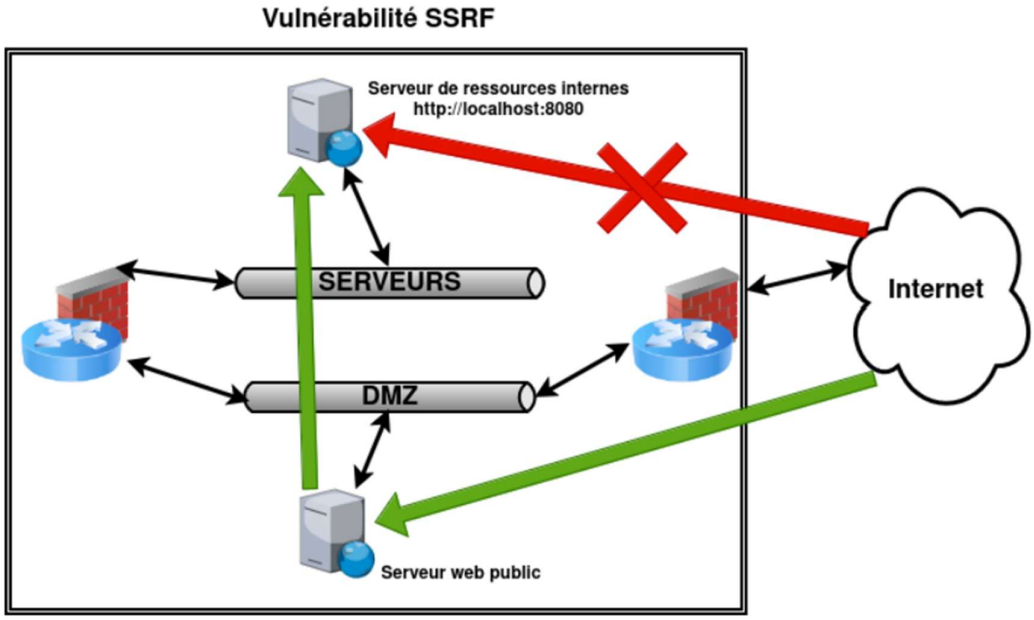
Présentation de la problématique

Les vulnérabilités de type SSRF (Server Side Request Forgery) permettent d'interagir avec le serveur afin d'en extraire des fichiers et de trouver d'autres services ouverts. Ainsi, lorsqu'une application web comporte une fonctionnalité permettant d'interagir avec des ressources internes via des URL (chargement d'une image, redirection vers une page située sur un serveur interne...), alors l'attaquant pourra tenter de modifier la requête en soumettant une URL différente afin d'accéder à la ressource visée. Cette ressource cible n'est normalement pas accessible directement depuis l'extérieur via la définition d'une politique de filtrage. Mais le serveur web est autorisé à y accéder. Lorsque les URL soumis ne sont pas vérifiées, leur falsification peut permettre des accès non autorisés.



Seul le serveur web peut accéder au serveur de ressources internes. Un attaquant peut tenter de falsifier une requête de redirection pour viser d'autres ressources internes qui sont inaccessibles directement depuis l'extérieur.

Exemple : localhost/admin au lieu de localhost:8080/ressource-interne

Exemple de code vulnérable :

```
$url=$_GET['url']
$content=file_get_contents($url)
echo $content ;
```

Dans cet exemple de code, le paramètre *url* n'est pas contrôlé avant son utilisation dans *file_get_contents($url)*

Conséquences possibles d'une vulnérabilité SSRF

- Accès à une fonctionnalité du serveur web :

http://monapli.fr?url=http:///localhost/server-status

- Inclusion de fichiers locaux (voir activité 6 de la même série sur l'inclusion de fichiers locaux et distants) :
  http://monapli.fr?url=file:///etc/passwd
- Accès à un service offert par le serveur web
  http://monappli.fr?url=http://localhost:8081
- Accès à une autre instance de serveur web
  http://monappli.fr?url=http://192.168.10.15

Les conséquences peuvent aller jusqu'à d'autres types d'exploitation (XXE, XSS, injection de commandes). XXE et XSS sont détaillés dans d'autres côtés labos de cette série. D'autres conséquences sont ainsi possibles : reconnaissance pour la préparation d'une attaque, déni de service.

Types de SSRF

Dans l'exemple de code cité plus haut, le contenu de la ressource est retourné par le serveur (echo $content;). Ce type de SSRF est appelé content-based. D'autres types de SSRF existent :

- Boolean-Based : la réponse du serveur est différente que la ressource existe ou non ;
- Error-Based : les codes erreurs HTTP (400 ou 500) permettent à l'attaquant de déduire si une ressource existe ou non ;
- Time-Based:Le temps de réponse du serveur peut varier et donner des indications sur l'existence ou non de ressources internes.

Il est ainsi potentiellement possible de cartographier des ressources existantes en vue de préparer une attaque plus aboutie.

Contre-mesures

Pour éviter les vulnérabilités SSRF, il faut mettre en place des contrôles sur les liens via la configuration d'une liste blanche de ressources dont le chargement est autorisé.

- Liste de ressources connues autorisées ;
- Définition de noms et d'adresses IP autorisées, domaines de confiance ;
- Liste de protocoles autorisés (file://,sftp://...).

Il convient donc de vérifier les ressources avant de les charger en testant leur contenu afin de ne pas les exploiter en l'état. Tout URL doit être vérifié avant d'être traité.

Exemple de code non vulnérable :

```
$url=$_GET['url']
$url=nettoyage($url) ;
if(nettoyage) {
$content=file_get_contents($url)
echo $content ;
}
```

La fonction de nettoyage permet de valider l'URL. Il convient donc d'implémenter des contrôles stricts sur les URL que l'application utilise pour charger des ressources. De plus, il faut configurer des en-têtes de sécurité (security-headers) qui limite l'origine du contenu d'une page web.

Ensuite, il faut prévoir des contrôles d'accès stricts sur les ressources internes sensibles de sorte que même si un attaquant réussi à en faire la demande, il ne pourra pas accéder aux données sensibles sans autorisation.

Enfin, la configuration d'un système de surveillance et de journalisation permet de suivre et d'analyser les demandes d'accès afin de repérer celles qui sont inhabituelles et de déclencher une alerte.