# Capítulo 28 · Seguridad industrial: IEC 62443, PKI, TLS y ACL

## Conceptos clave

Verifica la cadena de confianza y la identidad del servidor. Restringe la lectura y escritura de cada cliente a sus propios temas.



# GUÍA PASO A PASO DE LABORATORIO — MÓDULO 09

## Ciberseguridad IoT Industrial, Cifrado MQTTS ( `TLS 1.3 :8883` ), PKI X.509 ( `OpenSSL SAN` ), Cortafuegos `UFW` , `WireGuard` y Supervivencia 24/7 ( `log2ram` + `Watchdog` )

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

**Módulo:** 09 — Seguridad Ciberfísica, Endurecimiento de Servidores Edge y Alta Disponibilidad

---

## 1. Objetivos de Ingeniería del Laboratorio

- Construir y cablear el **circuito de seguridad física de gabinete y supervisión de hardware** (tanto en protoboard estilo Tinkercad como en placa impresa PCB FR-4 soldada), integrando un microinterruptor de apertura de puerta (*Tamper Switch* `NC` ), un selector de llave física de mantenimiento (*Keylock*), aislamiento por optoacopladores `PC817` y el circuito de latido (*Heartbeat*) del procesador.
- Desplegar una **Infraestructura de Clave Pública ( `PKI X.509` )** privada de planta con `OpenSSL` ( `aprovisionar_pki_x509_openssl_san.sh` ), generando una Autoridad Certificadora Raíz ( `RSA 4096 bits` ) y certificados de servidor/cliente con extensión **Subject Alternative Name ( `SAN` - RFC 9525)**.
- Reconfigurar el broker **Eclipse Mosquitto 2.x** para operar exclusivamente con cifrado `TLS v1.3` **en el puerto** `8883` hacia la red local y aplicar una **Lista de Control de Acceso ( `ACL` )** bajo el Principio de Mínimo Privilegio.
- Endurecer el sistema operativo Linux de la Raspberry Pi mediante el cortafuegos de estado `UFW` , protección contra fuerza bruta `Fail2ban` y acceso remoto seguro punto a punto mediante `WireGuard` / `Tailscale` sin abrir puertos en el router.
- Garantizar la supervivencia continua `24/7` durante años configurando el disco en memoria RAM `log2ram` (para evitar el desgaste de celdas Flash NAND de la microSD) y activando el **Watchdog Timer de Hardware Broadcom** ( `/dev/watchdog` a $15\ \text{s}$).

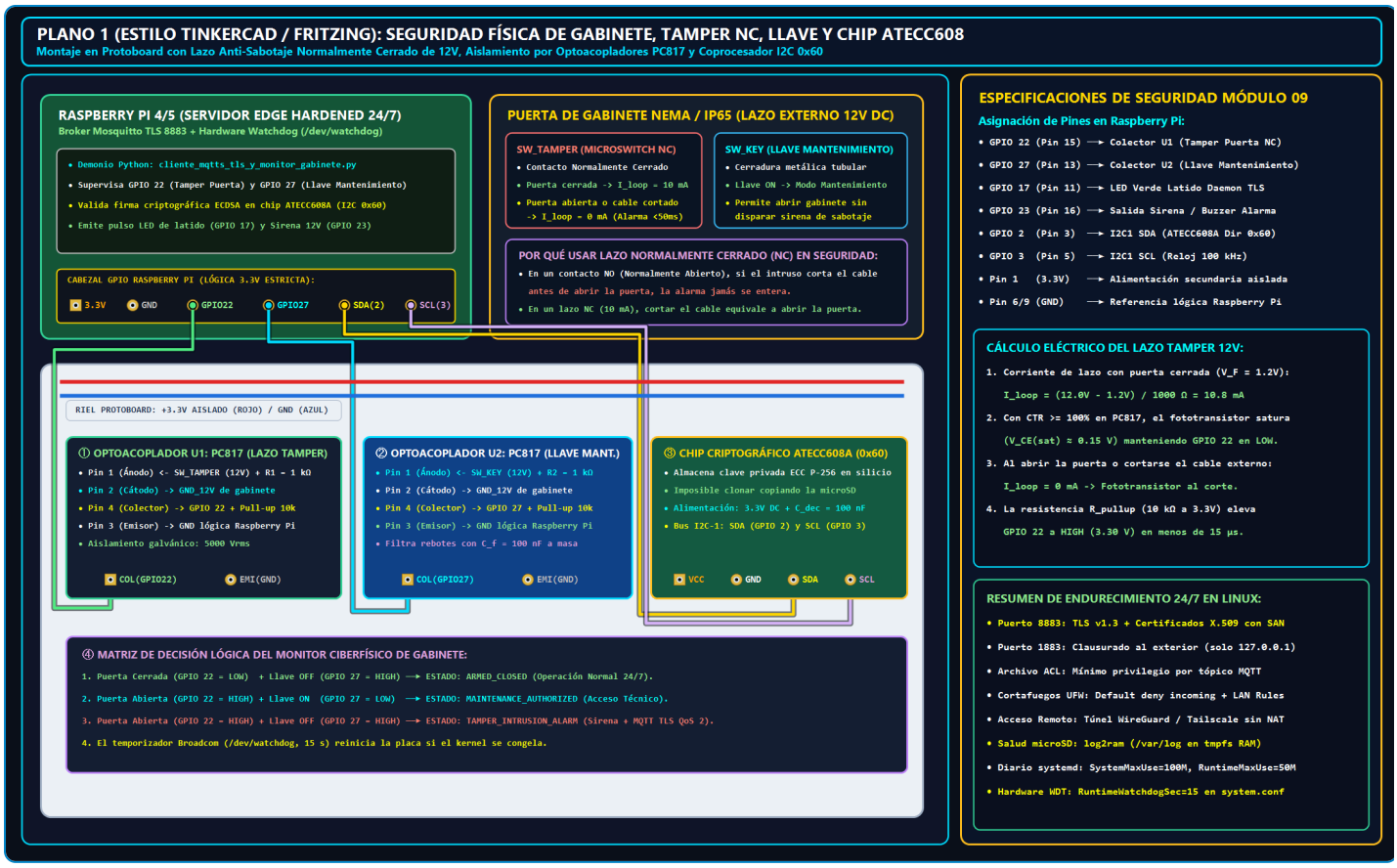
---

## 2. Fase 1: Montaje del Circuito de Seguridad Física de Gabinete (Tamper + Keylock)

Un servidor Edge en planta cuyos puertos físicos están al alcance de cualquiera sigue siendo vulnerable aunque su red esté cifrada. Por ello implementamos supervisión física de gabinete:

- **Microinterruptor de Sabotaje (*Tamper Switch `SW_TAMPER` *en configuración Normalmente Cerrada* `NC` *)**:
- Mientras la puerta del gabinete metálico permanece cerrada, el contacto `NC` conduce corriente a través de una resistencia limitadora de $1\ \text{k}\Omega$ hacia el LED infrarrojo del primer optoacoplador `PC817` **(** `U1` **)**, manteniendo `GPIO 22` en nivel lógico `LOW` ($0\ \text{V}$).
- Si alguien abre la puerta o corta el cable del sensor, el lazo de corriente se interrumpe instantáneamente y la resistencia *pull-up* a $3.3\ \text{V}$ eleva `GPIO 22` a `HIGH` , disparando una interrupción de intrusión.
- **Interruptor de Llave Mecánica de Mantenimiento ( `SW_KEY` en `GPIO 27` )**:
- Permite al ingeniero autorizado poner el gabinete en modo `MAINTENANCE_AUTHORIZED` antes de abrir la puerta, inhibiendo la sirena piezoeléctrica local y registrando en `InfluxDB` una ventana de mantenimiento legítima.

### 📐 Plano 1: Montaje en Protoboard del Nodo de Seguridad Física de Gabinete




---

## 3. Fase 2: Aprovisionamiento de Certificados X.509 con Extensión `SAN` (`OpenSSL`)

- Copia el script `aprovisionar_pki_x509_openssl_san.sh` a la Raspberry Pi y ejecútalo indicando la IP estática y el nombre de host local:

  ```bash
  chmod +x aprovisionar_pki_x509_openssl_san.sh
  sudo ./aprovisionar_pki_x509_openssl_san.sh 192.168.1.100 edge-gateway-01.local
  ```

- Verifica que el certificado firmado `server.crt` contenga la extensión `X509v3 Subject Alternative Name`:

  ```bash
  openssl x509 -in /opt/mecatrobotica/certs/server.crt -noout -text | grep -A 2 "Subject Alternative Name"
  ```

- Debe mostrar: `DNS:edge-gateway-01.local, DNS:localhost, IP Address:192.168.1.100, IP Address:127.0.0.1`.

---

## 4. Fase 3: Cifrado `MQTTS (8883)` y Reglas `ACL` de Mínimo Privilegio

- Genera el archivo de contraseñas cifradas con `PBKDF2-SHA512`:

  ```bash
  sudo mosquitto_passwd -c -b /opt/mecatrobotica/mosquitto/config/passwd bridge_service "ClaveBlindadaBridge2026!"
  sudo mosquitto_passwd -b /opt/mecatrobotica/mosquitto/config/passwd esp32_nodo01 "ClaveNodoCampo2026!"
  ```

sudo mosquitto_passwd -b /opt/mecatrobotica/mosquitto/config/passwd nodered_service "ClaveOrquestador2026!"

```

- Instala el archivo `mosquitto_tls_8883_y_acl_planta.conf` y reinicia el contenedor `iot-mosquitto`.
- **Prueba de Penetración y Auditoría de Reglas `ACL`**:
- Verifica la lectura autorizada con `TLS 1.3`:

  ```bash
  mosquitto_sub -h edge-gateway-01.local -p 8883 --cafile /opt/mecatrobotica/certs/ca.crt \
  -u nodered_service -P "ClaveOrquestador2026!" -t "planta/#" -v
  ```

- Intenta inyectar maliciosamente un cambio de consigna usando las credenciales del sensor exterior `esp32_nodo01`:

  ```bash
  mosquitto_pub -h edge-gateway-01.local -p 8883 --cafile /opt/mecatrobotica/certs/ca.crt \
  -u esp32_nodo01 -P "ClaveNodoCampo2026!" -t "planta/zona1/arduino01/cmd/rpc" -m '{"cmd":"SET_SP","val":45.0}' -q 1
  ```

- **Resultado certificado:** El broker `Mosquitto` rechaza la publicación por violación de `ACL` y el actuador permanece intacto.

---

## 5. Fase 4: Cortafuegos `UFW`, `log2ram` y `Hardware Watchdog` Broadcom

- Ejecuta el script de endurecimiento de producción:

  ```bash
  chmod +x endurecimiento_ufw_fail2ban_log2ram_watchdog.sh
  sudo ./endurecimiento_ufw_fail2ban_log2ram_watchdog.sh 192.168.1.0/24 100.64.0.0/10
  ```

- Verifica que el temporizador *Watchdog* de hardware del SoC Broadcom esté activo a $15\ s$:

  ```bash
  systemctl show | grep -E "RuntimeWatchdogUSec|RebootWatchdogUSec"
  ```
  ```

# LISTADO DE MATERIALES (BOM) Y HARDWARE DE SEGURIDAD — MÓDULO 09

## Supervisión Física de Gabinete (Tamper NC + Keylock), Criptografía y Placa PCB FR-4 Soldada

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

**Módulo:** 09 — Ciberseguridad IoT, Confiabilidad y Puesta en Producción

---

## 1. Tabla Maestra de Componentes de Hardware (BOM)

| Ítem | Designador | Componente / Especificación Técnica | Cantidad | Función en el Módulo 09 |
|---|---|---|---|---|
| **1** | `SBC1` | **Raspberry Pi 4 Model B (4 GB / 8 GB RAM)** en gabinete metálico de aluminio anodizado con montaje Riel DIN | 1 | Servidor Edge Linux endurecido ( `UFW` , `Mosquitto TLS 8883` , `WireGuard` , `log2ram` y `Hardware Watchdog` ). |
| **2** | `SW_TAMPER` | **Microinterruptor de Palanca con Rodillo ( `SPDT` usado en contacto Normalmente Cerrado `NC` )** $5\ A\ /\ 250\ V$ | 1 | Detecta instantáneamente la apertura física de la puerta del gabinete o el corte del cable del lazo de seguridad. |
| **3** | `SW_KEY` | **Interruptor Rotativo de Llave Metálica de 2 Posiciones ( `ON-OFF` con llave extraíble)** $Ø\ 16\ mm$ | 1 | Habilita el modo de mantenimiento autorizado ( `GPIO 27` ) para evitar falsas alarmas cuando el técnico abre el gabinete. |
| **4** | `U1, U2` | **Optoacopladores Fototransistor `PC817C` (DIP-4, aislamiento** $2500\ V_{rms}$**)** | 2 | Aíslan galvánicamente los cables largos de la puerta del gabinete respecto a los pines `GPIO 22` y `GPIO 27` ($3.3\ V$) de la Raspberry Pi. |
| **5** | `U3` | **Módulo Coprocesador Criptográfico `ATECC608A` ( `I2C 0x60` )** (Opcional / Industrial) | 1 | Almacenamiento en silicio a prueba de extracción de claves privadas ECDSA P-256 para autenticación mutua `mTLS` . |
| **6** | `TVS1, TVS2` | **Diodos Supresores de Transitorios `SMBJ15CA` / `P6KE15CA` bidireccionales** | 2 | Absorben descargas electrostáticas ( `ESD` ) al tocar la cerradura metálica o la puerta del gabinete. |
| **7** | `BZ1` | **Zumbador Piezoeléctrico Activo Industrial de** $12\ V\ DC$ **(**$85\ dB$ **a** $10\ cm$**)** + Transistor `2N2222A` | 1 | Emite alerta acústica disuasoria inmediata ante apertura de gabinete no autorizada ( `TAMPER_INTRUSION_ALARM` ). |
| **8** | `R1..R6` | **Resistencias de Película Metálica** $1/4\ W$ **(**$1\%$**):** $2\times\ 1\ k\Omega$ (lazo 12V), $2\times\ 10\ k\Omega$ (*pull-up* 3.3V), $2\times\ 330\ \Omega$ (LEDs) | 6 | Polarización de optoacopladores, referencias lógicas de $3.3\ V$ e indicadores visuales de estado. |
| **9** | `PCB1` | **Placa de Circuito Impreso FR-4 Doble Capa ( `MECATROBOTICA-SEC-WDG-V09` )** con borneras `KF301` soldadas | 1 | Implementación permanente libre de falsos contactos para montaje dentro del gabinete industrial. |

---

## 2. Presupuesto de Reducción de Escrituras en Almacenamiento Flash ( `log2ram` )

En un servidor Linux estándar sin optimizar, los demonios del sistema ( `syslog` , `journald` , `mosquitto` , `docker` ) ejecutan pequeñas escrituras síncronas constantes sobre la tarjeta microSD. Si el volumen de registros es de $V_{log} = 180$ $MB/día$ con un factor de amplificación de escritura en bloques Flash NAND de $WAF \approx 12$:

> $$W_{anual,\ sin\ optimizar} = 180\ MB/día \times 12 \times 365\ días \approx 788.4\ GB/año$$

Al activar `log2ram` **(** `tmpfs` **en RAM)** con volcado diferido compacto y mover las bases de datos al SSD externo:

> $$W_{anual,\ optimizado} < 18.5\ GB/año \quad \quad \Rightarrow \quad \quad \text{Reducción del } 97.6\% \text{ en desgaste de microSD}$$

# 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



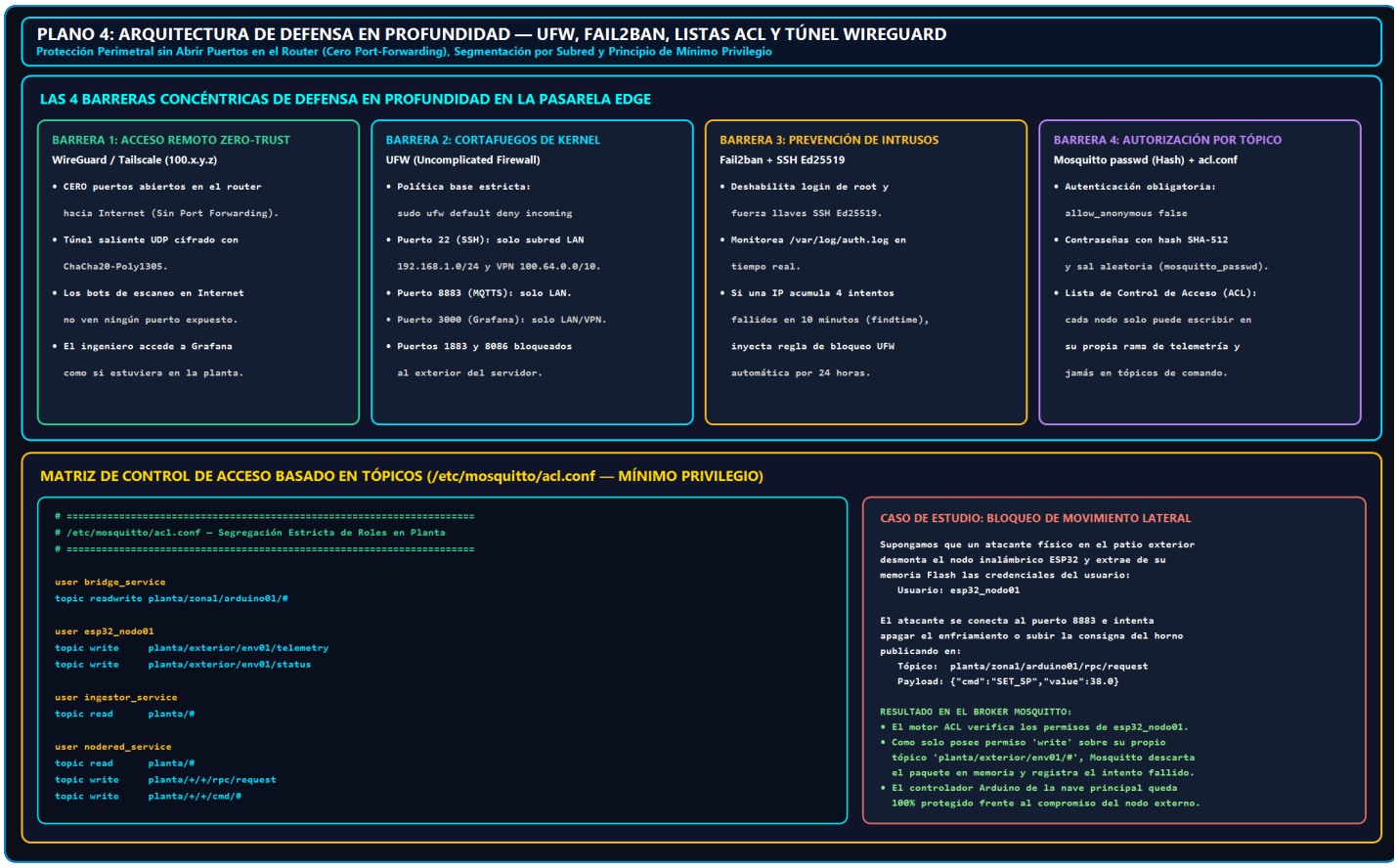

---

## 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.

# PLANOS Y ESQUEMÁTICOS DE CIRCUITO E INGENIERÍA — MÓDULO 09

## Planos en Protoboard (Estilo Tinkercad), Placa PCB FR-4 con Montaje por Soldadura y Diagramas de Ciberseguridad

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

**Módulo:** 09 — Ciberseguridad IoT, Confiabilidad y Puesta en Producción

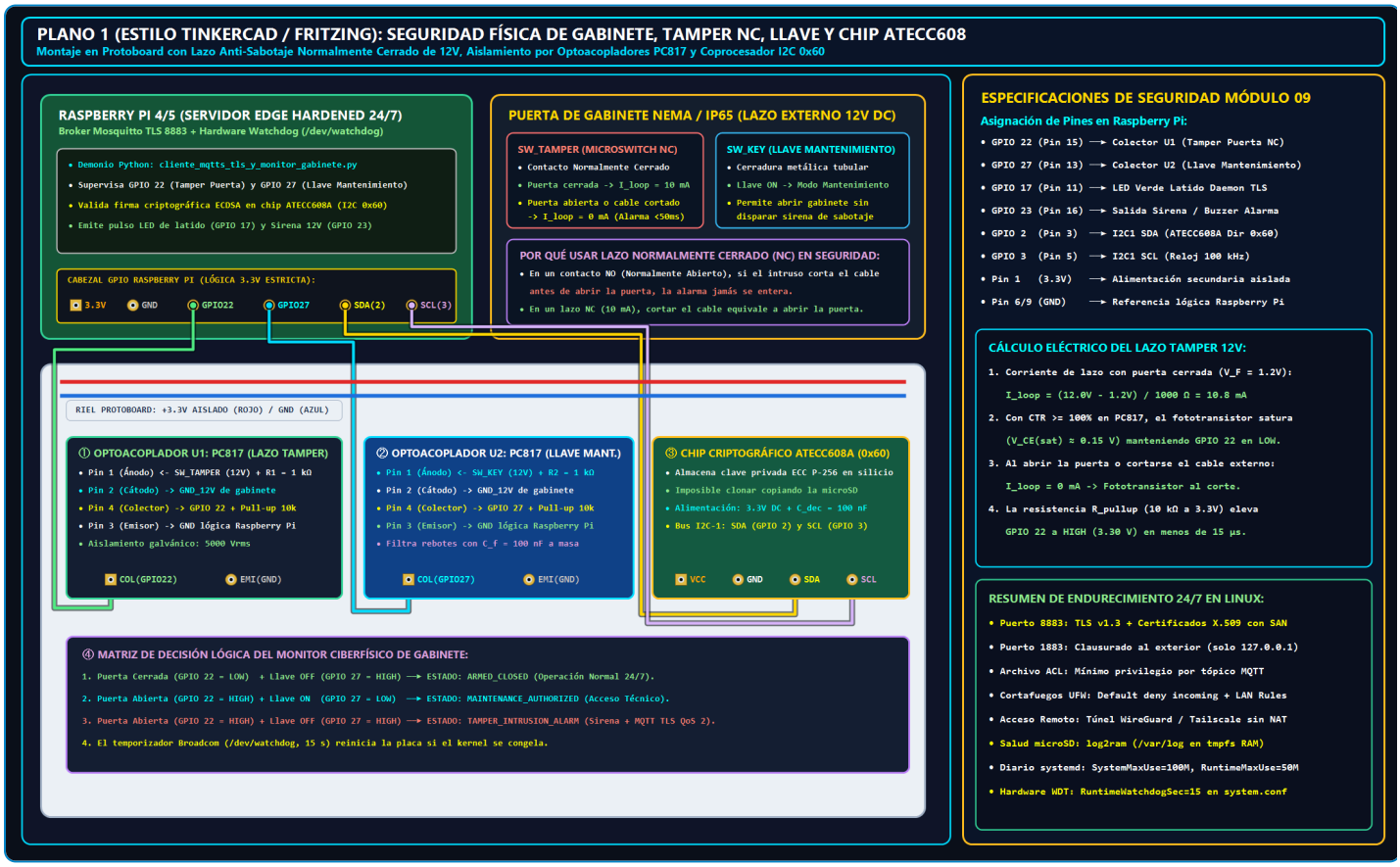
---

## 1. Plano 1 (Estilo Tinkercad / Fritzing): Montaje en Protoboard del Nodo de Seguridad Física de Gabinete y Latido Watchdog

Este primer plano muestra el conexionado práctico sobre protoboard del subsistema de protección física del gabinete Edge:

- **Lazo Normalmente Cerrado ( `NC` ) en `SW_TAMPER` :** Conecta el riel de $12\ V$ al ánodo del optoacoplador `PC817 (U1)` mediante una resistencia de $1\ k\Omega$. Si la puerta se abre o el cable es cortado por un intruso, el colector del fototransistor sube a $3.3\ V$ en `GPIO 22` .
- **Selector de Llave Física ( `SW_KEY` ):** Aislado mediante el segundo optoacoplador `PC817 (U2)` hacia `GPIO 27` , permite autorizar el mantenimiento físico en campo.
- **Coprocesador Criptográfico `ATECC608A` ( `I2C 0x60` ) y Alarma Acústica ( `BZ1` ):** Completan la defensa física y criptográfica del nodo.

### 📐 Plano 1: Montaje en Protoboard de Seguridad Física de Gabinete



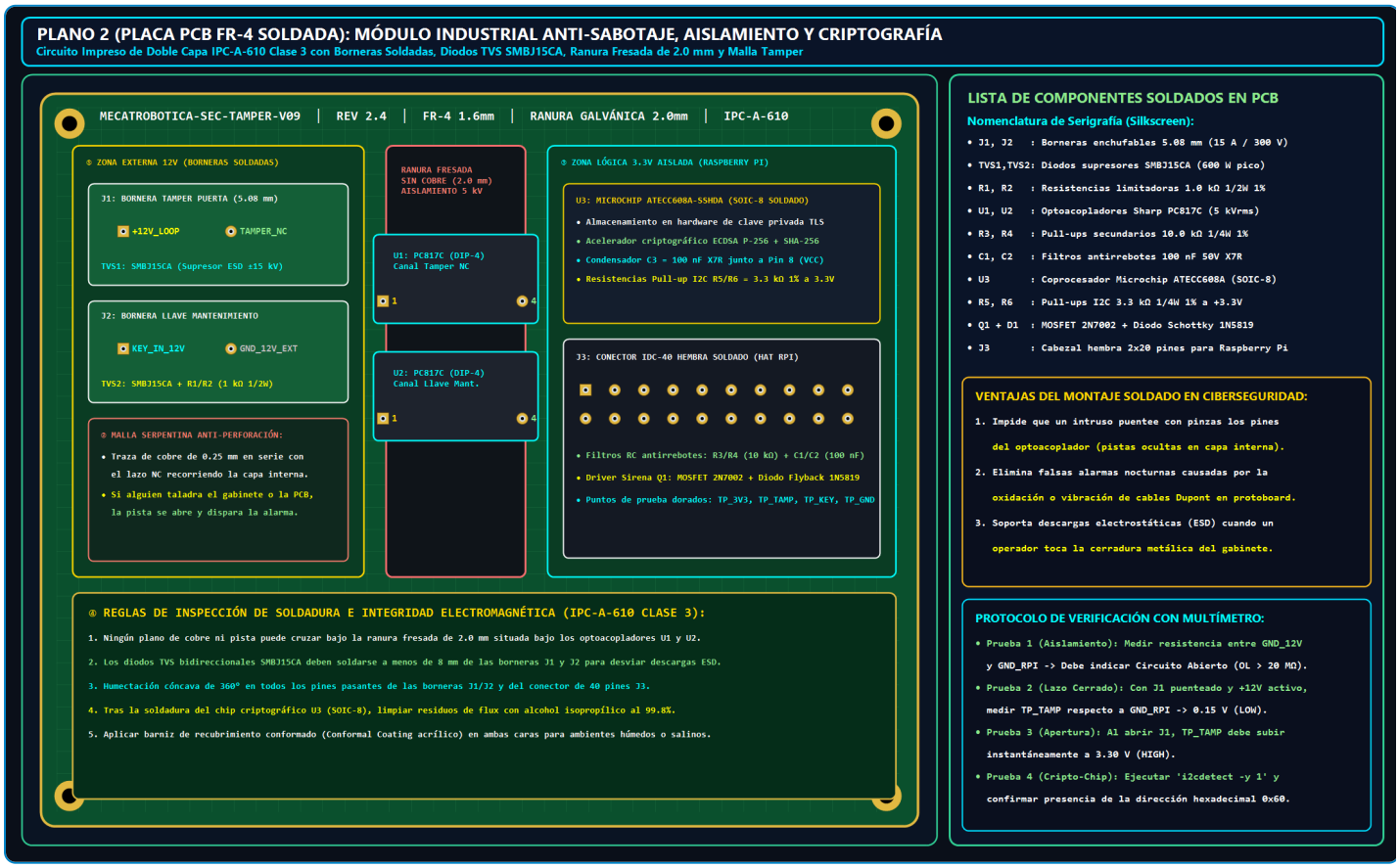

---

## 2. Plano 2 (Placa de Circuito Impreso PCB FR-4): Montaje por Soldadura del Módulo de Seguridad y Watchdog ( `MECATROBOTICA-SEC-WDG-V09` )

En un gabinete industrial real, un sistema anti-sabotaje jamás puede montarse con cables sueltos que puedan puentearse fácilmente. Este segundo plano detalla la placa de circuito impreso de doble capa en fibra de vidrio **FR-4** (`MECATROBOTICA-SEC-WDG-V09`) con montaje por soldadura:

- **Borneras de tornillo `TB1` y `TB2` (`KF301`) soldadas con diodos `TVS SMBJ15CA`:** Protegen las entradas del interruptor de puerta y de la cerradura metálica contra descargas electrostáticas (`ESD` de hasta $\pm 15\ kV$).
- **Ranura fresada de aislamiento galvánico (`2.0 mm`) bajo los optoacopladores `U1` y `U2` (`PC817`):** Separa físicamente el cableado perimetral del gabinete ($12\ V$) respecto al plano lógico de $3.3\ V$ de la Raspberry Pi.
- **Malla conductora anti-perforación (*Tamper Mesh*) en capa interna/superior:** Si se intenta perforar la placa, el lazo `NC` se abre inmediatamente.

### 📐 Plano 2: Placa PCB FR-4 Soldada de Seguridad de Gabinete y Watchdog




---

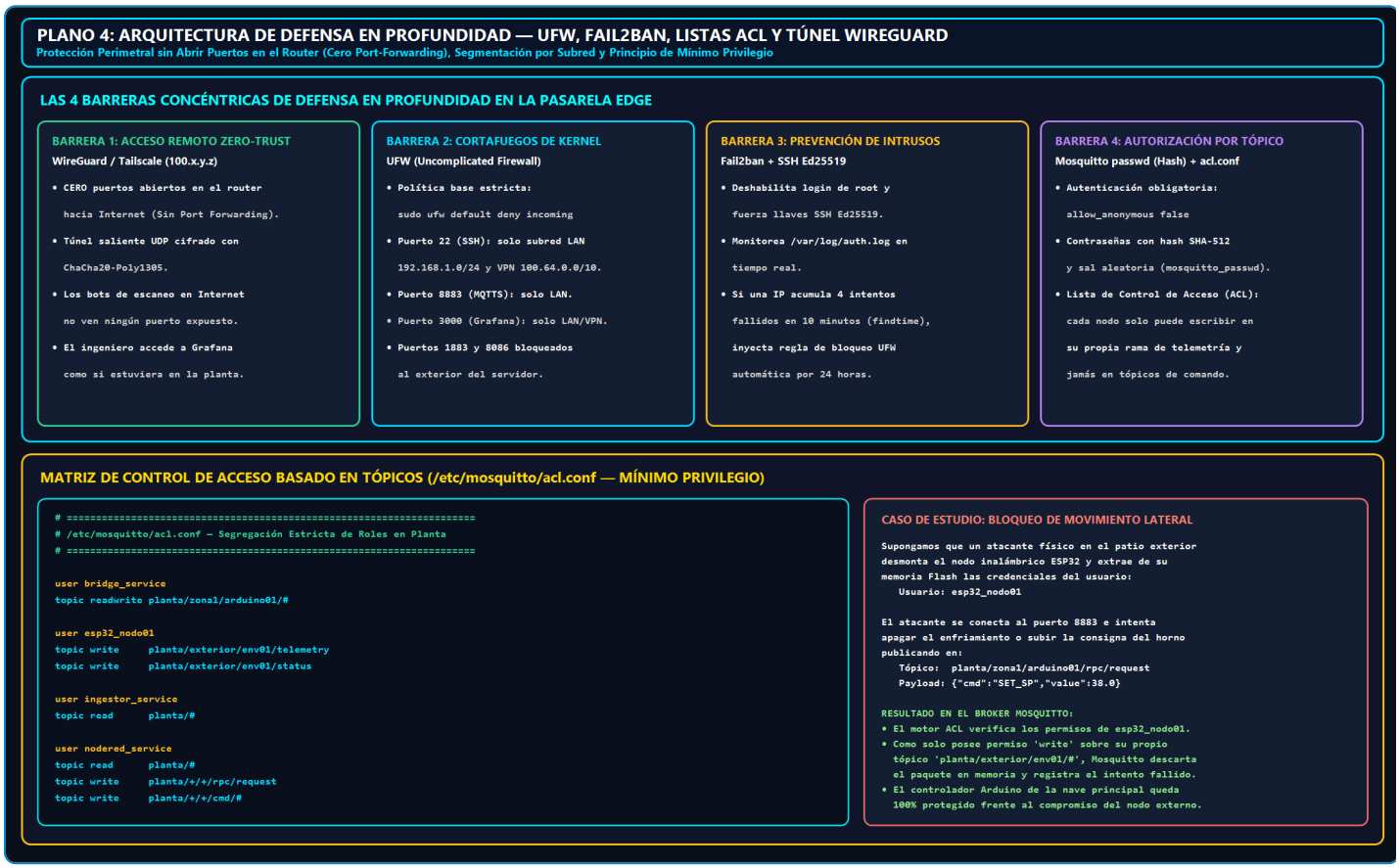
## 3. Plano 3: Infraestructura PKI X.509 con Extensión `SAN` (RFC 9525) y Negociación `MQTTS` `TLS 1.3`

### 📐 Plano 3: Cadena de Confianza PKI X.509 y Handshake TLS 1.3




# 4. Plano 4: Defensa en Profundidad ( `UFW` + `Fail2ban` + `ACL Mosquitto` + `WireGuard VPN` )

## 📐 Plano 4: Defensa en Profundidad y Segmentación de Red



# 5. Plano 5: Supervivencia 24/7 ( `log2ram` en Disco RAM `tmpfs` + `Hardware Watchdog` Broadcom `/dev/watchdog` )

## 📐 Plano 5: Arquitectura de Supervivencia 24/7 en Servidor Edge

