P2P Systems and Security - Initial Report
Team Onion-16
Team name: hAcKiNg_tHe_mAinFrAmE
Module: Onion (39943)

Langer, Valentin
valentin.langer@tum.de

Müller, Till
till.mueller@tum.de

15. May 2020

1 Languge and Operating System Choice

1.1 Language

We have chosen to use the Go [1] language for our implementation. This enables us to create an efficient implementation of our module without having to worry about garbage collection. Go also provides a solid standard library as a starting point for our endeavors, as well as an easy way of dealing with concurrency, which is especially relevant networking domain for creating high-throughput, functional software. Lastly, Google has provided excellent learning resources for new Go developers, allowing us to escape a lot of the headache of learning new languages.

1.2 Operating System

The operating system we have decided to use to base our system on is *GNU/Linux*. We will try to maintain compatibility with many distributions as possible, but might resort to using Docker [10] to maintain portability between differing systems.

Since both of us have significant experience working with *GNU/Linux* and are aware of its role within the application hosting and software development world, this decision was an easy one to make. Of course, the fact that it is free open-source software (FOSS) was also counted in its favor, as well as the good integration *GNU/Linux* has with the existing Go environment.

2 Build System

We will mainly focus on using the build tools included with Go. As mentioned, Docker could also be used to provide a consistent environment for building the application. Apart from that we do not expect to require further build tools to compile our software since we do not intent to stray too far away from the default libraries and support setups.

3 Measures to Guarantee Quality

For testing and quality assurance purposes, we will base our testing suite on the tools provided by the Go language itself [5]. After a brief read through their documentation, these tools should be sufficient for testing all functionality where we want to ensure they are working properly.

4 Helper Libraries

As part of our objective of keeping our code simple to compile and compatible to as many systems as possible, we will try to use as few non-standard libraries as possible. From the standard libraries [9], we plan on using these packages:

Library name	Use case within our application
net	networking tasks
crypto	cryptography functions and hashing
bytes	manipulating byte slices
time	for time-keeping tasks / timestamps
sys	provides syscall functionalities
text	formatting and working with strings for output

Apart from these we also intend to use the `ini` package for Go [8]. This is required to read from and write to configuration files as defined by the specification.

5 Licensing

We will use the MIT [3] license, since our only motivation for writing this software is learning about peer to peer architectures, completing this course, and potentially showcasing the source code to interested parties. As opposed to some copyleft licenses [4], it also frees us from any additional responsibilities like having to send the source code when prompted.

6 Previous Experience

Both of us have experience with low-level languages (C) and networking (Grundlagen: Rechnernetze und verteilte Systeme [GRNVS] lecture [6] and iLab1 [7]).

The basic concepts of multi-threading were a part of Praktikum: Grundlagen der Programmierung (PGdP) and reiterated in the Operating Systems course.

7 Planned Workload Distribution

We intend to split up the work into parts of equal effort, and will discuss and decide on architectural decisions together. To reduce errors in a langauge we are both not yet well versed in, we will use pair programming extensively. This will also ensure that we are always on the same page in terms of understanding the control flow of the program.

References

[1] `https://golang.org/`

[2] `https://ubuntu.com/`

[3] `https://opensource.org/licenses/MIT`

[4] `https://www.gnu.org/licenses/agpl-3.0.de.html`

[5] `https://golang.org/pkg/testing/`

[6] `https://grnvs.net.in.tum.de/`

[7] `https://ilab.net.in.tum.de/`

[8] `https://github.com/go-ini/ini`

[9] `https://golang.org/pkg/#stdlib`

[10] `https://www.docker.com/`