**Mode Opératoire**

# Investiguer l'activité réseau d'un poste pédagogique

| | |
|---|---|
| **Code :** | MO-SEC-007 |
| **Version :** | 1.0 |
| **Date :** | 7 octobre 2026 |
| **Auteur :** | Cédric LEGRAND |
| **Classification :** | USAGE INTERNE — Équipe BTS SIO |

## Historique des révisions

| Version | Date | Modifications |
|---|---|---|
| 1.0 | 07/10/2026 | Création initiale, éprouvée sur un cas réel (poste de la salle i111) |

# 1 Objet

Ce mode opératoire décrit la procédure d'investigation de l'activité réseau d'un poste pédagogique de l'infrastructure BTS SIO du Lycée Jean Lurçat : quels domaines ont été demandés par le poste, à quelles heures, et quel compte utilisateur était ouvert dessus.

La méthode exploite exclusivement les journaux **déjà produits** par l'infrastructure, en lecture seule : le journal DNS de Pi-hole (horodatage à la seconde, par adresse IP cliente), le journal Sécurité du contrôleur de domaine (événements 4624/4768) et le journal `eve.json` de Suricata (SNI TLS sur alertes). Aucune interception de contenu ni modification des systèmes n'est nécessaire.

Le document couvre l'identification du poste, l'extraction et la qualification des domaines (distinction visite effective / appel tiers), l'identification du compte ouvert, et la conservation des preuves.

# 2 Champ d'application

| | |
|---|---|
| **Public concerné** | Référent infrastructure de l'équipe BTS SIO |
| **Systèmes** | Pi-hole (CT `docker-srv` `10.0.112.20`), contrôleur de domaine `DC1` (`10.0.112.2`), pare-feu OPNsense (`10.0.112.1`) |
| **Postes couverts** | Postes fixes des salles (DNS via Pi-hole) ; le Wi-Fi étudiants (VLAN 240) n'est **pas** couvert par Pi-hole |
| **Durée estimée** | 15–30 minutes |

# 3 Prérequis

**Prérequis**

- **Demande motivée dans le cadre de l'administration déclarée du réseau (charte informatique signée par les utilisateurs) ; toute exploitation disciplinaire relève des procédures de l'établissement**
- **Accès SSH à Proxmox (`10.0.112.200`) et à OPNsense (clé du poste d'administration ; commande `bts`)**
- **Identifiants `Administrateur` du domaine et Pi-hole dans le coffre Vaultwar-**

**den de l'équipe**
— **Poste d'administration Linux avec `uv` (scripts PEP 723), `dig`, `jq`**

# 4 Procédure

## 4.1 Identifier le poste et son adresse IP

### 1 Correspondre nom de poste, IP et MAC

— **La convention de nommage encode la salle : `HPZ2S111PC04` = salle i111, poste 04**
— **IP depuis le nom : `dig +short @10.0.112.2 HPZ2S111PC04.bts.sio`**
— **Nom depuis l'IP : `dig +short @10.0.112.2 -x 10.0.232.17`**
— **MAC et fabricant : table ARP d'OPNsense (Interfaces → Diagnostics → ARP Table) ou API : `POST /api/diagnostics/interface/search_arp`**
— **Postes Wi-Fi du moment : baux Kea, `POST /api/kea/leases4/search` (VLAN 240 étudiants, plage `10.0.240.100-250`)**

**Quels postes passent par Pi-hole**

**Les postes fixes des salles (réseau plat `10.0.23x.0/24`) interrogent Pi-hole directement avec leur IP réelle. Les clients du Wi-Fi étudiants utilisent Unbound sur OPNsense : pour eux, seuls les flux vus par Suricata (alertes avec SNI) et le filtrage OPNsense sont exploitables.**

## 4.2 Extraire les requêtes DNS du poste (Pi-hole)

2 **Lire la base FTL de Pi-hole en lecture seule**

**Depuis le poste d'administration, via Proxmox (Pi-hole est un conteneur du CT 200 `docker-srv`) :**

1. **Pousser un script Python dans le CT :** `pct push 200 script.py /tmp/script.py`
2. **Exécuter :** `pct exec 200 -- python3 /tmp/script.py`
3. **Le script ouvre la base en lecture seule (URI SQLite `mode=ro`) :**

```
sqlite3.connect(
    "file:/opt/docker/pihole/etc-pihole/pihole-FTL.db?mode=ro",
    uri=True)
```

**Requête SQL de base (jour courant, un poste) :**

```
SELECT timestamp, domain, status, type FROM queries
WHERE client = '10.0.232.17'
  AND timestamp >= strftime('%s','now','start of day')
ORDER BY timestamp
```

**Colonnes utiles : `status` (1, 4, 5, 6–10 = bloqué par listes ; 2 = forwarded, 3 = cache), `type` (1 = A, 28 = AAAA, 65 = HTTPS).**

**Rapatrier plutôt que traiter en ligne**

**Filtrer par `client` dès la requête SQL, écrire le résultat en JSON dans `/tmp`, le rapatrier (`pct pull` + `scp`) puis supprimer les fichiers temporaires du CT. L'analyse se fait ensuite hors ligne, sans poids pour la production.**

## 4.3 Qualifier : visite effective ou appel tiers

Une requête DNS ne prouve pas une visite : elle peut venir d'un appel tiers (publicité, CDN, tracker), d'un préchargement ou d'un onglet restauré. Appliquer trois critères avant toute conclusion :

3 **Vérifier la nature de chaque domaine**

1. **Dispersion temporelle : des dizaines de domaines résolus dans les mêmes 2–3 secondes (typiquement au lancement du navigateur) relèvent d'un chargement automatique (restauration de session, page d'accueil à raccourcis), pas de visites humaines successives**
2. **Ouverture de grappe : une visite réelle résout d'abord le domaine racine, puis ses satellites (CDN, API) dans la seconde qui suit**
3. **Domaines de service : `accounts.*`, `login.*`, `api.*` ou les CDN spécialisés d'une console (ex. AWS) attestent une interaction effective**

**Exclure du constat : télémétrie OS (`*.microsoft.com`, `msftconnecttest...`), vérificateurs de certificats (`ocsp.*`), CDN génériques (Akamai, CloudFront, Fastly), sondes navigateur (`example.org`, `*-portal-detection.*`) et requêtes bloquées par Pi-hole (jamais abouties).**

**Ne pas sur-interpréter**

**Le maintien d'onglets ouverts génère des requêtes répétées pendant des dizaines de minutes sans aucune lecture active. Conclure sur le pattern (rafale initiale, récurrence, grappes séparées), jamais sur le seul volume. En cas de doute, qualifier « chargement automatique / onglets ouverts » plutôt que « visite ».**

## 4.4 Croiser avec les connexions TLS (Suricata)

4

### Extraire les SNI du poste depuis eve.json

**Sur OPNsense, `/var/log/suricata/eve.json` ne journalise que les événements `alert` et `anomaly` : seules les connexions TLS dont le SNI correspond à une signature chargée sont visibles (champ `tls.sni`). Filtrer par `src_ip` avec un script Python (pas de `jq` sur FreeBSD) ; l'API `POST /api/ids/service/query_alerts` offre une alternative paginée sans SSH.**

**Le SNI confirme une connexion TLS établie quand il est présent ; son absence ne prouve rien (couverture partielle des signatures). L'événement `www.msftconnecttest.com` marque le démarrage réseau d'un poste Windows.**

## 4.5 Identifier le compte ouvert sur le poste (DC1)

5

### Requêter le journal Sécurité de DC1 via WinRM

**Les agents Wazuh ne couvrent que la salle S110 ; pour les autres salles, le journal Sécurité de `DC1` fait foi. WinRM (5985) est ouvert sur `DC1` ; utiliser `pywinrm` (NTLM, compte `BTS\Administrateur`) depuis un script PEP 723 :**

```
Get-WinEvent -FilterHashtable @{LogName='Security';
  Id=4624,4768; StartTime=...; EndTime=...} |
  Where-Object { $_.Message -match '<ip_du_poste>' }
```

**Lire `TargetUserName`, `IpAddress`, `WorkstationName` dans les propriétés XML de l'événement. Un 4624 de type 3 dont**

`WorkstationName` **est le poste étudié, à l'heure du début d'activité DNS, établit la correspondance compte ↔ poste. Résoudre ensuite le nom complet :** `Get-ADUser -Identity <compte> -Properties DisplayName`.

### 4.6 Conserver les preuves

**6** **Sceller les extractions**

1. **Calculer les empreintes SHA-256 de chaque fichier d'extraction (`sha256sum`) et les consigner dans le rapport**
2. **Conserver les fichiers bruts hors dépôt Git (données nominatives)**
3. **Produire le rapport en LaTeX/PDF (hors dépôt), avec méthodologie, chronologie, qualification par domaine et limites probantes**
4. **Supprimer les fichiers temporaires des serveurs (`/tmp`) et verrouiller le coffre (`bw lock`)**

## 5 Vérification

Après avoir effectué les opérations de ce mode opératoire, vérifier les points suivants :

**Vérification**

- ☐ **La correspondance IP ↔ nom de poste est confirmée par le DNS inverse**
- ☐ **L'ouverture de session (4624) précède de quelques secondes le début de l'activité DNS**
- ☐ **Chaque domaine du constat est qualifié (visite / chargement automatique / appel tiers) avec son critère**

> ☐ **Les empreintes SHA-256 des extractions sont consignées**
> ☐ **Les fichiers temporaires sont supprimés des serveurs et le coffre est verrouillé**

# 6 Dépannage

**Aucune requête du poste dans Pi-hole** Le poste est éteint, utilise un DNS différent (vérifier la configuration distribuée), ou contourne en DoH/VPN — contrôler alors les alertes Suricata et les flux bloqués du pare-feu.

**Aucun événement 4624 pour l'IP** Élargir la fenêtre horaire et inclure les événements 4768 (TGT Kerberos) ; vérifier que le poste est bien joint au domaine.

**`eve.json` volumineux** Ne jamais rapatrier le fichier complet : filtrer sur place par `src_ip` avec un script Python déposé dans `/tmp`.

# 7 Documents liés

— **MO-PLT-019** — Gérer le filtrage DNS (Pi-hole)
— **MO-SEC-001** — Exploitation Wazuh (agents limités à la salle S110 à ce jour)
— **MO-SEC-002** — Consulter et traiter les alertes Suricata IDS
— **MO-NET-008** — Architecture du Wi-Fi pédagogique (DNS du VLAN 240 hors Pi-hole)