Justin
Ralite

# Automatisation

Basé sur https://github.com/RaliteJ/repo_script_elastic

## Installation de Elastic

Pour installer Elastic je git clone en premier temps le repo de JMP, si celui-ci est deja présent je le remplace au cas ou il y ait une mise à jour.

Ceci fait je lance ensuite la commande:

```
make clean
```

Qui nettoie toutes les installations faites depuis ce repo avec les make.

Je clone ensuite mon repo.

je modifie ensuite le **lance-ES.sh** de **Jean Marc Pouchoulon**.

En effet dans celui-ci la commande **Make curlES** est lancée à la fin, le probleme est que celle-ci m'empeche de continué le script Ansible car le make ne renvoie jamais un signe d'extinction.

Pour se faire j'execute un script qui remplace cette ligne **make curlES** par **exit 0**.

Je modife aussi le **print_password.sh** pour y ajouter une ligne qui permet de placer le MDP elastic dans un fichier.

```
#!/bin/bash

echo "echo \${ELASTIC_PASSWORD} > /opt/siem/repo_script_elastic/pass_elastic"
>> /opt/siem/scripts/print_password.sh

exit 0
```

Celui-ci sera stocké dans une variable avec register dans l'ansible pour l'envoyer dans les requettes de créations de policies etc.

```
---
- name: Install Elastic Security
  hosts: elastic
  tasks:
    - name: Clone siem repository
```

```
    git:
      repo: https://github.com/pushou/siem.git
      dest: /opt/siem
      force: yes
    tags: [clone]

  - name: Clean Siem
    shell: make clean
    args:
      chdir: /opt/siem/
    tags: [siem]

  - name: Modify sysctl.conf
    lineinfile:
      path: /etc/sysctl.conf
      line: 'vm.max_map_count=262144'
    tags: [sysctl]

  - name: Apply sysctl changes
    shell: sysctl -p
    tags: [sysctl]

  - name: Install dependencies
    package:
      name: "{{ item }}"
      state: present
    loop:
      - docker
      - docker-compose
      - jq
    tags: [dependencies]

  - name: Clone siem repository
    git:
      repo: https://github.com/RaliteJ/repo_script_elastic.git
      dest: /opt/siem/repo_script_elastic
      force: yes
    tags: [clone]

  - name: Change Permissions
    shell:
      cmd: chmod +x *
    args:
      chdir: /opt/siem/repo_script_elastic/

  - name: Change SIEM Build
    shell:
      cmd: "./modif_make_es.sh"
    args:
      chdir: /opt/siem/repo_script_elastic/

  - name: Build SIEM containers
    shell: make es
```

```
      args:
        chdir: /opt/siem/
      tags: [siem]

    - name: Start SIEM containers
      shell: make siem
      args:
        chdir: /opt/siem/
      tags: [siem]

    - name: Start Fleet containers
      shell: make fleet
      args:
        chdir: /opt/siem/
      tags: [fleet]

    - name: Change Make Pass
      shell:
        cmd: "./modif_make_pass.sh"
      args:
        chdir: /opt/siem/repo_script_elastic/

    - name: Display elastic user password
      shell: make pass
      args:
        chdir: /opt/siem/
      tags: [password]

    - name: copy_pass
      shell:
        cmd: cat pass_elastic
      args:
        chdir: /opt/siem/repo_script_elastic/
      register: elastic_pass

    - name: Create Elastic Policies DC
      shell:
        cmd: "./create_new_policies.sh DC {{ elastic_pass.stdout }}"
      args:
        chdir: /opt/siem/repo_script_elastic/

    - name: Get Elastic Token DC
      shell:
        cmd: "./grep_enrollment_token_of_policie.sh DC {{ elastic_pass.stdout
}}"
      args:
        chdir: /opt/siem/repo_script_elastic/
      register: tocken_dc

    - name: Add Integration Elastic Search DC
      shell:
        cmd: "./install_elastic_defend.sh DC {{ elastic_pass.stdout }}"
      args:
```

```
        chdir: /opt/siem/repo_script_elastic/

    - name: Add Integration Windows DC
      shell:
        cmd: "./install_windows.sh DC {{ elastic_pass.stdout }}"
      args:
        chdir: /opt/siem/repo_script_elastic/

    - name: Create Elastic Policies SRV
      shell:
        cmd: "./create_new_policies.sh SRV {{ elastic_pass.stdout }}"
      args:
        chdir: /opt/siem/repo_script_elastic/

    - name: Get Elastic Token SRV
      shell:
        cmd: "./grep_enrollment_token_of_policie.sh SRV {{ elastic_pass.stdout
}}"
      args:
        chdir: /opt/siem/repo_script_elastic/
      register: tocken_dc

    - name: Add Integration Elastic Search SRV
      shell:
        cmd: "./install_elastic_defend.sh SRV {{ elastic_pass.stdout }}"
      args:
        chdir: /opt/siem/repo_script_elastic/

    - name: Add Integration Windows SRV
      shell:
        cmd: "./install_windows.sh SRV {{ elastic_pass.stdout }}"
      args:
        chdir: /opt/siem/repo_script_elastic/
```

## Création des Policies

Crée la policy avec l'id et le nom voulu.

```
#!/bin/bash
#2 variables
#$1 is the id of the policy
#$2 is the password of elastic
curl --cacert /opt/siem/ca.crt -k  -X POST
"https://10.202.0.180:5601/api/fleet/agent_policies?sys_monitoring=true" --
header 'kbn-xsrf: true' --header 'Content-Type: application/json'  --data-raw
'{
  "name": "'$1'",
  "id": "'$1'",
  "description": "",
  "namespace": "default",
```

```
  "monitoring_enabled": [
    "logs",
    "metrics"
  ],
  "inactivity_timeout": 1209600
}' -K- <<< "--user elastic:$2"
```

## Ajout d'Intégrations aux Policies

Creer l'intégration Elastic defend ( avec les modification voulues )pour l'id $1 de la policy

```
#!/bin/bash
#Take 2 variables:
#$1: id of the user
#$2: pass of elastic
curl --cacert /opt/siem/ca.crt -k --request POST \
  --url 'https://10.202.0.180:5601/api/fleet/package_policies' \
  -H 'Accept: */*' \
  -H 'Accept-Language: en-US,en;q=0.9' \
  -H 'Connection: keep-alive' \
  -H 'Content-Type: application/json' \
  -H 'Sec-Fetch-Dest: empty' \
  -H 'Sec-Fetch-Mode: cors' \
  -H 'Sec-Fetch-Site: same-origin' \
  -H 'kbn-version: 8.9.0' \
  -d \
'
{
  "name": "Protect",
  "description": "",
  "namespace": "default",
  "policy_id": "'$1'",
  "enabled": true,
  "inputs": [
    {
      "enabled": true,
      "streams": [],
      "type": "ENDPOINT_INTEGRATION_CONFIG",
      "config": {
        "_config": {
          "value": {
            "type": "endpoint",
            "endpointConfig": {
              "preset": "EDRComplete",
              "tls": {
                "verify_peer": false,
                "verify_hostname": false,
                "ca_cert": false
              }
            }
```

```
          }
        }
      }
    }
  ],
  "package": {
    "name": "endpoint",
    "title": "Elastic Defend",
    "version": "8.11.0"
  }
}' -K- <<< "--user elastic:$2"
```

Creer l'intégration Windows pour l'id $1 de la policy

```
#!/bin/bash
#Take 2 variables:
#$1: id of the user
#$2: pass of elastic
curl --cacert /opt/siem/ca.crt -k --user elastic:$2 --request POST \
  --url 'https://10.202.0.180:5601/api/fleet/package_policies' \
  -H 'Accept: */*' \
  -H 'Accept-Language: en-US,en;q=0.9' \
  -H 'Connection: keep-alive' \
  -H 'Content-Type: application/json' \
  -H 'Sec-Fetch-Dest: empty' \
  -H 'Sec-Fetch-Mode: cors' \
  -H 'Sec-Fetch-Site: same-origin' \
  -H 'kbn-version: 8.9.0' \
  -d \
'
{
  "name": "Windows",
  "description": "",
  "namespace": "default",
  "policy_id": "'$1'",
  "enabled": true,
  "inputs": [],
  "package": {
    "name": "windows",
    "title": "Windows",
    "version": "1.43.0"
  }
}'
```

## Création du Fleet Server

Ici le but était seulement de creer un fleet server, pour ce faire, il m'a suffit d'écorcher les scripts de Jean-Marc Pouchoulon dans lequel il creer une policies avec "has_fleet_server"

## Configuration d'Elastic dans Settings

Ici le but était de modifier la configuration d'Elastic pour lui ajouter en OUTPUT le **PRCA** ainsi que le **Fingerprint**

```
#!/bin/bash
#$1 = cafingerprint
#$2 = yaml

ELASTICSEARCH_URL="https://10.202.0.180:9200"
KIBANA_URL="https://10.202.0.180:5601"

ENROLLMENT_TOKEN=$(curl -X POST "${KIBANA_URL}/api/fleet/enrollment-api-keys" \
  --header "kbn-xsrf: true" \
  --header "Content-Type: application/json" \
  --data-raw '{
    "name": "my-enrollment-token",
    "expiration": "1d",
    "type": "output"
  }' | jq -r '.item.api_key')

FLEET_SERVER_CONFIG=$(cat <<EOF
{
  "hosts": ["${ELASTICSEARCH_URL}"],
  "enrollment_token": "${ENROLLMENT_TOKEN}",
  "outputs": [
    {
      "name": "elasticsearch",
      "hosts": ["https://10.202.0.180:9200"],
      "ca_sha256": "'$1'",
    }
  ],
  "elasticsearch_ca_sha256": ["$1"],
  "yaml_configs": ["$2"]
}
EOF
)

FLEET_SERVER_ID=$(curl -X POST "${ELASTICSEARCH_URL}/_fleet/setup" \
  --header "Content-Type: application/json" \
  --data-raw "${FLEET_SERVER_CONFIG}" | jq -r '.id')
```

## Déploiement des Agents

Pour se faire il faut en premier temps récupérer le token enrollment de la policy en question, pour se faire:

```
#!/bin/bash
#2 variables
#$1 is the id of the policy
```

```
#$2 is the password of elastic
curl --cacert /opt/siem/ca.crt -k --user elastic:$2 --request GET --url
'https://10.202.0.180:5601/api/fleet/enrollment-api-keys'    -H 'kbn-xsrf: true'
| jq '.list[] | select(.policy_id == "$1") | .api_key'
```

Puis lancer le script elastic pour DC:

```
---
- name: "Installation Elastic on DC"
  hosts: dc
  gather_facts: false
  tasks:
    - name: "Download Elastic Agent"
      win_shell: |
        Invoke-WebRequest -Uri
https://artifacts.elastic.co/downloads/beats/elastic-agent/elastic-agent-8.9.0-
windows-x86_64.zip -OutFile elastic-agent-8.9.0-windows-x86_64.zip
      args:
        executable: powershell.exe

    - name: "Extract Elastic Agent Archive"
      win_shell: |
        Expand-Archive .\elastic-agent-8.9.0-windows-x86_64.zip -
DestinationPath .
      args:
       executable: powershell.exe

    - name: "Install in Archive"
      win_shell: |
        cd elastic-agent-8.9.0-windows-x86_64
      args:
       executable: powershell.exe

    - name: "Install Elastic Agent"
      win_shell: |
        ..\elastic-agent.exe install --url=https://10.202.0.180:8220 --
enrollment-token=YWVabEZvd0JIaU1GTExNaU02VXg6d2Q4MHZ6OW9UQzZabGJ0Z29MNlNrUQ== -
-insecure -f
      args:
        executable: powershell.exe
```