keys.txt

Service de rétention de clé kernel

Ce service autorise les clé cryptographiques, jetons d'authentification, mappage d'utilisateur inter-domaine, et similaire d'être mis en cache dans le kernel pour utiliser les systèmes de fichier et d'autres services kernel.

Les portes-clés sont permis ; ce sont des type spéciaux de clé qui peuvent maintenir des liens vers d'autres clés. Chaque processus a 3 souscription de porte clé standard qu'un service peut utiliser pour trouver des clés.

Le service de clé peut être configuré ou activé avec "Security options"/"Enable access key retention support" (CONFIG_KEYS)

Présentation

Dans ce contexte, les clés représentent des unités de données cryptographiques, des jetons d'authentification, porte clé, etc. Ils sont représenté dans le kernel pas la structure key.

Chaque clé a un nombre d'attribut :

- un numéro de série
- un type
- une description
- des informations de contrôle d'accès
- une date d'expiration
- Un payload
- l'état

Chaque clé reçois un numéro de série du type key_serial_t qui est unique pour la durée de vie de cette clé. Tous les numéros de série sont des entier 32bits positifs. Les programmes userspace peuvent utiliser le numéro de série d'une clé comme moyen d'y accéder, sujet à vérification des permissions.

Chaque clé est d'un type définis. Les types doivent être enregistrés dans le kernel par un service kernel (tel qu'un système de fichier) avant que des clés de ce type puissent être ajoutées ou utilisées. Les programmes userspace définissent les nouveaux type directement. Les types de clé sont représentés dans le kernel par la structure key_type. cela définis un nombre d'opérations qui peuvent être effectuées sur une clé de ce type.

Chaque clé a une description, qui devrait être une chaîne affichable. Le type de clé fournis une opération pour effectuer une correspondance entre la description dans un cré et une chaîne critère

Chaque clé peut être définie pour expirer à un moment donné par la fonction d'instantiation du type. Les clés peuvent être immortelles.

Chaque clé peut avoir un payload. C'est la quantité de données qui représente la clé. Dans le cas d'un porte-clé, c'est une liste de clés qui sont liées au trousseau ; dans le cas d'une clé définie par l'utilisateur, c'est un blob de donnée arbitraire Ce payload n'est pas requis, et le payload peut, en fait, être simplement stocké dans la structure de clé elle-même

Similairement, quand le userspace souhaite lire le contenu de la clé, si permis, une autre opération de type de clé sera appelée pour convertir le payload attaché à la clé en un blob de donnée.

Chaque clé peut être dans un des états de base suivant :

- non-instancié. La clé existe, mais n'a pas de données attachée. Les clés demandée depuis le userspace sont dans cet état
- instancié : C'est l'état normal. La clé est formée, et a des donnée attachées
- Négative. C'est un état relativement court. La clé agit comme une note disant que l'appel précédent du userspace a échoué, et agit comme un accélérateur de recherche de clé. une clé négative peut être mise à jours vers un état normal.
- expiré. Les clé ont dépassés la durée de vie définie. Une clé expirée pour être mise à jours pour revenir à un état normal
- révoqué. Une clé est placée dans cet état par une action userspace.
- dead. Le type de clé a été désenregistré

Vue générale du service de clés

Le service de clé fournis des fonctionnalité :

Le service de clé définis 3 types de clé spéciaux

keyring Keyring sont des clé spéciales qui contiennent une liste d'autres clé.

user Une clé de ce type a une description et un payload qui sont des blobs arbitraires de données. Ils peuvent être manipulés par le userspace et ne sont pas prévus pour être utilisés par les services kernel

logon Comme une clé user, une clé logon a un payload qui est un blob de données arbitraire. Il est prévue comme un emplacement pour stocker des secrets qui sont accessible au kernel mais pas aux programmes userspace

Chaque processus souscrit à 3 keyrings : un keyring spécifie au thread, un keyring spécifique au processus, et un keyring spécifique à une session.

Chaque user ID résident dans le système maintient 3 keyrings spéciaux : un spécifique à l'utilisateur et un spécifique à la session utilisateur.

Chaque utilisateur a 2 quotas avec lesquel les clés qu'ils possèdent sont suivis. Une limite le nombre total de clé et de trousseaux, et l'autre limite la quantité totale d'espace qui peut être consommé par les descriptions et payload

Il y a une interface d'appel système par laquelle les programmes peuvent créer et manipuler les clé et trousseaux.

Il y a une interface kernel par laquelle les services peuvent enregistrer des type et rechercher des clés

Il y a une manière pour qu'une recherche faite depuis le kernel revienne au userspace pour demander une clé qui ne peut pas être trouvée dans un trousseau de processus

Un système de fichier optionnel est disponible via lequel la base de clé peuvent vue et manipulée

Permissions d'accès aux clés

Les clé ont un propriétaire, un groupe d'accès, et un masque de permission. La masque a jusqu'à 8 bit chacun pour le processeur, utilisateur, groupe et les autres accès. Seul six de chaque jeu de 8bits sont définis. Ces permissions sont :

View Permet de voir une clé ou un trousseau, incluant le type et description

Read Permet de voir une le payload d'une clé, ou la liste des clé liées au trousseau

Write Permet d'instancier un payload de clé ou de le mettre à jours, ou d'ajouter/supprimer un lien vers un trousseau.

Search Permet de rechercher un trousseau et des clés.

Link Permet de lier une clé ou un trousseau. Pour créer un lien d'un trousseau vers une clé, un processus doit avoir la permission Write sur le trousseau, et la permission Link sur la clé

Set Attribute Permet de changer le masque de permissions

Support SELinux

La classe de sécurité "key" a été ajoutée à SELinux pour que le contrôle d'accès puisse être appliqué aux clés créées dans divers contextes. Actuellement, toutes les permissions de base ci-dessus sont fournis dans SELinux, SELinux est simplement invoqué une fois les vérification de permissions de base effectuées.

La valeur du fichier /proc/self/attr/keycreate influence le labéling des clés créées. Si le contenu de ce fichier correspont à un contexte de sécurité SELinux, la clé obtiendra ce contexte, sinon, la clé obtiendra le contexte courant de la tâche qui a invoqué la création de la clé. Les tâches doivent données explicitement les permissions à assigner à un contexte particulier aux clé nouvellement créées, en utilisant la permission "create" dans la classe de sécurité key.

Les trousseaux par défaut associés avec les utilisateurs sont labélisés avec le contexte par défaut de l'utilisateur si et seulement si les programmes login ont été configurés correctement pour initialiser keycreate durant le processus de login. Sinon, il vont être labélisé avec le contexte du programme login lui même.

Noter, cependant, que les trousseaux par défaut associés avec l'utilisateur root sont labélisés avec le contexte kernel par défaut, vu qu'ils sont créés très tôt dans le processus de boot, avant que root ait une chance de se connecter.

Les trousseaux associés avec de nouveaux threads sont chacun labélisés avec le contexte de leur thread associés, et les trousseaux de session et de processus sont gérés de manière similaire.

Nouveaux fichiers procfs

2 fichiers ont été ajoutés à procfs par lesquels un administrateur peut trouver le status du service de clé :

/proc/keys Liste les clé qui sont actuellement lisibles par la tâche lisant le fichier, en donnant les informations sur le type, description et permissions. Les clés incluse dans la liste sont celles dont la permission View est donné au processus qui lit le fichier. Noter que les vérifications de sécurité LSM sont effectuées, et peuvent filtrer les clés que le processus courant n'est pas autorisé à voir.

Le contenu ressemble à :

```
SERIAL FLAGS USAGE EXPY PERM UID GID TYPE DESCRIPTION: SUMMARY
00000001 I--- 39 perm 1f3f0000 0 0 keyring _uid_ses.0: 1/4
00000002 I--- 2 perm 1f3f0000 0 0 keyring _uid.0: empty
00000007 I--- 1 perm 1f3f0000 0 0 keyring _pid.1: empty
0000018d I--- 1 perm 1f3f0000 0 0 keyring _pid.412: empty
000004d2 I-Q- 1 perm 1f3f0000 32 -1 keyring _uid.32: 1/4
000004d3 I-Q- 3 perm 1f3f0000 32 -1 keyring _uid_ses.32: empty
00000892 I-QU- 1 perm 1f000000 0 0 user metal:copper: 0
00000893 I-Q-N 1 35s 1f3f0000 0 0 user metal:silver: 0
00000894 I-Q- 1 10h 003f0000 0 0 user metal:gold: 0
```

- **I** Instancié
- **R** Révoqué
- **D** Mort
- **Q** Contribut au quota de l'utilisateur
- **U** En construction
- **N** Clé négative

/proc/key-users Ce fichier liste les données pour chaque utilisateur qui a au moins une clé dans le système. De telles données incluent les informations de quota et des statistiques :

[root@andromeda root]# cat /proc/key-users

```
0: 46 45/45 1/100 13/10000
```

```
29: 2 2/2 2/100 40/10000
32: 2 2/2 2/100 40/10000
38: 2 2/2 2/100 40/10000
```

<UID> User ID

<usage> Structure refcount

<inst>/<keys> Nombre total de clés et le nombre instanciées

<key>/<max> Quota - compteur de clé

<bytes>/<max> Quota - taille de clé

4 nouveaux fichiers sysctl ont été ajoutés également pour contrôler les limites de quota :

/proc/sys/kernel/keys/root_maxkeys Nombre maximum de clés que root peut posséder

/proc/sys/kernel/keys/root_maxbytes Taille maximum en octets que root peut stocker dans ces clés

/proc/sys/kernel/keys/maxkeys Nombre total de clé qu'un utilisateur peut posséder

/pro/sys/kernel/keys/maxbytes Taille maximum en octets qu'une utilisateur peut stocker dans ces clés

Garbage collector

Les clés morets (pour lesquelles le type a été supprimé) sont automatiquement unlinked des trousseaux et supprimés dès que possible par un collector en tâche de fond.

Similairement, les clé révoquées et expirées sont collectées, mais seulement au bout d'un certain temps. Ce délai est définis en secondes dans :
/proc/sys/kernel/keys/gc_delay