Available online www.ejaet.com

European Journal of Advances in Engineering and Technology, 2023, 10(6):53-55

Research Article ISSN: 2394 - 658X

Ensuring the Secure Handling of Customer Secrets in Salesforce ISV Apps Utilizing Protected Hierarchy Custom Settings

Praveen Kotholliparambil Haridasan

ABSTRACT

This article provides a guide for Independent Software Vendors (ISVs) developing applications on the Salesforce 1 Platform and highlights the importance of handling customer generated confidential information. It underscores the importance of the Salesforce security review process in ensuring that software apps by these companies comply with security standards when working with data like API keys and passwords. The emphasis is on leveraging Protected Hierarchy Custom Settings within Salesforce for data management and security measures. The article also explains types of custom settings and their role, in maintaining data security. In addition to that are examples provided such as an Apex class for accessing settings and ideas for creating personalized user interfaces to manage these configurations effectively highlighted in the content which emphasizes the importance for ISVs partners to adhere to the security guidelines set by Salesforce in order to successfully navigate the AppExchange Security Review and present their applications, on the platform with impact.

Keywords: Independent Software Vendors (ISVs), Secure Handling of Customer Secrets, Protected Hierarchy Custom Settings

---

INTRODUCTION

Independent Software Vendors (ISVs) who develop applications on the Salesforce 1 Platform should ensure their solutions meet Salesforce security standards before publishing their package listings in the AppExchange. Salesforce ensures that ISV applications are secured for distribution through a rigorous security review process. One common mistake ISV partners make in their applications is mishandling sensitive customer data like API Keys, passwords, or tokens—often called "Secrets."

Improper management of these Secrets can have serious consequences. Unauthorized access to sensitive information can lead to data breaches, regulatory non-compliance, legal challenges, and significant harm to the ISV's reputation. Additionally, applications that fail to manage these Secrets securely are rejected during Salesforce's AppExchange Security Review, preventing them from being listed on the marketplace.

Secrets can be broadly categorized into customer-managed secrets and ISV-managed secrets. Each type comes with distinct responsibilities and requires specific strategies.

Customer-Managed Secrets

These are secrets that the end customer is responsible for managing. Examples include API keys, tokens, or passwords that customers generate and maintain. The ISV's application may need to access these secrets to integrate with the customer's other systems or services, but the customer retains control over their creation, storage, and lifecycle management.

Examples:

- API keys for third-party services that customers enter into the ISV application.
- Customers generate OAuth tokens to allow the ISV application to interact with the customer's infrastructure.

ISV-Managed Secrets

These are secrets that the ISV is responsible for managing. These might include the application's internal secrets, such as service account credentials, API keys for ISV-owned services, or encryption keys used within the application.

Examples:

- API keys for services that ISV controls are used within the application to enable certain functionalities.
- Database credentials or encryption keys are stored securely within the application's infrastructure.

This article focuses on securely managing Customer-Managed Secrets using Protected Hierarchy Custom Settings.

PROTECTED HIERARCHY CUSTOM SETTINGS

Salesforce Custom Settings offers an efficient solution for creating and accessing custom data in a Salesforce Org. One notable characteristic of Salesforce Custom Settings is their cache behavior. Salesforce caches Custom Settings data, making it highly efficient for read operations. This is particularly advantageous for Hierarchy Custom Settings, as it ensures that the data retrieval is fast and does not count against SOQL query limits. Custom Settings are recommended for storing application configurations or settings, Enabling you to configure and manage application behavior without requiring complex code. We should use standard or custom objects in Salesforce to store transactional data, not Custom Settings.

There are two primary types of Custom Settings: Hierarchy Custom Settings and List Custom Settings. Hierarchy Custom Settings allow you to store and retrieve data based on the Org, profile, and user levels, reflecting a hierarchical structure. In contrast, List Custom Settings store and retrieve data in a tabular format, similar to a traditional database table. Salesforce recommends Custom Metadata Types over List Custom Settings because they offer better deployability, allowing configuration data to be quickly packaged and transferred across environments.

Hierarchy Custom Settings are ideal for storing Customer-Managed Secrets because they allow you to manage them at different levels—such as Org, profile, or user—enabling more granular control over access and configuration. The hierarchical structure ensures that Customer-Managed Secrets can be securely stored and accessed based on specific needs, reducing the risk of unauthorized access.

Hierarchy Custom Settings are classified into two categories based on their access level: Protected Hierarchy Custom Settings and Non-Protected Hierarchy Custom Settings. Protected Hierarchy Custom Settings are designed to restrict access, meaning they can only be accessed by the Apex code within the managed package that defines them. This ensures that sensitive data, such as customer-managed secrets or configuration settings, are securely managed and hidden from external access. On the other hand, Non-Protected Hierarchy Custom Settings are accessible by any Apex code in the Org, allowing broader use but with less security control.

APEX CLASS TO READ PROTECTED HIERARCHY CUSTOM SETTING VALUES

```
public with sharing class CustomSettingService {

   /* Gets the value of CustomerSecret__c field from the ISVSecretCustomSetting
custom setting.
    * @return The value of CustomerSecret__c for the current user's hierarchy. user,
Profile and Org */
   public static String getCustomerSecret(){

       // Retrieve the custom setting for the current user's hierarchy
       ISVSecretCustomSetting__c customSetting =
ISVSecretCustomSetting__c.getInstance(UserInfo.getUserId());

       // If no user-specific custom setting exists, fall back to the profile
       if (customSetting == null) {
           customSetting =
ISVSecretCustomSetting__c.getInstance(UserInfo.getProfileId());
       }

       // If no profile-specific custom setting exists, fall back to the org-wide
default
       if (customSetting == null) {
           customSetting = ISVSecretCustomSetting__c.getOrgDefaults();
       }

       // Return the value of MyField
       if (customSetting != null) {
           return customSetting.CustomerSecret__c;
       }

       // Return a default value or null if the field isn't set
       return null;

   }

}
```

A Packaged Protected Hierarchy Custom Setting is accessible only by the Apex Classes that are part of the same package or other packages that share the same namespace. This makes the Protected Hierarchy Custom Settings more secure than the Non-Protected. Here is an example of Apex Class that ISV partners can use to read values from a protected Hierarchy Custom Setting.

It is essential to use the getInstance() method or getOrgDefaults() to get the custom settings values. Some ISV partners include the above code in a global Apex Class, so they can also access the value outside their managed package. So, Including custom settings to retrieve code in the global Apex Class is a security violation, and the Salesforce Security review team will reject such applications.

CUSTOM UI TO MANAGE PROTECTED HIERARCHY CUSTOM SETTING

Protected Hierarchy Custom Settings won't be visible for your customer administrator when setting values through setup. So, ISV partners must include a custom UI as part of their application to set values into the custom settings. As mentioned, Protected Hierarchy Custom Settings are used to manage customer-managed secrets.

ISV partners can build this Custom UI using a custom Lightning Web Component (LWC) component and custom Apex class. The partner should include the LWC and Custom Apex in their managed package. The Salesforce security review team does not recommend passing a User ID or Profile Id from LWC to Apex. So, it is recommended that an Org-Wide value or current user ID or profile ID be set through the UserInfo object.

Per Salesforce Security review guidance, ISV partners should perform CURD and FLS checks before performing any DML operations through Apex. There are no CRUD or FLS available for Protected Hierarchy Custom Settings, but the ISV partner should use getOrgDefaults() or getInstance() method to retrieve the Protected Hierarchy Custom Settings. These methods respect CRUD and FLS for Protected Hierarchy Custom Settings.

Another common mistake ISVs make is displaying customer secrets in the UI. ISVs should never display any of these values in the custom UI or any debug logs, even if only the system administrator has access to them. Instead, ISV can provide a validate feature on the UI that will validate whether they have the right secrets in the system.

CONCLUSION

Securing customer secrets in Salesforce ISV applications is crucial for maintaining data integrity and meeting Salesforces security standards. This document has outlined the use of Protected Hierarchy Custom Settings as an approach to safeguarding information like API keys, tokens and passwords essential for ISV solutions functionality and security. By implementing the strategies discussed such as utilizing Apex classes and custom user interfaces ISV partners can significantly reduce the risks associated with access to customer managed secrets. Additionally adhering to these security measures is vital, for navigating the Salesforce AppExchange Security Review process facilitating application deployment and building trust with end users.

REFERENCES

[1]. Salesforce Developer Documentation on Custom Settings: Salesforce Custom Settings
[2]. Salesforce Security Guide: Salesforce Security Guide
[3]. Salesforce AppExchange Security Review Guidelines: AppExchange Security Review