MANUAL TÉCNICO DE INGENIERÍA — MÓDULO 09

Ciberseguridad Industrial IoT, Criptografía X.509 (`TLS 1.3`), Listas de Control de Acceso (`ACL`), Defensa en Profundidad y Supervivencia 24/7

Curso: Internet de las Cosas (IoT) con Arduino y Raspberry Pi — Ingeniería Mecatrobótica

---

1. Modelo de Amenazas Ciberfísicas en Sistemas Mecatrónicos (STRIDE / IEC 62443)

A diferencia de un servidor web corporativo donde un incidente compromete únicamente datos digitales, en un sistema mecatrónico industrial (bajo el estándar **ISA/IEC 62443**) un ataque sobre la red de control tiene consecuencias **físicas y térmicas directas**:

- **Escucha Pasiva (*Sniffing*) en el Puerto* `1883` *:** En MQTT sin cifrar, el paquete `CONNECT` transmite el usuario y la contraseña en texto claro ASCII. Un atacante en la misma subred captura la trama en menos de un segundo.
- **Suplantación e Inyección de Comandos (*Spoofing & Tampering*): Sin cifrado ni listas* `ACL` *, cualquier dispositivo comprometido (por ejemplo, un sensor exterior económico) puede publicar en* `planta/zona1/arduino01/cmd/rpc` *para alterar el* Setpoint* térmico o desactivar el enfriamiento.
- **Ataque de Intermediario (*Man-in-the-Middle - *MitM): Si un nodo cliente no verifica criptográficamente el certificado* `X.509` *y la extensión* `SAN` *del broker, un atacante mediante* ARP Spoofing* puede hacerse pasar por la Raspberry Pi.

---

2. Infraestructura de Clave Pública (`PKI X.509`), `SAN` (RFC 9525) y `TLS 1.3` (`:8883`)

Para erradicar la escucha pasiva y la suplantación, establecemos una jerarquía criptográfica de tres niveles:

- **Autoridad Certificadora Raíz de Planta (`Root CA` , `ca.crt` / `ca.key` , RSA 4096 bits):** Su clave privada `ca.key` se mantiene fuera de línea; su certificado público `ca.crt` se distribuye a todos los clientes Python y microcontroladores como única raíz de confianza.
- **Certificado del Servidor Broker (`server.crt` / `server.key` , RSA 2048 bits):** Firmado por nuestra `Root CA` . Cumpliendo con el estándar **RFC 6125 / RFC 9525**, incluye obligatoriamente la extensión **`subjectAltName (SAN)`** con los nombres DNS (`edge-gateway-01.local` , `localhost`) y las direcciones IP (`192.168.1.100` , `127.0.0.1`).
- **Negociación* `TLS v1.3` *con Secreto Perfecto hacia Adelante (*Perfect Forward Secrecy - *PFS):** Mediante intercambio efímero de claves sobre curva elíptica (`ECDHE`), cada sesión genera una clave simétrica `AES-256-GCM` única. Aunque en el futuro alguien obtuviera una clave estática, resulta matemáticamente imposible descifrar capturas de tráfico pasadas.

📐 Plano 3: Cadena de Confianza PKI X.509 con SAN y Handshake MQTTS TLS 1.3



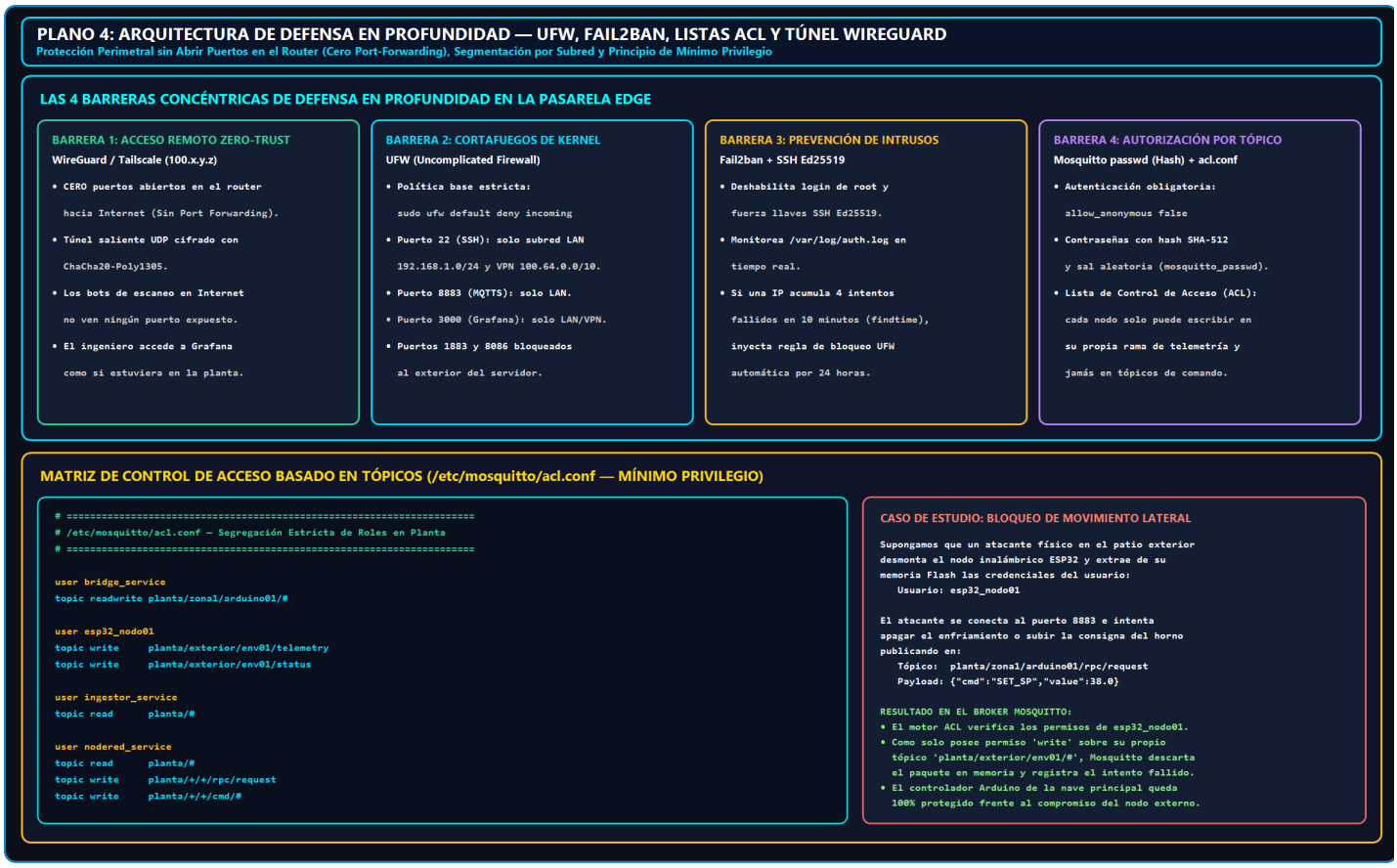
---

3. Defensa en Profundidad: `ACL`, Cortafuegos `UFW` y Red Mallada `WireGuard`

Aplicamos tres anillos concéntricos de protección en el servidor Edge:

- **Anillo de Aplicación (`Mosquitto ACL`):** Cada credencial está confinada estrictamente a sus tópicos mediante `read`, `write` o `readwrite`. Un nodo sensor jamás posee permiso `write` sobre tópicos `cmd/#`.
- **Anillo de Transporte (`UFW` + `Fail2ban`):** `ufw default deny incoming` cierra todos los puertos salvo `22/tcp`, `8883/tcp` y `3000/tcp` originados exclusivamente desde la subred LAN de planta o la interfaz VPN.
- **Anillo de Acceso Remoto Cero-Confianza (`WireGuard` / `Tailscale`):** Basado en criptografía moderna (`ChaCha20-Poly1305` + `Curve25519`), crea una red privada virtual punto a punto (`100.x.y.z`) atravesando NAT (`UDP Hole Punching`) **sin abrir ningún puerto público en el router de Internet**.

📐 Plano 4: Defensa en Profundidad, Segmentación UFW, ACL y Túnel WireGuard



---

4. Ingeniería de Supervivencia 24/7: `log2ram` y `Hardware Watchdog` (`/dev/watchdog`)

Para que la pasarela Edge opere de forma desatendida durante años sin degradación física ni bloqueos de sistema operativo:

- **Mitigación de Desgaste Flash NAND (`log2ram` + `journald`):** Montamos `/var/log` sobre un sistema de archivos temporal en memoria RAM (`tmpfs` de $128\ MB$) y limitamos `systemd-journald` a almacenamiento volátil (`RuntimeMaxUse=50M`), eliminando el 97.6% de las escrituras repetitivas sobre la tarjeta microSD.
- **Temporizador* Watchdog *de Hardware del SoC Broadcom (`bcm2835_wdt`): El procesador de la Raspberry Pi incorpora un contador regresivo en silicio conectado a la línea de reset físico de la placa. Al configurar `RuntimeWatchdogSec=15s` en `/etc/systemd/system.conf` , el proceso `PID 1 (systemd)` escribe un pulso de vida en `/dev/watchdog` cada $7.5\ s$. Si un* Kernel Panic* o bloqueo eléctrico congela Linux, el contador expira a los $15\ s$ y reinicia físicamente el servidor.

📐 Plano 5: Arquitectura de Supervivencia 24/7 con log2ram y Hardware Watchdog

PLANO 5: INGENIERÍA DE SUPERVIVENCIA 24/7 — MITIGACIÓN DE DESGASTE FLASH (log2ram) Y HARDWARE WATCHDOG

Modelado Matemático de Amplificación de Escritura en NAND Flash, Disco RAM tmpfs y Reinicio Autónomo Broadcom (/dev/watchdog a 15 s)

1. PROTECCIÓN DE CELDAS NAND FLASH EN TARJETA MICROSD (log2ram)

EL PROBLEMA: MUERTE PREMATURA DE TARJETAS MICROSD EN CAMPO

- Una celda NAND Flash comercial soporta ~1000 a 3000 ciclos de borrado (P/E).
- Los demonios Linux, Docker y Mosquitto escriben ~180 MB/día de bitácoras.
- Por el Factor de Amplificación de Escritura ($A_w \approx 12$ en bloques de 4 KB):

  $W_{anual} = 0.180\ GB/día \times 12 \times 365\ días = 788.4\ GB/año$ de desgaste físico.
- Consecuencia: Corrupción del sistema de archivos ext4 tras 6 a 8 meses.
- Meta de ingeniería: Reducir las escrituras físicas en más de un 97 %.

LA SOLUCIÓN: DISCO EN MEMORIA RAM (tmpfs) CON VOLCADO DIFERIDO

1. Servicio log2ram monta /var/log en memoria RAM volátil (tmpfs de 128 MB):
   - Todas las escrituras de Mosquitto, UFW, sshd y contenedores van a la RAM a velocidad de nanosegundos con CERO desgaste de silicio Flash.

2. Sincronización por lotes mediante rsync periódico o al apagar la placa:
   - En lugar de 50 000 escrituras aleatorias de 4 KB, se realiza un único volcado secuencial comprimido con algoritmo zstd.

3. Acotamiento estricto del diario de systemd (/etc/systemd/journald.conf):

```
[Journal]
Storage=volatile
RuntimeMaxUse=50M
SystemMaxUse=100M
```

4. Resultado medido en producción:
   - Desgaste anual baja de 788.4 GB/año a solo 18.5 GB/año (-97.6 %).
   - Vida útil estimada de la tarjeta microSD Industrial A2: > 12 años.

2. TEMPORIZADOR WATCHDOG DE HARDWARE EN SILICIO BROADCOM (15 s)

ARQUITECTURA DE AUTOCURACIÓN ANTE KERNEL PANIC O CONGELAMIENTO

- El SoC Broadcom (BCM2711 / BCM2712) incluye un registro contador autónomo en hardware (/dev/watchdog) gobernado por su propio oscilador de cristal.

- Paso 1: Habilitar el periférico en /boot/firmware/config.txt:

```
dtparam=watchdog=on
```

- Paso 2: Configurar el demonio central PID 1 en /etc/systemd/system.conf:

```
[Manager]
RuntimeWatchdogSec=15
RebootWatchdogSec=2min
```

- Funcionamiento en Tiempo Real:
  1. Cada 7.5 segundos ($T_{wdt} / 2$), systemd escribe un pulso en /dev/watchdog.
  2. Si una tormenta eléctrica o fuga de memoria congela el kernel Linux, systemd deja de refrescar el contador.
  3. Al llegar a 15.0 segundos exactos sin pulso, el hardware Broadcom fuerza un Hard Reset físico de la placa y recupera toda la estación.

3. ESTRATEGIA DE RESPALDO AUTOMATIZADO Y RECUPERACIÓN EN < 3 MINUTOS

¿Qué necesitamos respaldar cada noche (03:00 AM vía cron/systemd timer)?

1. Respaldo en caliente de InfluxDB v2:

```
docker exec iot-influxdb influx backup /var/lib/influxdb2/backup
```

2. Empaquetado cifrado de configuración declarativa y certificados:
   - /opt/mecatrobotica/capstone/docker-compose.yml y archivo .env
   - /opt/mecatrobotica/capstone/certs/ (CA y certificados X.509)
   - /opt/mecatrobotica/capstone/mosquitto/config/ (passwd + acl.conf)
   - Flujos de Node-RED (flows.json) y Dashboards de Grafana

3. Regla 3-2-1 de Recuperación ante Desastres:
   - Si la placa física o el almacenamiento sufren daño catastrófico, basta con tomar una Raspberry Pi nueva, copiar el respaldo y ejecutar:

```
sudo ./despliegue_automatizado_capstone.sh
```

   - Tiempo total de restauración (RTO): < 3 minutos.