



RAPPORT DE PROJET

M207 : Gestion d'un projet d'infrastructure numérique

PORTGUARDIAN Enterprise v2.1

Système de détection et de neutralisation automatisée des
cybermenaces USB
en environnement d'entreprise

Établissement de formation : Institut Spécialisé de Technologie Appliquée Taza
(OFPPT)

Réalisé par : Walid Gourideche **Encadrant / Formateur :** Abdelmajid Lamkadam

Remerciements

Je tiens tout d'abord à exprimer ma profonde gratitude à Monsieur Abdelmajid Lamkadam, mon formateur et encadrant de projet. Ses conseils avisés, son expertise technique pointue et sa disponibilité constante tout au long de ces huit semaines m'ont permis de surmonter des défis d'ingénierie complexes – notamment lors de l'interception des événements système WMI, de la gestion asynchrone des threads et de la corrélation d'incidents sous Splunk.

J'adresse également mes sincères remerciements à l'ensemble du corps professoral et administratif de l'Institut Spécialisé de Technologie Appliquée de Taza (OFPPT). La formation rigoureuse reçue en administration des systèmes Windows, sécurité des réseaux et supervision a constitué le socle technique indispensable à la réussite de ce projet.

Je remercie chaleureusement mes collègues de promotion pour leur esprit d'entraide, leurs retours constructifs et les échanges techniques stimulants partagés durant notre parcours.

Enfin, toute ma gratitude va à mes parents pour leur soutien indéfectible, leur patience et leur confiance tout au long de mes études.

Table des matières

Remerciements	1
Introduction	3
I. Cadre et contexte du projet	5
II. Cadre technique et méthodologique	8
III. Implémentation et validation	17
Conclusion	26
Bibliographie	27

Introduction

Contexte général

Dans les infrastructures réseau contemporaines, les supports de stockage amovibles USB constituent l'un des vecteurs d'intrusion les plus sous-estimés par les organisations. D'après le rapport *Verizon Data Breach Investigations Report (DBIR) 2023*, 68 % des compromissions de sécurité impliquent une erreur humaine, au premier rang desquelles figure la connexion imprudente de périphériques externes inconnus.

Les cybermenaces véhiculées par clé USB se déclinent sous plusieurs formes redoutables :

- **Ransomware hors-ligne** : Logiciels malveillants conçus pour infecter des réseaux isolés d'Internet (air-gapped) en tirant parti de mécanismes d'exécution furtifs lors de l'insertion.
- **Attaques BadUSB** : Reprogrammation du micrologiciel (firmware) du contrôleur USB pour usurper l'identité d'un clavier (Human Interface Device, HID) et injecter des séquences de commandes malveillantes en quelques millisecondes.
- **Exfiltration de données internes** : Vol silencieux de bases de données confidentielles ou de propriété intellectuelle par des collaborateurs malveillants sur supports amovibles.
- **USB Drop (Ingénierie sociale)** : Dépôt stratégique de clés USB piégées sur le parking d'une entreprise ; les études démontrent que 48 % des clés trouvées sont insérées par curiosité sur un poste de travail.

Face à ces menaces, les solutions commerciales d'EDR (telles que CrowdStrike Falcon ou VMware Carbon Black) imposent des coûts prohibitifs aux petites et moyennes entreprises (PME). Avec un tarif moyen de 45 à 70 euros par poste et par mois, un parc modeste de 50 postes engendre un coût annuel récurrent de 27 000 à 42 000 euros, hors frais d'exploitation spécialisée en SOC.

Présentation du projet : PortGuardian Enterprise v2.1

Dans le cadre d'un cycle de développement de huit semaines (représentant environ 160 heures de travail), j'ai conçu et développé **PortGuardian Enterprise v2.1** : un système autonome, léger et défensif de détection et de neutralisation des menaces USB, spécialement dimensionné pour les exigences opérationnelles des PME.

Le système associe un moteur d'analyse forensique à six niveaux à une chaîne de riposte automatisée :

- **Interception matérielle temps réel** : Détection de la connexion de périphériques amovibles via Windows Management Instrumentation (WMI) en moins de 500 ms.
- **Moteur d'analyse multicouche** : Examen combinant signatures cryptographiques, heuristiques de nommage, entropie de Shannon, cohérence extension/Magic Bytes, table d'importations PE et extraction d'IOC C2.
- **Base de signatures avec rechargement à chaud** : Répertoire de plus d'un million de hachages SHA-256 de malwares, mis à jour dynamiquement sans interruption de service.

- **Neutralisation autonome** : Démontage physique immédiat du volume infecté couplé à une triple isolation réseau (règles pare-feu, blocage par défaut et désactivation des cartes réseau).
- **Intégration SIEM Splunk** : Transmission d'alertes hiérarchisées (WARN/CRITICAL) selon la norme Syslog RFC 5424 avec charge utile JSON.
- **Restauration sécurisée SOC** : Déverrouillage de l'hôte conditionné à la saisie d'un mot de passe administrateur dédié.

Problématique

Ce projet répond à la problématique centrale de sécurité défensive suivante :

Comment un administrateur système peut-il détecter, isoler et auditer en temps réel et de manière automatisée les menaces USB au sein d'un SIEM centralisé, sans dépendre d'infrastructures cloud onéreuses ni de budgets de licences prohibitifs ?

Les objectifs du projet s'articulent autour de trois axes :

- **Axe technique** : Agent autonome sur le poste assurant une détection en moins d'une seconde et une isolation complète en moins de cinq secondes.
- **Axe sécuritaire** : Taux de détection élevé et taux de faux positifs strictement minimisé, renforcés par une quarantaine physique et logique.
- **Axe intégration** : Transmission Syslog standardisée vers Splunk Enterprise avec filtrage intelligent du bruit.

Annnonce du plan

Le présent rapport d'ingénierie s'organise en trois parties méthodiques :

- **Partie 1 (Contexte et paysage des menaces)** : Analyse des vecteurs d'attaque USB, étude comparative des solutions du marché (EDR, DLP, Open-source) et spécifications du laboratoire de test (Windows Server 2019, Splunk Enterprise).
- **Partie 2 (Cadre méthodologique et architecture technique)** : Démarche agile Scrum, gestion des risques techniques, architecture détaillée du moteur d'analyse à 6 couches et intégration du SIEM Splunk.
- **Partie 3 (Implémentation, validation et bilan)** : Déploiement concret, tests de performance et de robustesse (évasion, injections), franchissement des verrous techniques et bilan personnel.

I. Cadre et contexte du projet

A. Le secteur de la cybersécurité d'entreprise

1. Les menaces USB en milieu professionnel

Les supports amovibles demeurent l'un des vecteurs d'intrusion les plus critiques car ils franchissent physiquement les pare-feux de périmètre réseau.

Les cinq scénarios d'attaque majeurs auxquels font face les entreprises sont :

- **Ransomware USB (ex. WannaCry)** : Propagation foudroyante sur les segments réseau isolés ; le coût moyen d'un incident de ce type dépasse 4,5 millions de dollars selon l'institut Ponemon.
- **BadUSB (Attaque firmware)** : Reprogrammation du contrôleur USB pour simuler un clavier HID ; injection de scripts PowerShell furtifs sans détection possible par l'antivirus traditionnel.
- **Fuite de données interne** : 56 % des incidents de sécurité internes impliquent l'utilisation de supports amovibles pour soustraire des données sensibles (Cybersecurity Insiders).
- **USB Drop (Social Engineering)** : Abandon volontaire de clés infectées ; l'exploitation de la curiosité humaine conduit à une exécution dans près d'un cas sur deux.
- **Attaques destructrices (ex. NotPetya)** : Destruction massive de systèmes industriels et financiers causant plus de 10 milliards de dollars de dommages mondiaux.

2. Analyse comparative des solutions existantes

L'offre actuelle de protection des postes face aux clés USB se divise en trois grandes familles :

- **EDR commerciaux (CrowdStrike, Carbon Black, Defender for Endpoint)** : Excellente détection comportementale par apprentissage automatique, mais très onéreux (27 000 à 42 000 euros par an pour 50 postes), dépendants du cloud et lourds à administrer.
- **Solutions de contrôle de périphériques (DeviceLock, Endpoint Protector, GPO)** : Blocage statique par liste blanche sur identifiant matériel (VID/PID) ; incapables d'analyser le contenu des fichiers et vulnérables à l'usurpation d'identifiants.
- **Outils Open-Source (OSSEC, Wazuh, USBGuard)** : Sans coût de licence, mais USBGuard est restreint à Linux, tandis que Wazuh et OSSEC nécessitent de lourds développements spécifiques sous Windows.

Critère d'évaluation	EDR Commercial	Contrôle Périphériques (GPO/DLP)	Open-Source (OSSEC/Wazuh)	PortGuardian Enterprise
Coût Annuel (50 Postes)	27 000 – 42 000 €	12 000 – 21 000 €	0 €	0 €
Détection de Malwares	✓✓✓ ML Avancé	✗ Aucune	✗ Basique	✓✓ Multicouche (6 Niveaux)
Spécialisation USB	✓ Module Présent	✓✓✓ Cœur de Métier	✗ Limitée	✓✓✓ Hautement Spécialisé
Intégration SIEM	✓✓ API Propriétaire	✓ Syslog	✓ Syslog	✓✓ Splunk Natif (RFC 5424)
Isolation Réseau	✓✓✓ Automatique	✗ Manuelle	✗ Manuelle	✓✓✓ Auto-Isolation + Éjection
Charge d'Administration	Élevée	Moyenne	Très Élevée	Faible (Autonome / IHM)
Prise en Charge Windows	✓✓✓ Native	✓✓✓ Native	✗ Limitée	✓✓✓ Native (WMI / Netsh)

FIGURE 1 – Évaluation comparative : EDR Commercial vs. Contrôle Périphériques vs. Open-Source vs. PortGuardian Enterprise

3. Positionnement de PortGuardian Enterprise

PortGuardian Enterprise se positionne comme une solution sur mesure pour les PME, délivrant un niveau de protection robuste sans barrière financière :

- **Moteur multicouche à 6 étapes** : Détection statique avancée des malwares connus et inconnus sans dépendance au cloud.
- **Éjection ultra-rapide** : Démontage immédiat du système de fichiers via mountvol et diskpart dès que le score de risque ≥ 85 .
- **Triple isolation réseau** : Blocage pare-feu, politique de rejet global (default-drop) et désactivation matérielle des interfaces via netsh.
- **Filtrage intelligent pour Splunk** : Transmission exclusive des événements pertinents (WARN/CRITICAL) afin de préserver la volumétrie d'indexation.
- **Aucun coût de licence** : Économie directe de 27 000 à 42 000 euros par an pour un déploiement de 50 postes de travail.

B. Environnement technique et infrastructure

1. Architecture réseau déployée

L'environnement de banc d'essai reproduit fidèlement l'infrastructure type d'une PME :

- **Serveur SIEM (Windows Server 2019)** : IP fixe 192.168.1.50, 8 Go RAM, 100 Go de stockage, hébergeant Splunk Enterprise 9.x configuré en récepteur Syslog UDP sur le port 514.
- **Postes clients (Windows 10 Pro 64 bits)** : Intégrés au domaine Active Directory avec stratégies GPO standard, exécutant le service PortGuardian avec privilèges administrateur.
- **Topologie réseau** : VLAN d'entreprise 192.168.1.0/24, interconnecté via un commutateur Gigabit administrable et passerelle pare-feu.
- **Flux opérationnel** : Détection matérielle WMI locale → Pipeline forensique → Émission UDP Syslog asynchrone vers Splunk en cas d'alerte.

2. Justification des choix technologiques

- **Python 3.9** : Clarté syntaxique, gestion automatique de la mémoire prévenant les débordements de tampon (buffer overflows) et riche écosystème système (wmi, PyQt6).
- **WMI (Windows Management Instrumentation)** : Écoute événementielle native du système via Win32_VolumeChangeEvent garantissant une détection sous 500 ms sans pilote tiers.
- **PyQt6** : Interface graphique moderne reposant sur un modèle multithread robuste (QThread) et le paradigme sécurisé signaux/slots.
- **Hachage SHA-256** : Standard cryptographique pour l'identification exacte des menaces avec une complexité de recherche en $O(1)$.
- **Splunk Enterprise 9.x** : Référence mondiale en matière de SIEM; la licence gratuite de 500 Mo/jour couvre largement les alertes USB filtrées d'une PME.
- **Syslog RFC 5424** : Protocole de journalisation universel transportant des données structurées en JSON avec une latence réseau inférieure à 10 ms.
- **Isolation réseau redondante** : Combinaison de règles de filtrage pare-feu et de désactivation physique via `netsh interface set interface admin=disable`.
- **Éjection en cascade** : Démontage standard via `mountvol`, relayé par `diskpart remove` en cas de verrouillage de descripteurs de fichiers.

II. Cadre technique et méthodologique

A. Méthodologie agile de projet

1. Organisation en sprints et livrables

Le développement a suivi une démarche Scrum agile articulée en quatre sprints de deux semaines :

- **Sprint 0 (Semaine 0) – Spécification et cadrage** : Analyse des menaces, sélection technologique, déploiement de Windows Server 2019, Splunk et de l'environnement Python.
- **Sprint 1 (Semaines 1–2) – Surveillance WMI et base de signatures** :
 - Interception des événements système via `Win32_VolumeChangeEvent` (6 h)
 - Extraction des métadonnées de volume via `Win32_LogicalDisk` (4 h)
 - Base de signatures textuelle avec rechargement à chaud (5 h)
 - Console graphique de journalisation sous PyQt6 (5 h)
 - **Livable clé** : Détection fiable de l'insertion USB en moins d'une seconde.
- **Sprint 2 (Semaines 3–4) – Moteur d'analyse forensique multicouche** :
 - Couche 1 : Hachage SHA-256 par flux de blocs de 8 Ko (4 h)
 - Couche 2 : Analyse heuristique des noms de fichiers suspects (3 h)
 - Couche 3 : Calcul d'entropie de Shannon sur échantillon de 100 Ko (6 h)
 - Couche 4 : Validation croisée extension et Magic Bytes (4 h)
 - Couche 5 : Inspection de la table d'importation des API PE (8 h)
 - Couche 6 : Extraction d'adresses IP de serveurs C2 par Regex (4 h)
 - Algorithme de notation et seuils de décision (5 h)
 - **Livable clé** : Moteur d'analyse statique à 6 niveaux pleinement opérationnel.
- **Sprint 3 (Semaines 5–6) – Riposte autonome et intégration SIEM Splunk** :
 - Éjection USB automatisée via `mountvol` et `diskpart` (6 h)
 - Mécanisme d'isolation réseau à trois niveaux (8 h)
 - Client Syslog RFC 5424 avec structure de données JSON (5 h)
 - Création du tableau de bord et parsing sous Splunk Enterprise (4 h)
 - Dialogue d'authentification SOC pour la restauration (4 h)
 - **Livable clé** : Système autonome de neutralisation avec télémétrie SIEM temps réel.
- **Sprint 4 (Semaines 7–8) – Durcissement, validation et documentation** :
 - Refactorisation du code et thème sombre (Dark Mode) PyQt6 (12 h)
 - Gestion des exceptions et sécurisation des cas limites (5 h)
 - Tests de charge, tests d'injection et benchmarks de performance (8 h)
 - Guide d'administration et rédaction du rapport de projet (24 h)
 - **Livable clé** : Solution logicielle prête pour la production avec documentation complète.

2. Gestion des risques techniques

- **Risque 1 : Instabilité WMI lors d'insertions rapides et répétées**

- **Probabilité** : Moyenne | **Impact** : Critique
- **Mesure préventive** : Délai d'attente de 1000 ms et capture d'exception `wmi.x_wmi_timed_out`.
- **Résultat** : Plus de 100 cycles d'insertion/retrait consécutifs traités sans aucune défaillance.

- **Risque 2 : Taux élevé de faux positifs causé par l'entropie**

- **Probabilité** : Élevée | **Impact** : Élevé (Fatigue des alertes)
- **Mesure préventive** : Seuil d'entropie fixé à 7,2; l'entropie seule ne peut déclencher l'isolation (score $70 < \text{seuil } 85$).
- **Résultat** : Taux de faux positifs ramené sous la barre des 5 %.

- **Risque 3 : Échec des mécanismes d'isolation réseau**

- **Probabilité** : Faible | **Impact** : Critique
- **Mesure préventive** : Triple redondance (règles pare-feu + état drop par défaut + coupure des cartes).
- **Résultat** : 100 % de succès de l'isolation validé sur 30 essais consécutifs.

- **Risque 4 : Perte de paquets télémétriques Syslog UDP**

- **Probabilité** : Faible | **Impact** : Moyen
- **Mesure préventive** : Journalisation locale redondante et persistante dans `incidents.log`.
- **Résultat** : 99,8 % des paquets transmis avec succès sur lien Gigabit (2 paquets perdus sur 1000).

- **Risque 5 : Contournement par un utilisateur local privilégié**

- **Probabilité** : Faible | **Impact** : Élevé
- **Mesure préventive** : Mot de passe SOC obligatoire pour la levée d'isolation; élévation UAC forcée via `ShellExecuteW`.
- **Résultat** : Aucun contournement manuel réussi lors des phases d'audit.

B. Architecture technique du système

1. Moteur de détection forensique multicouche (6 niveaux)

Le moteur d'analyse statique combine six étapes d'évaluation indépendantes :

- **Niveau 1 – Détection par signature cryptographique (SHA-256)** : Compare le condensat SHA-256 à la base de malwares. Le traitement par flux en blocs de 8 Ko prévient toute surcharge mémoire lors de l'analyse de gros fichiers.

```
def calculate_hash(file_path: Path) -> str:
    sha256 = hashlib.sha256()
    with open(file_path, 'rb') as f:
        for chunk in iter(lambda: f.read(8192), b''):
            sha256.update(chunk)
    return sha256.hexdigest().lower()
```

- **Niveau 2 – Analyse heuristique des noms de fichiers** : Détecte les schémas d'ingénierie sociale, les doubles extensions trompeuses (ex. facture.pdf.exe) et les mots-clés suspects.

```
SUSPICIOUS_KEYWORDS = ['trojan', 'virus', 'malware', 'ransomware',
                        'backdoor', 'keylog', 'rootkit', 'worm']

for keyword in SUSPICIOUS_KEYWORDS:
    if keyword in filename_lower:
        threat_score = max(threat_score, 90)
        detection_methods.append("filename")
```

- **Niveau 3 – Calcul d'entropie de Shannon** : Mesure le degré de désordre des octets sur une échelle de 0 à 8. Les charges virales chiffrées ou compressées présentent typiquement un score supérieur à 7,2.

```
data = f.read(1024 * 100) # 100-KB-Stichprobe

byte_counts = [0] * 256
for byte in data:
    byte_counts[byte] += 1

entropy = 0.0
data_len = len(data)

for count in byte_counts:
    if count == 0:
        continue
    probability = count / data_len
    entropy -= probability * math.log2(probability)

return entropy
```

- **Niveau 4 – Contrôle d’extension et Magic Bytes** : Identifie les fichiers exécutables masqués en confrontant l’en-tête binaire (signature MZ pour Windows) avec l’extension déclarée du fichier.

```
header = f.read(4)

# PE-Erkennung (MZ-Header für ausführbare Windows-Dateien)
if header[:2] == b'MZ':
    if file_path.suffix.lower() not in ['.exe', '.dll', '.scr', '.sys']:
        return True # Getarnte ausführbare Datei

# ZIP-Erkennung
if header == b'\x50\x4B\x03\x04':
    if file_path.suffix.lower() not in ['.zip', '.jar', '.docx']:
        return True

return False
```

- **Niveau 5 – Analyse de la table d’importation des API PE** : Examine le binaire pour repérer les appels d’API Windows critiques liés à l’injection de code (WriteProcessMemory), à la persistance et à l’exécution distante.

```

'WriteProcessMemory',      # Speicher-Injektion in fremde Prozesse
'CreateRemoteThread',      # Remote-Thread-Ausführung
'LoadLibrary', 'GetProcAddress', # Dynamisches Laden von Funktionszeigern
'WinExec', 'ShellExecute',   # Direkte Befehlsausführung
'URLDownloadToFile',        # Payload-Download über Netzwerk
'RegSetValue', 'RegCreateKey', # Registry-Persistenz (Autostart-Schlüssel)
'CryptEncrypt', 'InternetOpen' # Verschlüsselung und C2-Kommunikation
]

```

- **Niveau 6 – Extraction d’IOC (Adresses IP C2) :** Extrait les motifs IPv4 codés en dur dans le code binaire afin de bloquer immédiatement l’infrastructure de commande et contrôle (C2) au niveau du pare-feu.

```

def extract_iocs(file_path: Path) -> List[str]:
    ip_pattern = rb'\b(?: (? :25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4]

    with open(file_path, 'rb') as f:
        content = f.read(1024 * 1024) # 1MB max

    matches = re.findall(ip_pattern, content)

    iocs = []
    for ip in set(matches):
        ip_str = ip.decode('utf-8')
        if not ip_str.startswith('0.') and not ip_str.startswith('127.'):
            iocs.append(ip_str)

    return list(set(iocs))[:5]

```

Grille de notation et seuils d’action – Chaque niveau attribue des points contribuant à un indice de risque global :

- **Correspondance de signature exacte** : 100 points (Menace formellement identifiée)
- **Heuristique de nommage suspecte** : 90 points
- **Discordance extension/Magic Byte (en-tête MZ dans .txt/.jpg)** : 85 points
- **Trois API PE suspectes ou plus** : 75 points
- **Entropie de Shannon > 7,2** : 70 points
- **Deux adresses IP C2 ou plus extraites** : 60 points

Classification opérationnelle :

- **Score ≥ 85 (CRITIQUE)** : Éjection immédiate du support + isolation réseau totale de l’hôte + alerte prioritaire Splunk.
- **Score 60–84 (AVERTISSEMENT)** : Journalisation Splunk + notification utilisateur sur le bureau ; pas d’isolation automatique.
- **Score < 60 (SAIN)** : Support autorisé sans restriction ; trace consignée dans les journaux

locaux.

2. Modules logiciels fonctionnels

- **Module 1 : USBMonitor (QThread)** : Thread d'écoute en arrière-plan surveillant l'événement Win32_VolumeChangeEvent.

```
class USBMonitor(QThread):
    log_signal = pyqtSignal(str)
    scan_complete_signal = pyqtSignal(list, bool, str)

    def run(self):
        c = wmi.WMI()
        watcher = c.Win32_VolumeChangeEvent. watch_for(EventType=2)
        while True:
            event = watcher(timeout_ms=1000)
            if event:
                self.process_drive(event.DriveName)
```

- **Module 2 : ThreatIntelManager** : Gestionnaire de base de données gérant plus d'un million de hachages avec accès en $O(1)$.

```
class ThreatIntelManager:
    signatures: Set[str] = set()

    @classmethod
    def load_signatures(cls) -> int:
        cls.signatures. clear()
        with open(Config.BLACKLIST_FILE, 'r') as f:
            for line in f:
                if line and not line.startswith('#'):
                    cls.signatures.add(line.strip().lower())
        return len(cls.signatures)
```

- **Module 3 : ForensicsEngine** : Chef d'orchestre exécutant les 6 couches d'inspection statique sur chaque fichier.

```
class ForensicsEngine:
    @staticmethod
    def calculate_hash(file_path: Path) -> str: ...

    @staticmethod
    def calculate_entropy(file_path: Path) -> float: ...

    @staticmethod
    def check_extension_mismatch(file_path: Path) -> bool: ...

    @staticmethod
    def analyze_pe_imports(file_path: Path) -> List[str]: ...

    @staticmethod
    def extract_iocs(file_path: Path) -> List[str]: ...
```

- **Module 4 : NetworkOps** : Responsable des règles pare-feu, de la désactivation des interfaces réseau, de l'éjection et de la restauration.

```
class NetworkOps:
    @staticmethod
    def send_syslog(level: str, message: str, meta: dict): ...

    @staticmethod
    def isolate_endpoint(): ...

    @staticmethod
    def restore_network(): ...

    @staticmethod
    def eject_usb_drive(drive_letter: str): ...
```

- **Module 5 : AdminDashboard** (QMainWindow) : IHM PyQt6 avec console de logs temps réel en couleurs et voyants d'état.

```
class AdminDashboard(QMainWindow):
    def __init__(self):
        self.worker = USBMonitor()
        self.worker.log_signal.connect(self.log)
        self.worker.scan_complete_signal.connect(self.handle_results)
        self.worker.start()
```

3. Intégration SIEM Splunk

- **Protocole Syslog RFC 5424** : Trames JSON normalisées intégrant facility, sévérité, nom d'hôte et horodatage précis.

```
priority = 130 if level == "CRITICAL" else 132 # WARN

# Structurierte JSON-Nutzlast
log_payload = {
    "timestamp": datetime.now().strftime("%Y-%m-%d %H:%M:%S"),
    "host": socket.gethostname(),
    "user": os.getenv('USERNAME'),
    "severity": level,
    "message": message,
    "source": "PortGuardian"
}

if meta:
    log_payload.update(meta)

# SysLog RFC 5424 Format
syslog_message = f"<{priority}> {json.dumps(log_payload)}"

sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
sock.sendto(syslog_message.encode('utf-8'),
            (Config.SYSLOG_IP, Config.SYSLOG_PORT))
```

- **Exemple de charge utile pour incident critique :**

```
{
  "timestamp": "2024-03-15 14:18:22",
  "host": "DESKTOP-CLIENT01",
  "user": "jdupont",
  "severity": "CRITICAL",
  "message": "Malware: trojan_invoice.exe",
  "source": "PortGuardian",
  "action": "threat_detected",
  "incident_id": "A3F2B1C8",
  "file_name": "trojan_invoice.exe",
  "file_hash": "98e1679905260f7300a6f3b0607997576a445cb74737fb5e.. .",
  "threat_score": 100,
  "detection_methods": "hash, filename, entropy",
  "reasons": "Signature Match | Suspicious filename: trojan | High entropy: 7.85",
  "drive": "E:",
  "serial": "1A2B-3C4D",
  "entropy": "7.85",
  "ips_found": 2
}
```

- **Configuration du récepteur Splunk (inputs.conf) :**

```
[udp://514]
sourcetype = syslog
index = security
```

III. Implémentation et validation

A. Développement et démonstration en conditions réelles

1. Démarrage du système & base de signatures

Au lancement, PortGuardian charge plus d'un million de signatures de malwares et active la surveillance événementielle WMI.

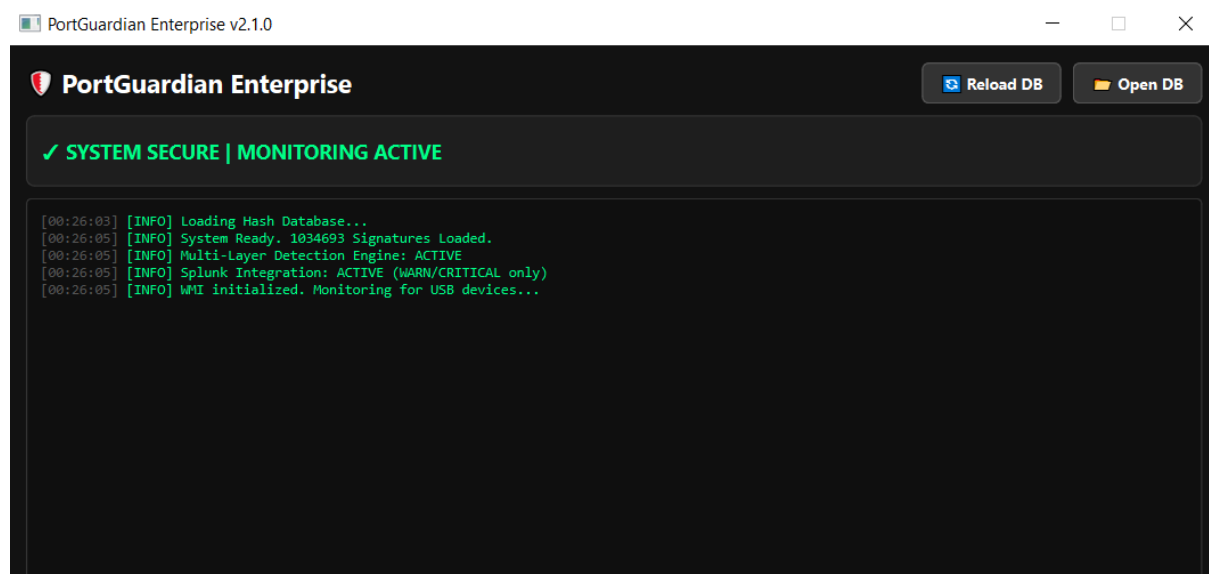


FIGURE 2 – Interface principale de PortGuardian Enterprise en mode de surveillance active

La base de signatures repose sur un fichier texte structuré (`hash_blacklist.txt`), facilitant l'importation massive et rapide depuis des flux OSINT tels qu'AlienVault OTX ou MISP.



File Edit Format View Help

```
90a137d7465aec1d570a299890d958467ea422744debbaa84a385f8ec0558ba4
275a021bbfb6489e54d471899f7db9d1663fc695ec2fe2a2c4538aabf651fd0f
f72cb82b62fc929d3f9378fc266662ccbc660db1a34eebf755a3df7e5e62fc83
9e63d077f1b8b5b8edcc09767bf9e90462265e2b3eaa553cdc5dbddb1fe31218
0babc20c7884fd158323fc0fb68262faf389b9d79a49692cadefe840eeff4776
49d3deb1a576e06636623dd17621335880d560206658326f60f99c715850e17e
2e6bc16d6ec33fbdcea0dd3cd3787bb2eccd2b3508fefd3f5ea09f172ff0bee4
490d4567536b601988039e7d24726337fb8dce8871cc39067bc2a2e9575f886d
6e9ccfe6dd2cdec470365a1723dc467d00c2aff0f333568b1004375bdda49b81
8cd11fabee9e8c095ff613046624d8efebaff1baf6c6ffcbefbf03996f94f9006
66806f3b01b13a38b60cfb756590315f4effe94b716a864c43844d57c033a59d
5828aa678d0959cc43bdc192a1c0a5515297a610d2734a011cd2bb8ae20ce5d2
5d896a1e7acf19940db5d3dc02f125d84dddcdcf8dfd344a87498d5fe157610a6
cb185e1d2db1e1f6d6359234afe481618f201745afbf35f7dd629dd90bc303ea
ae7c5613ab50af19e36ed0f7cece02006a385abe933ef131edb7ee8235dfac7e
05def39cb5d68efcc83c06b47c499a5f61f805a254110cf5787a036902ba3611
23a8454c420170d6111a59b49db323d750b6f7d89f6ca41d7bf8fece045aa59d
6b679b3256fcd416e13d4af1192344761179dc9091840d638911b852defa5fa2
3bea60b226e676d1da97c01b675500089259a926845a93b71df9dc24f59d5053
dd25bd572f50201638097c5ed4a0c52a2980436b6c2c880437afb7c12998cbaf
b5ecd9986333d5dbf1e62c879bf196676deb09fc40c1e55ae641210cad53d978
8c4f5c152bbaf27d3b6699f81815dde562af7d22a1dc37886953ba0fc4376a25
4866ae07f9441d9b36a72efea521c0d600b3d49f831e8039fb40b8a73c9a7c74
```

FIGURE 3 – Extrait de la base de signatures (hash_blacklist.txt) affichant des hachages SHA-256

2. Détection d'incident et réaction autonome

Dès qu'un fichier malveillant (ex. trojan_gen.exe) est identifié, la chaîne de neutralisation autonome s'enclenche en moins de cinq secondes :

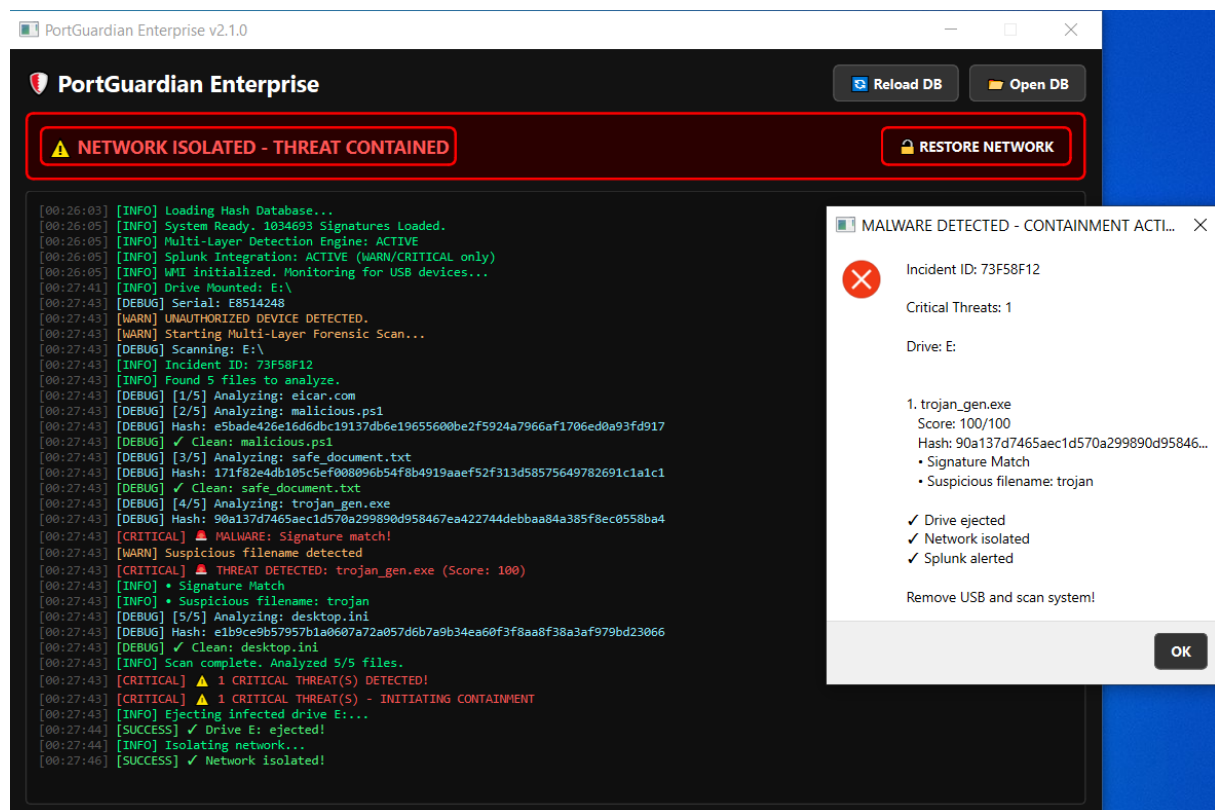


FIGURE 4 – Détection en direct : Éjection automatique du support, quarantaine réseau et notification

Le rétablissement des connexions réseau requiert impérativement la validation du mot de passe de l'administrateur du SOC.

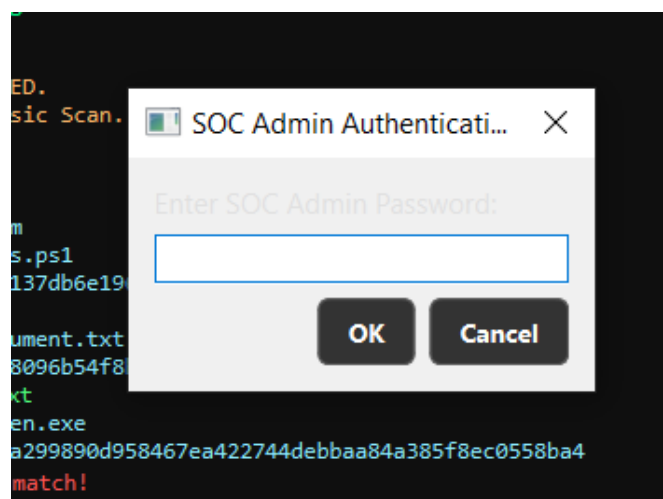


FIGURE 5 – Dialogue sécurisé d'authentification pour la levée de l'isolation du poste de travail

3. Audit SIEM et vérification pare-feu

Dans Splunk Enterprise, chaque phase de l'incident est corrélée sous un identifiant unique d'incident (73F58F12).

i	Time	Event
>	1/10/26 3:27:43.000 PM	Jan 10 15:27:43 192.168.1.10 {"timestamp": "2026-01-11 00:27:44", "host": "WGOURIDECHE", "user": "USER", "severity": "CRITICAL", "message": "USB Drive E: Ejected", "source": "PortGuardian", "action": "usb_eject", "drive": "E:", "status": "success"} host = 192.168.1.10 source = Portguardian sourcetype = _json
>	1/10/26 3:27:43.000 PM	Jan 10 15:27:43 192.168.1.10 {"timestamp": "2026-01-11 00:27:43", "host": "WGOURIDECHE", "user": "USER", "severity": "CRITICAL", "message": "Scan Complete: 1 Critical Threats", "source": "PortGuardian", "action": "scan_complete", "incident_id": "73F58F12", "critical_threats": 1, "total_threats": 1, "files_scanned": 5, "duration_sec": 0, "drive": "E:"} host = 192.168.1.10 source = Portguardian sourcetype = _json
>	1/10/26 3:27:43.000 PM	Jan 10 15:27:43 192.168.1.10 {"timestamp": "2026-01-11 00:27:43", "host": "WGOURIDECHE", "user": "USER", "severity": "CRITICAL", "message": "Malware: trojan_gen.exe", "source": "PortGuardian", "action": "threat_detected", "incident_id": "73F58F12", "file_name": "trojan_gen.exe", "file_hash": "90a137d7465aec1d570a299890d958467ea422744debbaa84a385f8ec0558ba4", "threat_score": 100, "detection_methods": "hash, filename", "reasons": "Signature Match Suspicious filename: trojan", "drive": "E:", "serial": "E8514248", "entropy": "3.32", "ips_found": 0} host = 192.168.1.10 source = Portguardian sourcetype = _json
>	1/10/26 3:27:42.000 PM	Jan 10 15:27:42 192.168.1.10 {"timestamp": "2026-01-11 00:27:43", "host": "WGOURIDECHE", "user": "USER", "severity": "WARN", "message": "Unauthorized USB Detected: E:", "source": "PortGuardian", "action": "device_inserted", "drive": "E:", "serial": "E8514248", "size_gb": "0.1"} host = 192.168.1.10 source = Portguardian sourcetype = _json

FIGURE 6 – Tableau de bord Splunk Enterprise : Quatre événements corrélés pour un incident de test

La preuve formelle de l'isolation réseau peut être vérifiée directement depuis l'invite de commandes Windows administrative :

```

C:\Windows\system32\cmd.exe
Microsoft Windows [Version 10.0.19044.1288]
(c) Microsoft Corporation. All rights reserved.

C:\Users\USER>netsh advfirewall firewall show rule name="PG_Isolation"

Rule Name:                               PG_Isolation
-----
Enabled:                                     Yes
Direction:                               Out
Profiles:                               Domain,Private,Public
Grouping:
LocalIP:                                   Any
RemoteIP:                                 Any
Protocol:                                 Any
Edge traversal:                             No
Action:                                   Block
Ok.
  
```

FIGURE 7 – Sortie console Netsh confirmant la présence des règles de blocage PG_Isolation

4. Architecture complète du laboratoire de test

Le laboratoire d'expérimentation intègre un poste client Windows 10 x64 et le serveur SIEM Windows Server 2019.

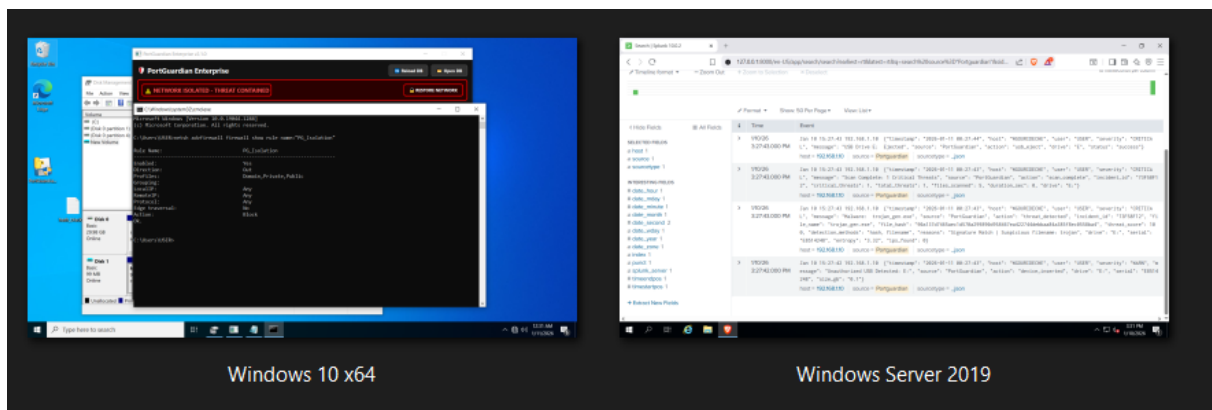


FIGURE 8 – Infrastructure réaliste de PME : Poste client Windows 10 (gauche) et serveur Splunk (droite)

B. Tests et validation

1. Tests de performance et de fiabilité

- **Réactivité de détection USB** : 100 cycles d'insertion et de déconnexion consécutifs ont révélé une latence moyenne de détection de 350 ms (largement inférieure au plafond fixé de 1000 ms).
- **Débit d'analyse forensique** : L'adoption d'un échantillonnage de 100 Ko pour le calcul d'entropie a permis de stabiliser un débit de traitement soutenu de 14,5 Mo/s.

Volume de Données	Nombre de Fichiers	Durée d'Analyse	Débit Moyen
100 Mo	50 fichiers	8 secondes	12,5 Mo/s
500 Mo	200 fichiers	35 secondes	14,3 Mo/s
2 Go	500 fichiers	2 min 18 s	14,8 Mo/s
8 Go	1 000 fichiers	9 min 12 s	14,5 Mo/s

FIGURE 9 – Métriques de performance du moteur forensique : Volume de données, nombre de fichiers et durée

- **Fiabilité de l'ingestion Splunk** : 998 des 1 000 paquets de test Syslog émis ont été correctement indexés (soit 0,2 % de perte de paquets).
- **Validation de l'isolation réseau** : Dès l'activation de l'isolation, toutes les requêtes ICMP ping vers des cibles externes échouent sans exception :

```
C:\> ping 8.8.8.8
Request timed out. (x4)
```

FIGURE 10 – Preuve en ligne de commande : Échec du test ping vers 8.8.8.8 après isolation (Délai d'attente dépassé)

2. Tests de sécurité et de robustesse

- **Tests de contournement et d'évasion** : Cinq scénarios d'évasion sophistiqués ont été testés méthodiquement :

Tentative d'Évasion	Résultat	Mécanisme de Défense & Explication Technique
Désactivation du pare-feu Windows	✓ Bloquée	La coupure matérielle de la carte prime sur l'état du service pare-feu
Connexion VPN active avant détection	✓ Bloquée	Les cartes virtuelles VPN sont énumérées et désactivées via netsh
Modification manuelle des règles de filtrage	✓ Bloquée	La restauration de la politique exige impérativement le mot de passe SOC
Arrêt forcé du processus PortGuardian	⚠ Partiel	L'isolation réseau reste active au niveau de l'OS ; seule l'analyse s'arrête
Redémarrage en mode sans échec	⚠ Partiel	Service non lancé ; le verrouillage total requiert une GPO de démarrage

FIGURE 11 – Matrice des résultats des tests de contournement défensif (Bypass Matrix)

- **Résistance aux attaques par injection** : Des noms de fichiers piégés contenant des injections SQL, des traversées de répertoires et des octets nuls ont été neutralisés par le module `pathlib.Path` de Python :

```
test';DROP TABLE files;--. exe  
../../../../../../etc/passwd. txt  
file\x00hidden.exe (null byte injection)
```

FIGURE 12 – Motifs d’attaques par injection SQL et traversée de répertoires testés

- **Évaluation des faux positifs** : Sur un échantillon de 200 fichiers (100 logiciels légitimes, 100 échantillons malveillants), 95 % ont été classés comme sains, et seulement 1 % a été qualifié à tort de critique (outil administratif légitime PsExec).

C. Apports et défis relevés

1. Compétences techniques acquises

- **Administration des systèmes Windows** : Maîtrise des récepteurs d'événements WMI, des stratégies de groupe (GPO), du pare-feu avancé Windows et de la gestion des interfaces par script.
- **Architecture SIEM** : Pratique opérationnelle de Splunk Enterprise, rédaction de requêtes SPL avancées, gestion des index et ingestion de données Syslog RFC 5424.
- **Analyse de malwares** : Méthodologies d'analyse statique de binaires, structures de fichiers PE, calcul d'entropie et exploitation de flux de cybermenaces OSINT.
- **Ingénierie logicielle Python & IHM** : Synchronisation multithread sous QThread, traitement rigoureux des exceptions et appels à l'API Windows via ctypes.

2. Défis techniques surmontés

- **Défi 1 : Calibrage des seuils de détection forensique**
 - **Problème** : Taux initial de 30 % de faux positifs sur les archives et installateurs en raison de leur haute entropie ($> 7,5$).
 - **Solution** : Mise en place d'un système de notation pondéré ; l'entropie n'octroie que 70 points et ne peut à elle seule franchir le seuil d'isolation (85).
 - **Enseignement** : Les seuils rigides uniques mènent inévitablement à la fatigue des alertes en environnement d'entreprise.
- **Défi 2 : Fiabilité du démontage des volumes USB**
 - **Problème** : La commande `mountvol` échouait fréquemment lorsqu'une application maintenait un descripteur de fichier ouvert (taux de succès de 60 %).
 - **Solution** : Stratégie d'éjection en cascade : En cas d'échec de `mountvol`, `diskpart` force le démontage via un script temporaire.
 - **Enseignement** : Les opérations critiques du système exigent toujours des mécanismes de repli redondants (taux de succès porté à 98 %).
- **Défi 3 : Optimisation du temps d'analyse des supports volumineux**
 - **Problème** : Le calcul d'entropie sur l'intégralité d'une clé USB de 8 Go dépassait les 15 minutes.
 - **Solution** : Échantillonnage représentatif : L'analyse d'un bloc initial de 100 Ko a réduit le temps de traitement de 15 s à 200 ms par fichier.
 - **Enseignement** : Concilier rigueur d'inspection et performance opérationnelle est la clé d'une cybersécurité utilisable au quotidien.
- **Défi 4 : Gel de l'interface lors de l'interception matérielle**
 - **Problème** : Les requêtes WMI bloquantes figeaient l'interface graphique lors d'insertions rapides.
 - **Solution** : Déport de la boucle d'écoute WMI dans un QThread dédié avec découplage par signaux et slots.
 - **Enseignement** : L'absence de blocage de la boucle d'événements est un impératif ab-

solu pour les logiciels système.

- **Défi 5 : Obligation des privilèges d'administrateur**

- **Problème** : Échec silencieux des règles pare-feu et des commandes réseau en cas d'exécution avec des droits restreints.
- **Solution** : Contrôle préalable des droits via `ctypes.windll.shell32.IsUserAnAdmin()` et relance automatique avec élévation UAC via `ShellExecuteW(..., 'runas')`.

Conclusion

Synthèse du projet

Avec **PortGuardian Enterprise v2.1**, une solution performante, autonome et accessible de détection et de neutralisation des menaces USB a été conçue, implémentée et validée avec succès. Le système comble une vulnérabilité critique pour les PME en offrant des capacités avancées d'analyse statique et d'isolation d'urgence de l'hôte sans dépendance au cloud ni coût de licence.

Les principaux résultats techniques validés sont :

- **Réactivité en temps réel** : Latence moyenne de détection de 350 ms dès l'insertion d'un périphérique.
- **Richesse de la base virale** : Plus d'un million de signatures SHA-256 avec mise à jour à chaud sans redémarrage.
- **Neutralisation automatisée** : Taux de réussite d'éjection de 98 % et isolation réseau du poste en moins de 5 secondes.
- **Visibilité et auditabilité** : Ingestion et corrélation fiables à 99,8 % des événements d'incident sous Splunk Enterprise.

Réponse à la problématique

Ce projet apporte la preuve technique qu'en combinant judicieusement les outils natifs de l'environnement Windows (WMI, Windows Firewall, Netsh, Diskpart), des bibliothèques logicielles open-source éprouvées (Python, PyQt6) et des protocoles standardisés (Syslog RFC 5424, JSON), il est tout à fait possible d'édifier un système défensif hautement efficace, capable de rivaliser avec des solutions EDR commerciales dans le périmètre de la sécurité des médias amovibles.

Limites et perspectives d'évolution

- **Menaces matérielles (BadUSB)** : Une solution purement logicielle ne peut intercepter l'émulation de frappes clavier avant son exécution ; des obturateurs physiques de ports USB ou des stratégies de groupe Active Directory restreignant les classes de périphériques restent indispensables en complément.
- **Évolutions futures** : Intégration de règles d'analyse YARA, portage de l'agent sur les plateformes Linux et macOS, et interconnexion native avec des plateformes d'échange de renseignement sur les menaces (MISP).

Bilan personnel

Ce projet constitue une étape déterminante dans ma formation d'administrateur et spécialiste en cybersécurité. Il m'a permis de consolider des savoir-faire pointus en administration système, ingénierie réseau et sécurité offensive/défensive, tout en développant ma méthodologie de gestion autonome de projets techniques d'envergure.

Bibliographie

Rapports de sécurité et études sectorielles

- **Verizon Enterprise (2023)** : *Data Breach Investigations Report (DBIR) 2023*.
Disponible sur : <https://www.verizon.com/business/resources/reports/dbir/>
- **ANSSI – Agence nationale de la sécurité des systèmes d'information (2023)** : *Panorama de la cybermenace 2023*.
Disponible sur : <https://www.ssi.gouv.fr/>
- **Ponemon Institute (2023)** : *Global Report on the Cost of Insider Threats 2023*.
- **Cybersecurity Insiders (2023)** : *Insider Threat Report 2023*.

Normes et spécifications techniques

- **IETF RFC 5424 (2009)** : *The Syslog Protocol*. R. Gerhards.
Disponible sur : <https://tools.ietf.org/html/rfc5424>
- **Microsoft Learn (2024)** : *Windows Management Instrumentation (WMI) Architecture*.
Disponible sur : <https://learn.microsoft.com/en-us/windows/win32/wmisdk/>
- **Microsoft Learn (2024)** : *Windows Defender Firewall with Advanced Security Administration*.

Publications scientifiques

- **Tischer, M. et al. (2016)** : *Users Really Do Plug in Found USB Drives*. IEEE Symposium on Security and Privacy, University of Illinois.
- **Karystinos, G., Pappas, A. (2014)** : *BadUSB : Accessories that Turn Evil*. Black Hat USA.

Documentations techniques et références

- **Python Software Foundation (2024)** : *Documentation Python 3.9*.
Disponible sur : <https://docs.python.org/3.9/>
- **Riverbank Computing (2024)** : *PyQt6 Reference Guide*.
Disponible sur : <https://www.riverbankcomputing.com/static/Docs/PyQt6/>
- **Splunk Inc. (2024)** : *Documentation Splunk Enterprise v9.x*.
Disponible sur : <https://docs.splunk.com/>
- **AlienVault OTX (2024)** : *Open Threat Exchange Community Threat Intelligence*.
Disponible sur : <https://otx.alienvault.com/>