Ansible Windows Guide

# Intro

Ansible?! Hvis du ikke har en ide om, hvad det er, så forklares det som del af det Red Hat Linux trainingsvideo kursus, som jeg varmt anbefaler. I videoen arbejder instruktøren fra en Mac, men det burde være lige så nemt at arbejde med det fra Windows – så jeg har lavet den her guide, så du får en god oplevelse med Ansible fra starten.

# Rationale

Guiden hjælper dig med at komme i gang med at lære Ansible at kende og bruge.

Det er først og fremmest beregnet til at sætte maskiner og containere op på gaming clusteret pcs-gc på en hurtig og nem måde.

Ideen er at bruge Ansible fra sin egen Windows maskine med VPN forbindelse til gaming clusteret, så man kan bruge sin yndlingseditor til at editere Ansible filer med. Den udnytter, at man på Windows 10 kan installere ”Windows Subsystem for Linux” og bruge Linux kommandoer fra sin Windows.

Guiden bygger på træningsvideoer fra RedHat Linux, som er gratis. Deres store Ansible kursus koster i øvrigt ca. 25.000 kr., så det må da være en skill der er værd at lære og efterspurgt! ☺

Det kunne godt gøres nemmere, men jeg synes det er en god ting at sætte det op så man nemt kan arbejde fra sin Windows Workstation og få en god start med Ansible!

# Overblik

Her er en oversigt over de steps du kommer igennem:

1. Windows Subsystem for Linux (WSL) med Ubutu Linux
2. WSL konfigueres til at gemme på elevdrev
3. Ansible installeres i Ubuntu WSL
4. Chocolatey pakke manager til Windows sættes op så du nemt kan installere software
5. Git for Windows og tortoise git installeres så du kan gemme dine projekter i gitlab hvis du vil arbejde i team
6. En demo ansible projektmappe sættes op med kofiguration og ssh nøgler til automatisk login på maskinerne du laver
7. En Ansible playbook skrives, der sætter nogle proxmox linux containere op på clusteret som du så kan øve dig på
8. Til sidst kan du gå i gang med training videoerne fra Red Hat med dine nye maskiner fra din Windows maskine!

# WSL - Windows Subsystem for Linux installation

Det er nemt at få en Ubuntu bash prompt på windows. Men med lidt ekstra kan man konfigurere sin bash prompt til at være lidt smartere!

Hvorfor fx ikke lave backup af din Linux home folder på dit elevdrev? Så har du alle dine Linux indstillinger med, når du skifter PC! Og hvordan får man WSL til at bruge elevdrevet som arbejdsmappe så man kan tilgå filerne fra Windows i stedet for at gemme alt inden i WSL hvor man kun kan bruge bash kommandoer? Let’s built a bridge over troubled water ☺

## Step 0 – er WSL slået til?

WSL ere n Windows feature der skal enables. Det kan være, at det allerede er sket under deployment, men der er en god chance for, at det ikke er tilfældet.

Åbn for en sikkerhed skyld en Powershell som admin (!) og kør

Enable-WindowsOptionalFeature -Online -FeatureName Microsoft-Windows-Subsystem-Linux

Det kan godt være, at din maskine skal genstartes, når featuren ikke er installeret!

## Step 1 – Get connected!

Heldigvis er der allerede noget Powershell script til opgaven af pcsyd server team! Det er gemt på PC-Syds Gitlab git-server.

Hvis du ikke er registreret derinde, så kan du oprette en account med din elevcampus.dk eller eucsyd.dk email adresse. Gå på <https://git.pcsyd.dk/users/sign_in#register-pane> and get it done!!

## Step 2 – Power up with Powershell!

Det script, du skal bruge, ligger på <https://git.pcsyd.dk/ServerTeam/wsl> . Når du er logget ind på gitlab burde du kunne se og downloade det.

Hent skriptet *WSLSetupBackupRestore.ps1* og åbn det i Powershell ISE – ikke som Administrator men som almindelig bruger!!!

Hvis du får problemer med at køre skriptet er det pga Powershell Execution Policy på computeren. Åbn så en Powershell som admin (!) og kør kommandoen

Set-Executionpolicy Bypass

Prøv så igen med wsl skriptet.

Når du kører skriptet viser det en menu, her vælger du ”INSTALL WSL” og så skulle det bare køre derudad!

Det er også en god ide at køre den menupunkt i skriptet der hedder ”SETUP WSL” – så får du bl.a. adgang til dit elevdrev fra din wsl linux!

Som du kan se på menuen kan du også bruge skriptet til at lave en sikring af din hjemmefolder og restore den. Disse funktioner bliver automatisk gemt i din WSL installation når du kører ”SETUP WSL” fra menuen. Så kan du senere køre ”wslbackup” og ”wslrestore” direkte fra din Linux prompt!

# WSL konfiguration

Som standard laver WSL en home folder som på en Linux maskine – gemt inde i WSL filsystemet. Det skal man bestemt ikke tilgå fra Windows (hvis man finder det), det ødelægger bare filerne for Linux!!

Skriptet der blev brugt til wsl installation sørgede istedet automatisk for, at dit elevdrev/hjemmedrev bliver monteret i din wsl linux hver gang den starter. Den sætter også et link op, så du kan tilgå dit elevdrev fra folderen ~/winhome.

Med kommandoen **cd** **~/**winhome skulle du gerne komme derind. Med **ls** **~/**winhome burde du kunne se indholdet af dit elevdrev. Prøv det nu!

# Ansible installation i WSL

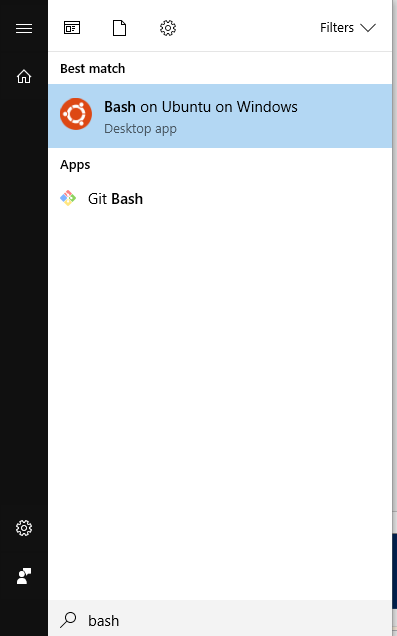
Nu skal der installeres Ansible!

Der er flere måder at gøre det på, men de to mest populære er at bruge pakkemanageren i Linux (apt-get osv.) eller at bruge Python programmeringssprogets ”pip”-pakkemanger.

Windows-programmører kender den tilsvarende ”nuget” i Visual Studio, ”pip” er noget lignende bare til Python pakker.

Her bruges ”pip” metoden, så får du altid den nyeste version af ansible!

Åbn din WSL Linux ved at åbne ”Bash” fra Windows menuen.



I bash kører du følgende kommandoer:

# opdater linux pakke fortegnelsen

**sudo** apt-get update

# installer python tools

**sudo** apt-get **-**y install python-setuptools

# installer pip

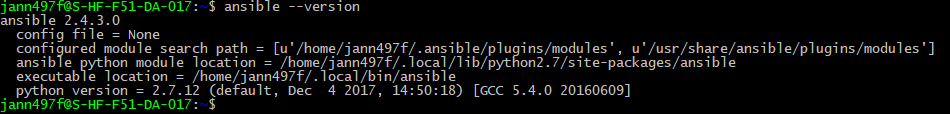
sudo -H easy\_install pip

# installer ansible med pip

sudo -H pip install ansible

# test ansible

ansible –-version



Voila! Ansible er klar til brug!

Hvis du har lyst, kan du selvfølgelig opdatere din WSL Linux som enhver ubuntu med apt-get upgrade, men det behøver du ikke for ansible.

# Chocolatey installation

Du skal også bruge en god tekst editor og en git klient hvis du gerne vil bruge gitlab.

Du kan selvfølgelig downloade installationspakker fra hjemmesiderne for de enkelte programmer, men der er en nemmere metode!

Den hedder ”Chocolatey package manager for Windows”, lidt som ”apt-get install”, bare for Windows programmer ☺ Mere info på <https://chocolatey.org/> hvis du er interesseret!

For at installere chocolatey på din Windows pc skal du åbne en Powershell – denne gang som Administrator og følge vejledning på <https://chocolatey.org/install> - du skal bare kopiere en lang string i en admin powershell og trykke enter ☺

OBS: De har også en metode til cmd, men det kan give problemer med F-Secure Firewall. Så brug Powershell, det virker!

# Windows programmer installation

Når du har installeret Chocolatey kan du bruge den til nemt at installere programmer. Du finder en oversigt på <https://chocolatey.org/packages> - Det ligner Ninite installer, bare bedre!

Her bruger du chocolatey til at installere nogle praktiske programmer:

* Sublime teksteditor (det er den de bruger i Red Hat videoerne)
* Notepad++ teksteditor (så har du noget at vælge imellem) – den vil vi gerne have som 32 bit version, det kræver lidt ekstra argumetation med chocolatey 😊
* Git for Windows og Tortoise Git Gui

Kør koden i en admin-cmd eller admin-powershell:

cinst sublimetext3 git git-lfs tortoisegit

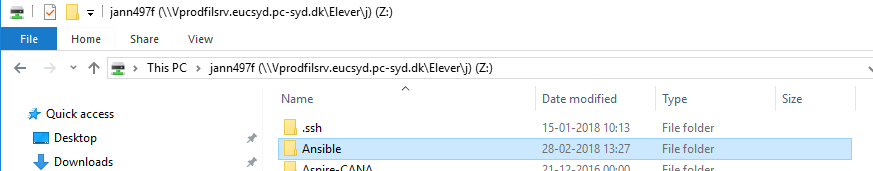
cinst -x86 notepadplusplus

choco feature enable -n useRememberedArgumentsForUpgrades

# Ansible training mappe setup

Lav en mappe I dit elevdrev til dine ansible projekter, så du kan tilgå den fra din WSL Linux. Min mappe hedder så ~/winhome/Ansible i WSL.

For eksempel:



Nu kan du bruge din yndlings teksteditor i Windows og samtidig bruge Linux kommandoer som ansible på det du har skrevet. Smart!

# Ansible container setup

Nu har du et sted du kan lægge dine Ansible projekter.

Til RedHats training skal du bruge nogle maskiner der kører Centos Linux. Dem kan du oprette som containere på gaming clusteret på forhånd, så du er klar. Der ligger en container template inde på clusteret til centos 7.

* Når du laver containerne, så brug defaults til cpu og ram, det betyder ikke noget!
* Brug vlan 10 til netværk, der er dhcp, så sparer du ip opsætning – husk at notere ip adresserne ☺
* Find selv ud af, hvordan du åbner for ssh forbindelser på en CentOS ☺ Ellers bruger du bare Ubuntu containere i stedet for!

Det er den samme linux de bruger i RedHats training videoer, så når du har afsluttet denne del ved du måske ikke hvad kommandoerne gør, men du er klar til at følge med i Redhats training hands on!

* Hvis du bestemmer dig for at bruge Ubuntu, skal du bare huske at pakke-manageren hedder ”yum” på RedHat og CentOS og ”apt” på Debian og Ubuntu Linux!

Sørg for at du har adgang med ssh til 3-5 linux containere inden på clustret inden du går i gang!

# Kom I gang med Red Hat training!

Du kan finde RedHat intro til ansible på:

<https://www.redhat.com/en/services/training/do007-ansible-essentials-simplicity-automation-technical-overview>

Du skal registrere dig, men det er gratis at bruge.

Andre gode websteder:

* Ansible hjemmeside: <https://www.ansible.com/>
* Ansible dokumentation: <http://docs.ansible.com/>
* Ansible modul indeks: <http://docs.ansible.com/ansible/latest/modules_by_category.html>

God fornøjelse!

# Prøv git versionssystem til dit projekt

Nu hvor du sandsynligvis har lavet nogle ansible filer, har du måske lyst til at lægge dem op på gitlab serveren, så andre kan være med. Her er en lille guide til at komme i gang.

WORK IN PROGRESS!

# Prøv ansible med csgo demo

Nu hvor du kan (eller i det mindste har prøvet) både ansible og git, hvorfor ikke lave en ansible rolle/role til dit spil? Så du kan give den rolle til en server eller container, og så får den automatisk installeret alt hvad der skal til for at køre en spilleserver. Husk at lægge dit resultat i gitlab, så andre kan klone det du har lavet!

Hvis du gerne vil se et eksempel, ligger der et i mappen "ansible-lgsm-demo". Læg mappen over i din ansible projektmappe og tilgå den i WSL bash. Du kan køre den med følgende kommando - husk, at du skal være i den mappe, hvor playbooken ligger!!

ansible-playbook csgoserver.yml

Demoen laver det hele for dig:

* Den opretter tre containere - en på hver gaming server node - som det står beskrevet i "hosts" filen. Du eller ansible kan bagefter logge ind på containeren med den ssh-nøgle, som ligger i "ssh" mappen. Smart, ikke?
* Den henter og installerer csgoserver med lgsm, som det står beskrevet på lgsms hjemmeside.

Hvis du mangler en udfordring, prøv at kigge på koden - lige nu ligger alt i en playbook.

* Kan du lave det om til en ansible rolle, der kan installere csgoserver på en hvilken som helst maskine?
* Prøv også at lave playbooks og senere roller til andre spil, som kan installeres med lgsm. Hvor meget ram, cpu og diskplads skal man regne med? Hvad er realistisk - ikke bare utrolig stor?
* Kan du tilføje et mount-point til containeren under oprettelse, så den får adgang til en folder under /mnt/pve/gamedata på clusteret, så spilleserverne kan være fælles om nogle filer?
* Kan du få ansible til at lægge en custom konfigurationsfil til spillet ind i containeren?
* Kan du måske få ansible til at lave konfigurationsfilen til dig med "template:" modulet?

Til sidst kan du køre playbooken "destroy.yml", så rives alle demo-containere ned igen og slettes fra clusteret. Kør den med "ansible-playbook destroy.yml" - også hvis du vil starte forfra"

HINT: hvis "dine" containere ikke skal hedde det samme som de andres, der prøver demoen, kan du bare ændre navnene i "inventory/hosts" filen!

HINT: I hosts filen kan du bruge jokertegn til at lave mange maskiner, fx laver "csgoserver[01:03]" tre container der hedder csgoserver01, csgoserver02 og csgoserver03 automatisk. Og der er mange flere jokertegn, man kan bruge!