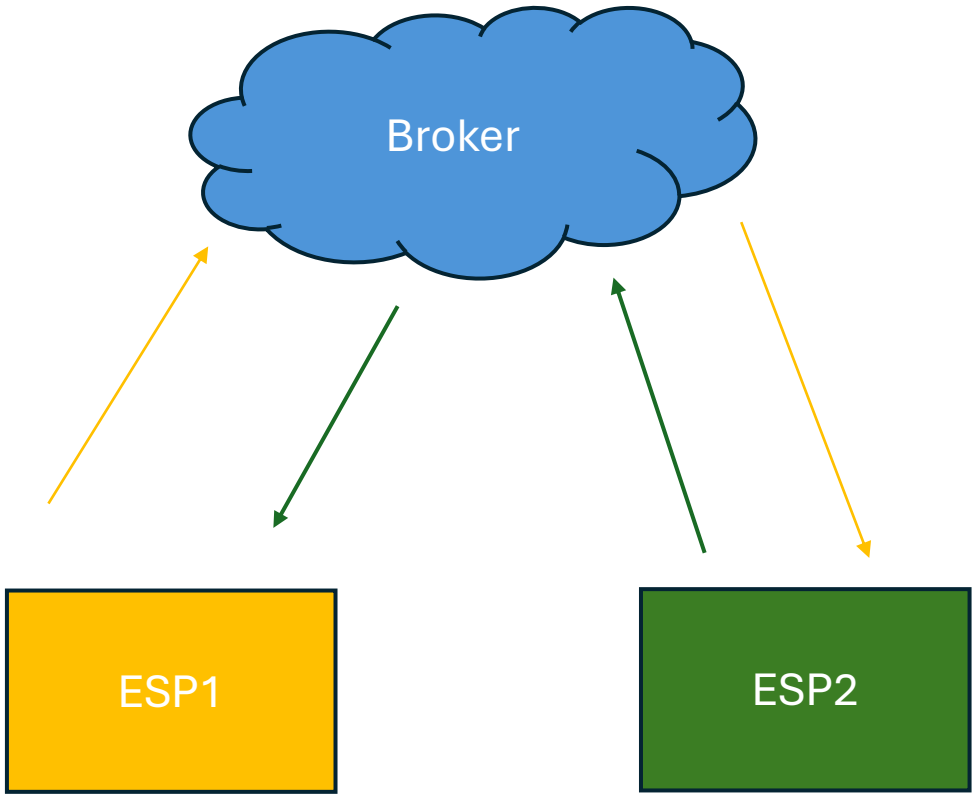
Exploring Vulnerabilities: A Penetration Testing Approach Using MQTT and ESP Devices

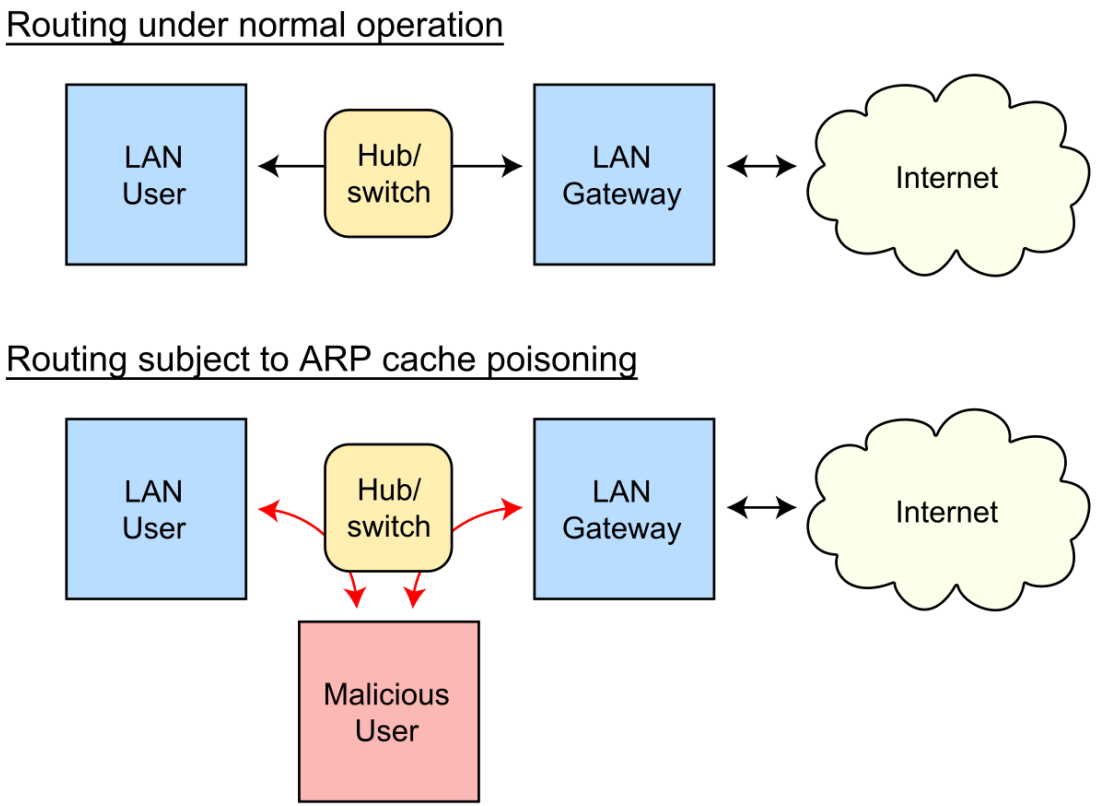
Grasso Emanuele:
0001141478

Rinaldi Simone:
0001140193



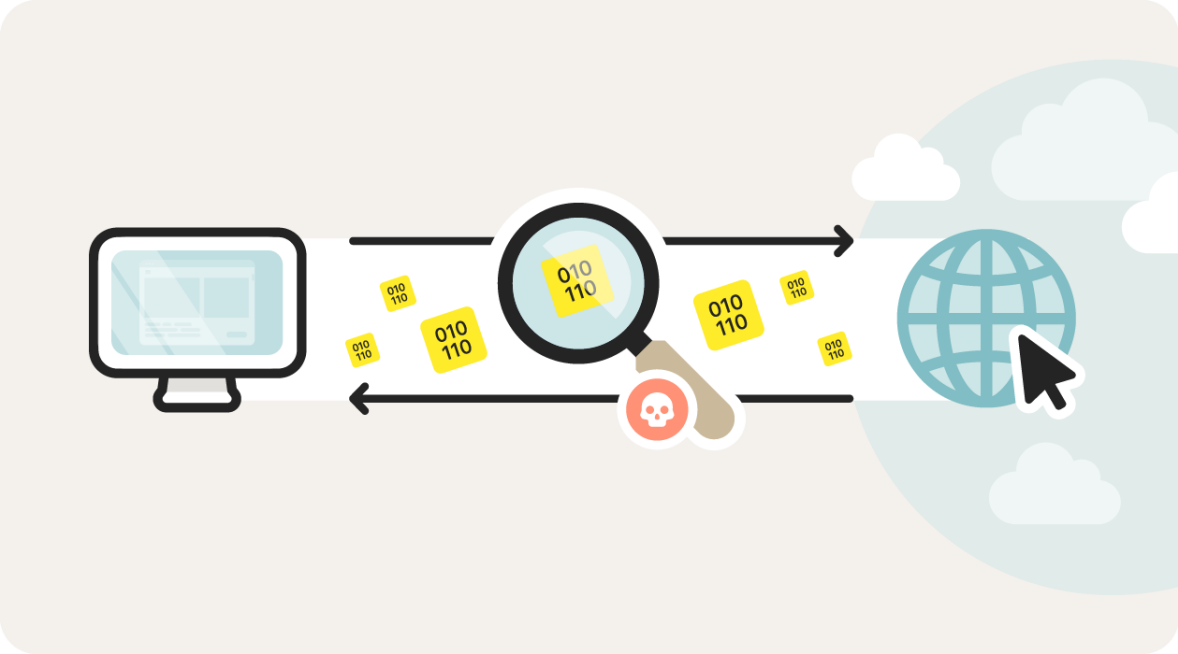
Goal:
Testing some cyber security attacks(most MitM) against some devices of a network communicating through MQTT protocol.

- 2 types of MQTT Implementation:
  - MQTT not encrypted
  - MQTT encrypted over TLS with authentication



ARP Spoofing

- Performs ARP spoofing to intercept MQTT messages
- Extracts and analyzes MQTT PUBLISH topics and payloads
- Uses Scapy for packet sniffing on TCP port 1883 (MQTT)
- Displays MQTT messages in both hex and string formats
- Restores ARP tables and disables IP forwarding on exit



SNIFFING

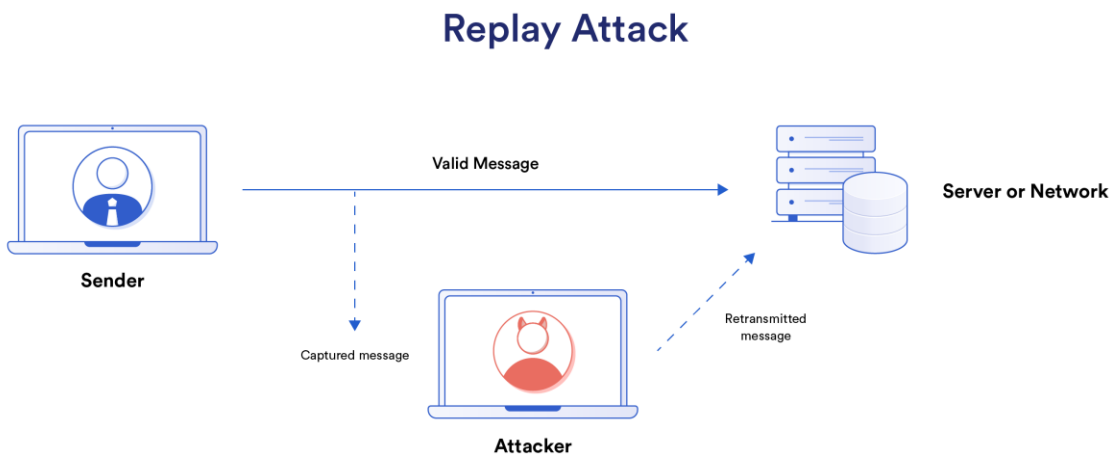
•Captures and analyzes traffic between two hosts

•Identifies MQTT packet types (CONNECT, PUBLISH, SUBSCRIBE, etc.)

•Extracts and logs MQTT PUBLISH topics and messages

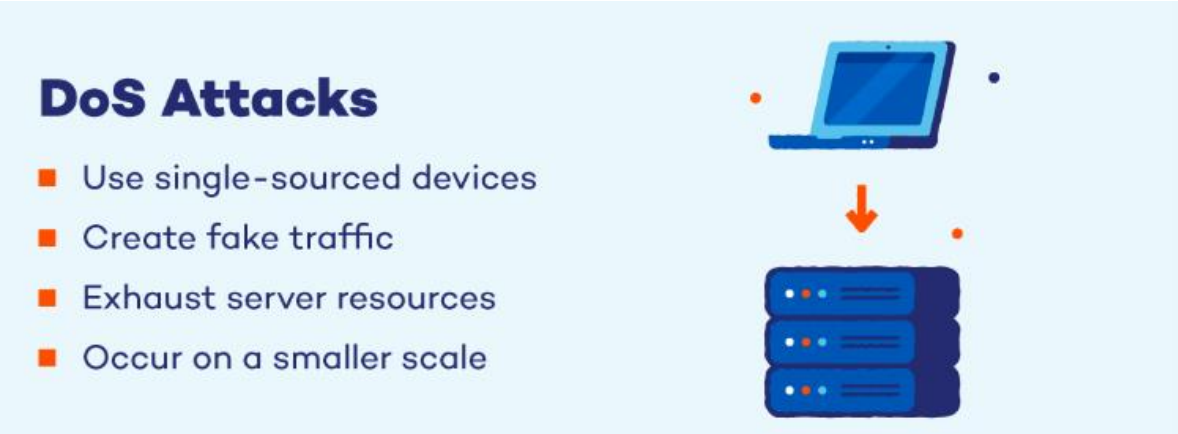
•Uses Scapy for packet sniffing on a specified network interface

•Provides real-time logging for security and troubleshooting



REPLAY

- Sniffs MQTT PUBLISH messages between a client and broker
- Extracts and prints MQTT topics and payloads
- Uses Scapy for packet capture and Paho-MQTT for message replay
- Implements MQTT remaining-length decoding for proper parsing
- Automatically republishes intercepted messages to the broker



DOS

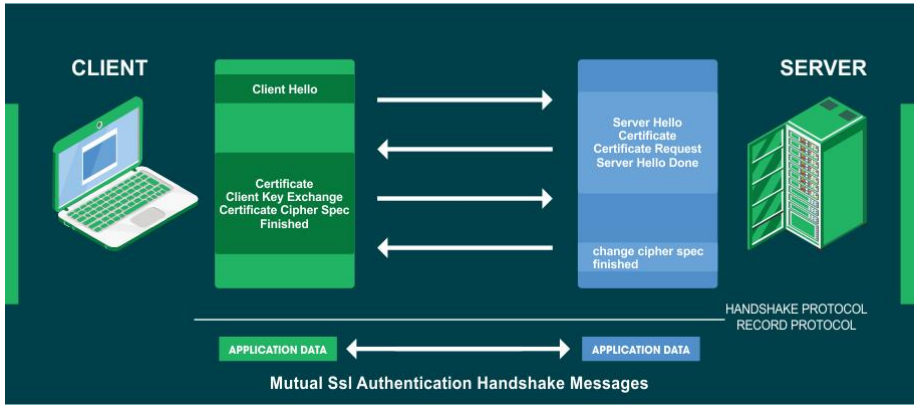
•**Setup & Initialization**: Configure the MQTT client and define message parameters.

•**Connect to Broker**: Establish a connection using the specified IP and port.

•**Message Transmission**: Publish a large number of messages in a loop.

•**Progress Tracking**: Display message count updates during transmission.

•**Disconnect & Error Handling**: Close the connection and manage potential errors.



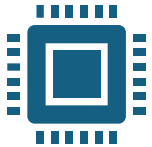
TLS

Transport Layer Security (**TLS**) is a cryptographic protocol designed to provide communications security over a computer network, such as the Internet. The protocol is widely used in applications such as email, instant messaging, and voice over IP, but its use in securing HTTPS remains the most publicly visible.

Results of the attacks

	MQTT UNSAFE	MQTT SAFE (TLS + Authentication)
ARP POISONING	**SUCCESS**	**FAIL**
SNIFFING	**SUCCESS**	**FAIL**
DOS	**SUCCESS**	**FAIL**
DEAUTHENTICATION	**SUCCESS**	**SUCCESS**
SESSION HIJACKING	**SUCCESS**	**FAIL**
REPLAY	**SUCCESS**	**FAIL**

Future Improvements



AI-Driven Anomaly Detection & Advanced Authentication

AI-based monitoring: Real-time analysis of MQTT traffic to detect/block suspicious patterns.

Hardware-backed authentication: Implement digital certificates or TPM for device identity verification.

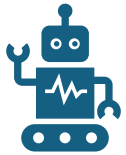
Enhanced session management: Enforce mutual TLS, persistent sessions, and automated token revocation.

2. Network-Level Security Improvements

WPA3 adoption: Mitigate deauthentication attacks via enterprise-grade Wi-Fi security.

Network segmentation & SDN: Dynamically control device access to limit lateral attacker movement.

Application-layer safeguards: Message integrity checks, signed tokens, and timestamp validation to block injection/replay.



3. Advanced Logging & Forensic Analysis

SIEM integration: Continuously monitor MQTT sessions for unauthorized access and live attacks.

AI-powered forensics: Correlate logs with anomaly detection for faster incident response.

4. Holistic Resilience Strategy

Combine **AI-driven detection**, **hardware-based authentication**, and **adaptive network policies** to counter evolving threats.

Prioritize proactive measures over reactive fixes for long-term IoT security