Sécurité MQTT et Modbus : pentest des protocoles IoT/OT industriels

> Pentest MQTT et Modbus : découvrez les vulnérabilités des protocoles IoT/OT industriels et les défenses à déployer selon NIS 2 et la LPM 2024.

Les protocoles MQTT et Modbus constituent l'épine dorsale de millions d'installations industrielles à travers le monde : usines automatisées, réseaux électriques intelligents, systèmes de traitement des eaux, plateformes pétrolières. Conçus il y a plusieurs décennies dans des environnements fermés et isolés, ces protocoles ont été pensés pour la fiabilité et la performance, jamais pour la sécurité. Aujourd'hui, la convergence IT/OT expose ces infrastructures critiques à des attaquants de plus en plus sophistiqués. Selon le rapport Dragos Year in Review 2025, 68 % des incidents ICS signalés impliquaient des vecteurs initiaux liés à des protocoles industriels non sécurisés ou à des équipements directement exposés sur Internet. Les conséquences d'une compromission OT vont bien au-delà d'une fuite de données : arrêt de production, sabotage physique, risques humains. Comprendre comment un attaquant peut exploiter MQTT, Modbus, OPC UA ou DNP3 est désormais une compétence fondamentale pour tout professionnel de la cybersécurité industrielle. Cet article vous guide, étape par étape, dans la reconnaissance, l'exploitation et la sécurisation de ces protocoles critiques.

À RETENIR

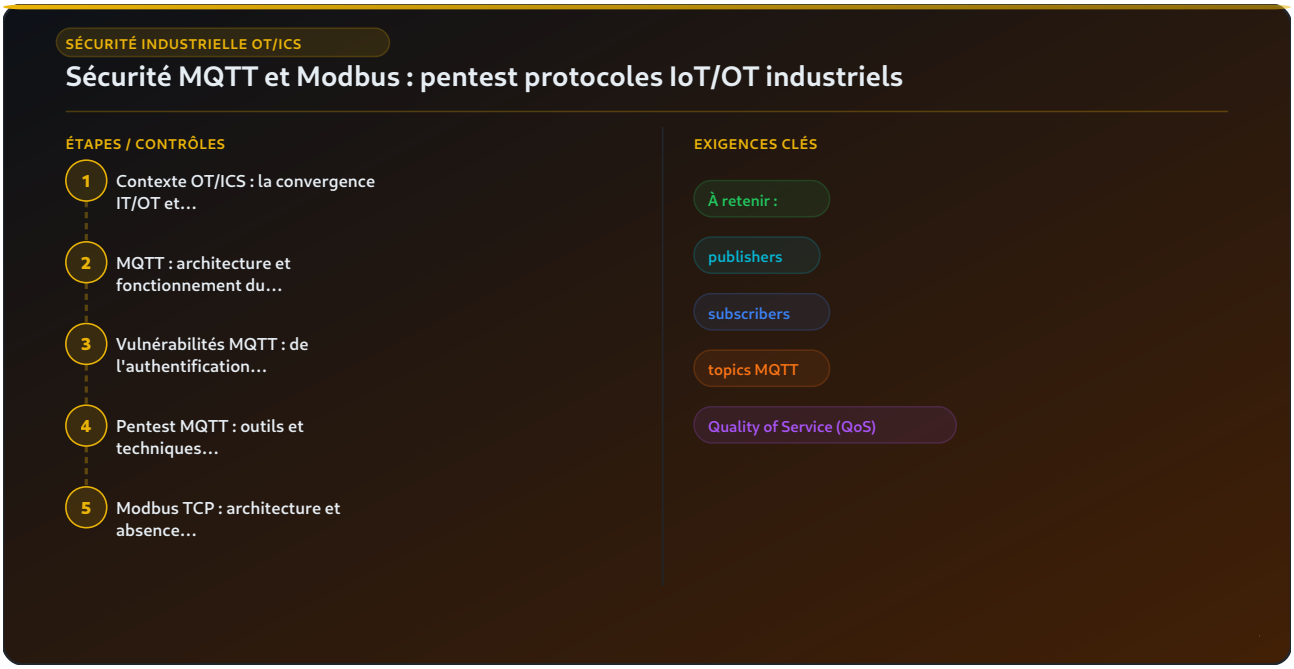
À retenir :

MQTT sans authentification permet à tout attaquant de s'abonner au topic # et d'intercepter l'intégralité du trafic IoT/OT d'un broker exposé.

Modbus TCP ne dispose d'aucun mécanisme d'authentification natif : lecture et écriture de registres arbitraires sont possibles sans credentials.

Des outils comme mqtt-pwn, pymodbus et les modules Metasploit ICS permettent d'automatiser l'exploitation de ces protocoles en quelques minutes.

> NIS 2, la directive CER et la LPM 2024 imposent aux opérateurs d'infrastructures critiques des obligations de sécurité OT désormais contraignantes.



Contexte OT/ICS : la convergence IT/OT et ses risques

Pendant des décennies, les environnements OT (Operational Technology) ont fonctionné en silo. Les automates programmables industriels (PLC), les systèmes SCADA et les DCS (Distributed Control Systems) évoluaient dans des réseaux propriétaires, isolés par des "air gaps" physiques. Cette isolation constituait en elle-même une forme de sécurité par obscurité : sans connectivité, le risque cybernétique restait théorique. Tout a changé avec la transformation numérique des industries.

La convergence IT/OT répond à des impératifs économiques légitimes : monitoring en temps réel de la production depuis des dashboards cloud, maintenance prédictive basée sur l'analyse de données IoT, intégration des systèmes ERP avec les lignes de fabrication. Les bénéfices sont réels, mais le prix à payer en termes de surface d'attaque est considérable. Un PLC qui échange des données avec un système MES (Manufacturing Execution System) lui-même connecté au

cloud expose potentiellement l'ensemble de la chaîne de contrôle à des menaces venues d'Internet.

Les statistiques sont alarmantes. Dragos signale dans son rapport 2025 une augmentation de 87 % des incidents OT/ICS par rapport à 2023. Les secteurs les plus touchés sont l'énergie (34 %), la fabrication (28 %) et les utilities (eau, gaz, 21 %). Les groupes APT spécialisés OT — VOLTZITE, KAMACITE, ELECTRUM — ciblent spécifiquement les protocoles industriels dans leurs intrusion sets. La menace n'est plus théorique : elle est documentée, répétée et en croissance.

Dans ce contexte, les protocoles MQTT et Modbus représentent des cibles de choix. Leur omniprésence, combinée à leur absence native de mécanismes de sécurité, en fait des vecteurs d'attaque privilégiés pour quiconque souhaite atteindre les couches OT d'une infrastructure industrielle.

MQTT : architecture et fonctionnement du protocole publish/subscribe

MQTT (Message Queuing Telemetry Transport) est un protocole de messagerie léger conçu en 1999 par IBM pour la supervision de pipelines pétroliers via des liaisons satellitaires à faible bande passante. Son modèle publish/subscribe s'est imposé comme le standard de facto des architectures IoT industrielles et grand public.

⚡ **Retour terrain**

Pour un groupe industriel agroalimentaire, l'audit OT a révélé que le réseau de supervision des lignes de production était connecté directement au réseau IT bureautique via un switch non managé — 'pour que les ingénieurs puissent accéder depuis leur PC'. La ségrégation via une DMZ industrielle avec des règles de flux explicites a été la première mesure. La

seconde : supprimer les 4 accès directs Internet depuis les automates programmables, configurés 'temporairement' il y a 5 ans.

L'architecture MQTT repose sur trois composants fondamentaux. Le **broker** est le serveur central qui reçoit tous les messages et les redistribue aux abonnés concernés. Les **publishers** sont les clients qui publient des données sur des topics précis. Les **subscribers** sont les clients qui s'abonnent à un ou plusieurs topics pour recevoir les messages correspondants. Cette architecture découplée permet une scalabilité horizontale et une résilience que le modèle point-à-point traditionnel ne peut offrir.

Les **topics MQTT** sont des chaînes de caractères hiérarchiques séparées par des slashes, similaires à des chemins de fichiers : `usine/ligne1/capteur_temperature`, `batiment/etage3/hvac/status`. Deux wildcards permettent des abonnements flexibles : le `+` représente un niveau unique (`usine/+/temperature` matche tous les capteurs de température de toutes les lignes), et le `#` représente tous les niveaux descendants (`usine/#` ou simplement `#` pour recevoir absolument tous les messages du broker).

Les **Quality of Service (QoS)** définissent la garantie de livraison : QoS 0 (at most once, fire-and-forget), QoS 1 (at least once, accusé de réception), QoS 2 (exactly once, handshake en 4 étapes). La **rétention de messages** (retained messages) permet au broker de stocker le dernier message publié sur un topic, disponible immédiatement pour tout nouvel abonné — une fonctionnalité qui peut devenir un vecteur d'information disclosure si des données sensibles sont retenues.

Les brokers MQTT les plus répandus sont Eclipse Mosquitto (open source, très déployé), HiveMQ (enterprise), EMQX et AWS IoT Core pour les déploiements cloud. Le port par défaut est 1883 (texte clair) et 8883 (TLS).

Vulnérabilités MQTT : de l'authentification absente au broker ouvert

MQTT version 3.1.1 — la version la plus déployée — ne requiert pas d'authentification. La connexion au broker peut s'effectuer sans username ni password. Même lorsqu'une authentification est configurée, de nombreuses implémentations acceptent des credentials vides ou utilisent des combinaisons par défaut (admin/admin, mqtt/mqtt). Cette situation reflète une réalité industrielle : les intégrateurs OT configurent l'authentification comme une option, pas comme un prérequis.

Les **ACL (Access Control Lists)** du broker définissent quels clients peuvent publier ou s'abonner à quels topics. En l'absence d'ACL ou avec des ACL mal configurées, n'importe quel client peut s'abonner au topic `#` et recevoir l'intégralité du trafic transitant par le broker : données de capteurs, états des actionneurs, commandes de contrôle. Dans un contexte industriel, cela équivaut à écouter toutes les communications d'une usine.

L'absence de **TLS** sur les déploiements MQTT est malheureusement courante. Les contraintes de ressources des équipements IoT (processeurs ARM basse consommation, mémoire limitée) sont souvent invoquées pour justifier l'absence de chiffrement. En pratique, des captures réseau passives permettent de récupérer l'intégralité du trafic MQTT en clair, y compris les credentials lors des phases de connexion CONNECT.

La **rétention de messages sensibles** constitue un vecteur d'information disclosure spécifique à MQTT. Des messages retained contenant des états critiques (position d'une vanne, niveau d'un réservoir, paramètres de configuration) restent stockés indéfiniment sur le broker, accessibles à quiconque s'y connecte ultérieurement.

Pentest MQTT : outils et techniques d'exploitation

La reconnaissance d'un broker MQTT exposé commence par Shodan. Le dork `port:1883 MQTT` retourne régulièrement des dizaines de milliers de brokers accessibles publiquement, dont une proportion significative ne requiert aucune authentification. La commande suivante permet de vérifier rapidement si un broker est accessible :

```
 # Test de connexion anonyme à un broker MQTT
mosquitto_sub -h  -p 1883 -t '#' -v --username '' --password ''

# Découverte de topics avec MQTT Explorer (GUI)
# ou avec mqtt-pwn pour l'automatisation
pip3 install mqtt-pwn
python3 -m mqtt_pwn
```

L'outil **MQTT Explorer** offre une interface graphique permettant de visualiser en temps réel l'arborescence des topics et les messages publiés. Pour un usage en pentest automatisé, **mqtt-pwn** propose des fonctionnalités d'énumération, de bruteforce de credentials et d'injection de messages. Le client CLI `mosquitto_pub` et `mosquitto_sub` d'Eclipse Mosquitto sont suffisants pour des tests manuels ciblés.

L'attaque la plus simple et la plus impactante consiste à s'abonner au topic wildcard `#` après une connexion anonyme réussie. En quelques secondes, le penester récupère l'ensemble du trafic IoT/OT transitant par le broker. Dans les cas documentés lors de missions terrain, ce trafic inclut régulièrement des états de capteurs industriels, des commandes d'actionneurs, des tokens d'authentification publiés par erreur, et parfois des credentials en clair pour des systèmes tiers.

```
 # Écoute de tous les topics (attaque passive)
mosquitto_sub -h  -t '#' -v

# Publication d'un message sur un topic de contrôle (attaque active)
mosquitto_pub -h  -t 'usine/ligne1/commande' -m 'STOP'

# Injection de retained message malveillant
mosquitto_pub -h  -t 'config/seuil_alerte' -m '9999' --retain
```

L'injection de commandes via MQTT représente l'escalade la plus critique : si le broker reçoit des commandes sans validation de source, un attaquant peut déclencher des actions physiques sur des équipements industriels. Des actionneurs, pompes ou systèmes de contrôle qui écoutent des topics MQTT sans vérification de l'authenticité de l'émetteur sont directement manipulables.

Modbus TCP : architecture et absence d'authentification native

Modbus est l'un des protocoles de communication industriels les plus anciens et les plus répandus. Créé en 1979 par Modicon pour ses automates programmables, il reste aujourd'hui déployé dans d'innombrables installations industrielles, de l'automatisation de bâtiments aux centrales électriques. Sa simplicité est à la fois sa force et sa faiblesse principale.

L'architecture Modbus repose sur un modèle maître/esclave (master/slave). Le maître (typiquement un SCADA ou un HMI) envoie des requêtes aux esclaves (PLC, capteurs, actionneurs). Chaque esclave est identifié par une adresse unitaire (Unit ID, 1-247). Les données sont organisées en quatre types de registres : les **Coils** (bobines, sorties digitales read/write, adresses 0x), les **Discrete Inputs** (entrées digitales read-only, adresses 1x), les **Input Registers** (registres 16 bits read-only, adresses 3x) et les **Holding Registers** (registres 16 bits read/write, adresses 4x).

Les **Function Codes** définissent l'opération demandée : FC01 (Read Coils), FC02 (Read Discrete Inputs), FC03 (Read Holding Registers), FC04 (Read Input Registers), FC05 (Write Single Coil), FC06 (Write Single Register), FC15 (Write Multiple Coils), FC16 (Write Multiple Registers). Sur Modbus TCP (port 502), ces fonctions sont encapsulées dans des trames TCP/IP sans aucun mécanisme d'authentification ni de chiffrement.

La conséquence directe est qu'**n'importe quel équipement sur le même réseau peut envoyer des commandes Modbus** valides à n'importe quel esclave accessible. Lire des registres de process, écrire des valeurs de consigne, activer ou désactiver des sorties : toutes ces opérations sont disponibles sans credentials, par simple construction de trames valides.

Pentest Modbus : lecture et écriture de registres arbitraires

Le pentest d'une infrastructure Modbus commence par l'énumération des esclaves accessibles sur le réseau. Le scanner `nmap` avec les scripts NSE Modbus, ou des outils spécialisés comme `ModbusPal`, permettent d'identifier les équipements répondant sur le port 502.

```
 # Scan Modbus avec nmap
nmap -p 502 --script modbus-discover

# Lecture de registres avec pymodbus
pip3 install pymodbus
python3 << 'EOF'
from pymodbus.client import ModbusTcpClient

client = ModbusTcpClient('', port=502)
client.connect()

# Lecture des 10 premiers holding registers (FC03)
result = client.read_holding_registers(0, 10, unit=1)
print("Holding Registers:", result.registers)

# Lecture des coils (sorties digitales)
result = client.read_coils(0, 8, unit=1)
print("Coils:", result.bits)

# Écriture d'un coil (activation sortie)
client.write_coil(0, True, unit=1)
client.close()
EOF
```

Les modules Metasploit dédiés à Modbus permettent d'automatiser la reconnaissance et l'exploitation. Le module `auxiliary/scanner/scada/modbusclient` offre un accès complet en lecture/écriture. Des outils comme **ModbusPal** fournissent une interface graphique pour simuler un maître Modbus et explorer les esclaves cibles.

L'impact d'une attaque Modbus réussie est directement physique. L'écriture d'une valeur incorrecte dans un holding register contenant un seuil de température peut désactiver des protections industrielles. La modification de coils peut activer ou désactiver des actionneurs électromécaniques. Dans des installations critiques (centrales, usines chimiques), ces actions peuvent provoquer des dommages matériels significatifs ou mettre en danger la sécurité humaine.

OPC UA : la réponse industrielle à la sécurité des protocoles

Face aux limitations sécuritaires de Modbus et des protocoles legacy, l'industrie a développé OPC UA (OPC Unified Architecture), standardisé par la OPC Foundation. OPC UA représente une évolution majeure : il intègre nativement des mécanismes de sécurité que ses prédécesseurs ignoraient complètement.

OPC UA propose trois **Security Modes** : *None* (pas de sécurité, équivalent Modbus), *Sign* (signatures numériques des messages, intégrité garantie mais pas de confidentialité), et *SignAndEncrypt* (signature + chiffrement asymétrique, niveau recommandé en production). Les certificats X.509 gèrent l'authentification mutuelle entre clients et serveurs OPC UA.

Malgré ces mécanismes, OPC UA n'est pas exempt de vulnérabilités. Des recherches publiées entre 2022 et 2025 ont documenté des failles dans des implémentations spécifiques : déni de service par fuzzing du parser, contournement de la validation de certificats dans certaines stacks open source, et exploitation de la fonctionnalité de découverte automatique pour énumérer des serveurs non exposés publiquement. Le mode *None* reste malheureusement utilisé dans de nombreux déploiements pour des raisons de compatibilité avec des équipements anciens.

DNP3 et IEC 104 : risques des protocoles SCADA

DNP3 (Distributed Network Protocol 3) est le protocole dominant dans les réseaux électriques nord-américains et australiens. Conçu pour les communications entre centres de contrôle SCADA et sous-stations, il souffre comme Modbus d'une absence historique d'authentification. La version Secure Authentication v5 (SAv5) a été développée pour corriger cela, mais son déploiement reste limité en raison des coûts de mise à niveau des équipements legacy.

IEC 60870-5-104 (IEC 104) est l'équivalent européen largement utilisé dans les réseaux électriques et gaziers. Il transporte le protocole IEC 101 sur TCP/IP (port 2404) sans sécurité native. Des outils comme `scapy` avec ses couches IEC 104, ou `61850-lib`, permettent de forger des trames malveillantes capables de déclencher des commandes dans des sous-stations électriques accessibles.

Des incidents réels illustrent la criticité de ces protocoles : l'attaque sur le réseau électrique ukrainien en 2015 (BlackEnergy/Industroyer) a utilisé DNP3 et IEC 104 pour émettre des commandes d'ouverture de disjoncteurs depuis des workstations SCADA compromises. Industroyer/Crashoverride reste à ce jour l'un des malwares OT les plus sophistiqués analysés publiquement.

Architecture sécurisée OT : modèle Purdue et segmentation

Le **modèle Purdue** (Purdue Enterprise Reference Architecture) définit une hiérarchie à 5 niveaux pour les architectures industrielles. Du niveau 0 (équipements physiques : capteurs, actionneurs) au niveau 4 (réseau d'entreprise IT), chaque niveau doit être isolé du suivant. La DMZ IT/OT entre les niveaux 3 (operations management) et 4 est le point critique où la segmentation doit être la plus stricte.

En pratique, l'implémentation d'une architecture Purdue robuste nécessite des **firewalls industriels** capables d'inspecter les protocoles OT spécifiques : Claroty, Nozomi Networks, Dragos Platform. Ces solutions comprennent le contexte protocolaire de Modbus, DNP3 ou OPC UA et peuvent générer des alertes sur des anomalies comportementales (lecture de registres inhabituels, modifications de coils en dehors des plages autorisées).

Les **Data Diodes** (unidirectional gateways) représentent la solution la plus robuste pour certains flux : des équipements matériels qui garantissent physiquement l'unidirectionnalité du trafic. Waterfall Security et Owl Cyber Defense sont les leaders de ce marché. Ils sont particulièrement adaptés pour les transferts de données de supervision depuis le réseau OT vers le réseau IT, sans risque de rebond.

Défenses MQTT : authentification, TLS et monitoring

La sécurisation d'un broker MQTT passe en premier lieu par l'activation obligatoire de l'authentification. Sur Eclipse Mosquitto, la configuration minimale sécurisée comprend :

```
 # /etc/mosquitto/mosquitto.conf - Configuration sécurisée
listener 8883
cafile /etc/mosquitto/ca.crt
certfile /etc/mosquitto/server.crt
keyfile /etc/mosquitto/server.key
require_certificate true
use_identity_as_username true

allow_anonymous false
password_file /etc/mosquitto/passwd
acl_file /etc/mosquitto/acl

# /etc/mosquitto/acl - ACL restrictif par pattern
user capteur_ligne1
topic read usine/ligne1/#
topic write usine/ligne1/capteur/+

user superviseur
topic read usine/#
```

Le **TLS mutuel** (mTLS) garantit l'authentification des deux parties : le client vérifie l'identité du broker via son certificat, et le broker vérifie l'identité du client. Cette approche élimine les risques d'usurpation d'identité et de man-in-the-middle. Les certificats doivent être renouvelés via une PKI interne, avec des durées de vie courtes (90 jours recommandés) pour limiter l'impact d'une compromission.

Le monitoring des topics sensibles via des sondes passives permet de détecter des abonnements anormaux (connexion d'un client inconnu au topic #) ou des publications sur des topics de contrôle depuis des sources non autorisées. Des règles Sigma ou des scripts Zeek ICS peuvent automatiser cette détection dans un SIEM. Pour approfondir la détection, consultez notre guide sur les règles Sigma pour la détection d'intrusion.

Défenses Modbus : unidirectional gateway et whitelisting

La protection des équipements Modbus repose sur plusieurs couches complémentaires. En premier lieu, la **segmentation réseau** doit isoler les équipements Modbus dans des VLANs dédiés accessibles uniquement depuis les contrôleurs SCADA légitimes, via des règles de firewall strictes. Aucun équipement IT (laptops, serveurs) ne devrait avoir d'accès direct au réseau Modbus.

Le **whitelisting des function codes** au niveau du firewall industriel permet de bloquer les fonctions d'écriture (FC05, FC06, FC15, FC16) pour les sources non autorisées, tout en autorisant la lecture de supervision. Des solutions comme Claroty ou Nozomi peuvent inspecter les trames Modbus en couche 7 et appliquer ces politiques granulaires.

Les **unidirectional gateways** constituent la solution ultime pour les flux qui n'ont besoin que d'une direction : données de télémétrie montantes vers le SCADA. En rendant physiquement impossible tout trafic descendant vers le réseau OT, elles éliminent toute possibilité d'injection de commandes Modbus depuis le réseau IT.

Pour la surveillance, les scripts Zeek ICS dédiés à Modbus permettent d'extraire et loguer toutes les transactions : function codes utilisés, unités adressées, valeurs de registres lues et écrites. Ces logs alimentent les règles de détection d'anomalies dans le SIEM. Pour comprendre les particularités de la réponse à incident en contexte OT, voir notre article sur l'incident response OT et ICS.

La corrélation avec des techniques d'attaque cloud comme le SSRF permet d'identifier des mouvements latéraux entre environnements IT et OT, documentés dans notre analyse des attaques SSRF sur les metadata services cloud.

Réglementation : NIS 2, directive CER et LPM 2024

Le cadre réglementaire européen et français impose désormais des obligations de sécurité OT contraignantes. La **directive NIS 2**, transposée en droit français via la loi du 7 février 2024,

étend significativement le périmètre des entités soumises à des exigences de cybersécurité. Les secteurs "hautement critiques" (énergie, transport, eau, infrastructure numérique) et "critiques" (services postaux, gestion des déchets, chimie, alimentation) doivent désormais implémenter des mesures de sécurité couvrant explicitement les systèmes OT.

La **directive CER** (Critical Entities Resilience), parallèle à NIS 2, s'applique à la résilience physique et cyber des entités critiques. Elle exige des évaluations de risques régulières intégrant les scénarios de compromission OT et des plans de continuité tenant compte de la nature spécifique des systèmes industriels.

La **LPM 2024** (Loi de Programmation Militaire) maintient et renforce les obligations des OIV (Opérateurs d'Infrastructures Vitales) en France. Les 12 secteurs d'activité d'importance vitale (SAIV) sont soumis aux arrêtés sectoriels de l'ANSSI qui spécifient les règles de sécurité applicables aux systèmes industriels, incluant la segmentation réseau, la gestion des accès aux équipements OT et les capacités de détection d'intrusion.

L'ANSSI publie régulièrement des guides spécifiques aux environnements industriels (Guide de la cybersécurité des systèmes industriels, recommandations pour les SCADA) qui constituent la référence technique en France pour sécuriser les protocoles OT. Pour comprendre l'architecture SSRF dans un contexte de pentest avancé, voir notre article sur la détection avec les règles Sigma.

FAQ — Questions fréquentes

Comment détecter un broker MQTT non sécurisé sur mon réseau industriel ?

La détection passe par plusieurs approches complémentaires. Un scan réseau avec `nmap -p 1883,8883 --script mqtt-subscribe` sur les plages d'adresses OT identifie les brokers accessibles. Tentez ensuite une connexion anonyme avec `mosquitto_sub -h <ip> -t '#' -v` : si vous recevez des messages sans authentification, le broker est non sécurisé. Au niveau SIEM, des règles Zeek détectant des connexions MQTT sans champ username/password dans le paquet

CONNECT permettent une surveillance continue. Nozomi Networks et Claroty intègrent des détecteurs spécifiques aux brokers MQTT non authentifiés dans leurs solutions de visibilité OT.

Modbus peut-il être sécurisé sans remplacer l'équipement legacy ?

Oui, plusieurs approches permettent de sécuriser un environnement Modbus legacy sans remplacer les équipements. La segmentation réseau via VLANs et firewalls industriels inspection L7 est la première ligne de défense. Des proxies de sécurité Modbus (Modbus Security Gateway) peuvent être intercalés entre les maîtres et les esclaves pour valider les requêtes sans modifier les équipements existants. La Modbus Security extension (Modbus/TCP Security RFC) définit un encapsulage TLS optionnel supporté par certains équipements modernes tout en conservant la compatibilité protocolaire. Pour les flux critiques, les unidirectional gateways restent la solution la plus robuste.

Quels sont les premiers réflexes lors d'un pentest OT pour éviter des dommages physiques ?

Un pentest OT exige des précautions radicalement différentes d'un pentest IT. Règle fondamentale : **ne jamais écrire sur des registres Modbus ou publier sur des topics MQTT de contrôle en production sans autorisation explicite et surveillance physique simultanée**. Les tests actifs d'exploitation doivent se faire sur des environnements de test physiquement isolés ou sur des maquettes. En phase de reconnaissance passive (lecture de registres, écoute MQTT), le risque est faible mais non nul (des lectures répétées peuvent perturber certains PLCs anciens). Toujours avoir un ingénieur process présent lors des phases actives. Documenter chaque action avec horodatage pour permettre une corrélation en cas d'incident. Définir des règles d'engagement précises avec le client incluant les plages horaires autorisées et les équipements hors périmètre.

Sources

[ANSSI CERT-FR — Publications et alertes]

MITRE ATT&CK — Framework des techniques d'attaque

CISA — Advisories de cybersécurité