Minutes of the oral examination of the lecture
Type Systems for Correctness and Security

Examiner	Dr.-Ing. Ben Hermann	Date:	21.08.2019
Course of studies:	M. Sc. Computer Science	Duration:	30 min
Language:	English	Grade (optional):	1.0

Preparation time: In total it would be roughly 4 weeks in addition to attending the exercises and doing the labs.

Literature: Lecture slides and exercises. In rare cases I looked things up in the referred literature.

Atmosphere & Further information: The atmoshpere was good and quite relaxed – at least for an exam ;)

Before starting the actual exam, Mr. Hermann shortly explained the procedure. He also explained that in some cases it may be easier to write down formulas instead of explaining things. Most of the time I was able to express my answers in a succint way.

Also noteworthy is that, while the exam contained some formal parts, writing things done was quite informal. In some cases my notation slightly diverted from the notation we used in the lecture but it was ok "as long as everybody understands it".

The exam was planned to take 45 minutes. However, we went quite fast and were done after 30 minutes.

The notes below only contain some answers. Don't want to present you with too many spoilers

---

Process of the exam:

- Why do we need type system?
- Evaluation rules for the language Bool (E-If, E-If-True, E-IF-False) presented on paper. We want to add natural numbers – can terms now get stuck? Where and why?
  - Yes. We would need to introduce typing rules to make it safe.
  - Had to define T-If. Used the simple definition with $T_2 == T_3$ and mentioned the join-variant.
- What would be now need to do to actually show that this is safe?
  - Proof Safety = Preservation + Progress. Induction on the structure of terms t using the typing and evaluation rules.
- What other properties can a typing system assure?
  - Control-flow and memory safety (TAL), access rights (security type systems)
- What is uniqueness? When did we drop it and why?
  - Types were unique until we introduced subtyping ($\rightarrow$ subtypes can take the place of their supertypes).
- Talked about joins before. Extend T-If to use joins. What are joins?
- How would you proof this subtype relationship?

  - Given two records on a paper. Before drawing the derivation tree I explained what rules to use and the tree was kept quite small

- What is the lambda calculus, What does it consist of?
  - Abstractions (function definitions) and applications (applying functions to values).
- What are extensions? How do they work?
  - Syntactic sugar that can be converted to the unextended form.
- After extensions we introduced references. What are they and why did we need them?
- Here are some reference rules (ref t_1, !$t1$, $t_1 := t_2$). Create typing rules for these.
- Switch to security type system. What is non-inference? Extend T-If with security typing.
  - Converted types to type tuples, e.g. (T_1, s_1). Explained that s_1 is a security level and how it works (implicitely used the security lattice; only high-low was required).
- Switch to Featherweight Java. What does this rule mean? (T-Invk)
- Here are all FW-Java term typing rules. Pick one and add security typing.
  - Picked T-Field. Added some security types, similar to the T-If extension.
- Does the security level of the class have any impact on field's security level?
  - It depends on how we define it. We can require $s_{class} \geq s_{field,i}$ for all i and extend the rule accordingly.
- Switch to TAL. Presented the semantic rules of TAL-0. We do *not* have typing rules yet. Where do we need to take care and why?
  - Every rule with jump is dangerous. We need to remember valid locations. Got confused with If-Neq and If-Eq, even if I explained it correctly (pointed to If-New which has no jump).
- How would we make it safe?
  - #typesystemsRule
- In TAL-1 there are special pointers s_p. What are they, why are they needed?
  - Shared pointers. Freely assignable. Ensure memory safety (type of a pointer is invariant).
- What are higher-order type systems? How are they used? What is Kinding?
- What are dependent types?
- Here is rule T-Conv. Why do we need it.
  - Generics.