## Go Migration Windows nach Ubuntu Blog

### Ausgangslage

Der HTTP Server stack.ch basiert auf der Programmiersprache Go (Golang) und wird seit 2017 auf einer Windows 32-Bit Plattform betrieben. Mit der zunehmenden Datenmenge soll auf eine Linux Ubuntu 64-Bit Umgebung migriert werden. Unter Windows wurde der Server als Windows Service betrieben. Die Registrierung als Windows Service ist direkt im Go Executeable integriert. Unter Unix soll das gleiche native kompilierte Programm als Unix Service systemctl Daemon installiert und betrieben werden.

### Native Compiled

Go bietet die native Kompiliation für die meisten gängigen Plattformen. Die Komplation erfolg in Abhängigkeit der Go Umgebungsvariablen GOARCH und GOOS. Das Windows 32-Bit Programm wird wie folgt native kompiliert:set GOARCH=386
set GOOS=windows
go build -o stack.ch.windows.x86.exe
set GOARCH=amd64
go build -o stack.ch.windows.x64.exe
Nach der erfolgreichen Kompilation prüfen wir wie folgt die Version unter Windows:stack.ch.windows.x86.exe -version

stack.ch Version 1.8.17Das Windows Executeable wird wie folgt unter Windows als Service installiert:stack.ch.windows.x86.exe -s -install
Das Go Programm native kompiliert für Unix basiert auf den folgenden Umgebungsvariablen:set GOARCH=386
set GOOS=linux
go build -o stack.ch.linux.x86
set GOARCH=amd64
go build -o stack.ch.linux.x64
Nach der erfolgreichen Kompilation prüfen wir wie folgt die Version unter Ubuntu:./stack.ch.linux.x64 -version

stack.ch Version 1.8.17Wir haben also aus dem gleichen Programmcode das Programm unter Windows und unter Ubuntu kompiliert und ausgeführt. Bis jetzt hat alles einwandfrei funktioniert.

### Ubuntu Service

Das Go Programm installieren wir unter Ubuntu als systemctl Service. Hierzu erstellen wir die Datei /etc/systemd/system/stack.ch.service wie folgt:[Unit]
Description=stack.ch server
After=network.target auditd.service
[Service]
User=to be defined
Group=to be defined
ExecStart=/srv/stack.ch/stack.ch.linux.x64
KillMode=process
Restart=on-failure
StandardOutput=null
Restart=on-failure
Type=simple
WorkingDirectory=/srv/stack.ch
[Install]
WantedBy=multi-user.target
Alias=stack.ch.service
Bevor wir das Programm ausführen können erlauben wir dem Program das öffnen der Ports < 1024 wie folgt:sudo setcap 'cap_net_bind_service=+ep' /srv/stack.ch/stack.ch.linux.x64Nun aktivieren wir den systemctl Service wie folgt:sudo systemctl daemon-reload
sudo systemctl enable stack.ch.service
sudo systemctl start stack.ch.service
sudo systemctl stop stack.ch.service
Der HTTP Server funktioniert einwandfrei. Vorher mussten wir aber einige Anpassungen am Programm und an den Dateien vornehmen.

### Probleme

Das folgende Listing zeigt die Probleme und Lösungen bei der Migration von Windows nach Ubuntu Unix auf:Unterschiedliche Fehlercodes bei os.Stat zwischen Windows und UnixKonsequenter Einsatz von filepath.SeparatorVerzeichnis- und File-Pfad AnpassungenFilesystem Case Sensitive AnpassungenAllgmein lohnt es sich, wenn man den Zugriff auf die os-Methoden kapselt und damit zentralisiert. So wurde z.B. die Existenzabfrage eines Pfades in die Methode PathExists ausgelagert:func PathExists(path string) bool {
  if _, err := os.Stat(path); err != nil {
    if os.IsNotExist(err) {
      return false
    }
  }
  return true
}Das grösste Problem waren die falsch definierten File Extensions z.B. für Bilder. Schreiben Sie alle Extensions konsequent Lowercase.

### Fazit

Go hält was es verspricht. Mit einem doch eher kleinen Aufwand konnten wir ein Programm mit dem gleichen Programmcode unter verschiedenen Plattformen betreiben und dies native kompiliert.Unser Urteil: phänomenal

**Feedback**

War dieser Blog für Sie wertvoll. Wir danken für jede Anregung und Feedback

## Kontakt

Simtech AG
Finkenweg 23
3110 Münsingen
Schweiz

## Impressum

Das Copyright für sämtliche Inhalte dieser Website liegt bei Simtech AG, Schweiz.
Beachten Sie auch unsere Hinweise zum Urheberrecht, Datenschutz und Haftungsauschluss.
Jeder Hinweis auf Fehler nehmen wir gerne entgegen.

## Copyright

2024 Simtech AG, All rights reserved, Powered by stack.ch written in Golang by Daniel Schmutz

## https://www.simtech-ag.ch/konsequent