



**ÉLIMINEZ LES DANGEREUX « PASSE-PARTOUT »
EN CRÉANT UN SYSTÈME DE GESTION DES CLÉS SSH**



www.splashtop.com/foxpass



QUE SONT LES PASSE-PARTOUT ?

Dans de nombreuses organisations, les ingénieurs ont un accès illimité à tous les serveurs et à toutes leurs données. On parle souvent d'accès « administrateur » ou « root ». Un tel accès sans restriction est considéré comme un « passe-partout », à l'image du « passe-partout » d'un immeuble de bureaux qui ouvre toutes les portes.

De même que la perte du « passe-partout » d'un immeuble de bureaux peut avoir des conséquences désastreuses, confier aux ingénieurs des « passe-partout » pour les serveurs et les données peut s'avérer encore plus catastrophique si l'un d'eux est perdu, volé ou détourné.

POURQUOI LES PASSE-PARTOUT SONT UNE ERREUR

CAS 1: INITIÉ MALVEILLANT – PEU PROBABLE MAIS FATAL

On entend par initié malveillant un employé qui dispose à la fois des moyens nécessaires pour accéder aux ressources d'une entreprise et d'un motif pour lui nuire. Ce cas est très peu probable, puisque seuls les employés ayant acquis un certain niveau de confiance au sein de l'organisation obtiennent un accès root, mais ce scénario ne peut pas être totalement exclu.

Prenons par exemple l'initié malveillant qui a dérobé des gigaoctets de données chez Tesla pour les transmettre à des tiers. Selon une étude, plus de 49 % des entreprises s'inquiètent des attaques malveillantes, car il s'agit d'une menace réelle et croissante.

CAS 2: INITIÉ NÉGLIGENT – PROBABLE ET INTRAÇABLE

Malgré toutes les formations dispensées aux professionnels de l'informatique, la plupart des entreprises restent vulnérables et risquent d'être victimes d'hameçonnage (phishing) et d'autres formes d'attaques d'ingénierie sociale. D'après une étude, plus de 34 % des violations de données survenues en 2018 ont d'ailleurs été causées par des initiés. De plus, ces attaques sont généralement intraquables, et l'infrastructure d'une entreprise peut déjà avoir été compromise avant que quiconque ne s'en aperçoive.

CAS 3: IDENTIFIANTS VOLÉS – PROBABLE ET TRAÇABLE

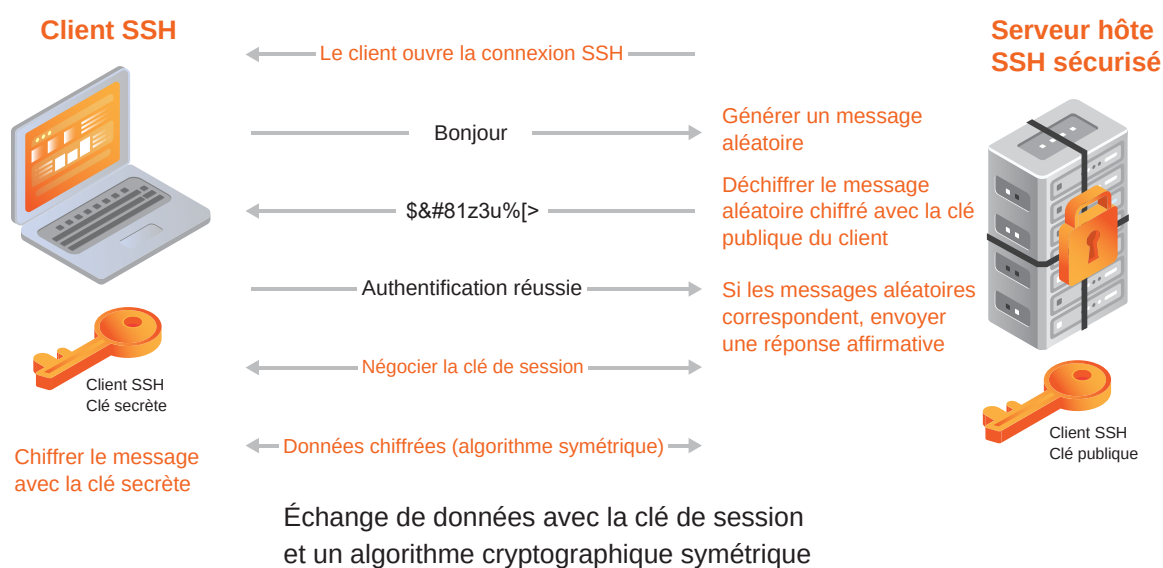
Le vol d'ordinateurs portables ou d'identifiants constitue également une attaque externe probable, qui cible les employés disposant des droits d'accès les plus élevés, c'est-à-dire les ingénieurs et l'équipe interne. Une fois ces identifiants dérobés en sa possession, un attaquant externe peut facilement accéder à l'infrastructure de l'entreprise. Si ces identifiants étaient partagés entre plusieurs employés, ils doivent être modifiés immédiatement, ce qui empêche tous les autres employés d'accéder aux serveurs jusqu'à ce qu'ils puissent être informés en toute sécurité des nouveaux identifiants. Si chacun disposait de ses propres identifiants pour le serveur, seuls ceux de la personne concernée devraient être réinitialisés.

COMMENT RÉDUIRE LE RISQUE DE VIOLATION DE DONNÉES ?

Les entreprises averties connaissent les dangers des attaques internes et limitent l'accès de leurs ingénieurs en fonction de leurs rôles. Le contrôle d'accès basé sur les rôles (RBAC) restreint l'accès des utilisateurs selon leur rôle au sein de l'organisation.

Déployer le RBAC signifie qu'un utilisateur n'a accès qu'à ce dont il a fréquemment besoin pour faire son travail. Toutefois, « fréquemment » ne signifie pas nécessairement chaque jour, chaque semaine, ni même chaque mois. L'option la plus sûre consiste à accorder aux ingénieurs le niveau d'accès le plus bas possible. Lorsqu'un accès de niveau supérieur est nécessaire, il doit être surveillé, contrôlé et limité dans le temps.

Comment mettre en place un mécanisme de contrôle d'accès bien surveillé, contrôlé et limité dans le temps ? La meilleure solution est un système où chaque ingénieur possède sa propre clé et qui autorise l'accès à un serveur pour une durée prédéfinie — la plus courte possible.



QUAND FAIRE EXPIRER LES AUTORISATIONS ?

En règle générale, un ingénieur se voit accorder des autorisations de son arrivée dans l'entreprise jusqu'à son départ. Une option plus dynamique consisterait toutefois à accorder des autorisations :

- Uniquement pendant une période d'astreinte
- Jusqu'à la clôture d'un ticket de type Jira
- Uniquement si l'autorisation a été utilisée au moins une fois au cours du dernier mois

QU'EST-CE QU'UNE CLÉ SSH ?

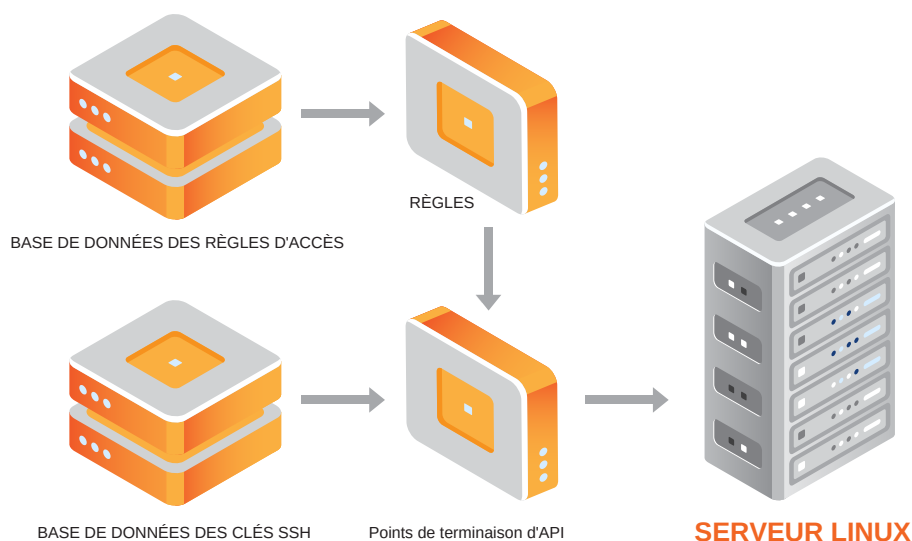
Une « clé SSH » est un mécanisme cryptographiquement sûr permettant de prouver son identité à un serveur lors d'une connexion via le réseau. En ce sens, elle ressemble à un long mot de passe, mais la comparaison s'arrête là. Une « clé SSH » se compose en réalité de deux parties : une clé privée dans un fichier et une clé publique dans un autre. Le contenu de ces deux fichiers est très différent, mais ils se complètent et sont donc créés en même temps.

La moitié privée de la clé SSH doit être protégée et ne jamais, par exemple, quitter le système sur lequel elle a été générée. La moitié publique, en revanche, peut être librement partagée. La clé privée ne peut pas être reconstituée à partir de la clé publique. En copiant la partie publique de la clé sur tout serveur distant auquel cet utilisateur doit pouvoir se connecter, on accorde l'accès au détenteur de la moitié privée.

COMMENT CRÉER UN SYSTÈME DE GESTION DES CLÉS SSH ?

Les composants nécessaires à la création d'un système de gestion des clés SSH sont :

-  Une base de données stockant la partie publique des clés SSH des utilisateurs
-  Une base de règles d'accès indiquant quels utilisateurs accèdent à quels serveurs
-  Un serveur d'applications gérant des applications web simples et des API





COMMENT LE METTRE EN PLACE :



Traditionnellement, le démon sshd recherche sur le disque le fichier `authorized_keys` de l'utilisateur dans son répertoire `~/.ssh`.

Dans cette configuration, on exploite toutefois la directive sshd `AuthorizedKeysCommand` (disponible à partir d'OpenSSH 6.2) pour indiquer à sshd d'utiliser le résultat du script situé dans `/usr/local/bin/get_keys.sh`. Pour ce faire, modifiez l'emplacement dans le fichier `sshd_config`. Notez qu'en cas d'échec, sshd se rabat toujours sur le fichier `authorized_keys` présent sur le disque. Pour cela, modifiez le fichier `sshd_config` comme suit :

```
ubuntu@host:~$ vi /etc/ssh/sshd_config

[. . .]
AuthorizedKeysCommand /path/to/get_keys.sh
AuthorizedKeysCommandUser nobody
```



Le script `/usr/local/bin/get_keys.sh` peut être écrit dans n'importe quel langage, mais un script Shell ou Python est recommandé. Le premier et unique argument de ce script est le nom d'utilisateur Linux de l'utilisateur qui demande les clés. Le script doit transmettre la même requête au point de terminaison d'API mis en place. Utilisez pour cela la commande `curl`, par ex. ``curl https://<endpoint>/get_keys/?user=$1``. Voici un exemple de `get_keys.sh` (spécifique à AWS) :

```
#!/bin/sh

user="$1"
hostname=`hostname`
instance_id=`curl -s -q -f
http://169.254.169.254/latest/meta-data/instance-id`
curl -s -q -f -m 5 "https://keyserver.example.com/sshkeys/?
user=${user}&hostname=${hostname}&instance_id=${instance_id}" 2>/dev/null

exit $?
```

Exemple de requête :

```
https://keyserver.example.com/sshkeys/?user=aren&hostname=prod-db-1&instance_id=i-0b263919b6498b123
```



Créez le point de terminaison `/sshkeys/` dans l'API afin qu'il :

- récupère le nom d'utilisateur passé dans la chaîne de requête ;
- recherche les clés SSH de cet utilisateur dans la base de données, puis
- renvoie les clés SSH au même format qu'un fichier `authorized_keys` : chaque clé publique SSH sur sa propre ligne.

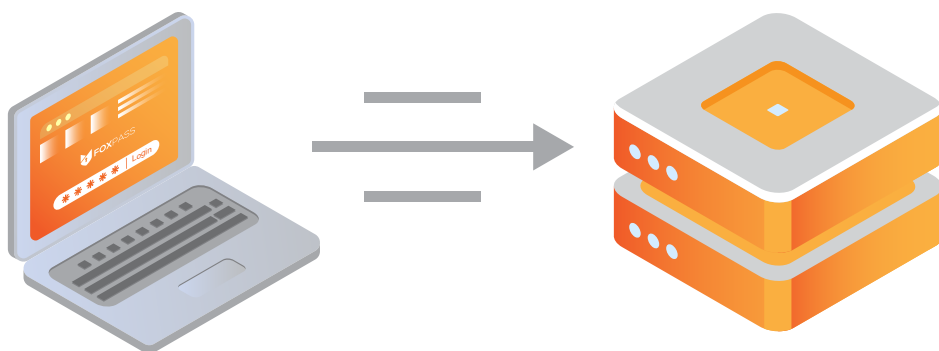
Exemple :

```
def sshkeys(request):  
    hostname = request.GET['hostname']  
    aws_instance_id = request.GET['aws_instance_id']  
    username = request.GET['user']  
  
    user = get_user(username)  
  
    keys = []  
    if is_user_allowed_on_host(user, hostname, aws_instance_id):  
        keys = user.keys  
  
    return HttpResponse("\n".join(key_text) for key in keys),  
    content_type='text/plain')
```



Créez une application web permettant aux utilisateurs de gérer leurs clés SSH. Il est facile d'authentifier un utilisateur dans l'application web via OAuth avec GSuite ou Office 365.

L'utilisateur doit pouvoir ajouter une nouvelle clé et désactiver une ancienne clé.



Interface web simple pour la connexion (OAuth)
et l'import des clés SSH (partie publique)

BASE DE DONNÉES DES CLÉS SSH

Cette configuration de base peut être étendue en ajoutant une base de données de règles d'accès qui régissent et contrôlent l'accès aux serveurs. Grâce à l'intégration d'API externes, l'accès peut être modifié en fonction de conditions, puis restreint ou supprimé en conséquence.

- Pour garantir que seules les clés SSH des employés actuels sont distribuées : effectuer un recoupement avec l'API de GSuite, Office365, Okta ou OneLogin
- Pour imposer la rotation des clés SSH : exclure les clés SSH de plus de 90/180/360 jours
- Pour restreindre les serveurs auxquels un utilisateur peut accéder : vérifier les droits d'accès définis pour chaque utilisateur. Si l'autorisation pour le serveur concerné n'existe pas, empêcher l'utilisateur d'accéder au système

Exemple :

<https://keyserver.example.com/sshkeys/?user=aren&hostname=prod-db-1>

```
hostname=prod-db-1
« Machine de production » match regex = ^prod-.*
« aren » a-t-il accès maintenant à une « Machine de production » ?

ou

hostname=prod-db-1
« Machine BDD » match regex = -db-\d+$
« aren » a-t-il accès maintenant à une « Machine BDD » ?
```

CONCLUSION

Les possibilités d'utilisation de cette configuration pour la gestion des clés SSH et le contrôle d'accès sont infinies. Les failles de sécurité et les attaques internes sont toujours une préoccupation majeure pour les organisations détenant des données sensibles susceptibles d'être partagées avec des ingénieurs, car leur fuite peut avoir un impact considérable. Il est toujours judicieux d'accorder aux employés le moins de privilèges possible et de limiter l'accès aux ressources critiques lorsque cet accès est accordé.

Si le site d'Ashley Madison, Target ou Tesla avaient disposé d'un meilleur contrôle d'accès, ces entreprises auraient peut-être évité des failles de sécurité aussi coûteuses que médiatisées. La gestion des clés SSH est un mécanisme utile pour sécuriser l'infrastructure et aider à empêcher les équipes internes de divulguer des informations confidentielles à l'extérieur.

COMMENT FOXPASS PEUT VOUS AIDER

Foxpass propose des produits complets et simples d'utilisation de gestion des clés SSH et de contrôle des accès à privilèges, qui permettent aux ingénieurs d'accéder aux serveurs nécessaires pendant une durée prédéfinie. En outre, Foxpass est un système complet de gestion des utilisateurs et des groupes pour Linux, qui se synchronise avec des annuaires courants tels que GSuite, Office365 et Okta.