

Ping d'IP et de site + Notifications tel

Guide de Monitoring HTTPS et SSL en Python - Niveau Lycée

Introduction :

Tu vas créer un système qui surveille automatiquement des sites web HTTPS pour vérifier :

1. **Si le site est accessible** (il répond ?)
2. **Si le certificat SSL est valide** (la connexion est sécurisée ?)
3. **Recevoir des notifications sur ton téléphone** via Ntfy !

C'est comme avoir un robot qui vérifie que tes sites préférés fonctionnent bien et t'alerte sur ton tel s'il y a un problème !

Partie 1 : Comprendre les concepts de base

1.1 Qu'est-ce qu'une adresse IP ?

Une **adresse IP** est comme l'adresse postale d'un ordinateur sur Internet.

Exemples :

192.168.1.1 ← Adresse IP locale (ton routeur)
8.8.8.8 ← Serveur DNS de Google
142.250.185.46 ← Un des serveurs de Google

1.2 Qu'est-ce qu'un nom de domaine ?

Un **nom de domaine** est le nom facile à retenir d'un site web.

Le **DNS** (Domain Name System) traduit ce nom en adresse IP.

1.3 HTTP vs HTTPS : La différence

Critère	HTTP	HTTPS
Sécurité	☐ Pas sécurisé	☑ Sécurisé
Port	80	443
Icône navigateur	⚠ Pas sécurisé	🔒 Cadenas
Données	En clair	Chiffrées
Certificat SSL	Non	Oui

Analogie simple :

- **HTTP** = Envoyer une carte postale (tout le monde peut lire)
- **HTTPS** = Envoyer une lettre dans une enveloppe scellée avec un sceau officiel

1.4 Qu'est-ce qu'un certificat SSL/TLS ?

Un **certificat SSL** est comme une **carte d'identité** pour un site web.

Il garantit :

- ☑ Le site est bien celui qu'il prétend être
- ☑ La connexion est chiffrée (personne ne peut intercepter tes données)
- ☑ Le certificat n'est pas expiré

Structure d'un certificat :

Certificat SSL de google.com

|

└─ Nom de domaine : *.google.com

└─ Émis par : Google Trust Services

└─ Valide du : 15/01/2025

└─ Expire le : 15/04/2025

└─ Clé publique : [longue suite de caractères]

└─ Signature numérique

1.5 Pourquoi plusieurs sites peuvent avoir la même IP ?

C'est ce qu'on appelle l'**hébergement mutualisé** (shared hosting).

Analogie de l'immeuble :

Immeuble au 123 rue du Web (IP : 185.12.34.56)

|

└─ Appartement 1 : site-a.com

└─ Appartement 2 : site-b.com

└─ Appartement 3 : site-c.com

└─ Appartement 4 : site-d.com

Le serveur sait quel site afficher grâce au **nom de domaine** que tu demandes (HTTP Host header).

Exemple concret :

```
# Ces 3 sites peuvent avoir la même IP
sites = [
    "https://blog-perso.fr",      # IP: 185.12.34.56
    "https://site-commerce.fr",  # IP: 185.12.34.56 (même IP !)
    "https://portfolio-john.fr"  # IP: 185.12.34.56 (même IP !)
]
```

Partie 2 : Installation du projet

2.1 Prérequis

Tu dois avoir installé :

- **Python 3.7 ou supérieur**
- **pip** (installateur de paquets Python)

Vérifier l'installation :

```
python --version    # Doit afficher Python 3.x.x
pip --version       # Doit afficher pip x.x.x
```

2.2 Installation des dépendances

```
# 1. Se placer dans le dossier du projet
cd script-ping-ips

# 2. Installer les bibliothèques nécessaires
pip install -r requirements.txt
```

Explication des bibliothèques :

- `requests` : Pour faire des requêtes HTTPS
- `python-dotenv` : Pour charger les variables du fichier `.env`

2.3 Configuration

Étape 1 : Copier le fichier d'exemple

```
cp .env.example .env
```

Étape 2 : Modifier le fichier `.env`

```
# Liste des sites à surveiller
HOSTS=https://google.com|Google,https://github.com|GitHub

# Timeout en secondes
TIMEOUT=10

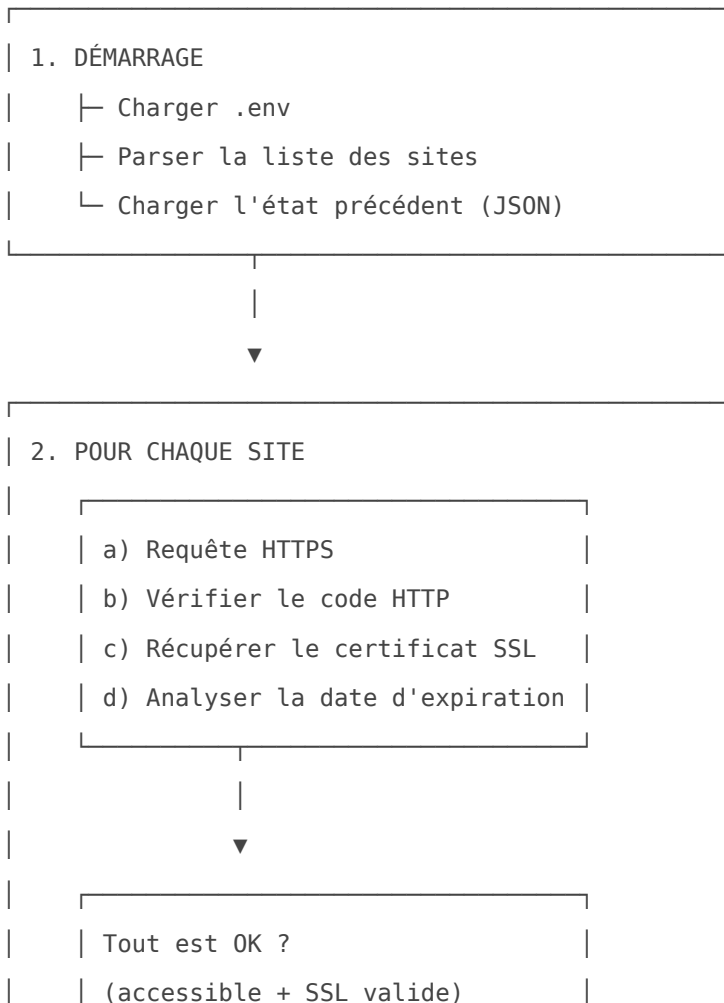
# Topic Ntfy (créer le tien sur https://ntfy.sh)
```

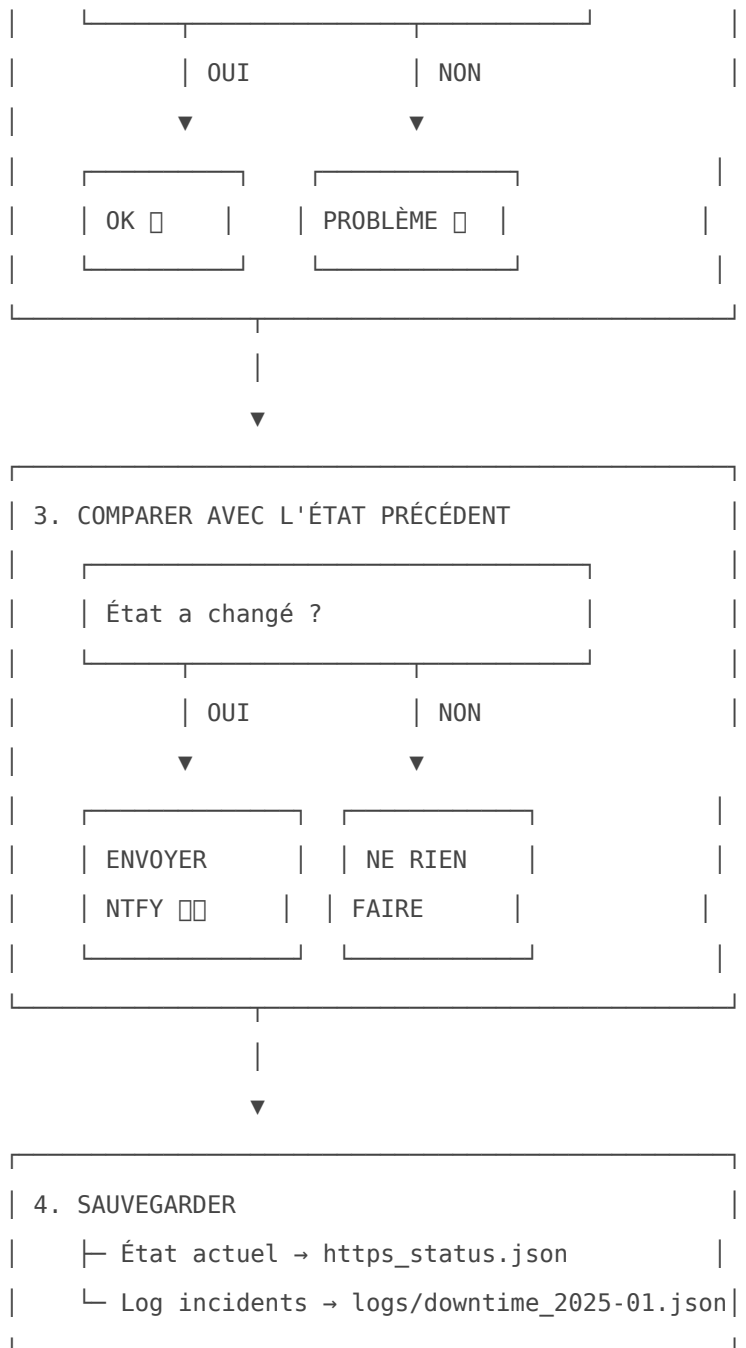
IMPORTANT pour Ntfy :

1. Va sur <https://ntfy.sh>
2. Invente un nom unique (ex: `monitoring-Ab4yteBE35Fwx`)
3. Note ce nom dans `.env`
4. Sur ton téléphone, installe l'app Ntfy et abonne-toi à ce topic

Partie 3 : Comment fonctionne le script Python

3.1 La pipeline complète





3.2 Structure des fichiers

```
script-ping-ips/
|
├─ check_https.py      ← Script Python principal
├─ requirements.txt    ← Liste des dépendances
├─ .env                ← Configuration (ne pas partager !)
├─ .env.example        ← Exemple de configuration
|
├─ https_status.json   ← État actuel (auto-généré)
```

```
| Exemple: {"https://google.com": true, "https://site-hs.com": false}
|
└─ logs/
    └─ downtime_2025-01.json ← Historique de janvier
    └─ downtime_2025-02.json ← Historique de février
```

Partie 4 : Comprendre le code Python

4.1 Vérifier un site HTTPS et son SSL

```
def check_https_and_ssl(url, timeout=10):
    """Vérifie un site HTTPS et son certificat SSL"""
    result = {
        'accessible': False,
        'ssl_valid': False,
        'ssl_expires': None,
        'ssl_days_remaining': None,
        'http_code': None,
        'response_time': None,
        'error': None
    }

    # ÉTAPE 1 : Requête HTTPS
    try:
        start_time = datetime.now()
        response = requests.get(url, timeout=timeout, verify=True)
        end_time = datetime.now()

        result['response_time'] = (end_time - start_time).total_seconds()
        result['http_code'] = response.status_code
        result['accessible'] = (200 <= response.status_code < 400)

    except requests.exceptions.SSLError:
        result['error'] = "Certificat SSL invalide ou expiré"
```

```

        return result

# ÉTAPE 2 : Vérifier le certificat SSL
parsed = urlparse(url)
hostname = parsed.hostname
port = 443

context = ssl.create_default_context()

with socket.create_connection((hostname, port), timeout=timeout) as sock:
    with context.wrap_socket(sock, server_hostname=hostname) as secure_sock:
        cert = secure_sock.getpeercert()

        # Analyser la date d'expiration
        not_after = cert['notAfter'] # 'Jan 15 23:59:59 2025 GMT'
        expires = datetime.strptime(not_after, '%b %d %H:%M:%S %Y %Z')

        days_remaining = (expires - datetime.now()).days

        result['ssl_valid'] = True
        result['ssl_expires'] = expires.strftime('%Y-%m-%d %H:%M:%S')
        result['ssl_days_remaining'] = days_remaining

return result

```

Explication pas à pas :

1. **Requête HTTPS** : On essaie de se connecter au site
 - Si ça marche → `accessible = True`
 - Si erreur SSL → On retourne directement une erreur
2. **Récupérer le certificat** : On se connecte en SSL et on demande le certificat
3. **Analyser la date** : On extrait la date d'expiration et on calcule combien de jours il reste

4.2 Envoyer une notification Ntfy

```

def send_ntfy(topic, title, message, priority='default', tags=''):
    """Envoie une notification sur ton téléphone via Ntfy"""
    url = f"https://ntfy.sh/{topic}"

    headers = {

```

```

        'Title': title,
        'Priority': priority, # urgent, high, default, low
        'Tags': tags         # warning, rotating_light, etc.
    }

    response = requests.post(url, data=message.encode('utf-8'), headers=headers)
    return response.status_code == 200

```

Utilisation :

```

# Envoyer une alerte
send_ntfy(
    topic='mon-monitoring',
    title='🚨 Alerte Site HS',
    message='Le site google.com ne répond plus !',
    priority='urgent',
    tags='rotating_light,warning'
)

```

4.3 Codes HTTP les plus courants

Code	Nom	Signification	Action
200	OK	Tout va bien	🟢 Normal
301	Moved Permanently	Redirection permanente	🟢 Normal
302	Found	Redirection temporaire	🟢 Normal
403	Forbidden	Accès interdit	🔴 Problème
404	Not Found	Page introuvable	🔴 Problème
500	Internal Server Error	Erreur serveur	🔴 Problème
502	Bad Gateway	Serveur indisponible	🔴 Problème
503	Service Unavailable	Service temporairement HS	🔴 Problème

Dans le script :

```

# On considère que c'est OK si le code est entre 200 et 399
result['accessible'] = (200 <= response.status_code < 400)

```

Partie 5 : Utiliser le script

5.1 Lancer le script manuellement

```
python check_https.py
```

Résultat attendu :

```
=====
[] Monitoring HTTPS & SSL - 02/12/25 14:30:15
=====

[] Vérification de https://google.com (Google)...
  └─ Accessible      : [] OUI
  └─ SSL valide      : [] OUI
  └─ Code HTTP       : 200
  └─ Temps réponse   : 0.15s
  └─ SSL expire le   : 2025-03-20 23:59:59
  └─ Jours restants  : 108 jours

[] Vérification de https://expired.badssl.com (Test SSL expiré)...
  └─ Accessible      : [] NON
  └─ SSL valide      : [] NON
  ▲[] Certificat SSL invalide ou expiré
  [] PANNE détectée!
  [] Notification Ntfy envoyée

=====
[] Monitoring terminé
=====
```

5.2 Automatiser avec Cron (Linux/macOS)

Vérifier toutes les 5 minutes :

```
# Ouvrir le crontab
crontab -e
```

```
# Ajouter cette ligne
*/5 * * * * cd /chemin/vers/script-ping-ips && /usr/bin/python3 check_https.py >>
logs/cron.log 2>&1
```

Explication du Cron :

```
*/5 * * * *
|   |   |   |   |
|   |   |   |   └─ Jour de la semaine (0-7, 0 et 7 = dimanche)
|   |   |   └─── Mois (1-12)
|   |   └────── Jour du mois (1-31)
|   └───────── Heure (0-23)
└────────── Minute (0-59)

*/5 = Toutes les 5 minutes
```

Exemples :

```
*/1 * * * *      # Toutes les minutes
*/15 * * * *     # Toutes les 15 minutes
0 * * * *        # Toutes les heures (à minute 0)
0 0 * * *        # Tous les jours à minuit
0 9 * * 1        # Tous les lundis à 9h00
```

5.3 Automatiser sur Windows (Task Scheduler)

Étape 1 : Créer un fichier `.bat`

Crée `run_monitoring.bat` :

```
@echo off
cd C:\Users\flavi\Documents\GitHub\script-ping-ips
python check_https.py >> logs\cron.log 2>&1
```

Étape 2 : Ouvrir le Planificateur de tâches

1. Recherche "Planificateur de tâches" dans Windows
2. Clique sur "Créer une tâche simple"

3. Nom : "Monitoring HTTPS"
 4. Déclencheur : "Répéter toutes les 5 minutes"
 5. Action : "Démarrer un programme"
 6. Programme : `C:\Users\flavi\Documents\GitHub\script-ping-ips\run_monitoring.bat`
-

Partie 6 : Recevoir les notifications sur ton téléphone

6.1 Installer Ntfy sur ton téléphone

Android :

1. Ouvre le Play Store
2. Cherche "Ntfy"
3. Installe l'application
4. Ouvre l'app et clique sur "+"
5. Entre ton topic (ex: `monitoring-flavi-2025`)

iOS :

1. Ouvre l'App Store
2. Cherche "Ntfy"
3. Installe l'application
4. Abonne-toi à ton topic

6.2 Types de notifications

Le script envoie 2 types de notifications :

1. Alerte (site HS) :

🔔🔔Alerte Site HS

Site: Google

URL: <https://google.com>

🔔🔔ALERTE

Site inaccessible ou SSL invalide

Date: 02/12/25 14:30:15

2. Récupération (site rétabli) :

☐ Récupération

Site: Google

URL: https://google.com

☐ Le site est de nouveau accessible!

Date: 02/12/25 14:35:20

6.3 Priorités des notifications

Priorité	Quand	Icône mobile	Son
min	Info sans importance	Silencieux	Non
low	Info	Notification normale	Non
default	Normal	Notification normale	Oui
high	Important	Notification insistante	Oui
urgent	Critique	Notification + vibre	Oui

Dans le script :

- Panne → `urgent` avec icône ☐
- Récupération → `default` avec icône ☐

Partie 7 : Exercices pratiques

Exercice 1 : Tester ton lycée

Ajoute le site de ton lycée dans `.env` :

```
HOSTS=https://google.com|Google,https://site-de-ton-lycee.fr|Mon Lycée
```

Puis lance :

```
python check_https.py
```

Questions :

- Le certificat SSL est-il valide ?
- Dans combien de jours expire-t-il ?
- Quel est le temps de réponse ?

Exercice 2 : Alerte d'expiration

Modifie le script pour envoyer une alerte si le certificat expire dans moins de 7 jours.

Indice : Regarde la ligne 145 du script (`check_https.py:145`)

```
# Trouve cette partie et modifie-la
if days_remaining < 30: # Change 30 en 7
    result['error'] = f"⚠️ Certificat expire dans {days_remaining} jours!"
```

Exercice 3 : Ajouter un site de test

Teste avec ce site qui a un certificat SSL expiré :

```
https://expired.badssl.com
```

Ajoute-le dans `.env` et vérifie que tu reçois une alerte Ntfy !

Partie 8 : Comprendre les erreurs courantes

Erreur 1 : ModuleNotFoundError

```
ModuleNotFoundError: No module named 'requests'
```

Solution :

```
pip install -r requirements.txt
```

Erreur 2 : Certificat SSL invalide

```
ssl.SSLError: [SSL: CERTIFICATE_VERIFY_FAILED]
```

Causes possibles :

- Le certificat est expiré
- Le certificat n'est pas émis pour ce domaine
- Problème de chaîne de certification

Erreur 3 : Timeout

```
requests.exceptions.Timeout: HTTPSConnectionPool(host='example.com', port=443)
```

Solutions :

- Augmente le timeout dans `.env` (ex: `TIMEOUT=20`)
- Vérifie ta connexion Internet
- Le site est peut-être vraiment lent ou HS

Erreur 4 : Ntfy ne reçoit pas les notifs

Vérifications :

1. Le topic est bien le même dans `.env` et dans l'app
2. Ton téléphone a Internet
3. Tu es bien abonné au topic dans l'app Ntfy
4. Teste manuellement : <https://ntfy.sh/ton-topic>

Partie 9 : Glossaire

Terme	Définition
IP	Adresse unique d'un ordinateur sur Internet
DNS	Système qui traduit les noms de domaine en IP

Terme	Définition
HTTPS	Protocole sécurisé pour naviguer sur le web
SSL/TLS	Protocoles de chiffrement pour sécuriser les connexions
Certificat	Document numérique prouvant l'identité d'un site
Port	Numéro identifiant un service (80=HTTP, 443=HTTPS)
Timeout	Temps maximum d'attente d'une réponse
Cron	Système d'automatisation de tâches sur Linux/macOS
JSON	Format de fichier pour stocker des données structurées
API	Interface permettant de communiquer avec un service (Ntfy)
Thread	Fil d'exécution permettant d'exécuter plusieurs tâches en parallèle

Partie 10 : Récapitulatif

Ce que tu as appris

□ Différence entre IP et nom de domaine □ Comment fonctionne HTTPS et SSL □ Pourquoi plusieurs sites peuvent avoir la même IP □ Comment vérifier qu'un site HTTPS est accessible en Python □ Comment vérifier la validité d'un certificat SSL □ Comment envoyer des notifications sur ton téléphone avec Ntfy □ Comment automatiser la surveillance avec Cron