# Pourquoi ces vulnérabilités

NEVER TRUST USER INPUT

Désérialisation de données python, avec le paquet Pickle. La sérialisation est le fait de coder une information sous une autre forme (souvent plus petites) et inverse pour la désérialisation. Pickle est utilisé pour sérialiser ou désérialiser des objets pythons. Par exemple, la sérialisation est très utile pour transférer des données en différent langage de programmation. Pour les applications web modernes, la sérialisation est utile pour stocker les données de session, ou bien des formulaires.

L’extraction d’archive en python, avec le paquet Zipfile. L’extraction d’archive est la décompression et l’extraction des fichiers d’un fichier compressé. La compression est souvent utilisée pour réduire la taille des fichiers afin d’économiser l’espace de stockage ou bien réduire le temps de transfert sur le réseau. Dans notre exemple, l’extraction d’archive sera utilisée pour un serveur interne de développement de production.

# Une présentation du code de votre site et de son architecture

# Une démonstration de la vulnérabilité

# Correction des vulnérabilités

**Désérialisation** : Il est important de s'assurer que la désérialisation de données dans l'application est effectuée de manière sûre et de ne pas faire confiance aux données entrantes sans une validation appropriée. Plus précisément, pour éviter les attaques d’injection de code, il est préférable de n’accepter que des types de données spécifiques, ou d’utiliser des règles de validation.  
De plus, ajouter une authentification et bloquer les requêtes non autorisées permettrait de contrôler l’accès à la désérialisation.  
Enfin utiliser des bibliothèques et langages de sérialisation plus sûrs, par exemple le JSON est plus sûr car sa bibliothèque n’inclut pas de fonctionnalités avancées.

**L’extraction d’archive** : il est nécessaire de restreindre les types de fichiers autorisés à être extraits en définissant une liste des types de fichiers autorisés ou non autorisés.   
L’utilisation d’un antivirus afin de détecter les fichiers malveillants avant de les extraire.  
Formater le nom des fichiers pour éviter le Path Traversal comme la fonctionnalité secure\_filename de Flask qui permet d’enlever les probables chemins de fichiers dans le nom.

En règle générale, maintenir à jour les bibliothèques et le code, mais encore réfléchir aux permissions accordées à l’exécuteur du code apporteront une sécurité passive à l’infrastructure.

# Cas concrets

## Attaque contre Cloudflare en 2017 :

En février 2017, Cloudflare a découvert une vulnérabilité dans leur application web qui utilisait la bibliothèque Python Flask. La vulnérabilité était due à une mauvaise gestion de la désérialisation des données, qui permettait à un attaquant d'exécuter du code malveillant à distance. Les attaquants ont utilisé cette vulnérabilité pour accéder à des informations sensibles des clients de Cloudflare, telles que des noms d'utilisateur, des adresses e-mail et des mots de passe. Cloudflare a rapidement corrigé la vulnérabilité et a recommandé à tous ses clients de changer leurs mots de passe.

## *Attaque* contre FireEye en 2020 :

En décembre 2020, FireEye, une société de sécurité informatique de renom, a découvert qu'elle avait été victime d'une cyberattaque sophistiquée. Les attaquants ont exploité une vulnérabilité de type "Pickle RCE" dans l'outil de surveillance réseau de FireEye pour accéder aux outils de piratage utilisés par FireEye pour surveiller les attaques informatiques. Les outils volés ont ensuite été utilisés dans d'autres attaques contre des entreprises du secteur de la défense, de la finance et du gouvernement. Cette attaque a été considérée comme particulièrement dangereuse en raison de la qualité des outils de piratage volés et du profil des entreprises ciblées. FireEye a rapidement corrigé la vulnérabilité et a travaillé avec d'autres sociétés de sécurité informatique pour identifier les autres entreprises victimes de l'attaque.