨테이너를 참조하십시오.

추가 리소스

- 자격 증명 모음에 대한 자세한 내용은 표준, 대칭 및 symmetric 자격 증명 모음을 참조하십시오.

1.3. 표준, 대칭 및 SYMMETRIC 자격 증명 모음

IdM은 보안 및 액세스 제어 수준에 따라 자격 증명 모음을 다음 유형으로 분류합니다.

표준 자격 증명 모음

Vault 소유자와 자격 증명 모음 멤버는 암호 또는 키를 사용하지 않고도 시크릿을 보관하고 검색할 수 있습니다.

대칭 자격 증명 모음

자격 증명 모음의 시크릿은 대칭 키로 보호됩니다. Vault 소유자와 멤버는 시크릿을 보관하고 검색할 수 있지만 자격 증명 모음 암호를 제공해야 합니다.

Vertical 자격 증명 모음

자격 증명 모음의 시크릿은 symmetric 키로 보호됩니다. 사용자는 공개 키를 사용하여 시크릿을 보관하고 개인 키를 사용하여 검색합니다. Vault 멤버는 시크릿만 보관할 수 있지만 자격 증명 모음 소유자는 시크릿을 모두 보관하고 검색할 수 있습니다.

1.4. 사용자, 서비스 및 공유 자격 증명 모음

소유권에 따라 IdM은 자격 증명 모음을 여러 유형으로 분류합니다. 아래 표에 는 각 유형, 소유자 및 사용에 대한 정보가 나와 있습니다.

표 1.1. 소유권에 따른 IdM 자격 증명 모음

유형	설명	소유자	참고
사용자 자격 증명 모음	사용자의 개인 자격 증명 모음	단일 사용자	IdM 관리자가 허용하는 경우 사용자는 하나 이상의 사용자 자격 증명 모음을 보유할 수 있습니다.
서비스 자격 증명 모음	서비스의 개인 자격 증명 모음	단일 서비스	IdM 관리자가 허용하는 경우 모든 서비스는 하나 이상의 사용자 자격 증명 모음을 보유할 수 있습니다.
공유 자격 증명 모음	여러 사용자 및 서비스에서 공유하는 자격 증명 모음	자격 증명 모음을 만든 자격 증명 모음 관리자	IdM 관리자가 허용하는 경우 사용자 및 서비스의 사용자 자격 증명 모음을 한 개 이상 보유할 수 있습니다. 자격 증명 모음을 생성한 자격 증명 모음 관리자 이외의 자격 증명 모음 관리자도 자격 증명 모음에 대한 전체 액세스 권한을 갖습니다.

1.5. VAULT 컨테이너

자격 증명 모음 컨테이너는 자격 증명 모음 컬렉션입니다. 아래 표에 는 IdM(Identity Management)에서 제공하는 기본 자격 증명 모음 컨테이너가 나열되어 있습니다.

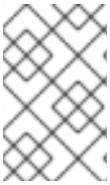
표 1.2. IdM의 기본 자격 증명 모음 컨테이너

유형	설명	목적
사용자 컨테이너	사용자의 개인 컨테이너	특정 사용자의 사용자 자격 증명 모음 저장
서비스 컨테이너	서비스의 개인 컨테이너	특정 서비스에 대한 서비스 자격 증명 모음 저장
공유 컨테이너	여러 사용자 및 서비스를 위한 컨테이너	여러 사용자 또는 서비스에서 공유할 수 있는 자격 증명 모음 저장

IdM은 사용자 또는 서비스의 첫 번째 개인 자격 증명 모음을 생성할 때 각 사용자 또는 서비스에 대해 사용자 및 서비스 컨테이너를 자동으로 생성합니다. 사용자 또는 서비스가 삭제되면 IdM에서 컨테이너와 해당 콘텐츠를 제거합니다.

1.6. 기본 IDM 자격 증명 모음 명령

아래에 설명된 기본 명령을 사용하여 IdM(Identity Management) 자격 증명 모음을 관리할 수 있습니다. 아래 표에 는 해당 목적에 대한 설명이 포함된 **ipa vault-*** 명령 목록이 포함되어 있습니다.



참고

ipa vault-* 명령을 실행하기 전에 IdM 도메인의 하나 이상의 서버에 KRA(키 복구 기관) 인증서 시스템 구성 요소를 설치합니다. 자세한 내용은 IdM에 키 복구 기관 설치를 참조하십시오.

표 1.3. 설명이 포함된 기본 IdM 자격 증명 모음 명령

명령	목적
IPA 도움말 자격 증명 모음	IdM 자격 증명 모음 및 샘플 자격 증명 모음 명령에 대한 개념 정보를 표시합니다.
ipa vault-add --help, **ipa vault-find --help**	특정 **ipa vault-*** 명령에 **--help** 옵션을 추가하면 해당 명령에 사용할 수 있는 옵션과 상세한 도움말이 표시됩니다.

ipa vault-show user_vault --user idm_user	자격 증명 모음 멤버로 액세스할 때 자격 증명 모음 소유자를 지정해야 합니다. vault 소유자를 지정하지 않으면 IdM에서 자격 증명 모음을 찾지 못했음을 알려줍니다.
	`[admin@server ~]$ ipa vault-show user_vault`
	`ipa: ERROR: user_vault: vault not found`

ipa vault-show shared_vault --shared	공유 자격 증명 모음에 액세스할 때 액세스하려는 자격 증명 모음이 공유 자격 증명 모음임을 지정해야 합니다. 그렇지 않으면 IdM에서 자격 증명 모음을 찾지 못했음을 알려줍니다.
	`[admin@server ~]$ ipa vault-show shared_vault`
	`ipa: ERROR: shared_vault: vault not found`

1.7. IDM에서 주요 복구 기관 설치

다음 절차에 따라 특정 IdM 서버에 KRA(Key Recovery Authority) 인증 시스템(CS) 구성 요소를 설치하여 IdM(Identity Management)에서 자격 증명 모음을 활성화합니다.

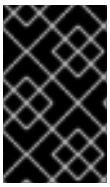
사전 요구 사항

- IdM 서버에 **root** 로 로그인되어 있습니다.
- IdM 인증 기관이 IdM 서버에 설치되어 있습니다.
- **Directory Manager** 자격 증명이 있습니다.

절차

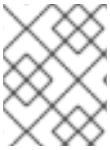
- KRA를 설치합니다.

```
# ipa-kra-install
```



중요

숨겨진 복제본에 IdM 클러스터의 첫 번째 KRA를 설치할 수 있습니다. 그러나 추가 KRA를 설치하려면 금지되지 않은 복제본에 KRA 복제본을 설치하기 전에 숨겨진 복제본을 일시적으로 활성화해야 합니다. 그런 다음 원래 숨겨진 복제본을 다시 숨길 수 있습니다.



참고

자격 증명 모음 서비스를 사용하고 탄력적으로 사용하려면 두 개의 IdM 서버에 KRA를 설치합니다. 여러 개의 KRA 서버를 유지 관리하면 데이터 손실이 발생하지 않습니다.

추가 리소스

- 숨겨진 복제본 데모 또는 승격을 참조하십시오.
- 숨겨진 복제본 모드를 참조하십시오.

2장. IDM 사용자 자격 증명 모음 사용: 시크릿 저장 및 검색

이 장에서는 Identity Management에서 사용자 자격 증명 모음을 사용하는 방법을 설명합니다. 특히 사용자가 IdM 자격 증명 모음에 시크릿을 저장하는 방법과 사용자가 이를 검색하는 방법을 설명합니다. 사용자는 두 개의 다른 IdM 클라이언트에서 저장 및 검색을 수행할 수 있습니다.

사전 요구 사항

- 키 복구 기관(KRA) 인증서 시스템 구성 요소가 IdM 도메인에 있는 하나 이상의 서버에 설치되어 있습니다. 자세한 내용은 IdM에 키 복구 기관 설치를 참조하십시오.

2.1. 사용자 자격 증명 모음에 시크릿 저장

중요한 정보가 있는 파일을 안전하게 저장하기 위해 하나 이상의 개인 자격 증명 모음으로 vault 컨테이너를 생성하려면 다음 절차를 따르십시오. 아래 절차에 사용된 예에서 **idm_user** 사용자는 표준 유형의 자격 증명 모음을 생성합니다. 표준 자격 증명 모음 유형을 사용하면 파일에 액세스할 때 **idm_user** 를 인증할 필요가 없습니다. **idm_user** 는 사용자가 로그인한 IdM 클라이언트에서 파일을 검색할 수 있습니다.

절차의 경우:

- **idm_user** 는 자격 증명 모음을 생성하려는 사용자입니다.
- **my_vault** 는 사용자의 인증서를 저장하는 데 사용되는 자격 증명 모음입니다.
- 자격 증명 모음 유형은 **표준** 이므로 보관된 인증서에 액세스하는 경우 사용자가 vault 암호를 제공할 필요가 없습니다.
- **secret.txt** 는 사용자가 자격 증명 모음에 저장하려는 인증서가 포함된 파일입니다.

사전 요구 사항

- **idm_user** 의 암호를 알고 있습니다.
- IdM 클라이언트인 호스트에 로그인되어 있습니다.

절차

1. **idm_user** 에 대한 TGT(K Kerberos 티켓 부여 티켓)를 받으십시오.

```
$ kinit idm_user
```

2. **ipa vault-add** 명령을 **--type standard** 옵션과 함께 사용하여 표준 자격 증명 모음을 생성합니다.

```
$ ipa vault-add my_vault --type standard
----------------------
Added vault "my_vault"
----------------------
  Vault name: my_vault
  Type: standard
  Owner users: idm_user
  Vault user: idm_user
```

중요

동일한 사용자가 첫 번째 사용자 자격 증명 모음을 만들어야 합니다. 사용자에 대한 첫 번째 자격 증명 모음을 생성하면 사용자의 자격 증명 모음 컨테이너도 생성됩니다. 생성 에이전트는 자격 증명 모음 컨테이너의 소유자가 됩니다.

예를 들어 **admin** 와 같은 다른 사용자가 **user1** 에 대해 첫 번째 사용자 자격 증명 모음을 만들고 사용자의 vault 컨테이너 소유자도 **admin** 이고 **user1** 은 사용자 자격 증명 모음에 액세스할 수 없거나 새 사용자 자격 증명 모음을 생성할 수 없습니다.

3. **secret.txt** 파일을 자격 증명 모음에 저장하려면 **ipa vault-archive** 명령을 **--in** 옵션과 함께 사용합니다.

```
$ ipa vault-archive my_vault --in secret.txt
----------------------------------
Archived data into vault "my_vault"
----------------------------------
```

2.2. 사용자 자격 증명 모음에서 시크릿 검색

IdM(Identity Management)으로 사용자 개인 자격 증명 모음의 시크릿을 사용자가 로그인한 모든 IdM 클라이언트로 검색할 수 있습니다.

다음 절차에 따라 **idm_user** 라는 IdM 사용자로 **my_vault** 라는 사용자 개인 자격 증명 모음에서 **idm_client.idm.example.com** 에 대한 시크릿을 검색합니다.

사전 요구 사항

- **idm_user** 는 **my_vault** 의 소유자입니다.
- **idm_user** 는 자격 증명 모음에 시크릿을 보관 했습니다.
- **my_vault** 는 표준 자격 증명 모음입니다. 즉, **idm_user** 는 자격 증명 모음의 콘텐츠에 액세스하기 위해 암호를 입력하지 않아도 됩니다.

절차

1. SSH to **idm_client** as **idm_user**:

```
$ ssh idm_user@idm_client.idm.example.com
```

2. **idm_user** 로 로그인 :

```
$ kinit user
```

3. 자격 증명 모음의 콘텐츠를 검색하고 **secret_exported.txt** 파일에 저장하려면 **ipa vault-retrieve --out** 명령을 --out 옵션과 함께 사용합니다.

```
$ ipa vault-retrieve my_vault --out secret_exported.txt
-------------------------------------
Retrieved data from vault "my_vault"
-------------------------------------
```

2.3. 추가 리소스

- Ansible을 사용하여 IdM 서비스 자격 증명 모음 관리: 시크릿 저장 및 검색을 참조하십시오.

3장. ANSIBLE을 사용하여 IDM 사용자 자격 증명 모음 관리: 시크릿 저장 및 검색

이 장에서는 Ansible 자격 증명 모음 모듈을 사용하여 Identity Management에서 사용자 자격 **증명 모음** 을 관리하는 방법을 설명합니다. 특히 사용자가 Ansible 플레이북을 사용하여 다음과 같은 세 가지 연속 작업을 수행하는 방법을 설명합니다.

- IdM에서 사용자 자격 증명 모음을 생성합니다 .
- 자격 증명 모음에 시크릿을 저장합니다 .
- 자격 증명 모음에서 시크릿을 검색합니다 .

사용자는 두 개의 다른 IdM 클라이언트에서 저장 및 검색을 수행할 수 있습니다.

사전 요구 사항

- 키 복구 기관(KRA) 인증서 시스템 구성 요소가 IdM 도메인에 있는 하나 이상의 서버에 설치되어 있습니다. 자세한 내용은 IdM에 키 복구 기관 설치를 참조하십시오.

3.1. ANSIBLE을 사용하여 IDM에 표준 사용자 자격 증명 모음이 있는지 확인

중요한 정보를 안전하게 저장하기 위해 Ansible 플레이북을 사용하여 하나 이상의 개인 자격 증명 모음 컨테이너를 생성하려면 다음 절차를 따르십시오. 아래 절차에 사용된 예제에서 **idm_user** 사용자는 **my_vault** 라는 표준 유형의 자격 증명 모음을 생성합니다. 표준 자격 증명 모음 유형을 사용하면 파일에 액세스할 때 **idm_user** 를 인증할 필요가 없습니다. **idm_user** 는 사용자가 로그인한 IdM 클라이언트에서 파일을 검색할 수 있습니다.

사전 요구 사항

- 절차의 단계를 실행하는 호스트인 Ansible 컨트롤러에 ansible-freeipa 패키지를 설치했습니다.
- **idm_user** 의 암호를 알고 있습니다.

절차

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** 디렉터리로 이동합니다.

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. 인벤토리 파일(예: **inventory.file**)을 생성합니다.

```
$ touch inventory.file
```

3. **inventory.file** 을 열고 **[ipaserver]** 섹션에서 구성할 IdM 서버를 정의합니다. 예를 들어, **server.idm.example.com** 을 구성하도록 Ansible에 지시하려면 다음을 입력합니다.

```
[ipaserver]
server.idm.example.com
```

4. **ensure-standard-vault-is-present.yml** Ansible 플레이북 파일을 복사합니다. 예를 들면 다음과 같습니다.

```
$ cp ensure-standard-vault-is-present.yml ensure-standard-vault-is-present-copy.yml
```

5. 편집을 위해 **ensure-standard-vault-is-present-copy.yml** 파일을 엽니다.
6. **ipavault** 작업 섹션에서 다음 변수를 설정하여 파일을 조정합니다.
   - **ipaadmin_principal** 변수를 **idm_user** 로 설정합니다.
   - **ipaadmin_password** 변수를 암호 **idm_user** 로 설정합니다.
   - **사용자** 변수를 **idm_user** 로 설정합니다.
   - **name** 변수를 **my_vault** 로 설정합니다.
   - **vault_type** 변수를 **표준** 으로 설정합니다.
     현재 예에 대해 수정된 Ansible 플레이북 파일입니다.

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_principal: idm_user
      ipaadmin_password: idm_user_password
      user: idm_user
      name: my_vault
      vault_type: standard
```

7. 파일을 저장합니다.
8. 플레이북을 실행합니다.

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-standard-vault-is-present-copy.yml
```

3.2. ANSIBLE을 사용하여 IDM의 표준 사용자 자격 증명 모음에 시크릿 보관

Ansible 플레이북을 사용하여 중요한 정보를 개인 자격 증명 모음에 저장하려면 다음 절차를 따르십시오. 사용된 예에서 **idm_user** 사용자는 **my_vault** 라는 자격 증명 모음에 민감한 정보가 있는 파일을 아카이브합니다 .

사전 요구 사항

- 절차의 단계를 실행하는 호스트인 Ansible 컨트롤러에 ansible-freeipa 패키지를 설치했습니다.
- **idm_user** 의 암호를 알고 있습니다.
- **idm_user** 는 **my_vault** 의 소유자 또는 적어도 멤버 사용자입니다.
- **my_vault** 에 보관할 시크릿인 **password.txt** 에 액세스할 수 있습니다.

절차

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** 디렉터리로 이동합니다.

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. 인벤토리 파일을 열고 구성할 IdM 서버가 **[ipaserver]** 섹션에 나열되어 있는지 확인합니다. 예를 들어, **server.idm.example.com** 을 구성하도록 Ansible에 지시하려면 다음을 입력합니다.

```
[ipaserver]
server.idm.example.com
```

3. **data-archive-in-symmetric-vault.yml** Ansible 플레이북 파일을 복사하고 "standard"으로 "symmetric"을 바꿉니다. 예를 들면 다음과 같습니다.

```
$ cp data-archive-in-symmetric-vault.yml data-archive-in-standard-vault-copy.yml
```

4. 편집을 위해 **data-archive-in-standard-vault-copy.yml** 파일을 엽니다.

5. **ipavault** 작업 섹션에서 다음 변수를 설정하여 파일을 조정합니다.

- **ipaadmin_principal** 변수를 **idm_user** 로 설정합니다.
- **ipaadmin_password** 변수를 암호 **idm_user** 로 설정합니다.
- **사용자** 변수를 **idm_user** 로 설정합니다.
- **name** 변수를 **my_vault** 로 설정합니다.
- 중요한 정보를 사용하여 **in** 변수를 파일의 전체 경로로 설정합니다.
- **action** 변수를 **member** 로 설정합니다.
  현재 예에 대해 수정된 Ansible 플레이북 파일입니다.

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_principal: idm_user
      ipaadmin_password: idm_user_password
      user: idm_user
      name: my_vault
      in: /usr/share/doc/ansible-freeipa/playbooks/vault/password.txt
      action: member
```

6. 파일을 저장합니다.

7. 플레이북을 실행합니다.

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file data-archive-in-standard-vault-copy.yml
```

3.3. ANSIBLE을 사용하여 IDM의 표준 사용자 자격 증명에서 시크릿 검색

Ansible 플레이북을 사용하여 사용자 개인 자격 증명 모음에서 시크릿을 검색하려면 다음 절차를 따르십시오. 아래 절차에 사용된 예제에서 **idm_user** 사용자는 **host01**. idm_user라는 IdM 클라이언트에 대한 표준 유형의 자격 증명 모음에서 중요한 데이터가 있는 파일을 검색합니다.**idm_user** 는 파일에 액세스할 때 인증할 필요가 없습니다. **idm_user** 는 Ansible이 설치된 모든 IdM 클라이언트에서 파일을 검색할 수 있습니다.

사전 요구 사항

- 다음 요구 사항을 충족하도록 Ansible 제어 노드를 구성했습니다.
  - Ansible 버전 2.14 이상을 사용하고 있습니다.
  - Ansible 컨트롤러에 **ansible-freeipa** 패키지가 설치되어 있습니다.
  - 이 예제에서는 *~/MyPlaybook/* 디렉터리에서 IdM 서버의 FQDN(정규화된 도메인 이름)을 사용하여 Ansible 인벤토리 파일을 생성했다고 가정합니다.
  - 이 예제에서는 secret.yml Ansible 자격 증명 모음이 **ipaadmin_password** 를 저장한다고 가정합니다.
- idm_user 의 암호를 알고 있습니다.
- idm_user 는 my_vault 의 소유자입니다.
- idm_user 는 my_vault 에 시크릿을 저장했습니다.
- Ansible은 시크릿을 검색하려는 IdM 호스트의 디렉터리에 쓸 수 있습니다.
- idm_user 는 시크릿을 검색할 IdM 호스트의 디렉터리에서 읽을 수 있습니다.

절차

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** 디렉터리로 이동합니다.

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. 인벤토리 파일을 열고 명확하게 정의된 섹션에서 시크릿을 검색할 IdM 클라이언트를 언급합니다. 예를 들어, host01.idm.example.com 에서 시크릿을 검색하도록 Ansible에 지시하려면 다음을 입력합니다.

```
[ipahost]
host01.idm.example.com
```

3. retrive-data-symmetric-vault.yml Ansible 플레이북 파일의 사본을 만듭니다. "symmetric"을 "standard"로 바꿉니다. 예를 들면 다음과 같습니다.

```
$ cp retrive-data-symmetric-vault.yml retrieve-data-standard-vault.yml-copy.yml
```

4. 편집을 위해 retrieve-data-standard-vault.yml-copy.yml 파일을 엽니다.

5. **hosts** 변수를 ipahost 로 설정하여 파일을 조정합니다.
6. **ipavault** 작업 섹션에서 다음 변수를 설정하여 파일을 조정합니다.
   - **ipaadmin_principal** 변수를 idm_user 로 설정합니다.
   - **ipaadmin_password** 변수를 암호 idm_user 로 설정합니다.
   - 사용자 변수를 idm_user 로 설정합니다.
   - **name** 변수를 my_vault 로 설정합니다.
   - 시크릿을 내보낼 파일의 전체 경로로 **out** 변수를 설정합니다.
   - **state** 변수를 검색된 로 설정합니다.
     현재 예에 대해 수정된 Ansible 플레이북 파일입니다.

```
---
- name: Tests
  hosts: ipahost
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_principal: idm_user
      ipaadmin_password: idm_user_password
      user: idm_user
      name: my_vault
      out: /tmp/password_exported.txt
      state: retrieved
```

7. 파일을 저장합니다.
8. 플레이북을 실행합니다.

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file retrieve-data-standard-vault.yml-copy.yml
```

검증 단계

1. user01 으로 host01 에 **SSH** 로 :

```
$ ssh user01@host01.idm.example.com
```

2. Ansible Playbook 파일의 **out** 변수로 지정된 파일을 확인합니다.

```
$ vim /tmp/password_exported.txt
```

내보낸 시크릿을 볼 수 있습니다.

- Ansible을 사용하여 IdM 자격 증명 및 사용자 시크릿을 관리하는 방법에 대한 자세한 내용은 **/usr/share/doc/ansible-freeipa/** 디렉토리에서 사용 가능한 README-vault.md Markdown 파일과 /usr/share/doc/ansible-freeipa /**playbooks/playbooks/vault/** 디렉토리에서 사용 가능한 샘

플 플레이북을 참조하십시오.

4장. IDM 서비스 시크릿 관리: 시크릿 저장 및 검색

이 섹션에서는 관리자가 **ansible-freeipa vault** 모듈을 사용하여 서비스 시크릿을 중앙 집중식 위치에 안전하게 저장하는 방법을 보여줍니다. 예제에 사용되는 자격 증명 모음 은 CloudEvent입니다. 즉, 관리자는 다음 단계를 수행해야 함을 의미합니다.

1. 예를 들어 **openssl** 유틸리티를 사용하여 개인 키를 생성합니다.
2. 개인 키를 기반으로 공개 키를 생성합니다.

관리자가 자격 증명 모음에 보관할 때 서비스 시크릿은 공개 키로 암호화됩니다. 이후 도메인의 특정 시스템에서 호스팅되는 서비스 인스턴스는 개인 키를 사용하여 시크릿을 검색합니다. 서비스와 관리자만 시크릿에 액세스할 수 있습니다.

보안이 손상되면 관리자는 서비스 자격 증명 모음에서 이를 교체한 다음 손상되지 않은 개별 서비스 인스턴스에 재배포할 수 있습니다.

사전 요구 사항

- 키 복구 기관(KRA) 인증서 시스템 구성 요소가 IdM 도메인에 있는 하나 이상의 서버에 설치되어 있습니다. 자세한 내용은 IdM에 키 복구 기관 설치를 참조하십시오.

이 섹션에는 다음 절차가 포함됩니다.

1. 대칭 자격 증명 모음에 IdM 서비스 시크릿 저장
2. IdM 서비스 인스턴스에 대한 서비스 시크릿 검색
3. 손상된 경우 IdM 서비스 자격 증명 모음 시크릿 변경

사용된 용어

절차의 경우:

- admin 은 서비스 암호를 관리하는 관리자입니다.
- private-key-to-externally-signed-certificate.pem 은 서비스 보안이 포함된 파일입니다(이 경우 외부 서명된 인증서에 대한 개인 키). 이 개인 키를 자격 증명 모음에서 시크릿을 검색하는 데 사용되는 개인 키와 혼동하지 마십시오.
- secret_vault 는 서비스에 대해 생성된 자격 증명 모음입니다.
- HTTP/webserver.idm.example.com 은 보안이 보관되는 서비스입니다.
- service-public.pem 은 password_vault 에 저장된 암호를 암호화하는 데 사용되는 서비스 공개 키입니다.
- service-private.pem 은 secret_vault 에 저장된 암호를 해독하는 데 사용되는 서비스 개인 키입니다.

4.1. 대칭 자격 증명 모음에 IDM 서비스 시크릿 저장

다음 절차에 따라 asymmetric 자격 증명 모음을 만들고 이를 사용하여 서비스 시크릿을 보관합니다.

사전 요구 사항

- IdM 관리자 암호를 알고 있습니다.

절차

1. 관리자로 로그인합니다.

```
$ kinit admin
```

2. 서비스 인스턴스의 공개 키를 가져옵니다. 예를 들어 **openssl** 유틸리티를 사용합니다.

   a. **service-private.pem** 개인 키를 생성합니다.

```
$ openssl genrsa -out service-private.pem 2048
Generating RSA private key, 2048 bit long modulus
.+++
.....................................+++
e is 65537 (0x10001)
```

   b. 개인 키를 기반으로 **service-public.pem** 공개 키를 생성합니다.

```
$ openssl rsa -in service-private.pem -out service-public.pem -pubout
writing RSA key
```

3. 서비스 인스턴스 자격 증명 모음으로 symmetric 자격 증명 모음을 생성하고 공개 키를 제공합니다.

```
$ ipa vault-add secret_vault --service HTTP/webserver.idm.example.com --type
asymmetric --public-key-file service-public.pem
----------------------------
Added vault "secret_vault"
----------------------------
Vault name: secret_vault
Type: asymmetric
Public key: LS0tLS1C...S0tLS0tCg==
Owner users: admin
Vault service: HTTP/webserver.idm.example.com@IDM.EXAMPLE.COM
```

   자격 증명 모음에 보관된 암호는 키로 보호됩니다.

4. 서비스 시크릿을 서비스 자격 증명 모음에 보관합니다.

```
$ ipa vault-archive secret_vault --service HTTP/webserver.idm.example.com --in
private-key-to-an-externally-signed-certificate.pem
---------------------------------
Archived data into vault "secret_vault"
---------------------------------
```

   이렇게 하면 서비스 인스턴스 공개 키로 보안이 암호화됩니다.

시크릿이 필요한 모든 서비스 인스턴스에 대해 이 단계를 반복합니다. 각 서비스 인스턴스에 대해 새 symmetric 자격 증명 모음을 만듭니다.

4.2. IDM 서비스 인스턴스에 대한 서비스 시크릿 검색

서비스 인스턴스를 사용하여 로컬에 저장된 서비스 개인 키로 서비스 자격 증명 모음 시크릿을 검색하려면 다음 절차를 따르십시오.

사전 요구 사항

- 서비스 주체를 소유하는 서비스 주체의 키 탭에 액세스할 수 있습니다(예: HTTP/webserver.idm.example.com).
- symmetric 자격 증명 모음을 생성하고 자격 증명 모음에 시크릿을 보관했습니다.
- 서비스 자격 증명 모음 시크릿을 검색하는 데 사용되는 개인 키에 액세스할 수 있습니다.

절차

1. 관리자로 로그인합니다.

```
$ kinit admin
```

2. 서비스에 대한 Kerberos 티켓을 받습니다.

```
# kinit HTTP/webserver.idm.example.com -k -t /etc/httpd/conf/ipa.keytab
```

3. 서비스 자격 증명 모음 암호를 검색합니다.

```
$ ipa vault-retrieve secret_vault --service HTTP/webserver.idm.example.com --private-key-file service-private.pem --out secret.txt
----------------------------------
Retrieved data from vault "secret_vault"
----------------------------------
```

4.3. 손상된 경우 IDM 서비스 자격 증명 모음 시크릿 변경

서비스 자격 증명 모음 시크릿을 변경하여 손상된 서비스 인스턴스를 분리하려면 다음 절차를 따르십시오.

사전 요구 사항

- IdM 관리자 암호를 알고 있습니다.
- 서비스 시크릿 을 저장하기 위해 symmetric 자격 증명 모음을 생성했습니다.
- 새 보안을 생성하고 이에 대한 액세스 권한이 있습니다(예: new-private-key-to-an-externally-signed-certificate.pem).

절차

1. 새 시크릿을 서비스 인스턴스 자격 증명 모음에 보관합니다.

```
$ ipa vault-archive secret_vault --service HTTP/webserver.idm.example.com --in new-private-key-to-an-externally-signed-certificate.pem
----------------------------------
Archived data into vault "secret_vault"
----------------------------------
```

이는 자격 증명 모음에 저장된 현재 시크릿을 덮어씁니다.

2. 확인되지 않은 서비스 인스턴스에서만 새 시크릿을 검색합니다. 자세한 내용은 IdM 서비스 인스턴스에 대한 서비스 시크릿 검색에서 참조하십시오.

4.4. 추가 리소스

- Ansible을 사용하여 IdM 서비스 자격 증명 모음 관리: 시크릿 저장 및 검색을참조하십시오.

5장. ANSIBLE을 사용하여 IDM 서비스 자격 증명 모음 관리: 시크릿 저장 및 검색

이 섹션에서는 관리자가 **ansible-freeipa vault** 모듈을 사용하여 서비스 시크릿을 중앙 집중식 위치에 안전하게 저장하는 방법을 보여줍니다. 예제에 사용되는 자격 증명 모음 은 CloudEvent입니다. 즉, 관리자는 다음 단계를 수행해야 함을 의미합니다.

1. 예를 들어 **openssl** 유틸리티를 사용하여 개인 키를 생성합니다.
2. 개인 키를 기반으로 공개 키를 생성합니다.

관리자가 자격 증명 모음에 보관할 때 서비스 시크릿은 공개 키로 암호화됩니다. 이후 도메인의 특정 시스템에서 호스팅되는 서비스 인스턴스는 개인 키를 사용하여 시크릿을 검색합니다. 서비스와 관리자만 시크릿에 액세스할 수 있습니다.

보안이 손상되면 관리자는 서비스 자격 증명 모음에서 이를 교체한 다음 손상되지 않은 개별 서비스 인스턴스에 재배포할 수 있습니다.

사전 요구 사항

- 키 복구 기관(KRA) 인증서 시스템 구성 요소가 IdM 도메인에 있는 하나 이상의 서버에 설치되어 있습니다. 자세한 내용은 IdM에 키 복구 기관 설치를 참조하십시오.

이 섹션에는 다음 절차가 포함됩니다.

- Ansible을 사용하여 IdM에 symmetric 서비스 자격 증명 모음이 있는지 확인
- Ansible을 사용하여 대칭 자격 증명 모음에 IdM 서비스 시크릿 저장
- Ansible을 사용하여 IdM 서비스의 서비스 시크릿 검색
- Ansible을 사용하여 손상된 경우 IdM 서비스 자격 증명 시크릿 변경

절차의 경우:

- admin 은 서비스 암호를 관리하는 관리자입니다.
- private-key-to-externally-signed-certificate.pem 은 서비스 보안이 포함된 파일입니다(이 경우 외부 서명된 인증서에 대한 개인 키). 이 개인 키를 자격 증명 모음에서 시크릿을 검색하는 데 사용되는 개인 키와 혼동하지 마십시오.
- secret_vault 는 서비스 시크릿을 저장하도록 생성된 자격 증명 모음입니다.
- HTTP/webserver1.idm.example.com 은 자격 증명 모음의 소유자입니다.
- HTTP/webserver2.idm.example.com 및 HTTP/webserver3.idm.example.com 은 자격 증명 모음 멤버 서비스입니다.
- service-public.pem 은 password_vault 에 저장된 암호를 암호화하는 데 사용되는 서비스 공개 키입니다.
- service-private.pem 은 secret_vault 에 저장된 암호를 해독하는 데 사용되는 서비스 개인 키입니다.

5.1. ANSIBLE을 사용하여 IDM에 SYMMETRIC 서비스 자격 증명 모음이 있는지 확인

중요한 정보를 안전하게 저장하기 위해 Ansible 플레이북을 사용하여 하나 이상의 개인 자격 증명 모음 컨테이너를 생성하려면 다음 절차를 따르십시오. 아래 절차에서 사용된 예에서는 관리자가 secret_vault 라는 symmetric 자격 증명 모음을 만듭니다. 이렇게 하면 자격 증명 모음 멤버가 개인 키를 사용하여 자격 증명 모음의 시크릿을 검색해야 합니다. 자격 증명 모음 멤버는 모든 IdM 클라이언트에서 파일을 검색할 수 있습니다.

사전 요구 사항

- 다음 요구 사항을 충족하도록 Ansible 제어 노드를 구성했습니다.
  - Ansible 버전 2.14 이상을 사용하고 있습니다.
  - Ansible 컨트롤러에 **ansible-freeipa** 패키지가 설치되어 있습니다.
  - 이 예제에서는 *~/MyPlaybook/* 디렉터리에서 IdM 서버의 FQDN(정규화된 도메인 이름)을 사용하여 Ansible 인벤토리 파일을 생성했다고 가정합니다.
  - 이 예제에서는 secret.yml Ansible 자격 증명 모음이 **ipaadmin_password** 를 저장한다고 가정합니다.
- IdM 관리자 암호를 알고 있습니다.

절차

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** 디렉터리로 이동합니다.

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. 서비스 인스턴스의 공개 키를 가져옵니다. 예를 들어 **openssl** 유틸리티를 사용합니다.
   a. **service-private.pem** 개인 키를 생성합니다.

```
$ openssl genrsa -out service-private.pem 2048
Generating RSA private key, 2048 bit long modulus
.+++
.....................................+++
e is 65537 (0x10001)
```

   b. 개인 키를 기반으로 **service-public.pem** 공개 키를 생성합니다.

```
$ openssl rsa -in service-private.pem -out service-public.pem -pubout
writing RSA key
```

3. 선택 사항: inventory 파일이 없는 경우 인벤토리 파일을 생성합니다(예: inventory.file).

```
$ touch inventory.file
```

4. 인벤토리 파일을 열고 **[ipaserver]** 섹션에서 구성할 IdM 서버를 정의합니다. 예를 들어, server.idm.example.com 을 구성하도록 Ansible에 지시하려면 다음을 입력합니다.

```
[ipaserver]
server.idm.example.com
```

5. ensure-asymmetric-vault-is-present.yml Ansible 플레이북 파일을 복사합니다. 예를 들면 다음과 같습니다.

```
$ cp ensure-asymmetric-vault-is-present.yml ensure-asymmetric-service-vault-is-present-copy.yml
```

6. 편집을 위해 ensure-asymmetric-vault-is-present-copy.yml 파일을 엽니다.

7. Ansible 컨트롤러에서 server.idm.example.com 서버에 service-public.pem 공개 키를 복사하는 작업을 추가합니다.

8. **ipavault** 작업 섹션에서 다음 변수를 설정하여 파일의 나머지 부분을 수정합니다.

   - **ipaadmin_password** 변수를 IdM 관리자 암호로 설정합니다.
   - name 변수를 사용하여 자격 증명 모음의 이름을 정의합니다(예: secret_vault).
   - **vault_type** 변수를 symmetric 으로설정합니다.
   - 서비스 변수를 자격 증명 모음을 소유하는 서비스 주체로 설정합니다(예: HTTP/webserver1.idm.example.com).
   - **public_key_file** 을 공개 키의 위치로 설정합니다.

   현재 예에 대해 수정된 Ansible 플레이북 파일입니다.

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false
  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Copy public key to ipaserver.
    copy:
      src: /path/to/service-public.pem
      dest: /usr/share/doc/ansible-freeipa/playbooks/vault/service-public.pem
      mode: 0600
  - name: Add data to vault, from a LOCAL file.
    ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      vault_type: asymmetric
      service: HTTP/webserver1.idm.example.com
      public_key_file: /usr/share/doc/ansible-freeipa/playbooks/vault/service-public.pem
```

9. 파일을 저장합니다.

10. 플레이북을 실행합니다.

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file ensure-asymmetric-service-vault-is-present-copy.yml
```

5.2. ANSIBLE을 사용하여 SYMMETRIC 자격 증명 모음에 멤버 서비스 추가

Ansible 플레이북을 사용하여 서비스 자격 증명 모음에 멤버 서비스를 추가하여 모두 자격 증명 모음에 저장된 시크릿을 검색할 수 있도록 다음 절차를 따르십시오. 아래 절차에 사용된 예제에서 IdM 관리자는 HTTP/webserver2.idm.example.com 및 HTTP/webserver3.idm.example.com 서비스 주체를 HTTP/webserver1.idm.example.com 에서 소유한 secret_vault 자격 증명 모음에 추가합니다.

사전 요구 사항

- 다음 요구 사항을 충족하도록 Ansible 제어 노드를 구성했습니다.
  - Ansible 버전 2.14 이상을 사용하고 있습니다.
  - Ansible 컨트롤러에 **ansible-freeipa** 패키지가 설치되어 있습니다.
  - 이 예제에서는 *~/MyPlaybook/* 디렉터리에서 IdM 서버의 FQDN(정규화된 도메인 이름)을 사용하여 Ansible 인벤토리 파일을 생성했다고 가정합니다.
  - 이 예제에서는 secret.yml Ansible 자격 증명 모음이 **ipaadmin_password** 를 저장한다고 가정합니다.
- IdM 관리자 암호를 알고 있습니다.
- 서비스 시크릿 을 저장하기 위해 symmetric 자격 증명 모음을 생성했습니다.

절차

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** 디렉터리로 이동합니다.

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. 선택 사항: inventory 파일이 없는 경우 인벤토리 파일을 생성합니다(예: inventory.file).

```
$ touch inventory.file
```

3. 인벤토리 파일을 열고 **[ipaserver]** 섹션에서 구성할 IdM 서버를 정의합니다. 예를 들어, server.idm.example.com 을 구성하도록 Ansible에 지시하려면 다음을 입력합니다.

```
[ipaserver]
server.idm.example.com
```

4. data-archive-in-asymmetric-vault.yml Ansible 플레이북 파일의 사본을 만듭니다. 예를 들면 다음과 같습니다.

```
$ cp data-archive-in-asymmetric-vault.yml add-services-to-an-asymmetric-vault.yml
```

5. 편집을 위해 data-archive-in-asymmetric-vault-copy.yml 파일을 엽니다.
6. **ipavault** 작업 섹션에서 다음 변수를 설정하여 파일을 수정합니다.
   - **ipaadmin_password** 변수를 IdM 관리자 암호로 설정합니다.
   - **name** 변수를 자격 증명 모음의 이름으로 설정합니다(예:secret_vault).

- 서비스 변수를 자격 증명 모음의 서비스 소유자(예:HTTP/webserver1.idm.example.com)로 설정합니다.
- services 변수를 사용하여 자격 증명 모음 시크릿에 액세스할 서비스를 정의합니다.
- action 변수를 member 로 설정합니다.
  현재 예에 대해 수정된 Ansible 플레이북 파일입니다.

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      service: HTTP/webserver1.idm.example.com
      services:
      - HTTP/webserver2.idm.example.com
      - HTTP/webserver3.idm.example.com
      action: member
```

7. 파일을 저장합니다.
8. 플레이북을 실행합니다.

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file add-services-to-an-asymmetric-vault.yml
```

5.3. ANSIBLE을 사용하여 대칭 자격 증명 모음에 IDM 서비스 시크릿 저장

나중에 서비스에서 검색할 수 있도록 Ansible 플레이북을 사용하여 서비스 자격 증명 모음에 시크릿을 저장하려면 다음 절차를 따르십시오. 아래 절차에서 사용된 예제에서 관리자는 secret_vault 라는 symmetric 자격 증명 모음에 보안이 포함된 PEM 파일을 저장합니다. 이렇게 하면 서비스가 개인 키를 사용하여 자격 증명 모음에서 시크릿을 검색해야 합니다. 자격 증명 모음 멤버는 모든 IdM 클라이언트에서 파일을 검색할 수 있습니다.

사전 요구 사항

- 다음 요구 사항을 충족하도록 Ansible 제어 노드를 구성했습니다.
  - Ansible 버전 2.14 이상을 사용하고 있습니다.
  - Ansible 컨트롤러에 ansible-freeipa 패키지가 설치되어 있습니다.
  - 이 예제에서는 *~/MyPlaybook/* 디렉터리에서 IdM 서버의 FQDN(정규화된 도메인 이름)을 사용하여 Ansible 인벤토리 파일을 생성했다고 가정합니다.
  - 이 예제에서는 secret.yml Ansible 자격 증명 모음이 ipaadmin_password 를 저장한다고 가정합니다.
- IdM 관리자 암호를 알고 있습니다.

- 서비스 시크릿 을 저장하기 위해 symmetric 자격 증명 모음을 생성했습니다.
- 보안은 Ansible 컨트롤러에 로컬로 저장됩니다(예: /usr/share/doc/ansible-freeipa/playbooks/vault/private-key-to-an-externally-signed-certificate.pem).

절차

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** 디렉터리로 이동합니다.

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. 선택 사항: inventory 파일이 없는 경우 인벤토리 파일을 생성합니다(예: inventory.file).

```
$ touch inventory.file
```

3. 인벤토리 파일을 열고 **[ipaserver]** 섹션에서 구성할 IdM 서버를 정의합니다. 예를 들어, server.idm.example.com 을 구성하도록 Ansible에 지시하려면 다음을 입력합니다.

```
[ipaserver]
server.idm.example.com
```

4. data-archive-in-asymmetric-vault.yml Ansible 플레이북 파일의 사본을 만듭니다. 예를 들면 다음과 같습니다.

```
$ cp data-archive-in-asymmetric-vault.yml data-archive-in-asymmetric-vault-copy.yml
```

5. 편집을 위해 data-archive-in-asymmetric-vault-copy.yml 파일을 엽니다.
6. **ipavault** 작업 섹션에서 다음 변수를 설정하여 파일을 수정합니다.
   - **ipaadmin_password** 변수를 IdM 관리자 암호로 설정합니다.
   - **name** 변수를 자격 증명 모음의 이름으로 설정합니다(예:secret_vault).
   - 서비스 변수를 자격 증명 모음의 서비스 소유자(예:HTTP/webserver1.idm.example.com)로 설정합니다.
   - **in** 변수를 "{{ lookup('file', 'private-key-to-an-externally-signed-certificate.pem') | b64encode }}" 로 설정합니다. 이렇게 하면 Ansible에서 IdM 서버가 아닌 Ansible 컨트롤러의 작업 디렉터리에서 개인 키를 사용하여 파일을 검색할 수 있습니다.
   - **action** 변수를 **member** 로 설정합니다.

   현재 예에 대해 수정된 Ansible 플레이북 파일입니다.

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
```

```
    name: secret_vault
    service: HTTP/webserver1.idm.example.com
    in: "{{ lookup('file', 'private-key-to-an-externally-signed-certificate.pem') |
b64encode }}"
    action: member
```

7. 파일을 저장합니다.

8. 플레이북을 실행합니다.

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file data-
archive-in-asymmetric-vault-copy.yml
```

5.4. ANSIBLE을 사용하여 IDM 서비스의 서비스 시크릿 검색

Ansible 플레이북을 사용하여 서비스 대신 서비스 자격 증명 모음에서 시크릿을 검색하려면 다음 절차를 따르십시오. 아래 절차에 사용된 예제에서 플레이북을 실행하면 secret_vault라는 secret_vault 의 시크릿을 사용하여 **PEM** 파일을 검색하고 Ansible 인벤토리 파일에 나열된 모든 호스트의 지정된 위치에 해당 파일을 **ipaservers**.

서비스는 keytab을 사용하여 IdM에 인증하고 개인 키를 사용하여 자격 증명 모음에 인증합니다. **ansible-freeipa** 가 설치된 IdM 클라이언트에서 서비스를 대신하여 파일을 검색할 수 있습니다.

사전 요구 사항

- 다음 요구 사항을 충족하도록 Ansible 제어 노드를 구성했습니다.
  - Ansible 버전 2.14 이상을 사용하고 있습니다.
  - Ansible 컨트롤러에 ansible-freeipa 패키지가 설치되어 있습니다.
  - 이 예제에서는 *~/MyPlaybook/* 디렉터리에서 IdM 서버의 FQDN(정규화된 도메인 이름)을 사용하여 Ansible 인벤토리 파일을 생성했다고 가정합니다.
  - 이 예제에서는 secret.yml Ansible 자격 증명 모음이 **ipaadmin_password** 를 저장한다고 가정합니다.
- IdM 관리자 암호를 알고 있습니다.
- 서비스 시크릿 을 저장하기 위해 symmetric 자격 증명 모음을 생성했습니다.
- 자격 증명 모음에 시크릿을 보관했습니다.
- Ansible 컨트롤러의 **private_key_file** 변수에서 지정한 위치에서 서비스 자격 증명 모음 시크릿을 검색하는 데 사용되는 개인 키를 저장했습니다.

절차

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** 디렉터리로 이동합니다.

```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. 선택 사항: inventory 파일이 없는 경우 인벤토리 파일을 생성합니다(예: inventory.file).

```
$ touch inventory.file
```

3. 인벤토리 파일을 열고 다음 호스트를 정의합니다.

   - **[ipaserver]** 섹션에 IdM 서버를 정의합니다.
   - **[webservers]** 섹션에서 시크릿을 검색할 호스트를 정의합니다. 예를 들어, Ansible에 webserver1.idm.example.com,webserver2.idm.example.com, webserver3.idm.example.com 에 시크릿을 검색하도록 지시하려면 다음을 입력합니다.

```
[ipaserver]
server.idm.example.com

[webservers]
webserver1.idm.example.com
webserver2.idm.example.com
webserver3.idm.example.com
```

4. retrieve-data-asymmetric-vault.yml Ansible 플레이북 파일의 사본을 만듭니다. 예를 들면 다음과 같습니다.

```
$ cp retrieve-data-asymmetric-vault.yml retrieve-data-asymmetric-vault-copy.yml
```

5. 편집을 위해 retrieve-data-asymmetric-vault-copy.yml 파일을 엽니다.

6. **ipavault** 작업 섹션에서 다음 변수를 설정하여 파일을 수정합니다.

   - **ipaadmin_password** 변수를 IdM 관리자 암호로 설정합니다.
   - **name** 변수를 자격 증명 모음의 이름으로 설정합니다(예:secret_vault).
   - 서비스 변수를 자격 증명 모음의 서비스 소유자(예:HTTP/webserver1.idm.example.com)로 설정합니다.
   - **private_key_file** 변수를 서비스 자격 증명 모음 시크릿을 검색하는 데 사용되는 개인 키의 위치로 설정합니다.
   - private-key-to-an-externally-signed-certificate.pem 시크릿을 검색하려는 IdM 서버의 위치로 **out** 변수를 설정합니다(예: 현재 작업 디렉터리).
   - **action** 변수를 **member** 로 설정합니다.
     현재 예에 대해 수정된 Ansible 플레이북 파일입니다.

```
---
- name: Retrieve data from vault
  hosts: ipaserver
  become: no
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Retrieve data from the service vault
    ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      service: HTTP/webserver1.idm.example.com
```

```
    vault_type: asymmetric
    private_key: "{{ lookup('file', 'service-private.pem') | b64encode }}"
    out: private-key-to-an-externally-signed-certificate.pem
    state: retrieved
```

7. IdM 서버에서 Ansible 컨트롤러로 데이터 파일을 검색하는 플레이북에 섹션을 추가합니다.

```
---
- name: Retrieve data from vault
  hosts: ipaserver
  become: no
  gather_facts: false
  tasks:
[...]
  - name: Retrieve data file
    fetch:
      src: private-key-to-an-externally-signed-certificate.pem
      dest: ./
      flat: yes
      mode: 0600
```

8. Ansible 컨트롤러의 검색된 private-key-to-externally-signed-certificate.pem 파일을 인벤토리 파일의 tag 섹션에 나열된 6443으로 전송하는 플레이북을 플레이북에 추가합니다.

```
---
- name: Send data file to webservers
  become: no
  gather_facts: no
  hosts: webservers
  tasks:
  - name: Send data to webservers
    copy:
      src: private-key-to-an-externally-signed-certificate.pem
      dest: /etc/pki/tls/private/httpd.key
      mode: 0444
```

9. 파일을 저장합니다.

10. 플레이북을 실행합니다.

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file retrieve-
data-asymmetric-vault-copy.yml
```

5.5. ANSIBLE을 사용하여 손상된 경우 IDM 서비스 자격 증명 시크릿 변경

서비스 인스턴스가 손상될 때 서비스 자격 증명 모음에 저장된 시크릿을 변경하기 위해 Ansible 플레이북을 재사용하려면 다음 절차를 따르십시오. 다음 예제의 시나리오에서는 webserver3.idm.example.com 에서 검색된 보안이 손상되었지만 secret을 저장하는 symmetric 자격

증명의 키가 손상되지 않은 것으로 가정합니다. 이 예제에서 관리자는 **symmetric** 자격 증명 모음에 시크릿을 저장할 때 사용된 **Ansible** 플레이북을 재사용하고 **symmetric** 자격 증명 에서 **IdM** 호스트로 시크릿을 검색합니다. 절차가 시작될 때 **IdM** 관리자는 대칭 자격 증명 모음에 새 시크릿을 사용하여 새 **PEM** 파일을 저장하고, 손상된 웹 서버인 **webserver3.idm.example.com** 에 새 시크릿을 검색하지 않도록 인벤토리 파일을 조정한 다음 두 절차를 다시 실행합니다.

사전 요구 사항

- 다음 요구 사항을 충족하도록 **Ansible** 제어 노드를 구성했습니다.
  - **Ansible** 버전 **2.14** 이상을 사용하고 있습니다.
  - **Ansible** 컨트롤러에 **ansible-freeipa** 패키지가 설치되어 있습니다.
  - 이 예제에서는 ~/***MyPlaybook***/ 디렉터리에서 **IdM** 서버의 **FQDN**(정규화된 도메인 이름)을 사용하여 **Ansible** 인벤토리 파일을 생성했다고 가정합니다.
  - 이 예제에서는 **secret.yml Ansible** 자격 증명 모음이 **ipaadmin_password** 를 저장한다고 가정합니다.
- **IdM** 관리자 암호를 알고 있습니다.
- 서비스 시크릿 을 저장하기 위해 **symmetric** 자격 증명 모음을 생성 했습니다.
- 손상된 이전 키를 교체하기 위해 **IdM** 호스트에서 실행되는 웹 서비스에 대한 새 **httpd** 키를 생성했습니다.
- 새 **httpd** 키는 **Ansible** 컨트롤러에 로컬로 저장됩니다(예: **/usr/share/doc/ansible-freeipa/playbooks/vault/private-key-to-an-externally-signed-certificate.pem**).

절차

1. **/usr/share/doc/ansible-freeipa/playbooks/vault** 디렉터리로 이동합니다.

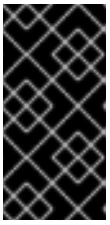
```
$ cd /usr/share/doc/ansible-freeipa/playbooks/vault
```

2. 인벤토리 파일을 열고 다음 호스트가 올바르게 정의되었는지 확인합니다.

- [ipaserver] 섹션의 IdM 서버.
- [webservers] 섹션에서 시크릿을 검색할 호스트입니다. 예를 들어, Ansible에 webserver1.idm.example.com 및 webserver2.idm.example.com 에 시크릿을 검색하도록 지시하려면 다음을 입력합니다.

```
[ipaserver]
server.idm.example.com

[webservers]
webserver1.idm.example.com
webserver2.idm.example.com
```



중요

현재 예제 webserver3.idm.example.com 에서 목록에 손상된 웹 서버가 포함되어 있지 않은지 확인합니다.

3. 편집을 위해 data-archive-in-asymmetric-vault-copy.yml 파일을 엽니다.

4. ipavault 작업 섹션에서 다음 변수를 설정하여 파일을 수정합니다.

- ipaadmin_password 변수를 IdM 관리자 암호로 설정합니다.
- name 변수를 자격 증명 모음의 이름으로 설정합니다(예: secret_vault).
- 서비스 변수를 자격 증명 모음의 서비스 소유자(예: HTTP/webserver.idm.example.com)로 설정합니다.
- in 변수를 "{{ lookup('file', 'new-private-key-to-an-externally-signed-certificate.pem') | b64encode }}" 로 설정합니다. 이렇게 하면 Ansible에서 IdM 서버가 아닌 Ansible 컨트롤러의 작업 디렉터리에서 개인 키를 사용하여 파일을 검색할 수 있습니다.

- action 변수를 member 로 설정합니다.

  현재 예에 대해 수정된 Ansible 플레이북 파일입니다.

```
---
- name: Tests
  hosts: ipaserver
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      service: HTTP/webserver.idm.example.com
      in: "{{ lookup('file', 'new-private-key-to-an-externally-signed-certificate.pem') |
b64encode }}"
      action: member
```

5. 파일을 저장합니다.

6. 플레이북을 실행합니다.

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file data-
archive-in-asymmetric-vault-copy.yml
```

7. 편집을 위해 retrieve-data-asymmetric-vault-copy.yml 파일을 엽니다.

8. ipavault 작업 섹션에서 다음 변수를 설정하여 파일을 수정합니다.

- ipaadmin_password 변수를 IdM 관리자 암호로 설정합니다.
- name 변수를 자격 증명 모음의 이름으로 설정합니다(예: secret_vault).
- 서비스 변수를 자격 증명 모음의 서비스 소유자(예: HTTP/webserver1.idm.example.com)로 설정합니다.
-

private_key_file 변수를 서비스 자격 증명 모음 시크릿을 검색하는 데 사용되는 개인 키의 위치로 설정합니다.

- **new-private-key-to-an-signed-certificate.pem** 시크릿을 검색하려는 **IdM** 서버의 위치로 설정합니다(예: 현재 작업 디렉터리).

- **action** 변수를 **member** 로 설정합니다.

현재 예에 대해 수정된 **Ansible** 플레이북 파일입니다.

```
---
- name: Retrieve data from vault
  hosts: ipaserver
  become: no
  gather_facts: false

  vars_files:
  - /home/user_name/MyPlaybooks/secret.yml
  tasks:
  - name: Retrieve data from the service vault
    ipavault:
      ipaadmin_password: "{{ ipaadmin_password }}"
      name: secret_vault
      service: HTTP/webserver1.idm.example.com
      vault_type: asymmetric
      private_key: "{{ lookup('file', 'service-private.pem') | b64encode }}"
      out: new-private-key-to-an-externally-signed-certificate.pem
      state: retrieved
```

9. **IdM** 서버에서 **Ansible** 컨트롤러로 데이터 파일을 검색하는 플레이북에 섹션을 추가합니다.

```
---
- name: Retrieve data from vault
  hosts: ipaserver
  become: true
  gather_facts: false
  tasks:
[...]
  - name: Retrieve data file
    fetch:
      src: new-private-key-to-an-externally-signed-certificate.pem
      dest: ./
      flat: yes
      mode: 0600
```

10. 검색된 **new-private-key-to-externally-signed-certificate.pem** 파일을 **Ansible** 컨트롤러에

서 **inventory** 파일의 **webservers** 섹션에 나열된 **6443**으로 전송하는 섹션을 플레이북에 추가합니다.

```
---
- name: Send data file to webservers
  become: true
  gather_facts: no
  hosts: webservers
  tasks:
  - name: Send data to webservers
    copy:
      src: new-private-key-to-an-externally-signed-certificate.pem
      dest: /etc/pki/tls/private/httpd.key
      mode: 0444
```

11. 파일을 저장합니다.

12. 플레이북을 실행합니다.

```
$ ansible-playbook --vault-password-file=password_file -v -i inventory.file retrieve-data-asymmetric-vault-copy.yml
```

5.6. 추가 리소스

- **/usr/share/doc/ansible-freeipa/** 디렉토리에서 **README-vault.md** 마크다운 파일을 참조하십시오.

- **/usr/share/doc/ansible-freeipa/playbooks/vault/** 디렉터리에서 샘플 플레이북을 참조하십시오.