Laboratuvar: Gizli Bilgileri Güvenli Bir Şekilde Saklama

Tahmini Çaba: 35 dk.

Giriş

Gizli bilgiler, özel verileri korumak ve insanların uygulamanıza yetkisiz erişim sağlamasını önlemek için güvenli bir şekilde saklanmalıdır. Vault, gizli bilgileri yönetmek için token tabanlı bir depolama çözümüdür. Vault için dört güvenlik aşaması vardır:

1. Kimlik Doğrulama
2. Doğrulama
3. Yetkilendirme
4. Erişim

Vault'un kullanıcıları nasıl kimlik doğruladığını ve gizli bilgileri nasıl yönettiğini Python kullanarak daha yakından inceleyeceğiz.

Öğrenme Hedefleri

Bu laboratuvar sırasında şunları yapacaksınız:

- Terminal kullanarak Vault UI'yi yapılandırma, başlatma ve erişme
- Vault Python API'sini (`hvac`) kullanarak Vault'ta anahtar-değer gizli bilgilerini okuma, yazma ve silme

Vault'ı Kurun

Bu laboratuvar boyunca Hashicorp'tan bir araç olan Vault'ı kullanacaksınız. Vault'ı herhangi bir sistemde kullanmadan önce, onu kurmalısınız.

Göreviniz

Vault'ı kurmak için terminalde aşağıdaki komutları çalıştırın:

```
wget -O- https://apt.releases.hashicorp.com/gpg | gpg --dearmor | sudo tee /usr/share/keyrings/hashicorp-archive-keyring.gpg >/dev/null
echo "deb [signed-by=/usr/share/keyrings/hashicorp-archive-keyring.gpg] https://apt.releases.hashicorp.com $(lsb_release -cs) main" | sudo tee /etc/apt/sources.list.d/hashicorp.list
sudo apt update
sudo apt install vault
```

Bunun doğru bir şekilde kurulduğunu, bir terminalde `vault` komutunu çalıştırarak doğrulayabilirsiniz:

```
vault
```

Bu, araçla kullanabileceğiniz komutların bir listesini göstermelidir.

Not: `vault` komutunu çalıştırdıktan sonra herhangi bir hata ile karşılaşırsanız, Vault kurulumunu kaldırmayı ve aşağıdaki komutları kullanarak yeniden yüklemeyi deneyin:

```
sudo apt remove vault
```

```
sudo apt install vault
```

Geliştirici Sunucusunu Kurma

Artık, Vault için Geliştirici Sunucusunu başlatacaksınız, böylece işlevlerin nasıl göründüğüne dair bir fikir edinebilirsiniz. Gerçekte, bu çok güvenli değildir, ancak aracı yerel olarak keşfetmek için faydalıdır. Yine de, tüm veriler şifrelenir ve Geliştirici Sunucusu için bellekte saklanır.

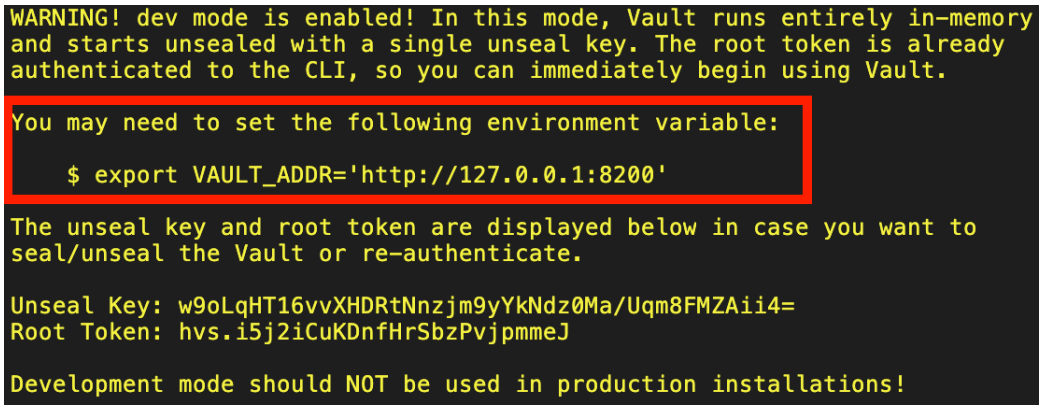
Göreviniz

Geliştirme modunda vault sunucusunu çalıştırmak için `-dev` bayrağı ile `vault server` komutunu kullanın:

```
vault server -dev
```

Çıktının sonunda bize daha fazla bilgi veren bir uyarı mesajı görmelisiniz.

Öncelikle, ortam değişkeninin ayarlanması gerektiğini bildirir. Bu, Vault'a yerel olarak erişebileceğiniz adrestir:

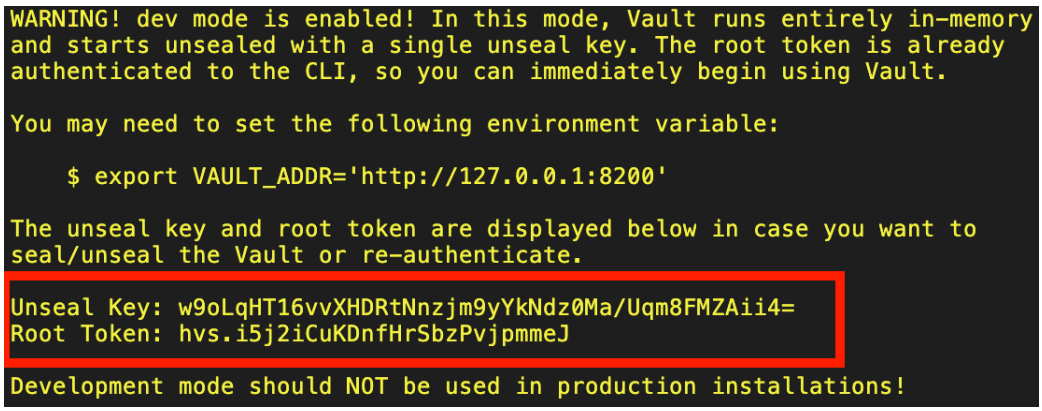


Ortam Değişkenini Ayarlayın

Sonra, üst menüden `Terminal > Yeni Terminal` seçeneğini kullanarak yeni bir terminal açmalısınız ve Vault için 8200 numaralı portu belirtmek üzere aşağıdaki shell komutunu çalıştırmalısınız:

```
export VAULT_ADDR='http://127.0.0.1:8200'
```

Açık anahtar ve kök jetonu (aşağıda vurgulanmış) çıktı mesajında da verilmiştir:



Bunlar kullanıcıyı kimlik doğrulamak için kullanılır - Vault'taki güvenliğin ilk adımıdır. Kök jetonu, sunucuyu her başlattığınızda **yeniden üretilir**, bu nedenle **en güncel kök jetonuna** göre **aşağıdaki ifadeyi düzenleyip çalıştırdığınızdan emin olun**:

```
export VAULT_TOKEN="YOUR ROOT TOKEN HERE"
```

Not: Terminalde değerleri kopyalamak ve yapıştırmak için sağ tıklayın.

Sunucuyu Kontrol Et

Sunucunun doğru çalışıp çalışmadığını bilmek önemlidir, bu nedenle bu adımda `vault status` komutunu kullanarak sunucu durumunu kontrol edeceksiniz.

Göreviniz

Sunucunun çalıştığını kontrol etmek için `vault status` komutunu kullanın:

```
vault status
```

```
Key             Value
---             -----
Seal Type       shamir
Initialized     true
Sealed          false
Total Shares    1
Threshold       1
Version         1.11.2
Build Date      2022-07-29T09:48:47Z
Storage Type    inmem
Cluster Name    vault-cluster-8bf1e69c
Cluster ID      4397bb37-f34e-3627-db08-3001f542936b
HA Enabled      false
```

Şu anda bunun ne anlama geldiğini bilmenize gerek yok - sadece bunun, sırlarınızı yönetmek için çalışan bir vault sunucumuz olduğunu gösterdiğini anlayın!

Kasa Arayüzüne Erişim

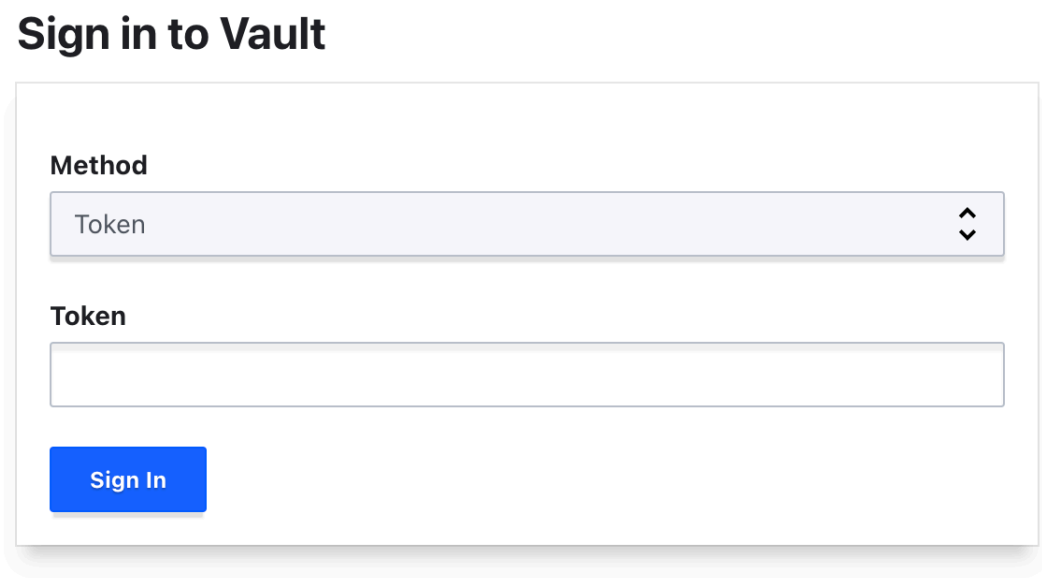
Kasa'ya erişmenin çeşitli yolları vardır. Kullanıcı dostu bir yol, yerel sunucunuzda web Kullanıcı Arayüzü (UI) başlatmaktır.

Kasa UI'yi Başlat

Bu, uygulama için bu IDE'de yeni bir sekme açacaktır. Karşınıza bir giriş sayfası çıkacaktır. Buradan, giriş yapmak için `VAULT_TOKEN` ortam değişkenine kopyaladığınız kök token'ı girebilirsiniz.

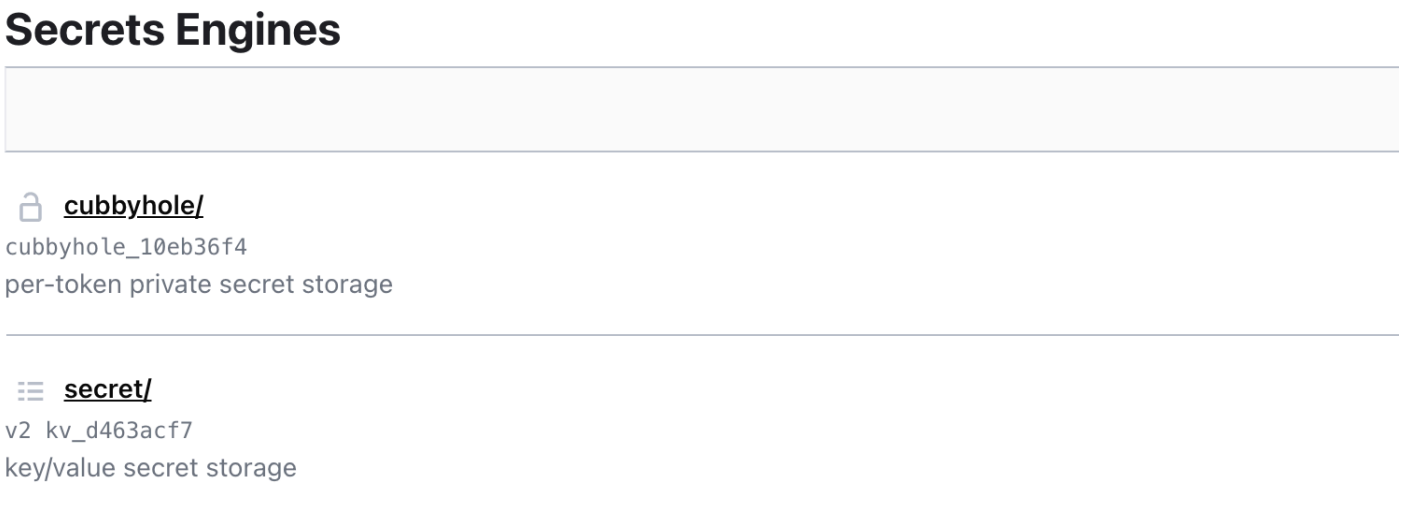
Token'ınızı bir terminalden şu şekilde görüntüleyebilirsiniz:

```
echo $VAULT_TOKEN
```

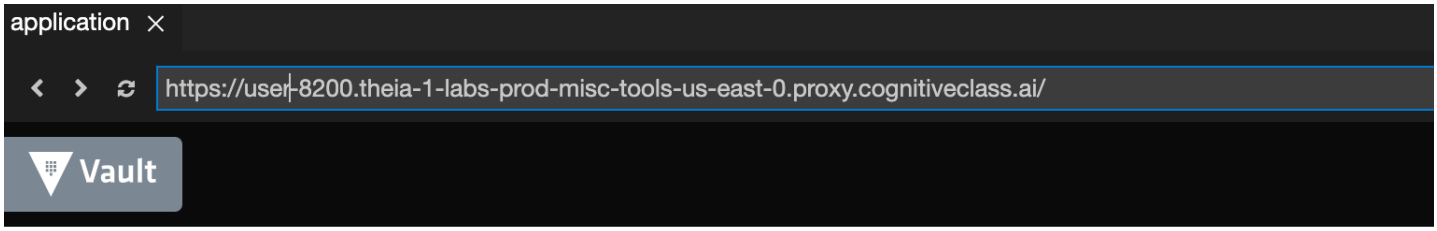


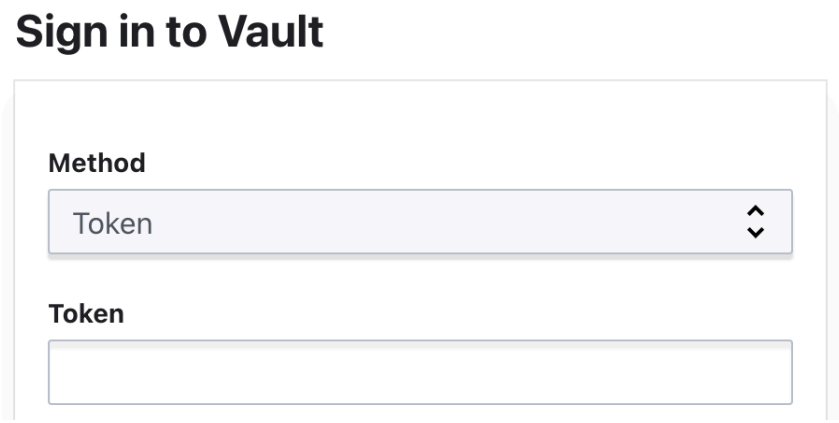
Vault web UI

Giriş yaptıktan sonra, `secret/` içinde saklanan sırları erişmek için bağlantıya tıklayabilirsiniz:



Alternatif olarak, sağ üst köşedeki butona (aşağıdaki sarı kutuyla vurgulanan) tıklayarak sayfayı yeni bir tarayıcı sekmesinde açabilirsiniz:





HVAC Kurulumu

Varsayılan olarak, Vault'un geliştirme sunucusu, `secret/` yolunda KV v2 Secrets Engines içerir ve sırları yapılandırılmış fiziksel depolama alanında saklar. Sırlar, arka uç depolamaya yazılmadan önce şifrelenir, bu nedenle Vault olmadan bu değerleri asla çözemez.

Videoda gösterildiği gibi, `hvac` kütüphanesinin yardımıyla Python kullanarak sunucuya sırları okuma ve yazmayı öğreneceksiniz. Bunu yapmadan önce, Python Paket Yöneticisi (pip) kullanarak `hvac` Python paketini kurmalıyız.

Göreviniz

Aşağıdaki komutla `hvac` paketini kurmak için `pip` komutunu kullanın:

```
python3 -m pip install hvac
```

Not: Gizli anahtarları yönetmek için Python API yerine Vault komut satırını da kullanabilirsiniz. Daha fazla bilgi için buraya göz atın.

Gizli Bilgiyi Vault'a Yaz

Artık gizli bilgileri yazmaya başlayarak vault'u kullanmaya hazırsınız.

Başlamadan önce, Vault'tan gizli bilgileri yazmak, okumak ve silmek için kullanılacak Python dosyasını edinmelisiniz:

Göreviniz

1. Aşağıdaki `wget` komutunu kullanarak `read_write_vault.py` programını indirin:

```
wget https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/IBM-CD0267EN-SkillsNetwork/labs/module3/read_write_vault.py
```

2. Dosyayı incelemek için IDE'de açalım:

Open **read_write_vault.py** in IDE

Programı İnceleyin

Bu programda dört fonksiyon bulunmaktadır:

```
def init_server():
def write_secret(secret_path, key, value):
def read_secret(secret_path):
def delete_secret(secret_path):
```

İlk olarak, belirtilen URL ile yeni bir hvac istemci örneği oluşturan `init_server()` (satır 4) fonksiyonunu kullanacağız. Bu laboratuvarı kendi bilgisayarınızda çalıştırırsanız, bu `http://localhost:8200` olacaktır; burası Vault geliştirme sunucunuzun bulunduğu yerdir. Cloud IDE ortamında ise uygulamayı başlatırken kullanılan URL'yi kullanacağız. İstemci örneğini daha sonra kullanmak üzere geri döndürüyoruz.

Bir gizli bilgi yazma

Şimdi, bir gizli bilgi yazmayı deneyin! Videodan bildiğiniz gibi, `client.secrets.kv.v2.create_or_update_secret()` fonksiyonu, istemci için `secret/path` altındaki bir anahtar/değer çifti yazmak için kullanılabilir. Bunu, oluşturulan yeni gizli bilginin bilgilerini de yazdıran `write_secret()` (satır 12) adlı özel bir fonksiyonda kullanıyoruz. Satır 35-65'te belirtildiği gibi, bu fonksiyonu doğrudan çağırabilirsiniz.

1. `python3` kullanarak `read_write_vault.py` programını `write_secret` fonksiyon adı ve `myapp alice mypassword` parametreleri ile çağırın:

```
python3 read_write_vault.py write_secret myapp alice mypassword
```

Bu, yolu `secret/myapp` olarak ayarlayacak ve burada gizli anahtar/değer çifti `alice=mypassword` yazacaktır.

Sonuçlar

Çıktı aşağıdaki gibi görünmelidir:

```
theia@theiadocker-rofrano:/home/project$ python3 read_write_vault.py write_secret myapp alice my
 Is client authenticated: True
{'request_id': '753fe955-1539-fb4b-6efb-c7edd53f257b', 'lease_id': '', 'renewable': False, 'leas
 {'created_time': '2022-09-09T18:22:11.239145121Z', 'custom_metadata': None, 'deletion_time': ''
version': 1}, 'wrap_info': None, 'warnings': None, 'auth': None}
```

Yazılı Sırrı Tekrar Kontrol Et

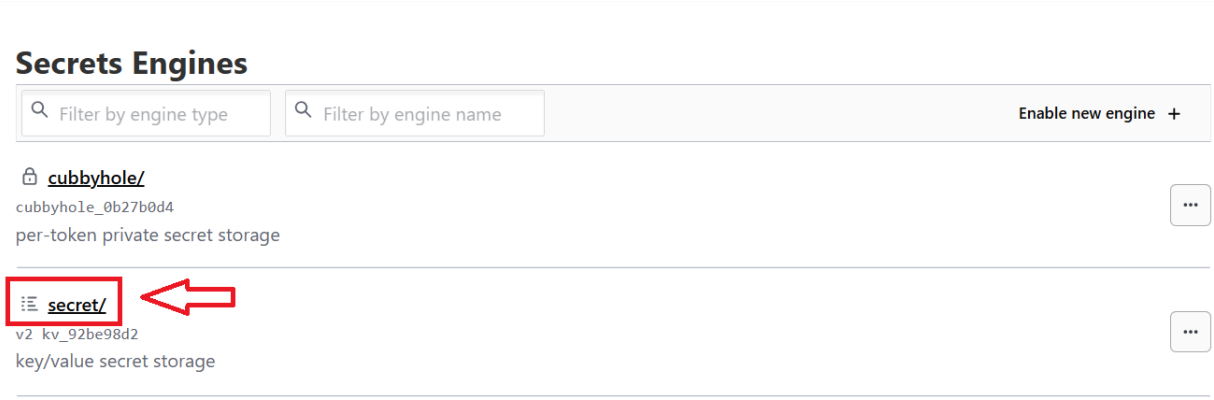
Anahtar/değer çiftini yazdıktan sonra, sırrın doğru yerde saklandığını tekrar kontrol etmek için Vault UI'yi açalım:

Göreviniz

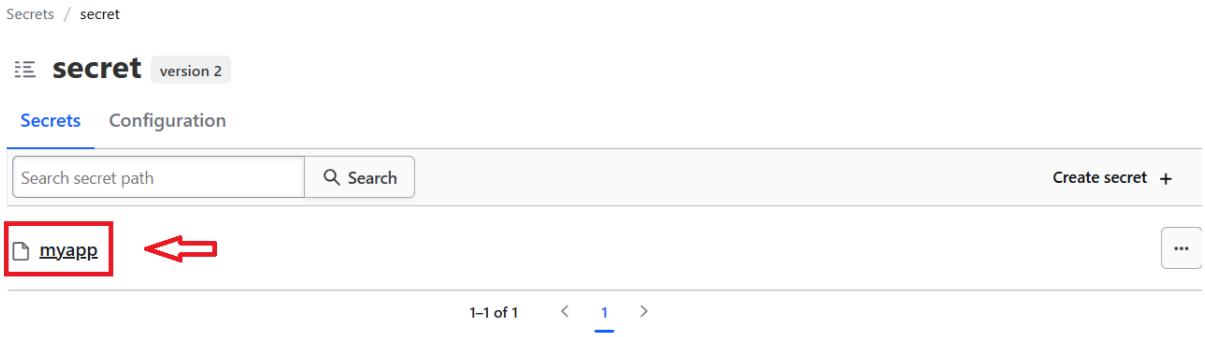
Vault UI'ye geri dönün.

Vault UI'yi Başlat

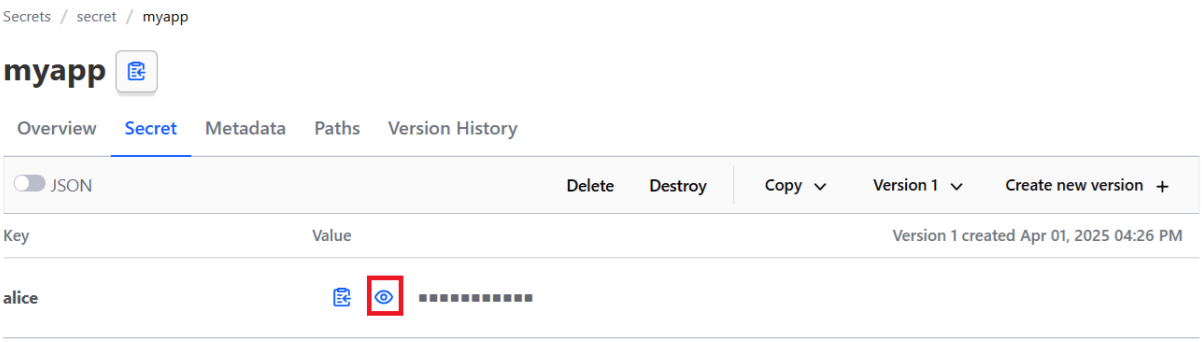
1. `secret/` bağlantısına tıklayın. Bu, tüm sırları gösterecektir.



2. Ardından `myapp/` bağlantısına tıklayın. Bu, `myapp` altındaki sırları gösterecektir.



3. Saklanan `alice` sırrını görmelisiniz ve göz simgesine tıkladığınızda `mypassword` değerini göreceksiniz.



Vault'tan Sırrı Okuma

Bu adımda, sırrı vault'tan geri okuyacaksınız. Bunu Python ile programatik olarak yapmak için `client.secrets.kv.v2.read_secret_version()` fonksiyonunu kullanırsınız. Bu, `read_secret()` (21. satırda) içinde yer almaktadır ve sırrın yolunu argüman olarak alır ve okunan sırrın detaylarını çıktılar.

Göreviniz

Terminalde `secret/myapp` yolundan bir sırrı okumak için aynı sözdizimini kullanacaksınız.

1. `read_secret` fonksiyonunu çağırarak `myapp` parametresini geçerek `read_write_vault.py` programını çalıştırmak için `python3` kullanın:

```
python3 read_write_vault.py read_secret myapp
```

Sonuçlar

Çıktı, `secret/myapp` altında saklanan tüm veriler hakkında bilgi döndürür. Özellikle, yeni yazdığınız anahtar/değer çiftini görebilirsiniz:

```
theia@theiadocker-rofrano:/home/project$ python3 read_write_vault.py read_secret myapp
 Is client authenticated: True
{'request_id': '80814815-1b6d-696d-df82-13b389e0c9b0', 'lease_id': '', 'renewable': False, 'leas
 {'data': {'alice': 'mypassword'}, 'metadata': {'created_time': '2022-09-09T18:22:11.239145121Z'
ne, 'deletion_time': '', 'destroyed': False, 'version': 1}}, 'wrap_info': None, 'warnings': None
```

Not: Geçersiz bir yoldan okuma yaparsanız, hvac bir hata döndürecektir.

Vault'tan Gizli Bilgiyi Silme

Vault'tan gizli bilgileri silmek, `client.secrets.kv.v2.delete_latest_version_of_secret()` ile de mümkündür; bu, belirttiğiniz bir yolda gizli bilginin en son sürümünü silecektir.

Terminalde `delete_secret()` (satır 29) fonksiyonunu benzer bir sözdizimi ile çağırabilirsiniz.

Göreviniz

1. `delete_secret` fonksiyonunu `myapp` parametresi ile çağırarak `read_write_vault.py` programını `python3` ile çalıştırın:

```
python3 read_write_vault.py delete_secret myapp
```

Program bu fonksiyon için herhangi bir şey yazdırmaz veya döndürmez, ancak silindiğini kontrol etmek için Vault UI'ye geri dönebilirsiniz.

Silinen Sırrı Kontrol Et

Her zaman Vault web UI'sını kullanarak hangi sırların mevcut olduğunu kontrol edebilirsiniz, ancak bunu programatik olarak bir python programı kullanarak `myapp` için sırrın silinip silinmediğini kontrol edeceğiz.

Göreviniz

1. `read_secret` fonksiyonunu çağırarak `myapp` parametresini geçerek `read_write_vault.py` programını çalıştırmak için `python3` kullanın:

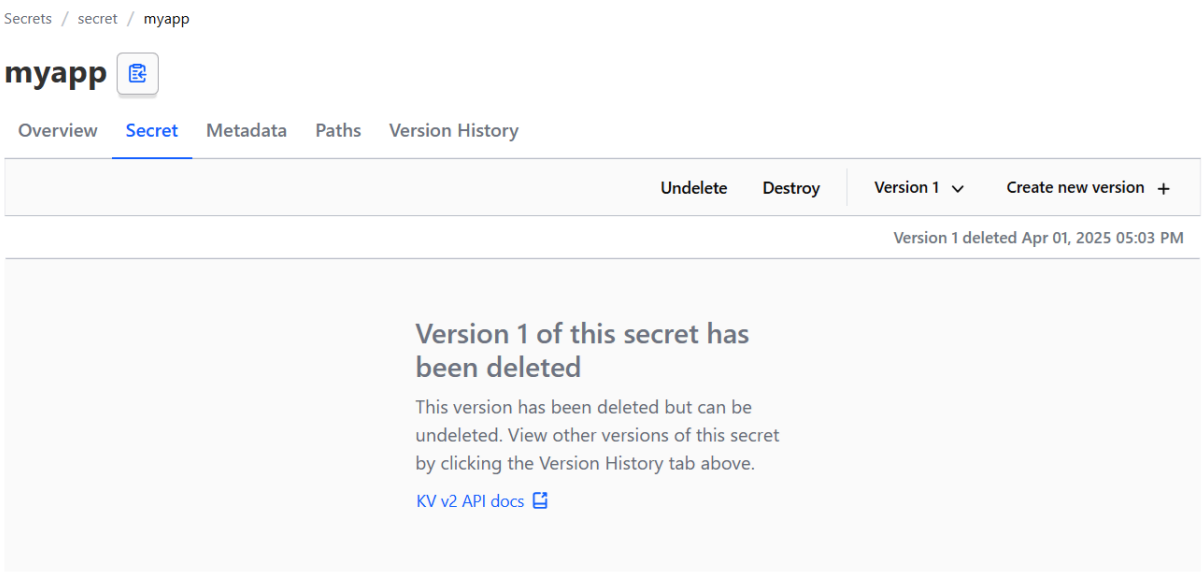
```
python3 read_write_vault.py read_secret myapp
```

Sonuçlar

`InvalidPath` hatası görmelisiniz, ayrıca sırrın ne zaman silindiğini gösteren "deletion_time" adlı bir anahtara sahip bir sözlük de görünecektir:

Vault UI'yi Kontrol Edin

Vault UI'ye geri döndüğünüzde, bu değişikliğin de yansıtıldığını görebilirsiniz.



Sonuç

Tebrikler, Store Secrets Securely laboratuvarını başarıyla tamamladınız.

Bu laboratuvarda, Hashicorp Vault'u nasıl kuracağınızı, bir gizli anahtar oluşturmayı ve bunu vault'ta güvenli bir şekilde saklamayı öğrendiniz. Ayrıca, vault'tan bir gizli anahtarı nasıl alacağınızı ve artık ihtiyaç duymadığınızda gizli anahtarları nasıl sileceğinizi de öğrendiniz. Tüm bunlar, bu kodu bir sonraki projenize entegre edebilmeniz için Python kodu kullanarak gerçekleştirildi.

Sonraki Adımlar

Neredeyse tüm projelerin bir tür gizli bilgisi vardır. Vault'u nasıl kullanacağınızı bildiğinize göre, bir sonraki meydan okumanız, kimlik bilgilerini veya diğer gizli bilgileri korumak için bunu projelerinizden birinde kullanmaya çalışmak olacaktır ve bunları güvenli bir şekilde saklamaktır.

Author(s)

Cindy Huang IBM Skills Network ekibinde bir veri bilimi uzmanıdır. Kullanıcı deneyimini geliştirmek için makine öğrenimine tutkulu olup, özellikle hesaplamalı dilbilim alanında çalışmaktadır.

Other Contributor(s)

Sam Prokopchuk

John J. Rofrano IBM T.J. Watson Araştırma Merkezi'nde Kıdemli Teknik Personel Üyesi ve DevOps Şampiyonu, ayrıca New York Üniversitesi Courant Enstitüsü ve Stern İşletme Okulu'nda DevOps ve Agile Yöntemleri üzerine lisansüstü dersi veren bir Öğretim Üyesidir.