

REPORTE DE AVANCE DE PRÁCTICAS PROFESIONALES 2

1. Datos Generales

- Nombre del Estudiante: Sergio Andres Cantor Moya
- Carrera: Ingeniería de sistemas
- Institución/Empresa: Universidad de los Andes
- Supervisor en la Empresa: Maria Jose Afanador
- Fecha de Entrega del Reporte: 11/11/2025

2. Actividades Realizadas

#	Actividad	Descripción de la tarea realizada	Producto o Evidencia
1	Automatización de Inventario	Implementación de sistema que actualiza automáticamente el inventario de equipos desde reportes en GitHub	Script YAML de actualización de inventario
2	Configuración de Autologin	Habilitación de inicio de sesión automático para usuario "estudiantes" en equipos de la sala T-201	Playbook de autologin y registro de ejecución
3	Preparación de Instalaciones	Desarrollo de sistema para instalación remota y silenciosa de aplicaciones en los equipos	Playbook de instalación de software

3. Productos y Evidencias Entregadas

- **Sistema de Gestión de Inventario Automatizado:** Script en Ansible que sincroniza las direcciones IP de los equipos desde reportes generados automáticamente en GitHub
- **Configuración de Autologon para Sala T-201:** Playbook que habilita el inicio de sesión automático para el usuario estudiantes, incluyendo reinicio automático de equipos
- **Sistema de Despliegue Remoto de Software:** Conjunto de playbooks para la instalación masiva de aplicaciones en los equipos de cómputo de manera desatendida

4. Porcentaje de Cumplimiento (Autovaloración del estudiante y calificación del supervisor sobre el cumplimiento de las actividades asignadas)

Aspecto Evaluado	Autoevaluación (%)	Evaluación del Supervisor (%)
Puntualidad y asistencia	100	100
Calidad de trabajo	100	100
Responsabilidad y compromiso	100	100
Cumplimiento de tareas asignadas	100	100
Trabajo en equipo y comunicación	100	100

Aspecto Evaluado	Autoevaluación (%)	Evaluación del Supervisor (%)
Total de cumplimiento promedio (%)	100	100

5. Comentarios y Observaciones (Espacio para sugerencias o comentarios tanto del estudiante como del supervisor)

Sergio está avanzando satisfactoriamente. Le recomiendo contactar a otros pares de la universidades que manejen salas de cómputo para conversar sobre retos del proyecto de práctica y conocer posibilidades para solucionar los retos para cumplir completamente con los objetivos de la práctica.

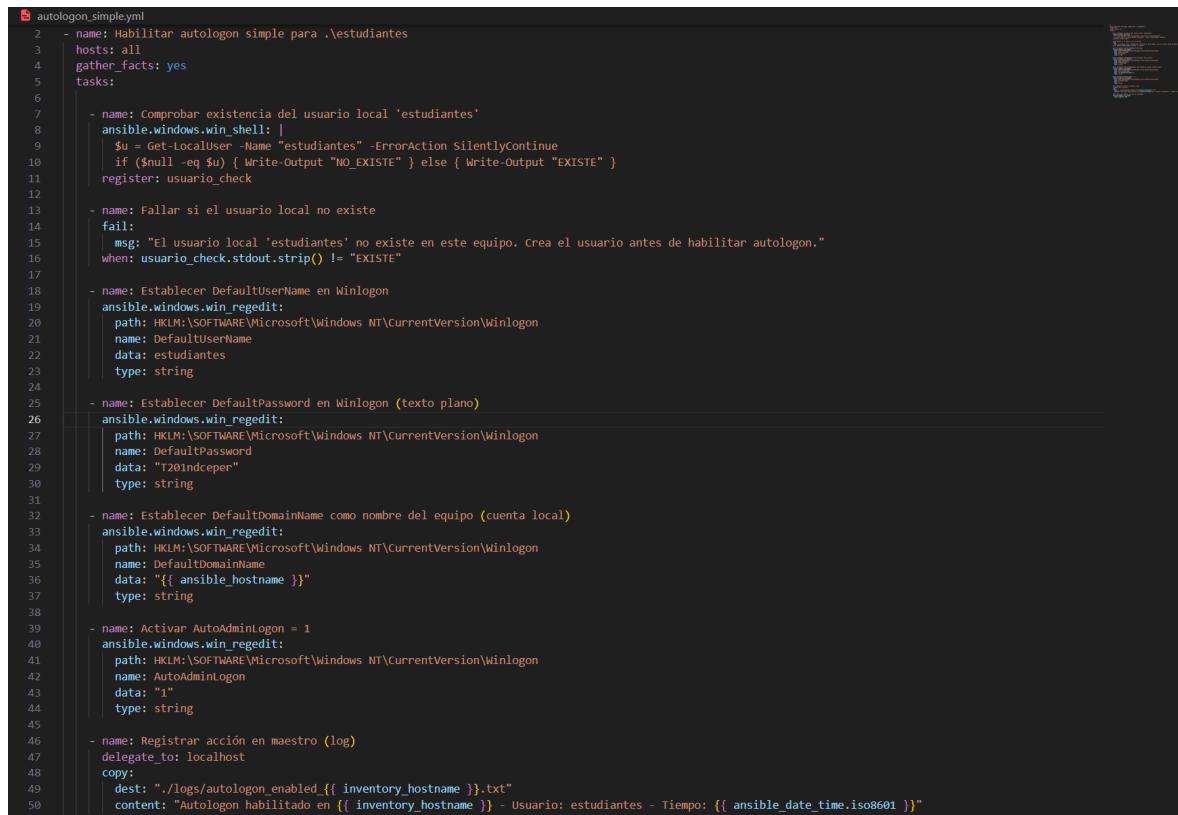
Firma del Estudiante: 

Firma del Supervisor: 

Avance del Proyecto O.R.I.O.N. - Análisis Técnico de Playbooks

1. autologon_simple.yml

Propósito: Configurar inicio de sesión automático para el usuario local "estudiantes".



```
autologon_simple.yml
1 - name: Habilitar autologon simple para ./estudiantes
2 hosts: all
3 gather_facts: yes
4 tasks:
5
6     - name: Comprobar existencia del usuario local 'estudiantes'
7       ansible.windows.win_shell:
8         $u = Get-LocalUser -Name "estudiantes" -ErrorAction SilentlyContinue
9         if ($null -eq $u) { write-Output "NO EXISTE" } else { Write-Output "EXISTE" }
10        register: usuario_check
11
12
13     - name: Fallar si el usuario local no existe
14       fail:
15         msg: "El usuario local 'estudiantes' no existe en este equipo. Crea el usuario antes de habilitar autologon."
16         when: usuario_check.stdout.strip() != "EXISTE"
17
18
19     - name: Establecer DefaultUserName en Winlogon
20       ansible.windows.win_regedit:
21         path: HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
22         name: DefaultUserName
23         data: estudiantes
24         type: string
25
26     - name: Establecer DefaultPassword en Winlogon (texto plano)
27       ansible.windows.win_regedit:
28         path: HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
29         name: DefaultPassword
30         data: "T201ndeper"
31         type: string
32
33     - name: Establecer DefaultDomainName como nombre del equipo (cuenta local)
34       ansible.windows.win_regedit:
35         path: HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
36         name: DefaultDomainName
37         data: "{{ ansible_hostname }}"
38         type: string
39
40     - name: Activar AutoAdminLogon = 1
41       ansible.windows.win_regedit:
42         path: HKLM:\SOFTWARE\Microsoft\Windows NT\CurrentVersion\Winlogon
43         name: AutoAdminLogon
44         data: "1"
45         type: string
46
47
48     - name: Registrar acción en maestro (log)
49       delegate_to: localhost
50       copy:
51         dest: "./logs/autologon_enabled_{{ inventory_hostname }}.txt"
52         content: "Autologon habilitado en {{ inventory_hostname }} - Usuario: estudiantes - Tiempo: {{ ansible_date_time.iso8601 }}"
```

Explicación:

- Utiliza PowerShell para verificar la existencia del usuario mediante el cmdlet Get-LocalUser, manejando silenciosamente errores y registrando el resultado
- Modifica el registro de Windows en la ruta específica de Winlogon para definir el usuario que iniciará sesión automáticamente.
- Configura la contraseña del usuario en texto plano en el registro (consideración de seguridad: esto expone la contraseña en el registro).

- Define el dominio como el nombre del equipo local, indicando que es una cuenta local y no de dominio.
- Habilita la funcionalidad de inicio de sesión automático estableciendo el valor en 1.

2. test_inv.yml

Propósito: Configurar el reporte automático de direcciones IP hacia GitHub al inicio de sesión.

```

test_inv.yml
1  ---
2  - name: Reporter GitHub autosuficiente v6 (token público legible)
3    hosts: all
4    gather_facts: no
5
6    tasks:
7      - name: Asegurar carpetas necesarias
8        ansible.windows.win_file:
9          path: "{{ item }}"
10         state: directory
11        loop:
12          - "C:\Temp"
13          - "C:\Users\Public"
14
15      - name: Limpiar token anterior
16        ansible.windows.win_file:
17          path: "C:\Users\Public\.github_token"
18          state: absent
19
20      - name: copiar token (legible por todos)
21        ansible.windows.win_copy:
22          src: "./archivos/github_token"
23          dest: "C:\Users\Public\.github_token"
24
25      - name: Ajustar permisos del token
26        ansible.windows.win_shell:
27          args:
28            executable: cmd.exe
29
30      - name: Copiar script principal
31        ansible.windows.win_copy:
32          src: "./archivos/report_ip_to_github.ps1"
33          dest: "C:\Temp\report_ip_to_github.ps1"
34
35      - name: Crear wrapper .cmd en Startup comun (v6)
36        ansible.windows.win_copy:
37          content: |
38            @echo off
39            echo [%date% %time%] Iniciando wrapper > C:\Temp\report_startup_log.txt
40            powershell.exe -ExecutionPolicy Bypass -NoProfile -WindowStyle Hidden -File "C:\Temp\report_ip_to_github.ps1" -RepoOwner "Ceper-Ansible" -RepoName "ip-reports" -TokenPath "C:\Users\Public\.github_token"
41            echo [%date% %time%] Wrapper finalizado >> C:\Temp\report_startup_log.txt
42            dest: "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup\run_report_at_logon.cmd"
43
44      - name: Confirmar creación del wrapper
45        ansible.windows.win_shell:
46          dir: "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\Startup"
47          register: startup_check
48
49      - debug:
50        var: startup_check.stdout
51
52

```

Explicación:

- Copia el token de GitHub a una ubicación pública en el equipo remoto.
- Configura permisos NTFS para que todos los usuarios puedan leer el token:
 - /inheritance:e: Habilita herencia
 - /grant *S-1-1-0:R: Otorga permiso de lectura a todos los usuarios (SID S-1-0)

- Crea un script por lotes en la carpeta de inicio automático que ejecuta el script

PowerShell al loguearse, con parámetros:

- Bypass: Omite la política de ejecución
- Hidden: Ejecución en segundo plano
- Redirección de salida a archivo de log

3. actualizar_inventario.yml

Propósito: Sistema automatizado que mantiene actualizado el inventario de equipos

consultando reportes de IP almacenados en GitHub.

```

actualizar_inventario.yml > ...
1  ---
2  - name: Actualizar inventario desde GitHub (v2 robusta con depuración)
3    hosts: localhost
4    gather_facts: no
5
6    vars:
7      repo_owner: "Ceper-Ansible"
8      repo_name: "ip-reports"
9      branch: "main"
10     token_path: "./archivos/.github_token"
11     inventory_path: "./inventory.ini"
12     temp_dir: "/tmp/github_reports"
13
14    tasks:
15
16      - name: Leer token desde archivo local
17        ansible.builtin.slurp:
18          src: "{{ token_path }}"
19          register: github_token_file
20
21      - name: Convertir token a texto
22        ansible.builtin.set_fact:
23          github_token: "{{ github_token_file['content'] | b64decode | trim }}"
24
25      - name: Crear carpeta temporal para reportes
26        ansible.builtin.file:
27          path: "{{ temp_dir }}"
28          state: directory
29
30      - name: Obtener lista de archivos desde la API de GitHub
31        ansible.builtin.uri:
32          url: "https://api.github.com/repos/{{ repo_owner }}/{{ repo_name }}/git/trees/{{ branch }}?recursive=1"
33          headers:
34            Authorization: "token {{ github_token }}"
35            Accept: "application/vnd.github.v3+json"
36            return_content: yes
37        register: repo_content
38
39      - name: Mostrar nombres detectados en el repositorio (debug)
40        ansible.builtin.debug:
41          msg: "{{ item.path }}"
42          loop: "{{ repo_content.json.tree | selectattr('type', 'equalto', 'blob') | list }}"
43          when: repo_content.json.tree is defined

```

```

45   - name: Descargar archivos detectados (sin filtrar extensión)
46     ansible.builtin.get_url:
47       url: "https://raw.githubusercontent.com/{{ repo_owner }}/{{ repo_name }}/{{ branch }}/{{ item.path }}"
48       headers:
49         Authorization: "token {{ github_token }}"
50       dest: "{{ temp_dir }}/{{ item.path | basename }}"
51       force: yes
52     loop: "{{ repo_content.json.tree | selectattr('type', 'equalto', 'blob') | list }}"
53     register: downloads
54     when: repo_content.json.tree | length > 0
55
56   - name: Regenerar inventario
57     ansible.builtin.copy:
58       dest: "{{ inventory_path }}"
59       content: |
60         [windows]
61         {% for f in downloads.results if f.dest is defined %}
62         {% for line in lookup('file', f.dest).splitlines() %}
63           {{ line }}
64         {% endfor %}
65         {% endfor %}
66     when: downloads is defined
67

```

Explicación:

- Lee el token de autenticación de GitHub desde un archivo local en formato binario y lo almacena en una variable temporal.
- Decodifica el token desde base64 a texto plano y elimina espacios en blanco, preparándolo para su uso en la API.
- Realiza una petición HTTP GET a la API de GitHub para obtener la estructura completa del repositorio, incluyendo todos los archivos y directorios.
- Itera sobre cada archivo encontrado en el repositorio (filtrando solo archivos, no directorios) y los descarga usando URLs directas de GitHub Raw.
- Construye dinámicamente el archivo inventory.ini concatenando el contenido de todos los archivos descargados, manteniendo la sección [windows] y agregando cada línea de los reportes.

4. apagar.yml

Propósito: Ejecutar apagado remoto simultáneo de todos los equipos en el inventario.

```
apagar.yml > {} o > [ ]tasks > {} o
1   ---
2   - name: Apagar equipos de la sala T-201
3     hosts: all
4     gather_facts: no
5     tasks:
6       - name: Ejecutar apagado remoto
7         win_shell: shutdown /s /t 0
8
```

Explicación:

- Define que el playbook se ejecutará contra todos los hosts del inventario sin recopilar información del sistema para mayor velocidad.
- Ejecuta el comando de Windows shutdown con parámetros:
 - /s: Apagar el sistema
 - /t 0: Sin tiempo de espera (0 segundos)

5. hostname.yml

Propósito: Obtener y mostrar los nombres de host de todos los equipos en la red.

```
(hostname.yml) 0 > [ ]tasks > {} 1 > {} debug
  1  <-- name: Obtener hostname de Windows remoto
  2    hosts: windows
  3    gather_facts: no
  4  <-- tasks:
  5    - name: Ejecutar comando hostname
  6      win_command: hostname
  7      register: resultado
  8
  9    - name: Mostrar salida del hostname
 10      debug:
 11        var: resultado.stdout
 12
```

Explicación:

- Se ejecuta específicamente sobre el grupo "windows" del inventario, deshabilitando la recolección de facts.
- Ejecuta el comando nativo hostname de Windows y almacena el resultado en una variable.
- Muestra en la salida de Ansible el nombre del host obtenido del comando anterior.

6. wake.yml

Propósito: Encender equipos remotamente usando Wake-on-LAN.

```
wake.yml > {} o > [ ] tasks > {} o
1   ---
2   - name: Encender equipos de la sala T-201 mediante Wake-on-LAN
3     hosts: all
4     gather_facts: no
5
6     tasks:
7       - name: Enviar paquete mágico para encender el equipo
8         community.general.wakeonlan:
9           mac: "{{ mac_address }}"
10          delegate_to: localhost
11          run_once: false
12
```

Explicación:

- Utiliza el módulo wakeonlan de Ansible para enviar paquetes "Magic Packet" a cada dirección MAC definida en el inventario. El parámetro run_once: false asegura que se ejecute para cada host individual.
- **Problema identificado:** Los switches de red de la universidad posiblemente bloquean los paquetes Wake-on-LAN o los equipos no tienen habilitada esta funcionalidad en BIOS.

7. winget.yml

Propósito: Actualizar automáticamente aplicaciones usando el gestor de paquetes Winget de Windows.

```
1 winget.yml > ...
2   - name: Actualización automática y silenciosa con Winget
3     hosts: all
4     gather_facts: yes
5
6     tasks:
7       - name: Crear directorio local para logs
8         delegate_to: localhost
9         file:
10           path: "./logs"
11           state: directory
12
13       - name: Buscar ruta real de winget
14         win_shell: |
15           $paths = @(
16             "$env:LOCALAPPDATA\Microsoft\WindowsApps\winget.exe",
17             "$env:ProgramFiles\WindowsApps\Microsoft.DesktopAppInstaller*\winget.exe",
18             "C:\Program Files\WindowsApps\Microsoft.DesktopAppInstaller_**\winget.exe"
19           )
20           foreach ($p in $paths) {
21             $resolved = Get-Item -Path $p -ErrorAction SilentlyContinue
22             if ($resolved) { Write-Output $resolved.FullName; break }
23           }
24     register: winget_path
25
26     - name: Ejecutar actualización si winget está disponible
27       when: winget_path.stdout | trim != ""
28       win_shell: |
29         & "{{ winget_path.stdout | trim }}" upgrade --all
30         --accept-package-agreements --accept-source-agreements --silent;
31         Write-Output "Actualización completada correctamente."
32       register: resultado
33       ignore_errors: yes
34
35     - name: Registrar salida de actualización
36       when: resultado is defined
37       debug:
38         msg: "{{ resultado.stdout | default('sin salida visible (modo silencioso)') }}"
39
40     - name: Registrar si winget no está disponible
41       when: winget_path.stdout | trim == ""
42       debug:
43         msg: "⚠️ Winget no encontrado en {{ inventory_hostname }}. Instálalo con App Installer desde Microsoft Store."
44
45     - name: Guardar log de ejecución
46       delegate_to: localhost
47       copy:
48         content: |
49           === LOG DE ACTUALIZACIÓN WINGET ===
50           Equipo: {{ inventory_hostname }}
51           Fecha: {{ ansible_date_time.date }} {{ ansible_date_time.time }}
52           Ruta Winget: {{ winget_path.stdout | default('No detectado') }}
53
54           {{ resultado.stdout | default('sin salida (modo silencioso)') }}
55           Error:
56           {{ resultado.stderr | default('sin errores') }}
57           dest: "./logs/winget_{{ inventory_hostname }}.txt"
58
59     - name: Crear tarea programada para actualizaciones nocturnas
60       when: winget_path.stdout | trim != ""
61       win_scheduled_task:
62         name: "WingetAutoUpdate"
63         description: "Actualiza aplicaciones automáticamente cada noche a las 2:00 AM"
64         actions:
65           - path: "{{ winget_path.stdout | trim }}"
66             arguments: "upgrade --all --accept-package-agreements --accept-source-agreements --silent"
67         triggers:
68           - type: daily
69             start_boundary: '2025-10-29T02:00:00'
70             username: "{{ ansible_user }}"
71             password: "{{ ansible_password }}"
72             state: present
73             enabled: yes
```

Explicación:

- Busca winget.exe en múltiples rutas comunes donde se instala, manejando la variabilidad de rutas con caracteres comodín.
- Ejecuta winget con parámetros de actualización masiva:
 - upgrade --all: Actualiza todas las aplicaciones instaladas
 - --accept-*-agreements: Acepta automáticamente licencias
 - --silent: Modo no interactivo
- **Problema identificado:** Aunque Winget está presente en Windows, la ejecución remota no encuentra el ejecutable en las rutas esperadas, posiblemente por restricciones de AppX deployment o contextos de usuario.

8. inventario.yml

Propósito: Recolectar información detallada del hardware y configuración de todos los equipos.

```
inventory.yml > ...
1 - name: Recolectar información detallada de los equipos de la sala T-201
2 hosts: all
3 gather_facts: yes
4
5 tasks:
6   - name: Crear directorio local para resultados
7     delegate_to: localhost
8     file:
9       path: "./resultados"
10      state: directory
11
12   - name: Obtener nombre del procesador
13     win_shell: "Get-CimInstance Win32_Processor | Select-Object -ExpandProperty Name"
14     register: cpu_info
15
16   - name: Obtener dirección MAC
17     win_shell: "(Get-NetAdapter | Where-Object {$_._Status -eq 'Up'} | Select-Object -First 1).MacAddress"
18     register: mac_info
19
20   - name: Obtener dirección IPv4
21     win_shell: "(Get-NetIPAddress -AddressFamily IPv4 | Where-Object {$_._InterfaceAlias -notmatch 'Loopback|Virtual|VPN'} | Select-Object -First 1).IPAddress"
22     register: ip_info
23
24   - name: Guardar información del equipo
25     delegate_to: localhost
26     copy:
27       content: |
28         Hostname: {{ ansible_hostname }}
29         SO: {{ ansible_distribution }} {{ ansible_distribution_version }}
30         Procesador: {{ cpu_info.stdout | trim | default('No detectado') }}
31         Memoria RAM: {{ ansible_memtotal_mb }} MB
32         IP: {{ ip_info.stdout | trim | default('No detectada') }}
33         MAC: {{ mac_info.stdout | trim | default('No detectada') }}
34         Fecha de recolección: {{ ansible_date_time.date }} {{ ansible_date_time.time }}
35         dest: "./resultados/{{ inventory_hostname }}.txt"
36
```

Explicación:

- Utiliza WMI a través de PowerShell para obtener el modelo exacto del procesador.
- Obtiene la dirección MAC del primer adaptador de red con estado "Up" (conectado).
- Filtra las direcciones IPv4 excluyendo interfaces virtuales, de loopback y VPN, tomando la primera dirección válida.
- Crea un archivo de texto local con formato legible que consolida toda la información recolectada, usando variables de Ansible y resultados de comandos.

9. instalar_apps.yml

Propósito: Instalar aplicaciones de manera remota y simultánea en todos los equipos.

```
instalar_apps.yml > ...
1  ---
2  - name: Instalar todas las aplicaciones desde la carpeta 'archivos'
3    hosts: all
4    gather_facts: no
5
6    tasks:
7      - name: Crear carpeta temporal en el equipo remoto
8        ansible.windows.win_file:
9          path: C:\Temp
10         state: directory
11
12      - name: Obtener lista de instaladores locales (.exe y .msi)
13        find:
14          paths: ./archivos
15          patterns: "*.exe,*.msi"
16          register: lista_instaladores
17          delegate_to: localhost
18
19      - name: Copiar instaladores al equipo remoto
20        ansible.windows.win_copy:
21          src: "{{ item.path }}"
22          dest: "C:\\Temp\\{{ item.path | basename }}"
23          loop: "{{ lista_instaladores.files }}"
24
25      - name: Instalar los archivos .msi en modo silencioso
26        ansible.windows.win_package:
27          path: "C:\\Temp\\{{ item.path | basename }}"
28          arguments: "/qn /norestart"
29          state: present
30          loop: "{{ lista_instaladores.files | selectattr('path', 'search', '.msi$') | list }}"
31          register: resultado_msi
32          ignore_errors: yes
33
34      - name: Mostrar resumen de instalación
35        debug:
36          msg:
37            - "Instalaciones MSI: {{ resultado_msi.results | map(attribute='rc') | list }}"
```

Explicación:

- Busca recursivamente en la carpeta local "archivos" todos los archivos con extensiones .exe y .msi, registrando la lista encontrada.
- Transfiere cada instalador encontrado al directorio C:\Temp del equipo remoto, manteniendo el nombre original del archivo.
- Para cada archivo MSI, ejecuta la instalación con parámetros:
 - /qn: Interfaz silenciosa sin interacción del usuario
 - /norestart: Evita reinicios automáticos