Learning Connections

school.learningconnections.it
info@learningconnections.it

Introduzione ai linguaggi e agli strumenti per la network automation

Workbook

Panoramica

Learning Connections organizza una serie di webinar ed eventi formativi gratuiti su temi legati alla network automation e al software defined networking.

Questo workbook contiene gli esercizi illustrati dagli istruttori nel corso dell'erogazione dei webinar di livello basic/associate erogati allo scopo di accompagnare gli studenti dei nostri corsi CCNA "tradizionali" nei primi passi verso il mondo della Network Automation.

Repository: https://github.com/Learning-Connections/netdevops-intro

Ultimo aggiornamento: 6 Novembre 2021

Introduzione

Immaginate di dover accedere ad un dispositivo di rete (ad esempio un Cisco IOS-XE) per verificarne lo stato. Come procediamo ? Le risposte più comuni sarebbero “accedere in SSH”, oppure "visualizzare la GUI tramite browser”; alcuni inoltre potrebbero suggerire l’utilizzo di SNMP.

Da oggi proveremo ad accedere ai dati di configurazione, di stato, agli eventi e alle operazioni (RPC) del dispositivo in conformità con il protocollo RESTCONF/NETCONF.

Accedendo in VPN al nostro laboratorio, potrai eseguire interrogazioni RESTCONF utilizzando curl (oppure Postman o altro ambiente di esecuzione in grado di confezionare richieste HTTP) come ad esempio:

curl -k https:// [IP-Addr] /restconf/data/Cisco-IOS-XE-native:native/hostname

E’ questo un ambito applicativo dove si fa utilizzo ad un gran numero di tecnologie ben note agli sviluppatori software ma, quasi sicuramente, meno a chi opera nel settore networking da diversi anni e opera su architetture e utilizza strumenti “legacy”.

Per questo motivo, proponiamo una serie di esercizi guidati di livello base per accompagnare gli studenti nello studio con il consueto approccio “learning by doing”.

Buon divertimento !

Exercise #1

durata: 30 min.

Obiettivi

- ☐ **Codifica Base64**.
- ☐ **Codifica UTF-8.**

Introduzione

La codifica base64 è ampiamente utilizzata per rappresentare con caratteri stampabili ASCII sequenze arbitrarie di byte.

La codifica UTF-8 è lo standard per i linguaggi JSON, YAML, XML, ecc.. E' necessario comprenderne le caratteristiche di base.

Attività:

- ❖ base64

  Convertire in base64 le sequenze binarie { 0xF5, 0xA401, 0x010203 }

  La codifica base64 rappresentante una sequenza di byte può avere una lunghezza arbitraria di cifre ? Ad esempio, è possibile rappresentare una sequenza di byte tramite i codici “abcde”, “abcde=”, “abcde==” ?

- ❖ UTF-8

  Deriviamo la codifica binaria del carattere Unicode ’✉’, corrispondente al code-point U+2709 esadecimale, 9993 in decimale.

Utilizzare il seguente servizio online per verificarne la correttezza:

https://www.cogsci.ed.ac.uk/~richard/utf-8.cgi.

Utilizzare il sito https://hexed.it/ per visualizzare/editare file contenenti caratteri UTF-8 e verificarne la corrispondente rappresentazione esadecimale dei byte.

Exercise #2

durata: 90 min.

Obiettivi

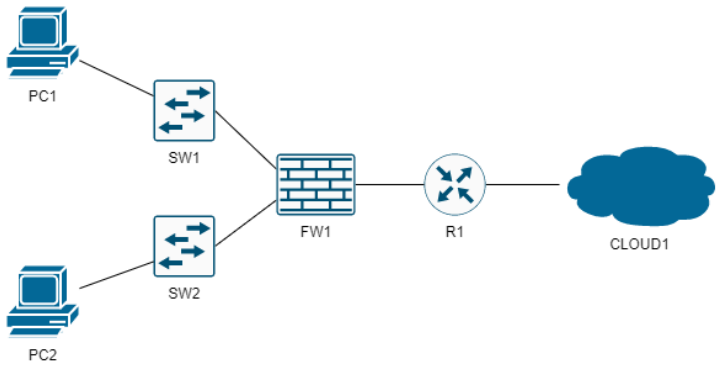
- [ ] **XML**
- [ ] **JSON**
- [ ] **YAML.**

Introduzione

XML, JSON e YAML sono i linguaggi utilizzati per “serializzare” oggetti, ovvero per rappresentarli con una sequenza di caratteri spesso definito “stream”. Differiscono per livello di leggibilità e predisposizione al “parsing” .

Attività:

Si propone la seguente rappresentazione grafica di una topologia di rete LAN:



- rappresentazione XML, JSON e YAML

  Con riferimento alla topologia in figura, produrre tre file di testo nei linguaggi XML, JSON e YAML. Si scelga di rappresentare le informazioni minime.

- Validatori e “Linter”

  Convalidare la **sintassi** dei documenti prodotti utilizzando i seguenti strumenti online:

  https://www.w3schools.com/xml/xml_validator.asp

  https://jsonlint.com/

  http://www.yamllint.com/

- da YAML a JSON

  convertire in JSON gli esempi pubblicati in https://netplan.io/examples/

  verificare la correttezza degli elaborati tramite il sito https://www.json2yaml.com/

Exercise #3

durata: 30 min.

Obiettivi

- ☐ **Git basics**

Introduzione

Git è lo strumento per il controllo di versione più diffuso nella comunità degli sviluppatori. Con esso è possibile gestire repository in forma collaborativa. Questa scheda operativa si propone di mostrare le operazioni di base eseguite da linea di comando.

Attività:

- ❖ Primi passi con Git

  Avviare Git Bash

  configurare lo username e password:

  git config --global user.name "<nome>"

  git config --global user.email "<email>"

  inizializzare un repository a partire da una directory

  git init

  verificare lo stato del repository tramite il comando

  git status

  creare il file vuoto README.md tramite il comando

touch README.md

ripetere il comando ***git status***

aggiungere il file appena creato alla 'staging area'

git add README.md

eseguire il primo commit:

git commit -m "create README.md"

eseguire il comando

git log

ripristinare la versione iniziale del repository

git checkout

annullare quest'ultima operazione tramite :

git checkout -

mostrare l'elenco dei branch: si osservi la presenza dell'HEAD detached

git branch

ripristinare la versione "master"

git checkout master

creare un nuovo branch dal nome "Day1" e mostrare il log

git branch Day1

eseguire delle modifiche ai file ed eseguire il commit

fondere nel branch master le modifiche del Day1

git checkout master

git merge Day1

- Per approfondire le tecniche di branching
  https://learngitbranching.js.org/?locale=it_IT

Exercise #4

durata: 60 min.

Obiettivi

- ☐ **Curl e REST API**

Introduzione

Le API Restful sono ampiamente utilizzate per abilitare una comunicazione “stateless” tra client e server. In questa scheda operativa eseguiremo i primi passi con **curl**, il client HTTP a riga di comando

Attività:

- ❖ Primi passi con curl

  Visualizzare la pagina https://reqres.in/ . Questo sito espone una API Restful utilizzabile a scopi didattici.

  Utilizzando **curl** dalla **Git Bash**, si eseguano delle chiamate API descritte nella pagina.

  Si utilizzi l’autorizzazione Basic (non richiesta dalla API) aggiungendo all’header il parametro seguente:

  ```
  -H "Authorization: Basic TG9MOnN1cGVyc2VjcmV0"
  ```

  Utilizzando il comando **for** della shell Bash, mostrare i primi 10 utenti.

Exercise #5

durata: 60 min.

Obiettivi

- ☐ **Guestshell IOS-XE : setup iniziale**

Introduzione

La guest shell di IOS-XE è un ambiente Linux disponibile in forma di Linux Container (LXC) e dotato di un interprete python che per questo viene definito **'on-box'**.

L'utilizzo principale della guest shell è l'automazione della prima installazione del dispositivo di rete (router o switch con IOS-XE), abilitando così lo Zero-touch Provisioning.

Attività:

- ❖ Setup IOS-XE, abilitazione di IOx e della guestshell

  Una configurazione funzionante è disponibile all'indirizzo:
  https://github.com/Learning-Connections/netdevops-intro/blob/main/guestshell/router-config

  Occore eseguire la configurazione iniziale del router CSR1000v abilitando i seguenti servizi:

  - ☐ AAA
  - ☐ HTTP server
  - ☐ accesso SSH
  - ☐ interfaccia VirtualPortGroup
  - ☐ app-hosting
  - ☐ NAT

Exercise #6

durata: 60 min.

Obiettivi

- ☐ **Eseguire uno script Python dalla guest shell**

Introduzione

Avendo installato Git, possiamo clonare un repository contenente i nostri script di esempio.

Eseguiremo il nostro primo script python per eseguire dei comandi della shell IOS

Attività:

- ❖ Esecuzione di uno script Python tramite guestshell

  Entriamo nella guest shell del router ed eseguiamo il seguente comando:

  git clone https://github.com/Learning-Connections/netdevops-intro/

  qualora fosse già presente una vecchia versione del repository, è possibile rimuoverlo tramite il comando **sudo rm netdevops-intro -Rf**

  Visualizzare lo script netdevops-intro/guestshell/script1.py

  Eseguire lo script tramite il comando

  [guestshell@guestshell ~]$ **python netdevops-intro/guestshell/script1.py loop55**

  Uscire dall'ambiente guest shell per rientrare nella CLI di IOS-XE.

  Eseguire nuovamente lo script tramite il comando:

  Cisco-IOS-XE# **guestshell run python netdevops-intro/guestshell/script1.py loop55**

Exercise #7

durata: 60 min.

Obiettivi

- ☐ **Aggiungere uno script Python ad un'applet EEM**

Introduzione

Embedded Event Manager è un tool ben conosciuto dalla comunità dei network engineer perché consente di eseguire script TCL in risposta a vari eventi, quale una notifica cron, un messaggio syslog, una esecuzione manuale ecc..

Oltre al linguaggio TCL, grazie alla guestshell, è adesso possibile eseguire script in Python

Attività:

- ❖ Setup IOS-XE, abilitazione di IOx e della guestshell

Si vuole eseguire il backup della configurazione del router su un server TFTP ogni qualvolta un utente esegue il login al dispositivo:

Configuriamo la seguente applet EEM:

```
event manager applet myTestApplet
event syslog pattern "Login Success"
action 0.0 cli command "enable"
action 1.0 cli  command "guestshell run python /bootflash/myTest.py"
```

Accedendo alla guestshell, copiare rinominandolo il file netdevops-intro/guestshell/script2.py in /bootflash/myTest.py

Il file myTest.py va editato in modo che punti al server TFTP corretto.

Eseguire il login in SSH al router da un dispositivo remoto e verificare l'avvenuta esecuzione dello script tramite il comando:

show event manager history events