Titre

Internal Server Error triggered via malformed callbackUrl parameter in authentication endpoint

---

Actif concerné

https://sign.test.sitetesté.com/api/auth/signin/pingfed

Type de vulnérabilité

- Improper Input Validation
- Unhandled Exception / Server Error

Sévérité

Low (potentiellement Information Disclosure / Stability issue)

Description

L'endpoint d'authentification accepte le paramètre callbackUrl.
Lorsqu'une valeur malformée est fournie (ex : utilisation de backslash \), le serveur retourne une erreur **500 Internal Server Error** au lieu de gérer correctement l'entrée utilisateur.

Cela indique une **mauvaise validation ou un traitement incorrect des entrées**, pouvant révéler des problèmes dans la logique backend ou provoquer des comportements inattendus.

Étapes pour reproduire

1. Exécuter la requête suivante : curl -i

"https://sign.test.sitetesté.com/api/auth/signin/pingfed?callbackUrl=\/evil.com"

2. Observer la réponse serveur :

HTTP/1.1 500 Internal Server Error

3. Le serveur retourne une page d'erreur indiquant un problème de configuration :

There is a problem with the server configuration.

Méthodologie ayant conduit à la découverte

1. **Reconnaissance initiale**

Identification d'un endpoint d'authentification exposé :

/api/auth/signin/pingfed

Cet endpoint semble gérer un flux SSO (PingFederate).

---

2. **Identification des paramètres** Observation de l'utilisation du paramètre : callbackUrl typiquement utilisé pour gérer les redirections après authentification.

---

3. **Tests de redirection (Open Redirect)**

Plusieurs payloads ont été testés afin d'identifier une potentielle redirection ouverte :

- URL externe :

callbackUrl=https://evil.com

- URL encodée :

callbackUrl=https:%2f%2fevil.com

- Double encodage :

callbackUrl=https:%252f%252fevil.com

- Sous-domaine trompeur :

callbackUrl=https://sign.test.sitetesté.com.evil.com

Résultat :

- Requêtes bloquées (403 – WAF Akamai) ou
- Redirection vers une page d'erreur contrôlée

---

4. **Tests de contournement de validation**

Tentatives de bypass via :

- caractères spéciaux
- variations de format d'URL

Exemple :

callbackUrl=//evil.com

Résultat : redirection vers une page d'erreur interne (validation backend active)

---

5. **Fuzzing des formats d'entrée**

Injection de formats non standards pour tester la robustesse du parsing :

Exemple :

callbackUrl=\/evil.com

---

6. **Détection du comportement anormal**

Cette entrée spécifique provoque

: HTTP/1.1 500 Internal Server

Error avec un message :

There is a problem with the server configuration.

---

7. **Validation de la reproductibilité** ○ Le comportement est reproductible
    - D'autres payloads similaires ne déclenchent pas nécessairement une erreur 500

---

8. **Conclusion**

La découverte résulte d'une approche progressive :

- Tests de sécurité classiques (open redirect) ○ Contournement de validation
- Fuzzing ciblé des formats d'entrée

Ce processus a permis d'identifier une mauvaise gestion des entrées menant à une erreur serveur non contrôlée.

Outils et techniques utilisés

1. **Fuzzing de répertoires – FFUF**

Utilisation de l'outil ffuf pour explorer les endpoints disponibles :

ffuf -u https://target/FUZZ -w C:\Users\famil\go\bin\wordlist.txt -H "User-Agent: Mozilla/5.0" -rate 2 t 5 -timeout 20

Objectif :

- Découvrir des endpoints cachés
- Identifier des points d'entrée intéressants (notamment /api/auth/...)

---

2. **Tests manuels HTTP – cURL**

Utilisation de curl pour envoyer des requêtes personnalisées :

curl -i "https://sign.test.sitetesté.com/api/auth/signin/pingfed?callbackUrl=..."

Objectif :

- Analyser les réponses HTTP (codes, headers, redirections)

- Tester différents payloads sur le paramètre callbackUrl

---

3. **Tests de redirection (Open Redirect)**

Envoi de plusieurs variantes de payloads :

- URL externe ○ URL encodée ○ Double encodage
- Sous-domaines malveillants

Objectif :

- Identifier une éventuelle redirection ouverte
- Tester la validation côté serveur

---

4. **Bypass de filtres / WAF**

Tests pour contourner les protections (Akamai) :

- Encodage URL (%2f, %252f)
- Format alternatif (//evil.com)
- Manipulation du schéma (https:)

Objectif :

- Vérifier la robustesse des filtres en place
- Atteindre le backend applicatif

---

5. **Fuzzing de paramètres**

Injection de valeurs non standards dans callbackUrl :

\/evil.com

Objectif :

- Tester le parsing des entrées
- Déclencher des comportements inattendus

---

6. **Analyse des réponses serveur**

Observation des réponses :

- 403 Forbidden → filtrage WAF ○ 302 Redirect → validation backend
- 500 Internal Server Error → anomalie détectée

Objectif :

- Comprendre la logique de traitement
- Identifier les points faibles

Résultat attendu

Le serveur devrait :

- Valider correctement le paramètre callbackUrl
- Rejeter les valeurs invalides avec une réponse contrôlée (ex : 400 Bad Request)

Résultat actuel

- Le serveur retourne une erreur **500**
- Le comportement indique une exception non gérée côté backend

---

Impact

- Révélation de mauvaise gestion des entrées utilisateur
- Potentiel point d'entrée pour :
  - tests de fuzzing ciblés
  - identification de bugs backend plus profonds
- Peut affecter la stabilité du service si abusé

Recommandations

- Implémenter une validation stricte du paramètre callbackUrl
- Rejeter les entrées invalides avec une réponse 4xx appropriée
- Ajouter une gestion d'erreur robuste pour éviter les 500
- Logger les erreurs côté serveur sans exposer d'information côté client

---

Preuve

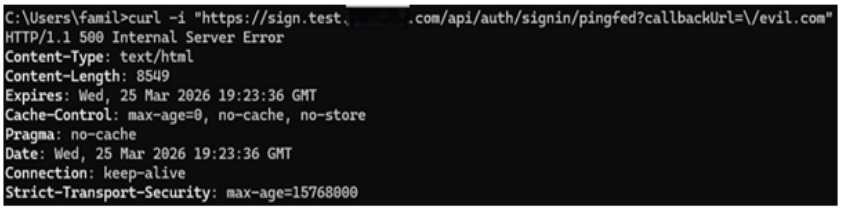
Requête :

GET /api/auth/signin/pingfed?callbackUrl=\/evil.com HTTP/1.1

Host: sign.test.sitetesté.com

Réponse :

HTTP/1.1 500 Internal Server Error



Notes supplémentaires

- Les autres tentatives (URL externes, encodage, subdomain trick) sont correctement bloquées
- Ce comportement semble spécifique à certaines entrées malformées
- Indique une surface d’attaque potentielle autour du parsing des URLs