School of Computation, Information and Technology
Technische Universität München

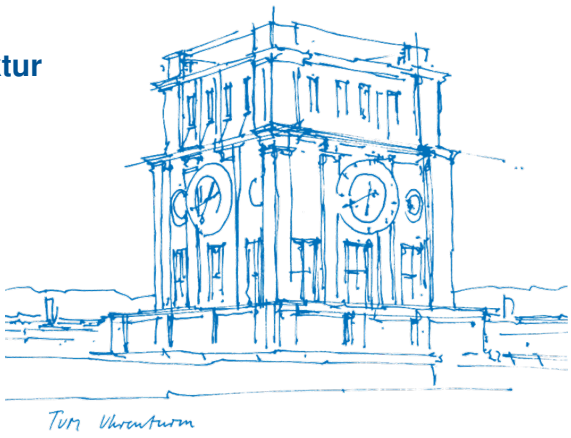
Übung 02: Sichere Programmierung und Nutzereingaben

Grundlagenpraktikum Rechnerarchitektur

Niklas Ladurner

School of Computation, Information and Technology
Technische Universität München

27. April 2024



Keine Garantie für die Richtigkeit der Tutorfolien: Bei Unklarheiten/Unstimmigkeiten haben VL-Folien Recht!

Organisatorisches

- Präferenzabgabe zur Wahl des Vertiefungszweigs auf Artemis verfügbar
- Syntaxtest zeigt an, ob Abgabe im korrekten Format
- Präferenzen von 1-3
- Es ist nicht garantiert, dass ihr eure erste Wahl bekommt!
- 3er-Gruppen, Gruppen mit randoms sind nicht zu empfehlen!

Sichere Programmierung

- häufige Fehler:
  - Buffer Overflows
  - Memory Leaks
  - Zugriff auf ungültige Adressen
- Undefined Behavior: Verhalten bei nicht im C-Standard definiertem Code
  - Double free bzw. use after free
  - Schreibzugriff auf string literal
  - Division durch 0
  - Lesen uninitialisierter Variablen
  - ...
- Hilfreich: `fsanitize=address`, `-fsanitize=leak`, `-fsanitize=undefined`
- Achtung: Sanitizer bringen erhebliche Performanceeinbußen mit sich, nicht in production verwenden!

Kommandozeilenargumente

- `argc`... Anzahl Argumente, `argv`... String-Array leerzeichengetrennter Argumente
- `getopt` für effiziente Verarbeitung von kurzen Argumenten (bspw. `-h`)
- `getgetopt_long` für lange Argumente (bspw. `--help`).
  Achtung: kann bei falscher Verwendung sehr leicht zu einem SEGFAULT führen

Fragen?

Links

- Zulip: „GRA Tutorium - Gruppe 20“ bzw. „GRA Tutorium - Gruppe 22“
- GRA-Artemis-Kurs
- gcc Program Instrumentation Options

School of Computation, Information and Technology
Technische Universität München

Übung 02: Sichere Programmierung und Nutzereingaben

Grundlagenpraktikum Rechnerarchitektur

Niklas Ladurner

School of Computation, Information and Technology
Technische Universität München

27. April 2024