# My Little Certificate

Synthèse de résolution – Certificate Transparency (CT) / TLS

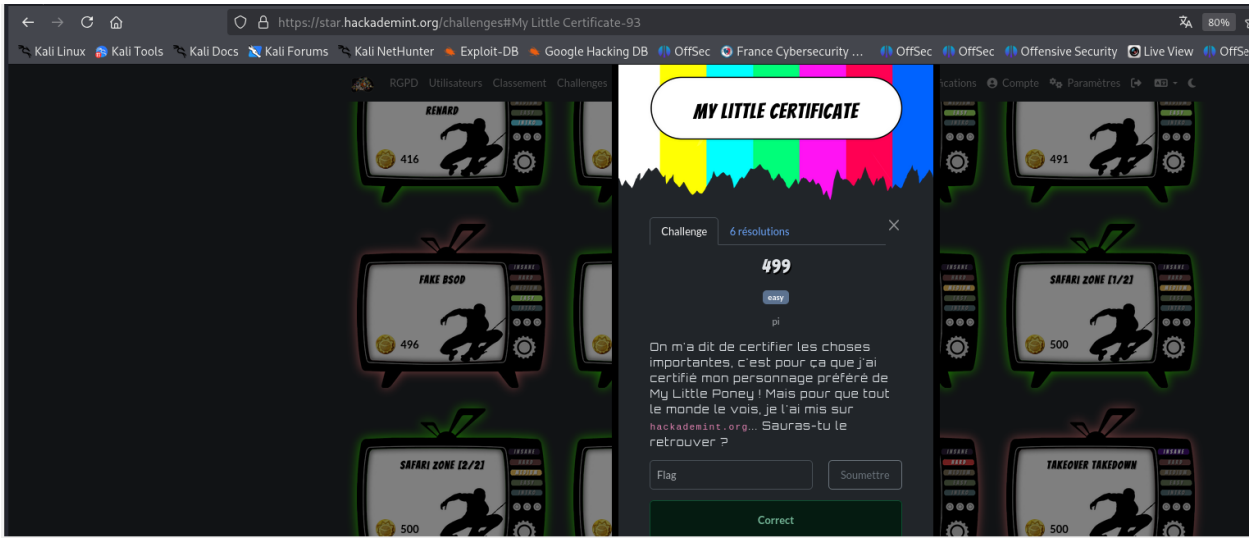

## 1. Résumé du challenge

Le challenge « My Little Certificate » part d'une histoire volontairement naïve : quelqu'un a été invité à « certifier les choses importantes » et a donc émis un certificat TLS pour son personnage préféré de *My Little Pony*, puis l'a « mis sur hackademint.org ». L'objectif consiste à retrouver ce personnage/indice à partir des journaux publics de Certificate Transparency (CT), puis à le convertir en un flag au format `Star{...}`.

Indice implicite : le titre « easy pi » renvoie au vocabulaire de la **PKI** (Public Key Infrastructure) et au monde des certificats X.509, plutôt qu'à un accès web direct.

Point clé : un sous-domaine peut apparaître dans un certificat (donc dans CT) sans pour autant être réellement déployé en DNS/HTTP. Un retour `502 Bad Gateway` ou un domaine non résolu n'invalide pas l'indice : le certificat, lui, existe bien.

**Capture – énoncé (référence)**




## 2. Notions utiles

Les éléments suivants suffisent à résoudre le challenge :

- Certificate Transparency (CT) : journaux publics qui enregistrent les certificats émis pour un domaine.
- SAN (Subject Alternative Name) : extension X.509 listant les noms DNS couverts par le certificat.
- CN (Common Name) : nom principal du certificat (souvent redondant avec le SAN).
- Chaîne de confiance : signature de l'AC (ex. Let's Encrypt) et période de validité (notBefore/notAfter).
- My Little Pony (MLP) : franchise animée. Les noms de personnages (ex. Rainbow Dash, Pinkie Pie) servent d'indice lisible.

## 3. Collecte : interroger les logs CT

Deux approches pratiques (équivalentes) permettent d'obtenir les certificats émis pour un domaine : crt.sh (interface CT) et l'API CertSpotter (SSLMate).

### 3.1 Recherche via crt.sh

```
# Rechercher tous les certificats contenant des sous-domaines hackademint.org

# (le caractère % fonctionne comme un wildcard SQL sur crt.sh)

# À ouvrir dans un navigateur :

crt.sh/?q=%25.hackademint.org
```

### 3.2 Recherche via CertSpotter

```
# API CertSpotter (JSON)

curl -s 'https://api.certspotter.com/v1/issuances?domain=hackademint.org&include_subdomains=true&expand=dns_names' | jq .
```

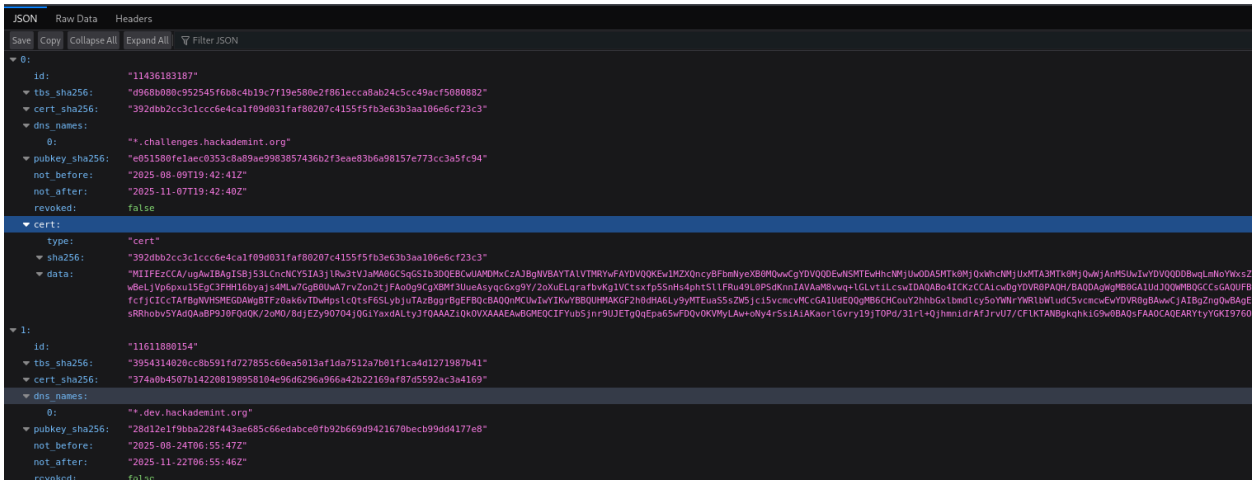
Dans les deux cas, l'objectif est d'identifier un nom de domaine « parlant » dans la liste des DNS names (SAN), contenant un indice lié à My Little Pony.

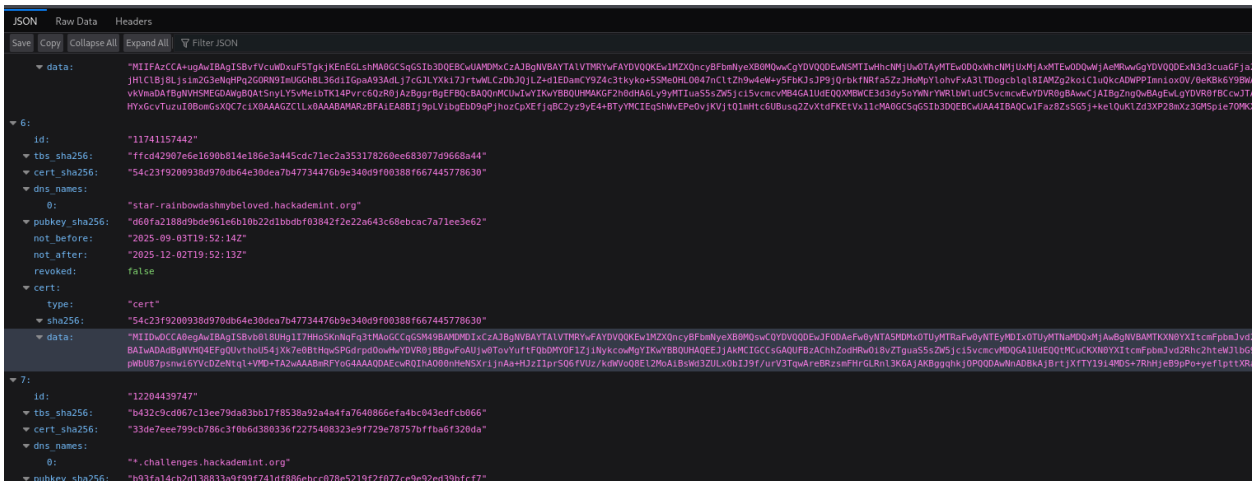
## 4. Filtrer et isoler le certificat pertinent

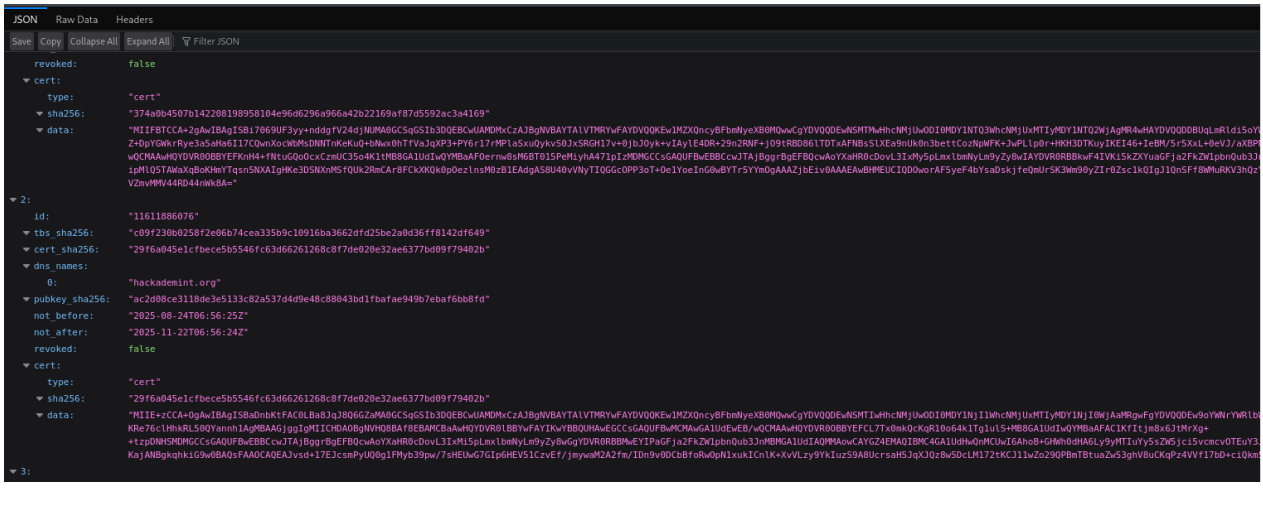
À partir du JSON, on filtre les entrées dont les SAN contiennent des mots-clés (nom de personnage). Dans vos résultats, un nom ressort : `star-rainbowdashmybeloved.hackademint.org`.

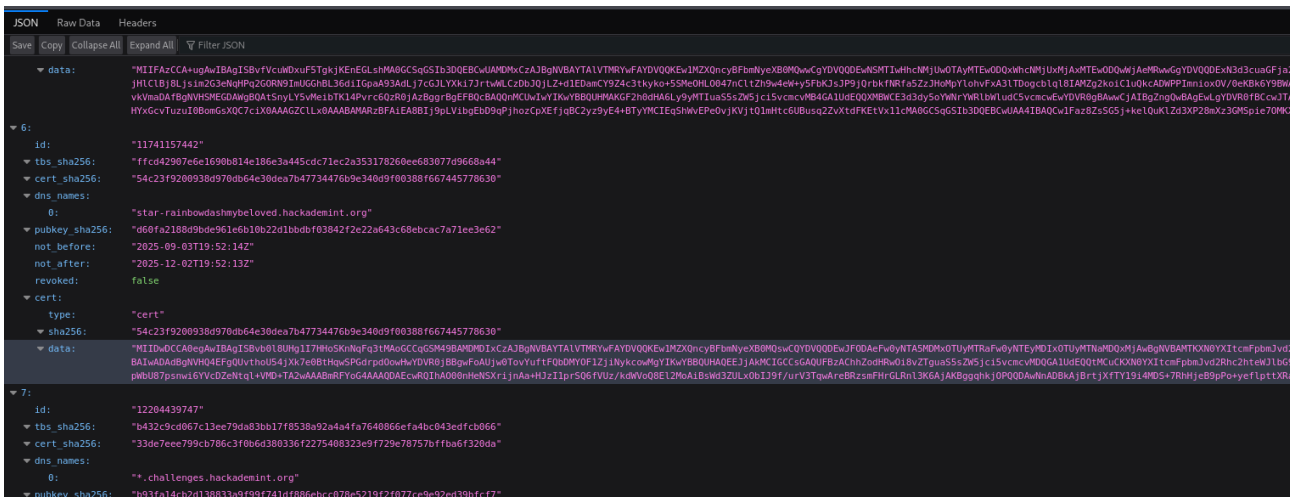
```
# Exemple de filtrage (CertSpotter + jq)

curl -s 'https://api.certspotter.com/v1/issuances?domain=hackademint.org&include_subdomains=true&expand=dns_names' \

| jq -r '.[] | select(.dns_names[]? | test("rainbow|pony|pinkie|twilight|flutter|rarity|applejack"; "i")) | .dns_names[]'
```

### Captures – extraction du sous-domaine dans le JSON (référence)






## 5. Prouver que le certificat est valide (même si le site ne répond pas)

Un certificat présent dans CT atteste qu'une AC l'a émis et qu'il est publiquement journalisé. La « validité » se vérifie ensuite via (1) la période `notBefore/notAfter`, (2) l'absence de révocation, et (3) une chaîne de confiance cohérente vers l'AC.

Vérification locale de CN/SAN (à partir de `cert.data` en base64) :

```
# 1) Décoder en DER puis afficher CN/SAN

base64 -d cert.b64 > cert.der

openssl x509 -inform der -in cert.der -noout -subject -issuer -dates -ext subjectAltName
```

Optionnel : vérifier la chaîne (si vous récupérez l'intermédiaire/CA) :

```
# Exemple (selon le bundle AC utilisé)

# openssl verify -CAfile chain.pem cert.pem
```

Vous devez retrouver dans `subjectAltName` (SAN) le nom DNS complet `star-rainbowdashmybeloved.hackademint.org`.

## 6. Déduction du flag

Le sous-domaine contient un message lisible. L'interprétation la plus cohérente avec l'énoncé est de récupérer le **nom du personnage** : *Rainbow Dash*.

Normalisation recommandée :

- Partir du FQDN trouvé dans le SAN.
- Retirer le suffixe `.hackademint.org`.
- Retirer le préfixe technique éventuel (`star-`).
- Extraire le nom du personnage (ici `rainbowdash`) puis adapter la casse si nécessaire.

Candidats typiques à tester (du plus « sémantique » au plus « littéral ») :

- Star{rainbowdash}
- Star{RainbowDash}
- Star{rainbowdashmybeloved}
- Star{star-rainbowdashmybeloved}

Si la plateforme refuse un candidat, cela signifie uniquement que la normalisation attendue diffère (casse, présence du préfixe, ou extraction stricte du nom).

## 7. Pièges et erreurs fréquentes

- Confondre indisponibilité HTTP (502) et absence du certificat : CT prouve l'émission, indépendamment du service web.
- Répondre avec une empreinte SHA-256 (cert/pubkey) alors que l'énoncé attend une chaîne lisible.
- Inclure le wildcard `*.` ou le domaine complet alors que seule l'étiquette utile est attendue.
- Oublier de tester la casse (RainbowDash vs rainbowdash).

## Annexe A – Extrait indiciel

Dans nos résultats CT, l'entrée qui contient l'indice mentionne explicitement : `dns_names: ["star-rainbowdashmybeloved.hackademint.org"]`.

**Certificate Transparency (CT)**

CT est un mécanisme de "journalisation publique" des certificats TLS/SSL : lorsqu'une autorité émet un certificat, celui-ci est généralement **enregistré dans des logs CT append-only** (gérés par divers opérateurs).

Objectifs principaux :

- **Détecter** les certificats émis par erreur ou de manière frauduleuse (ex. pour un domaine qui ne vous appartient pas).
- **Rendre auditable** l'écosystème des certificats : on peut rechercher quels certificats ont été émis pour un domaine, même si le site n'est pas accessible.

En pratique, CT permet donc de retrouver des **noms de sous-domaines** présents dans le certificat (souvent via le champ SAN), même si ces sous-domaines ne répondent pas en DNS/HTTP.

**AC (Autorité de Certification)**

Une AC (en anglais CA, Certificate Authority) est l'organisme qui **émet et signe** des certificats numériques X.509 (TLS). Elle :

- vérifie (à un certain niveau) que le demandeur **contrôle le domaine** concerné (validation DV/OV/EV selon le type),
- signe le certificat avec sa clé privée, ce qui permet aux navigateurs/systèmes de **faire confiance** au certificat via une chaîne de certification (AC intermédiaire → AC racine).

En résumé : **l'AC émet/signe**, et **CT publie/consigne** pour audit.