Mettre en place WAZUH comme SIEM : entre théorie et réalité

ANNE-SOPHIE LEDOUX | UMR 1283/8199 EGENODIA EGID PRECIDIAB

…quand un SIEM transforme un labo… et ses admins

- Contexte : UMR8199, données sensibles, ISO 15189
- Pourquoi un SIEM : complexité croissante, logs dispersés
- Choix du SIEM : Wazuh, open-source, multi-OS
- Approche : pragmatique, progressive, adaptée aux petites équipes
- Objectif du RETEX : partager la réalité terrain
- **Pas de démo live (sécurité + contraintes infra)**

Le labo : un terrain exigeant

- Domaine : génétique du diabète et de l'obésité, métabolomique, IA/ML-DL
- Données sensibles
- Accès multi-profils : chercheurs, ingénieurs, cliniciens, support/soutien
- Environnement ISO 15189 → forte exigence de traçabilité
- Volume de données massif → logs volumineux
- **Contraintes réglementaires (CNRS/INSERM/université)**

Une petite équipe pour une grande infra

- 2 personnes pour tout gérer
- Support utilisateurs
- Réseau / VLAN / firewall (en collaboration avec l'université)
- HPC (GPU + CPU)
- Stockage haut débit multi-Po
- Sécurité + qualité + projets transversaux
- **Objectif : faire plus avec peu**

Une infrastructure hétérogène et massive

- 7 Po de stockage dont 1,2 Po sur le PRA
- 150 serveurs (Windows/Linux/RHEL/Debian/VM/physiques)
- 120 postes (Windows/macOS/Linux)
- 12 GPU + 2000 CPU + 10 To RAM pour le HPC
- Virtualisation VMWare & Proxmox
- Multiplicité des sources de logs
- **Besoin d'une vision transversale**

Pourquoi la cybersécurité est critique

- Données santé → attractivité pour attaquants
- Multiples comptes et profils → complexité accrue
- Peu d'admins → pas de monitoring manuel possible
- Alertes fréquentes : brute-force, scans, vulnérabilités
- ISO 15189 : preuve de maîtrise du SI
- **SIEM* = renforcement de la posture sécurité**
  **Security Information and Event Management*

Pourquoi un SIEM : impossible de suivre manuellement

- Volume : plus de 4 millions de logs par jour
- Logs dispersés (firewall, serveurs, HPC, applications)
- Absence de corrélation → angles morts
- Humainement impossible à analyser
- SIEM = normalisation + filtrage + alerting
- Nécessité pour ISO 15189 (journalisation)

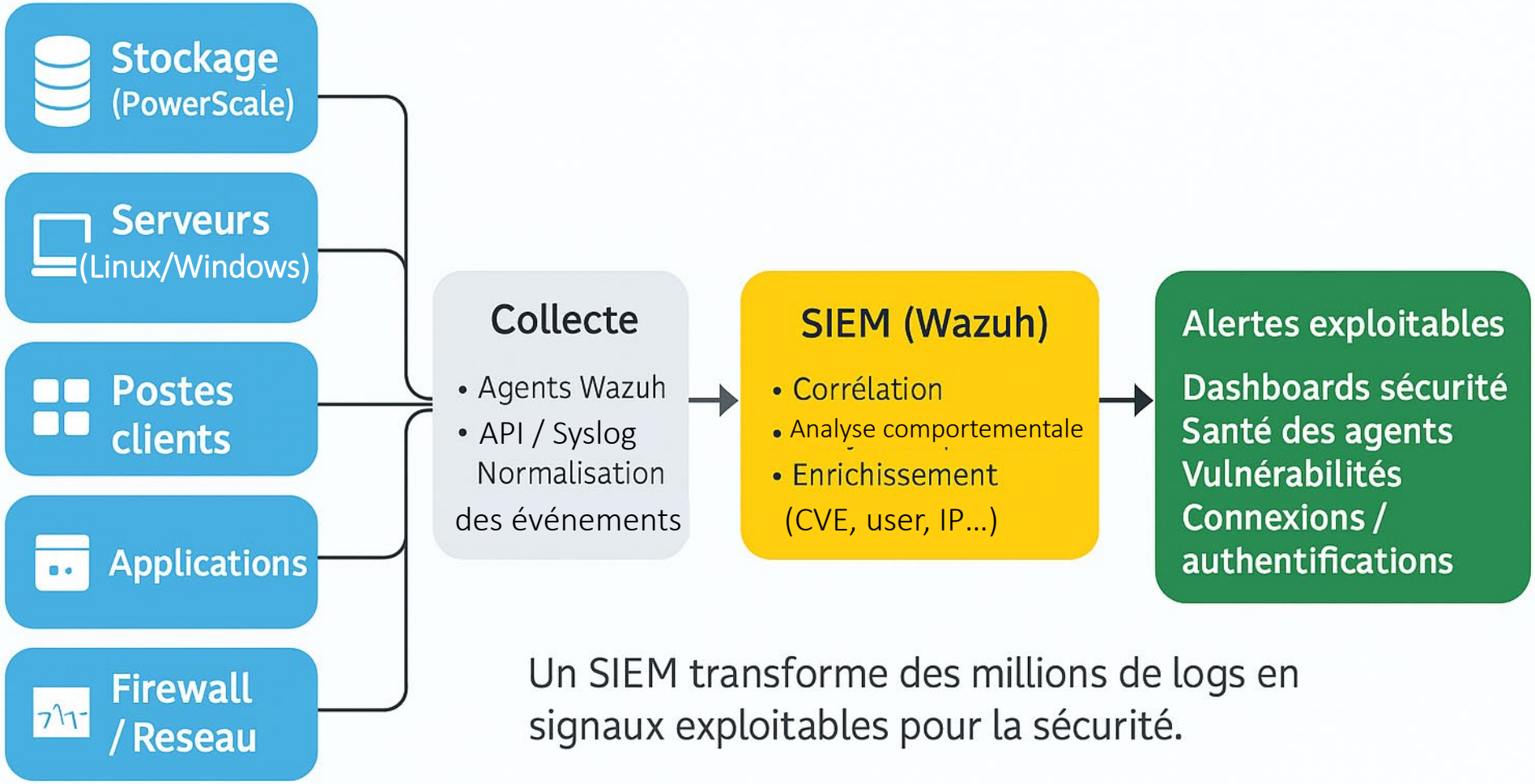
Pourquoi un SIEM *en plus* ?

- Un log seul ne raconte rien
- Corrélation = détection d'événements invisibles à l'œil nu
- Multi-OS → nécessité d'unification
- Détection comportementale → pas juste du "grep"
- Réduction du bruit via règles adaptées
- Vision chronologique des événements
- Exemple : 3 logs anodins → 1 incident réel

Qu'est ce qu'un SIEM fait concrètement ?

- Collecte des événements
- Décodage et normalisation
- Corrélation intelligente
- Détection d'anomalies
- Enrichissement (CVE, géolocalisation IP…)
- Alertes priorisées
- Dashboards décisionnels

• Collecter • Corréler • Analyser • Détecter • Alerter • Visualiser



Un SIEM transforme des millions de logs en signaux exploitables pour la sécurité.

Avant Wazuh : un patchwork d'outils

- Zabbix → supervision
- Grafana → métriques
- Graylog → certains logs
- Firewall → logs réseau
- HPC → logs propres
- Vision en silos = Aucun lien entre eux
- Analyse incidente = lente et manuelle

pas de corrélation, pas de sécurité centralisée

Après Wazuh : une vision unifiée et exploitable

- Centralisation des logs
- Corrélation automatique multi-OS (Linux / Windows / macOS)
- Alertes priorisées, moins de bruit que prévu après tuning
- Détection immédiate d'anomalies invisibles auparavant
- Vision "temps réel" + historique récent
- Gain énorme en réactivité lors d'incidents
- Première fois qu'on a une photo complète du SI

Pivot de sécurité = sécurité centralisée et corrélée

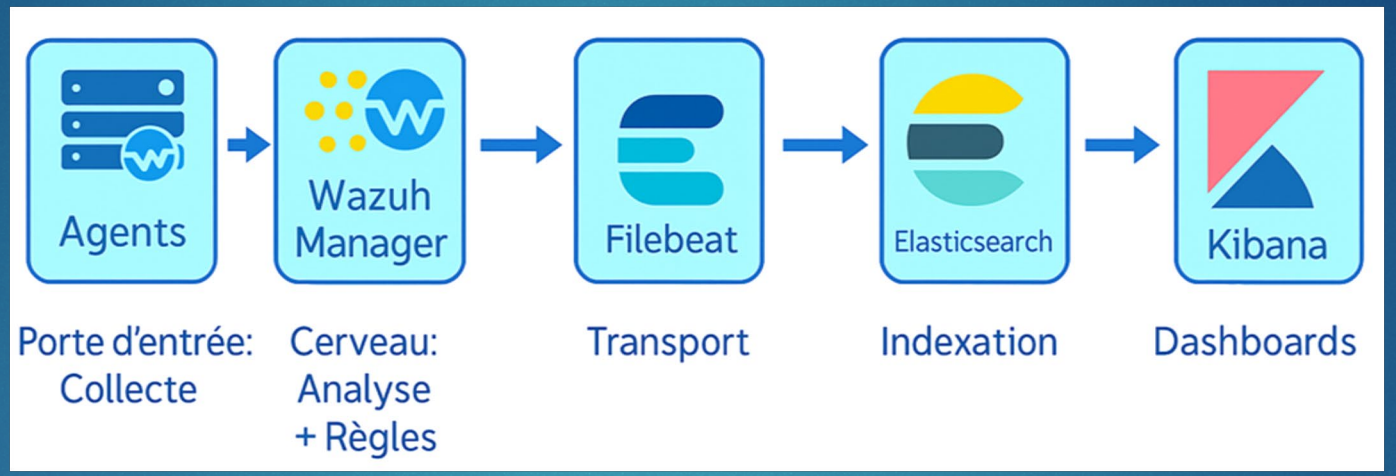
Pourquoi Wazuh ? Un choix pragmatique

- Open-source → coûts maîtrisés
- Multi-OS natif : Linux, Windows, macOS
- Modules intégrés : FIM, CVE, vulnérabilités
- Intégration transparente avec Elasticsearch/Kibana
- Très adapté aux petites équipes (simplicité infrastructure)
- Scalabilité progressive, sans cluster dès le début
- Communauté active + documentation correcte

Alternatives étudiées : pourquoi pas elles ?

- ELK seul → puissant mais pas orienté sécurité
- Graylog → simple mais corrélation limitée
- SIEM commerciaux → excellent → SOC, mais hors budget + trop lourds
- Pas adapté au contexte labo (petite équipe, forte autonomie)
- Coût humain et technique non soutenable pour l'unité
- Wazuh = cohérence + simplicité + couverture sécurité
- Le plus adapté à l'UMR8199

Architecture Wazuh (vue simple) : légère et efficace

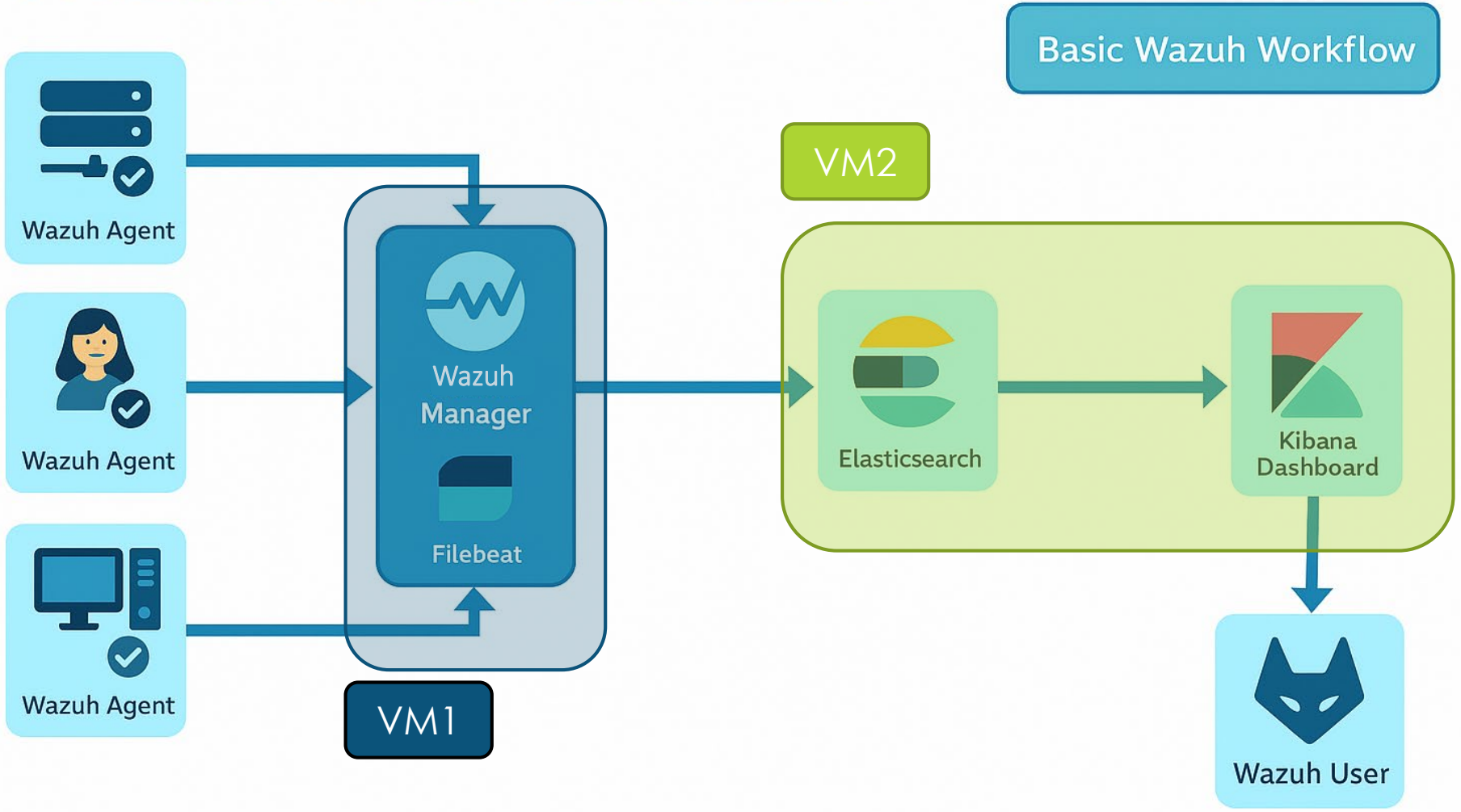


- Manager = cœur de la corrélation
- Filebeat transfère les événements vers Elastic
- Elastic = moteur de recherche et stockage
- Kibana = lecture humaine des événements
- Simple à déployer, simple à maintenir

Architecture détaillée : un pipeline robuste

- Filebeat transfère les événements vers Elastic
- Pipelines d'ingestion : normalisation, enrichissement
- Indexation optimisée pour la recherche
- Gestion fine des index (rotation, rétention)
- Architecture résiliente, même dans une VM
- Possibilité d'ajouter des nœuds Elastic plus tard
- Fiabilité assurée même en cas de charge

Wazuh et la stack ELK



https://github.com/mriazx/wazuh-setup

Pipeline agents → manager : la chaîne de traitement

- Collecte → décodage → corrélation → classification → alerte
- Filtrage automatique des faux positifs
- Ajout de métadonnées utiles (user, process, IP)
- Alignement chronologique des événements
- Consolidation multi-sources (SSH + sudo + FIM + CVE…)
- Visibilité sur comportements répétitifs
- Base essentielle pour les dashboards

Dimensionnement : un SIEM réaliste pour une infra de recherche

- 2 VMs recommandées pour séparer Manager et Elasticsearch
- VM Wazuh Manager : 4–8 vCPU • 8–16 Go RAM • 50–100 Go
- VM Elasticsearch/Kibana : 6–8 vCPU • 16–32 Go RAM • 300–500 Go
- Jusqu'à 10 millions d'évènements/jour sans degradation
- Capacité : ≈ 500 sources (HPC + stockage + serveurs)
- Index : 15–20 Go/jour
- Rétention 30 jours → ≈ 450–600 Go
- Snapshots réguliers sur stockage externe (PRA)

Rétention recommandée (500 Go)

- Rétention recommandée : 30 jours
- Rétention minimum acceptable : 15 jours
- Rotation : quotidienne
- Archivage prolongé : snapshots externes si besoin > 30 j
- Elasticsearch gourmand : au-delà, saturation & perte de performance
- 30 jours = zone de confort pour le labo

Rétention dans Wazuh ?

- La rétention se configure dans Elasticsearch / Wazuh-Indexer
- Utilisation des politiques ILM (Index Lifecycle Management)
- min_age définit la durée de conservation (ex. : 30 jours)
- Rotation automatique des index quotidienne
- Suppression ou archivage automatique selon la policy
- Wazuh applique la politique ILM aux data streams (alertes)

Architecture réseau : isoler sans compliquer

- VLAN dédié au SIEM
- Ports agents strictement filtrés
- Accès Kibana restreint à IP internes
- Accès SSH limité
- Pas d'exposition externe (zero-public)
- Monitoring du trafic agent ↔ manager
- Segmentation = sécurité + tranquillité

Sécurisation du Wazuh Manager

- SSH restreint (IP whitelisting)
- Certificats obligatoires pour les agents
- Mise à jour Wazuh/Elastic régulière
- Rotation automatique des index
- Snapshots Elastic (PRA)
- Journaux d'administration conservés
- Hardened OS (Lynis)

Déploiement massifs des agents : zéro manuel

- Linux : apt / yum automatisé
- Windows : MSI silencieux
- macOS : pkg standardisé
- Intégration automatique au manager via clé
- Déploiement par vagues (groupe par groupe)
- Vérification automatique via dashboards
- Objectif à atteindre : 0 installation manuelle

Pourquoi Ansible : industrialiser ou s'épuiser

- Parc hétérogène → impossible de gérer manuellement
- Idempotence : même résultat partout, sans erreur humaine
- Déploiement d'agents en masse
- Templates centralisés = cohérence
- Réduction drastique du temps de maintenance
- Compatible Linux / Windows / macOS
- Base indispensable pour un SIEM *propre*

Organisation des groupes : une structure claire et durable

- umr8199 = règles communes du labo
- Règles par OS : linux, windows, macos
- Groupes = héritage propre et lisible
- Chaque agent appartient à plusieurs groupes
- Facile à auditer → ISO
- Ajouts faciles pour futurs groupes (DMZ, MinIO…)
- Évite les configs “fourre-tout”

Structure des fichiers : un agent.conf par groupe

- ossec.conf = configuration globale
- agent.conf = configuration par groupe
- 1 fichier = 1 rôle = 1 responsabilité
- Lecture claire lors du debug
- Évite les conflits entre règles
- Templates Jinja2 = flexibilité
- Maintenance facilitée même en rotation d'équipe

Exemple de playbook : simple, efficace, reproductible

- Installation Wazuh agent
- Dépôt sécurisé (clé GPG)
- Copie des templates de config
- Redémarrage propre du service
- Vérification de l'enregistrement
- Journalisation de l'exécution
- Compatible "dry-run" pour tests

Workflow complet : de l'inventaire au dashboard

- Inventaire Ansible → groupes → templates
- Génération dynamique des confs
- Déploiement automatique des agents
- Connexion sécurisée au manager
- Vérification dans Kibana (dashboard "Santé agents")
- Pipeline complet industrialisé
- **Reproductible à l'infini (nouvelle machine = 1 commande)**

Les galères rencontrées (et assumées)

- Clé GPG Wazuh obsolète (404)
- Dépôts Debian cassés
- Variable oubliée dans un template (wazuh_manager_ip)
- Documentation parfois en retard
- Versions Elastic / Wazuh désynchronisées
- Règles par défaut trop bavardes
- **Charge mentale au début : “où commencer ?”**

Les solutions mises en place : stabiliser le système

- Mise à jour de la clé GPG + fallback local
- Ajout des variables manquantes dans les templates
- Documentation interne systématique
- Monitoring des URLs Wazuh/Elastic
- Tests automatiques d'intégrité (Ansible --check)
- Règles de tuning par groupe
- **Processus de MCO défini (mensuel)**

Organisation des règles : la clé d'un SIEM efficace

- Groupe commun : base sécurité + FIM + vulnérabilités
- Linux : SSH, sudo, fichiers sensibles
- Windows : RDP, AD, installation d'applis
- macOS : logs adaptés (restrictions Apple)
- Stratégie "peu de règles, bien réglées"
- Priorités définies par criticité
- **Réduction drastique des faux positifs**

Exemple Linux : détections vraiment utiles

- Brute-force SSH
- Sudo suspects (exécution inhabituelle)
- Modifications /etc/passwd ou /etc/shadow
- Suppression de logs
- Fichier sensible modifié hors maintenance
- Processus anormal (ex : script dans /tmp)
- **Indicateurs combinés = comportement suspect**

Exemple Windows : comprendre ce qui se passe vraiment

- Tentatives RDP depuis IP inattendues
- Échecs AD répétés
- Installation / désinstallation d'applications
- Création de comptes admin locaux
- Changements de stratégies Windows
- Processus inconnus au démarrage
- **Rassemblement d'événements dispersés**

macOS : un terrain particulier

- Wazuh fonctionne sur macOS mais avec des limitations
- Unified Logging → format complexe, difficile à décoder
- Restrictions Apple : SIP / TCC = accès limité à certains événements
- Peu ou pas d'accès aux logs système "sensibles"
- FIM fonctionne bien (fichiers sensibles, binaires modifiés)
- Auth locale / sudo remontent correctement
- **Nécessite un tuning spécifique → pas du "copier-coller" des règles Linux**

Incident réel #1 : brute-force SSH détecté

- 150 tentatives SSH/minute
- Comptes ciblés : admin, test, db…
- Détection via règle Wazuh 5710
- Corrélation entre logs d'échec + pattern d'attaque
- Alertes remontées en niveau élevé
- Bannissement automatique
- **Incident détecté ≠ machine compromise → énorme valeur**

Incident réel #2 : modification anomalie apt / Debian

- Alerte FIM sur /etc/apt/sources.list
- Changement soudain → CDN Google ajouté
- Checksum différent de d'habitude
- Wazuh corrèle FIM + logs APT → suspicion
- Après enquête : changement légitime (miroir temporaire)
- Mais aurait pu être le signe d'un dépôt compromis
- **Exemple parfait d'alerte "à investiguer", pas "à ignorer"**

Autres incidents détectés dans la vraie vie

- Sudo suspects (heures inhabituelles / utilisateurs inattendus)
- Suppression volontaire de fichiers logs (tentative de dissimulation)
- Modifications de fichiers système hors procédure
- Tentatives d'accès RDP répétées
- Scans réseau internes (recherche de ports ouverts)
- Processus suspects lancés dans /tmp
- **Détections corrélées = incidents identifiés plus tôt**

Pipeline : du log brut à l'alerte utile

- Log brut collecté par l'agent
- Décodage par règles Wazuh
- Corrélation multi-sources (SSH + sudo + FIM…)
- Enrichissement (CVE, IP lookup, user…)
- Classification : faible / moyen / élevé
- Stockage dans Elasticsearch
- Visualisation immédiate dans Kibana

Pourquoi les dashboards sont essentiels ?

- Vérification rapide de la santé du SI
- Tendances d'attaque (jour/semaine/mois)
- Priorisation des actions de sécurité
- Identification des machines silencieuses (potentiellement à risque)
- Suivi vulnérabilités → plan de patch
- Support des audits ISO
- Communication avec direction / responsables qualité

Dashboard « Santé des agents » : le baromètre quotidien

- Agents actifs / inactifs
- Date du dernier check-in
- Agents silencieux = machines potentiellement compromises
- Erreurs de communication → à investiguer rapidement
- Très utile lors des coupures électriques / incidents réseau
- Support essentiel pour ISO : machines réellement monitorées
- Utilisé chaque matin (revue rapide)

Dashboard Vulnérabilités : CVE / criticité / priorisation

- Basé sur NVD + CVSS
- Classement par criticité (critique / élevé / moyen)
- Liste des machines vulnérables
- Top 10 des CVE à corriger
- Impact direct sur le plan de patch management
- Visualisation facile pour la direction
- Très utile pour arbitrer les priorités

Dashboard SSH / RDP / Auth

- Brute-force SSH → corrélation automatique
- Tentatives RDP suspectes
- Échecs AD répétés = potentielle compromission
- IP suspectes / localisation inattendue
- Détection anomalies horaires (activité la nuit)
- Vue claire des comportements d'authentification
- Sécurité renforcée sur accès sensibles

Dashboard FIM : les modifications critiques sous surveillance

- Suivi des modifications /etc/passwd / /etc/shadow
- Détecte suppression/altération de logs
- Sur Windows : suivi des clés de registre critiques
- Sur macOS : modifications des fichiers sensibles autorisés
- Permet de détecter une compromission *avant* les dégâts
- Support indispensable pour ISO 15189 (traçabilité)
- Combine changement + heure + utilisateur → preuves auditables

Limite 1 : un SIEM est bavard (au début)

- Alertes par défaut beaucoup trop nombreuses
- Règles génériques → bruit inutile
- FIM très sensible → alerte sur chaque déplacement de fichier
- Corrélation encore imparfaite sans tuning
- Risque de “fatigue d’alertes” si mal calibré
- Obligation d’ajuster le périmètre des règles
- Nécessité de filtrer selon l’usage réel du labo

Limite 2 : maintenance & ressources (réaliste mais nécessaire)

- Elasticsearch = gourmand en RAM
- Index → rotation impérative
- Stockage → se remplit vite si pas de rétention
- Mises à jour Wazuh/Elastic fréquentes
- Sauvegardes d'index obligatoires
- Nécessite un MCO mensuel
- Rien d'insurmontable, mais non négligeable

Limite 3 : courbe d'apprentissage (mais pas la mer à boire)

- Règles XML → pas intuitives au début
- Décoders Wazuh → syntaxe spécifique
- Pipelines Elastic → concepts nouveaux
- Documentation parfois en retard sur les versions
- Nécessite de comprendre la logique interne
- Après 2–3 semaines → tout devient clair
- Le gain en expertise est réel et durable

Limite 4 : le cas macOS (le cas particulier)

- API et journaux limités
- Unified Logging difficile à exploiter
- Restrictions SIP/TCC → champ d'action réduit
- Peu de visibilité sur certains événements système
- FIM OK mais logs sécurité incomplets
- Tuning très spécifique nécessaire
- Résultat : utile mais partiel, pas un OS prioritaire

Leçons apprises : ce que je referais à l'identique

- **Commencer petit** : un périmètre réduit, bien maîtrisé
- **Automatiser tout de suite** : pas de config manuelle
- **Prioriser le tuning** : moins de règles → plus efficaces
- **Préparer les dashboards en amont**
- **Documenter chaque étape**
- **Adapter Wazuh au labo (pas l'inverse)**
- **Monter en compétence progressivement**

Bénéfice 1 : une vision centralisée et cohérente

- Une seule interface pour tout voir
- Corrélation multi-OS
- Détection d'incidents autrement invisibles
- Vue synthétique pour décider vite
- Réduction énorme des angles morts
- Meilleure gestion du risque global
- Photo en temps réel de l'état du SI

Bénéfice 2 : un atout fort pour l'ISO 15189

- Traçabilité systématique des événements
- Preuves auditables en un clic
- Historique exploitable pour les non-conformités
- Journalisation complète et normée
- Appui pour la maîtrise du risque SI
- Transparence lors des audits COFRAC
- Alignement avec les exigences qualité

Bénéfice 3 : montée en compétence (et en maturité)

- Compréhension des pipelines Elastic
- Meilleure lecture des logs
- Expertise CVE / vulnérabilités
- Automatisation avancée via Ansible
- Structuration du MCO
- Meilleure coopération interne (IT ↔ plateforme)
- Maturité du SI du labo clairement renforcée

Roadmap : ce qui arrive ensuite

- **Court terme :**
  - Tuning avancé
  - Nouveaux dashboards (DMZ)
  - Amélioration du FIM
- **Moyen terme :**
  - Intégration MinIO (logs S3)
  - Intégration Active Directory et W11
  - IDS réseau possible
- **Long terme :**
  - Cluster Elasticsearch pour résilience
  - Automatisation totale du pipeline de sécurité
  - Echanges / RETEX inter-labos

Conclusion : un SIEM réaliste, utile et adapté aux labos

- Wazuh = puissant + open-source
- Déploiement réaliste même en petite équipe
- Demande du tuning, mais résultats excellents
- Vision centralisée → meilleure sécurité
- Support ISO 15189
- Incidents détectés = bénéfice immédiat
- Un projet structurant pour le labo

Merci pour votre attention