Activité : mise en place d'un authentification SSH 2FA avec mot de passe et OTP sur un serveur DEBIAN

CONTEXTE

Suite à un test d'intrusion réalisé par un prestataire et au rapport d'audit qui vous a été transmis, plusieurs mots de passe utilisateurs ont été récupérés. Cela a mis en évidence les limites de

l'authentification appliquée actuellement au sein de votre entreprise. Votre responsable a pris le temps de lire les recommandations relatives à l'authentification multifacteur et aux mots de passe publiées par l'ANSSI . Ainsi, il vous est demandé de proposer des solutions permettant de renforcer la sécurité liée à l'authentification.

PLATEFORME NECESSAIRE

- 1 smartphone Android avec l'application FreeOTP+ ou Authy (ou autre app d'authentification) ou un iPhone avec l'application FreeOTP Authenticator ou Authy (ou autre app d'authentification).
- 1 machine virtuelle cliente sous Windows 10/11 Professionnel/Education.
- 1 machine virtuelle serveur sous Debian 12 avec OpenSSH installé et fonctionnel.

L'intégralité des VM doit disposer d'un accès réseau complet.

Présentation de l'infrastructure

Prérequis sur le smartphone et la machine cliente

- Sur votre smartphone, installez l'application d'authentification OTP (FreeOTP+, Authy, etc.).
- Sur le serveur Debian,
  - créez un compte etudiant qui pourra se connecter en SSH avec uen authentifiation 2FA
  - installez le paquet qrencode qui permet de générer un QR code afin de paramétrer l'application OTP en lien avec le service oath présent sur le serveur Debian 12.

```
root@serveur:~# apt install qrencode
```

- Vérifiez que la connexion SSH entre votre ordinateur et le serveur Debian est pleinement opérationnel avec mot de passe.

```
clientwindows> ssh etudiant@192.168.1.90
etudiant@192.168.1.90's password:
Linux serveur 6.1.0-17-amd64 #1 SMP PREEMPT_DYNAMIC Debian 6.1.69-1 (2023-12-30) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Wed Jan  3 19:34:10 2024 from 192.168.1.85
etudiant@serveur:~$

===== Configuration du serveur =====
Dans un premier temps, il sera nécessaire d'installer les paquets permettant de mettre en œuvre le
mécanisme d'OTP nommé OATH  sur le serveur.
<code>
root@serveur:~# apt install libpam-oath oathtool
```

Définissez un secret sous forme hexadécimal qui sera utilisé par le générateur TOTP/HOTP en lien avec l'utilisateur étudiant et qui sera soumis à la double authentification.

Attention ! Ce secret doit être jalousement gardé, car c'est la clé de voûte servant à la génération des mots de passe à usage unique. S'il est compromis, c'est l'ensemble de l'authentification OTP qui sera impacté.

```
root@serveur:~# KEY=$(openssl rand -hex 20)
root@serveur:~# echo "HOTP/T30/6 etudiant - ${KEY}" >> /etc/security/users.oath
root@serveur:~# chown root /etc/security/users.oath
root@serveur:~# chown 600 /etc/security/users.oath
```

La première commande stocke dans une variable nommée KEY le résultat de la commande openssl permettant de générer 20 caractères hexadécimaux.

La deuxième commande ajoute une ligne dans le fichier /etc/security/users.oath contenant « HOTP/T30/6 etudiant - » et le contenu de la variable $(KEY) :

- Au niveau du premier champ, T30 signifie que l'OTP généré aura une durée de validité de 30 secondes (on est donc en présence d'un TOTP du fait de la notion de temps) et qu'il sera composé de 6 chiffres.
- Le deuxième champ fait référence à l'utilisateur. Il y a autant de lignes que d'utilisateurs.
- Le quatrième champ est constitué d'un « - » (tiret) pour l'absence de code PIN.
- Le cinquième champ est constitué des 20 caractères hexadécimaux généré aléatoirement.

Si nous étions en présence d'une authentification HOTP la syntaxe serait du type : HOTP user - 12345678909876543210

Les deux dernières commandes permettent de définir le super-administrateur root comme propriétaire du fichier /etc/security/users.oath puis d'attribuer les droits de lecture/écriture à ce dernier et aucun droit pour les autres utilisateurs du système.

Nous allons maintenant configurer PAM (Pluggable Authentication Modules), le service qui contrôle les authentifications sur le serveur Debian.

root@serveur:~# nano /etc/pam.d/sshd

PAM configuration for the Secure Shell service

Standard Un*x authentication.

@include common-auth

auth required pam*unix.so nullok*secure auth required pam_oath.so usersfile=/etc/security/users.oath window=20 digits=6

Nous commentons la ligne @include common-auth car elle empêche l'authentification OTP même en cas de connexion avec le bon mot de passe.

La ligne suivante impose l'authentification par mot de passe stocké en local sur le système et interdit les mots de passe vides.

La dernière impose une fois l'authentification par mot de passe réussie, une deuxième authentification par OTP. Nous faisons référence au fichier contenant le (ou les) nom d'utilisateur concerné ainsi que le secret servant à générer des mots de passe à usage unique. Ce mot de passe disposera de 6 chiffres et il sera possible de générer 20 codes à usage unique valides.

Il nous reste maintenant à éditer le fichier de configuration du service SSH afin de définir l'usage de l'authentification 2FA.

root@serveur:~# nano /etc/ssh/sshd_config

ChallengeResponseAuthentication yes

KbdInteractiveAuthentication no

La ligne ci-dessous est normalement déjà décommentée

UsePAM yes

Nous nous assurons de commenter la ligne KdbInteractiveAuthentication et d'avoir les deux autres lignes activées avec la valeur yes. Nous

pouvons ensuite redémarrer le service SSH.

root@serveur:~# systemctl restart ssh

Enfin, il nous reste à récupérer le secret en base 32 qui nous permettra ensuite de générer sur le poste client un QR code pour notre application Android.

root@serveur:~# cat /etc/security/users.oath HOTP/T30/6 etudiant – 65f43c705ce51c9c058ec8bb4b7f64b656681866 root@serveur:~# oathtool -v -d 6 65f43c705ce51c9c058ec8bb4b7f64b656681866 Hex secret: 65f43c705ce51c9c058ec8bb4b7f64b656681866 Base32 secret: MX2DY4C44UOJYBMOZC5UW73EWZLGQGDG Digits: 6 Window size: 0 Start counter: 0x0 (0)

• Attention ! Pour que la partie TOTP soit pleinement fonctionnelle, vous devez vous assurer que les horloges des différentes machines sont synchronisées et à l'heure.

Configuration du client et de l'application Android FreeOTP+

Il s'agit maintenant de paramétrer correctement la machine cliente et l'application Android FreeOTP+ afin de rendre opérationnel l'authentification SSH 2FA.

Nous allons d'abord générer sur le poste client (Ubuntu ou Kali) un fichier png contenant un QR code que nous soumettrons à l'application FreeOTP+.

etudiant@client:~$ qrencode -o etudiant.png 'otpauth:*totp/etudiant@192.168.1.90?secret=MX2DY4C44UOJYBMOZC5UW73EWZLGQGDG*' *etudiant@client:~$ ls -l … -rw-r–r– 1 etudiant etudiant 471 3 janv. 23:08 etudiant.png Nous pouvons ouvrir ce fichier PNG sur le poste client puis ouvrir l'application FreeOTP+ sur le smartphone. Sélectionner l'icône « Appareil photo » en bas à droite. Puis prenez en photo le QR code présent sur l'écran du client. Une nouvelle configuration pour votre utilisateur et votre serveur est automatiquement créée. Nous pouvons ensuite supprimer le fichier PNG contenant le QR code, car il contient le secret à ne pas compromettre. Sur le client, nous pouvons lancer une connexion SSH vers le serveur avec le compte etudiant. Après avoir entré votre mot de passe, un OTP vous est demandé. Dans l'application FreeOTP+, sélectionnez la nouvelle configuration. Celle-ci vous fournit un code de 6 chiffres valable 30 secondes. etudiant@client:~$ ssh etudiant@192.168.1.90 (etudiant@192.168.1.90) Password: (etudiant@192.168.1.90) One-time password (OATH) for `etudiant': Linux serveur 6.1.0-17-amd64 #1 SMP PREEMPTDYNAMIC Debian 6.1.69-1 (2023-12-30) x8664 The programs included with the Debian GNU/Linux system are free software; the exact distribution terms for each program are described in the individual files in /usr/share/doc/*/copyright. Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent permitted by applicable law. Last login: Wed Jan 3 22:17:13 2024 from 192.168.1.85 etudiant@serveur:~$ Q11. Expliquer quel est le type d'authentification utilisé ici. ====== Retour Accueil Authentification multifacteur ====== * Authentification multifacteur*