Network Automation Showdown: Go vs. Python

NANOG 90 – February 13th, 2024
Moderator: Cat Gurinsky

Panelists:
Brandon Bennett, Roblox
Ryan Hamel, i3D.net
Daniel Hertzberg, Arista Networks
Frank Seesink, UNC Chapel Hill
Claus Töpke, Telcomanager Technologies
Go vs. Python

The goals of this panel
What we’ll cover

For each language we’ll discuss:
• Pros & Cons
• What the language excels at
• What the language struggles with
• What modules / libraries exist for network purposes
• Who should consider using it and why
Go vs Python quick comparison

**Python:**
- Ecosystem: lots of special libraries
- Learning Curve: more intuitive for beginners
- Dynamically Typing: streamlines the coding process

**Go:**
- Compiled Nature: Simplifies deployments
- Concurrency: great performance at scale
- Statically Typed: more predictable with upfront declarations
- Error Handling: proactive approach for better resilience

Taken from: [https://www.packetcoders.io/python-vs-go-for-network-automation/](https://www.packetcoders.io/python-vs-go-for-network-automation/)
Static vs. Typed
Interpreted vs. Compiled

- **Dynamic typing**: Used by Python, type checking happens at runtime. Types don’t have to be specified.
- **Static typing**: Used by Go, type checking happens when compiling. Types should be specified.
- **Interpreted Language**: Python, the source code of a program is converted into bytecode that is then executed by the interpreter.
- **Compiled Language**: Go, converted directly into machine code that the processor can execute, stand alone and the resulting binary doesn’t require installing dependencies.
Concurrency & Parallelism

• CPython GIL (Global Interpreter Lock)
  • Limited to a single core (work being done in PEP703)
  • Threads
  • Multiprocessing
  • concurrent.futures
  • asyncio
    • Coroutines
• Goroutines are not the same as coroutines
  • Green thread based scheduler
  • Can be spread across cores

https://docs.oracle.com/cd/E36784_01/html/E36868/mtintro-6.html
Performance

• At what scale does performance matter?
Easy vs Simple

• Python is easy. Go is simple.
  Simple is not easy.
• Python → Go cheat sheets

```go
type CityTemperature struct {
    City     string
    Temp float64
}

// ...

temperatures := []CityTemperature{
    {"City1", 19},
    {"City2", 22},
    {"City3", 21},
}

filtered_temps := {
    entry["city"] == entry["temp"] for entry in temperatures if entry["temp"] > 20
}
```

https://preslav.me/2023/11/27/python-is-easy-golang-is-simple-simple-is-not-easy/
Deployments & Dependencies

- **Python**
  - Plenty of tool chain based helpers
  - Jupyter notebooks
  - REPL: Read-Eval-Print-Loop
  - Requirements
  - Virtual Environments

- **Go**
  - No external dependencies
  - After compilation it’s a single binary
  - Can cross compile for other OS
  - Built in unit testing
  - Formatting
  - Typing
Network libraries Go vs Python

- **Python**
  - [Paramiko](https://paramiko.readthedocs.io/) (SSH)
  - [Netmiko](https://github.com/rap2hpoutre/Netmiko) (SSH network devices)
  - [Nornir](https://nornir.readthedocs.io/) (automation framework)
  - [NAPALM](https://napalm.readthedocs.io/) (Network Automation and Programmability Abstraction Layer with Multivendor support)
  - [pyGNMI](https://github.com/napalm-extras/pygnmi)
  - [netaddr](https://github.com/rap2hpoutre/netaddr)

- **Go**
  - Openconfig Go Modules ([yGOT](https://github.com/ygot/ygot), [yGNMI](https://github.com/ygot/ygnmi), [gRIBI](https://github.com/ygot/gribi), [gNMI](https://github.com/ygot/gnmi), [goYANG](https://github.com/ygot/goyang))
  - [goBGP](https://github.com/go-bgp)
  - [Netaddr](https://github.com/rap2hpoutre/netaddr)
  - [Prometheus](https://github.com/prometheus)

---

NANOG
## Dev Time vs. Execution Time

<table>
<thead>
<tr>
<th>Language</th>
<th>Development Time</th>
<th>Execution Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>Assembler</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Go</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Python</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Thank you

13-FEB-2024
Resources

• Python -> Go Cheat Sheet Examples:
  • https://www.353.solutions/py2go/index.html
• Getting started with Go tutorial
  • https://go.dev/doc/tutorial/getting-started
• Getting started with Python
  • https://www.python.org/about/gettingstarted/