

# Certificate Validation HTTPS — AAP Role

## cert\_monitor

---

### 1. Objetivo

Esta automatización valida certificados TLS (HTTPS) de una lista de URLs (F5/HAProxy/aplicaciones web) y genera:

- **Inventario CSV** con vigencia de:
  - Certificado **principal** (leaf)
  - Certificado **intermedio** (si existe)
  - Certificado **CA** (si existe)
- **Notificación por correo** con la política acordada:
  - Si existe **algún certificado en umbral** (por defecto  $\leq 90$  días): enviar alerta **cada 10 días**
  - Si **no hay alertas**: enviar reporte **mensual**
- Persistencia de estado para controlar reenvíos.

El playbook está diseñado para ejecutarse de forma periódica mediante **Scheduler en Ansible Automation Platform (AAP)** y también de manera manual cuando se requiera.

---

### 2. Alcance funcional (cliente)

- Se consultan URLs HTTPS (balanceadores o apps web) y se obtiene la **cadena de certificados** expuesta por el servicio.
- Se evalúan **tres niveles**:
  1. **Principal** (primer certificado entregado por el servidor)
  2. **Intermedio** (segundo certificado, si existe)
  3. **CA** (último certificado, si existe)
- Para cada uno se calcula:
  - `subject`
  - `notAfter` (fecha de expiración)
  - `days` (días restantes)
- Se determina un estado:
  - **ALERTA** si cualquiera de los certificados está dentro del umbral ( $\leq$  `dias_alerta`)
  - **OK** en caso contrario

Nota: En balanceadores como F5/HAProxy y apps web típicamente el certificado se obtiene por TLS handshake (no por rutas/paths locales). Esta solución se basa en TLS handshake con SNI.

---

### 3. Arquitectura / diseño

#### 3.1 Bastion (pivote)

La ejecución se realiza contra un host pivote denominado **bastion** (RHEL 9), que:

- Tiene conectividad DNS y de red hacia las URLs objetivo
- Ejecuta el playbook y guarda la persistencia/outputs en disco
- Corre con el usuario SSH **ec2-user** (en lab) o el usuario que aplique en cliente

## 3.2 Persistencia y salidas

En el bastion se guardan:

- **Estado** (control de reenvíos):
  - `{{ pivot_state_dir }}/{{ estado_filename }}`
  - Ejemplo: `/home/ec2-user/cert_monitor/state/estado_notificacion.json`
- **Salida CSV:**
  - `{{ pivot_output_dir }}/{{ csv_filename }}`
  - Ejemplo: `/home/ec2-user/cert_monitor/output/inventario_certificados.csv`

---

## 4. Política de notificación (lógica)

La automatización corre de manera **diaria** (recomendado) y decide internamente si envía correo:

### 4.1 Umbral de alerta

- Un certificado se considera “próximo a vencer” si le quedan  $\leq$  **dias\_alerta** días (default: 90)

### 4.2 Frecuencia de notificación

- **Con alertas** (hay al menos un URL en **ALERTA**):
  - Se envía correo **cada dias\_reenvio\_alerta días** (default: 10)
- **Sin alertas:**
  - Se envía correo **mensual** (primer run del mes o si el estado no existe)

### 4.3 Persistencia

Se guarda timestamp de último envío para:

- `last_alert_iso`
- `last_monthly_iso`

De esta forma, aunque el scheduler corra diario, el correo sólo se envía cuando corresponde por política.

## 5. Estructura del repositorio

```
playbooks/  
  certs-validate.yml  
roles/  
  cert_monitor/  
    defaults/  
      main.yml
```

```
vars/  
  config.yml  
tasks/  
  main.yml  
  url_process.yml  
  mail_send.yml  
templates/  
  inventario_certificados.csv.j2  
  mail_body.j2
```

## 5.1 Playbook

`playbooks/certs-validate.yml` es intencionalmente “mínimo”; solo invoca el role.

## 5.2 Role

El role `cert_monitor` contiene:

- `tasks/main.yml` → orquestación completa del flujo
- `tasks/url_process.yml` → procesamiento individual de cada URL
- `tasks/mail_send.yml` → envío de correo SMTP (solo si aplica)
- `templates/*.j2` → plantillas editables para CSV y correo

---

# 6. Componentes AAP (Ansible Automation Platform)

## 6.1 Project (Bitbucket)

- Configurar un **Project** que apunte al repositorio Bitbucket.
- Sincronizar/actualizar el Project para traer:
  - `playbook`
  - `role`
  - `templates`
  - `vars/config.yml`

## 6.2 Inventory

- Crear un Inventory (ej. `cert-monitor-inventory`)
- Añadir host **bastion** (1 host recomendado)
  - Ejemplo: `bastion.example.com`

Recomendación: 1 bastion por ambiente para asegurar consistencia de persistencia.

## 6.3 Credentials

### 6.3.1 Credencial SSH del bastion

- Tipo: Machine

- Usuario: `ec2-user` (lab) / `ansible_usr` o el que aplique
- Key/Password: según políticas del ambiente
- Asociar esta credencial al Job Template

### 6.3.2 Credenciales SMTP (recomendado: Credential Type personalizado)

Para no exponer user/password en Git o Extra Vars, se recomienda un **Credential Type** personalizado:

**Administration** → **Credential Types** → **Add**

- Nombre: `SMTP Credentials`

#### Input Configuration

```
{
  "fields": [
    { "id": "smtp_username", "type": "string", "label": "SMTP Username" },
    { "id": "smtp_password", "type": "string", "label": "SMTP Password" },
    "secret": true
  ],
  "required": []
}
```

#### Injector Configuration

```
{
  "extra_vars": {
    "smtp_username": "{{ smtp_username }}",
    "smtp_password": "{{ smtp_password }}"
  }
}
```

Después:

- **Resources** → **Credentials** → **Add**
  - Tipo: `SMTP Credentials`
  - Cargar `smtp_username` / `smtp_password`

Si el relay SMTP no requiere autenticación, esta credencial no es necesaria.

## 6.4 Job Template

Crear un Job Template (ej. `Certificate Validation HTTPS`)

- Playbook: `playbooks/certs-validate.yml`
- Inventory: el del bastion

- Credentials:
  - SSH bastion
  - SMTP credentials (si aplica)
- Extra Vars: opcional (ver sección de variables)

## 6.5 Scheduler (Schedule)

**Recomendación:** ejecutar **diario** (ej. 02:00 AM)

- Pestaña Schedules → Add → Recurring
- Frequency: Daily
- Time: 02:00
- Timezone: la correspondiente al cliente/ambiente

### Motivo:

- Permite detectar el primer día que un certificado entra al umbral ( $\leq$  `dias_alerta`)
- El correo no se envía diario, se controla por la persistencia y reglas

## 6.6 Concurrency

Recomendado:

- No permitir ejecuciones simultáneas del mismo template, para evitar colisiones de escritura en:
  - `estado_notificacion.json`
  - `inventario_certificados.csv`

---

# 7. Variables y configuración

## 7.1 Config editable (Git)

`roles/cert_monitor/vars/config.yml`

Recomendado para:

- URLs
- Umbrales
- Paths
- Parámetros generales

Ejemplo:

```
urls_certificados:
- https://console-openshift-console.apps.ocp-dc2.example.com
- https://api.ocp-dc2.example.com:6443/
- https://app.example.com

dias_alerta: 90
```

```
dias_reenvio_alerta: 10
ambiente_nombre: "PROD - CERT MONITOR"

pivot_state_dir: /home/ec2-user/cert_monitor/state
pivot_output_dir: /home/ec2-user/cert_monitor/output

modo_sin_smtp: false

smtp_host: "smtp.example.com"
smtp_port: 587
smtp_use_starttls: true
correo_from: "cert-monitor@example.com"
correo_to:
  - "ops@example.com"
  - "security@example.com"
```

**IMPORTANTE:** No guardar `smtp_password` en Git.

## 7.2 Defaults (Role)

`roles/cert_monitor/defaults/main.yml` contiene valores por defecto.

## 7.3 Extra Vars (AAP)

Se pueden usar Extra Vars para sobrescribir valores sin modificar Git.

- Extra Vars tienen mayor prioridad que defaults y vars/config.yml.

Casos recomendados:

- pruebas puntuales
- overrides por ambiente
- habilitar/deshabilitar smtp temporalmente:
  - `modo_sin_smtp: true/false`

---

## 8. Templates (Jinja2)

### 8.1 CSV

`roles/cert_monitor/templates/inventario_certificados.csv.j2`

Define el formato final del inventario con columnas:

- URL
- Cert\_Principal / Vigencia\_Principal / Dias\_Principal
- Cert\_Intermedio / Vigencia\_Intermedio / Dias\_Intermedio
- Cert\_CA / Vigencia\_CA / Dias\_CA
- Estado

## 8.2 Cuerpo del correo

`roles/cert_monitor/templates/mail_body.j2`

Incluye:

- Resumen de ejecución
- Detalle de URLs en umbral (cuando aplica)
- Mención de adjunto CSV
- Recordatorio de política de reenvío

Editar aquí el correo es seguro y no requiere tocar el flujo del role.

---

## 9. Funcionamiento técnico (cómo se obtiene la información)

Por cada URL:

1. Se normaliza host y puerto (SNI):

- extrae hostname de la URL
- si no hay puerto explícito, se usa `443`

2. Se ejecuta:

- `openssl s_client -servername <host> -connect <host>:<port> -showcerts`

3. Se parsea la salida para obtener bloques PEM:

- `-----BEGIN CERTIFICATE----- ... -----END CERTIFICATE-----`

4. Se calcula `notAfter` y días restantes con `openssl x509 -enddate`

5. Se clasifica:

- `ALERTA` si cualquier certificado tiene `days <= dias_alerta`

---

## 10. Validación y troubleshooting

### 10.1 Validación en bastion

• CSV generado:

- `cat /home/ec2-user/cert_monitor/output/inventario_certificados.csv`

• Estado:

- `cat /home/ec2-user/cert_monitor/state/estado_notificacion.json`

### 10.2 Si un URL no responde / handshake falla

El task no interrumpe la ejecución (se maneja con `failed_when: false` en el TLS fetch). Se recomienda en fase de hardening:

- distinguir “sin cert” vs “error conexión”
- agregar código/estado adicional en CSV

### 10.3 SMTP pendiente

Mientras no haya SMTP:

- mantener `modo_sin_smtp: true`
- validar que el CSV y el estado se generan correctamente
- una vez entregado SMTP, cambiar a `modo_sin_smtp: false`

---

## 11. Uso

### 11.1 Ejecución manual (AAP)

- Launch Job Template
- Revisar output y artifacts/logs
- Validar CSV en bastion

### 11.2 Ejecución programada

- Scheduler diario (02:00 AM)
- Corre todo el flujo y decide envío según política

---

## 12. Seguridad y buenas prácticas

- No almacenar contraseñas SMTP en Git.
- Preferir Credential Type personalizado para SMTP.
- Evitar concurrencia en Job Template.
- Mantener el bastion con conectividad requerida.
- Versionar cambios de URLs en `vars/config.yml` mediante PR/approval cuando aplique.

---

## 13. Parámetros clave (resumen)

| Variable                         | Descripción                                     | Default  |
|----------------------------------|---|--|
| <code>urls_certificados</code>   | Lista de URLs HTTPS objetivo                    | <code>[]</code>                                |
| <code>dias_alerta</code>         | Umbral de “próximo a vencer” en días            | <code>90</code>                                |
| <code>dias_reenvio_alerta</code> | Reenvío de alertas cuando hay umbral            | <code>10</code>                                |
| <code>modo_sin_smtp</code>       | Deshabilita envío SMTP (solo genera CSV/estado) | <code>true</code>                              |
| <code>pivot_state_dir</code>     | Directorio persistente estado                   | <code>/home/ec2-user/cert_monitor/state</code> |

| Variable                       | Descripción                        | Default   |
|--------------------------------|------------------------------------|---|
| <code>pivot_output_dir</code>  | Directorio persistente salidas     | <code>/home/ec2-user/cert_monitor/output</code> |
| <code>smtp_host</code>         | Host SMTP                          | <code>" "</code>                                |
| <code>smtp_port</code>         | Puerto SMTP                        | <code>25</code>                                 |
| <code>smtp_use_starttls</code> | STARTTLS                           | <code>false</code>                              |
| <code>smtp_username</code>     | Usuario SMTP (AAP credential)      | <code>" "</code>                                |
| <code>smtp_password</code>     | Password SMTP (AAP credential)     | <code>" "</code>                                |
| <code>correo_from</code>       | Remitente                          | <code>" "</code>                                |
| <code>correo_to</code>         | Lista destinatarios                | <code>[]</code>                                 |
| <code>ambiente_nombre</code>   | Nombre ambiente mostrado en correo | <code>CERT MONITOR</code>                       |

## 14. Notas finales

- Esta solución está preparada para escalar:
  - agregar más URLs sin tocar el playbook (solo `vars/config.yml`)
  - ajustar cuerpo del correo y CSV sin tocar lógica (`templates/`)
- El enfoque scheduler diario + control interno de reenvío garantiza:
  - detección oportuna del umbral
  - cumplimiento de frecuencia requerida sin duplicar schedules

