UNIVERSITÄT
DES
SAARLANDES

Faculty of Mathematics and Computer Science

Module Descriptions

Cybersecurity MSc

6th March 2024

List of module categories and modules

Module Category 1

Core Lectures (Stammvorlesungen)

Algorithms and Data Structures — AlgoDat

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Kurt Mehlhorn

lecturers Prof. Dr. Raimund Seidel
Prof. Dr. Kurt Mehlhorn

entrance requirements For graduate students: C, C++, Java

assessments / exams
- Regular attendance of classes and tutorials
- Passing the midterm and the final exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

The students know standard algorithms for typical problems in the area's graphs, computational geometry, strings and optimization. Furthermore, they master a number of methods and data-structures to develop efficient algorithms and analyze their running times.

content

- graph algorithms (shortest path, minimum spanning trees, maximal flows, matchings, etc.)
- computational geometry (convex hull, Delaunay triangulation, Voronoi diagram, intersection of line segments, etc.)
- strings (pattern matching, suffix trees, etc.)
- generic methods of optimization (tabu search, simulated annealing, genetic algorithms, linear programming, branch-and-bound, dynamic programming, approximation algorithms, etc.)
- data-structures (Fibonacci heaps, radix heaps, hashing, randomized search trees, segment trees, etc.)
- methods for analyzing algorithms (amortized analysis, average-case analysis, potential methods, etc.

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Artificial Intelligence — AI

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Jörg Hoffmann

lecturers Prof. Dr. Jörg Hoffmann

entrance requirements *Programming 1*, *Programming 2*, *Fundamentals of Data Structures and Algorithms*, and *Elements of Machine Learning* or other courses in machine learning are recommended.

assessments / exams
- Regular attendance of classes and tutorials
- Solving of weekly assignments
- Passing the final written exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from the performance in exams. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

Knowledge about basic methods in Artificial Intelligence

content

Search:

- Uninformed- and informed search procedures
- Monte-Carlo tree search

Planning:

- Formalism and complexity
- Critical-path heuristics
- Delete relaxation heuristics
- Abstraction heuristics

Markov decision processes:

- Discounted reward and expected cost
- Value iteration
- Informed search
- Reinforcement learning

Games:

- Adversarial search
- Learning from self-play

literature & reading

Russel & Norvig Artificial Intelligence: A Modern Approach;
further reading will be announced before the start of the course on the course page on the Internet.

Audio/Visual Communication and Networks AVCN

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr.-Ing. Thorsten Herfet

lecturers Prof. Dr.-Ing. Thorsten Herfet

entrance requirements Solid foundation of mathematics (differential and integral calculus) and probability theory. The course will build on the mathematical concepts and tools taught in TC I while trying to enable everyone to follow and to fill gaps by an accelerated study of the accompanying literature. *Signals and Systems* as well as *Digital Transmission and Signal Processing (TC I)* are strongly recommended but not required.

assessments / exams Regular attendance of classes and tutorials Passing the final exam
Oral exam directly succeeding the course. Eligibility: Weekly excersises / task sheets, grouped into two blocks corresponding to first and second half of the lecture. Students must provide min. 50% grade in each of the two blocks to be eligible for the exam.

course types / weekly hours

```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload

```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Final Exam Mark

language English

aims / competences to be developed

AVCN will deepen the students' knowledge on modern communications systems and will focus on wireless systems.

Since from a telecommunications perspective the combination of audio/visual data – meaning inherently high data rate and putting high requirements on the realtime capabilities of the underlying network – and wireless transmission – that is unreliable and highly dynamic with respect to the channel characteristics and its capacity – is the most demanding application domain.

content

As the basic principle the course will study and introduce the building blocks of wireless communication systems. Multiple access schemes like TDMA, FDMA, CDMA and SDMA are introduced, antennas and propagation incl. link budget calculations are dealt with and more advanced channel models like MIMO are investigated. Modulation and error correction technologies presented in Telecommunications I will be expanded by e.g. turbo coding and receiver architectures like RAKE and BLAST will be introduced. A noticeable portion of the lecture will present existing and future wireless networks and their extensions for audio/visual data. Examples include 802.11n and the terrestrial DVB system (DVB-T2).

literature & reading

Will be announced before the start of the course on the course page on the Internet.

additional information

This module was formerly also known as *Telecommunications II*.

Automated Reasoning AR

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Christoph Weidenbach

lecturers Prof. Dr. Christoph Weidenbach

entrance requirements *Introduction to Computational Logic*

assessments / exams
- Regular attendance of classes and tutorials
- Weekly assignments
- Practical work with systems
- Passing the final and mid-term exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

The goal of this course is to provide familiarity with logics, calculi, implementation techniques, and systems providing automated reasoning.

content

Propositional Logic – CDCL, Superposition - Watched Literals
First-Order Logic without Equality – (Ordered) Resolution,
Equations with Variables – Completion, Termination
First-Order Logic with Equality – Superposition (SUP) - Indexing

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Compiler Construction — CC

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Sebastian Hack

lecturers Prof. Dr. Sebastian Hack

entrance requirements For graduate students: none

assessments / exams
- Regular attendance of classes and tutorials
- Written exam at the end of the course, theoretical exercises, and compiler-laboratory project.
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

The students learn, how a source program is lexically, syntactically, and semantically analyzed, and how they are translated into semantically equivalent machine programs. They learn how to increase the efficiency by semantics-preserving transformations. They understand the automata-theoretic foundations of these tasks and learn, how to use the corresponding tools.

content

Lexical, syntactic, semantic analysis of source programs, code generation for abstract and real machines, efficiency-improving program transformations, foundations of program analysis.

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Complexity Theory CT

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Markus Bläser

lecturers Prof. Dr. Raimund Seidel
Prof. Dr. Markus Bläser

entrance requirements undergraduate course on theory of computation (e.g. *Grundzüge der Theoretischen Informatik*) is highly recommend.

assessments / exams
- Regular attendance of classes and tutorials
- assignments
- exams (written or oral)

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be calculated from the results in the assignments and/or exams, as announced by the Lecturer at the beginning of the course

language English

aims / competences to be developed

The aim of this lecture is to learn important concepts and methods of computational complexity theory. The student shall be enabled to understand recent topics and results in computational complexity theory.

content

Relation among resources like time, space, determinism, nondeterminism, complexity classes, reduction and completeness, circuits and nonuniform complexity classes, logarithmic space and parallel complexity classes, Immerman-Szelepcsenyi theorem, polynomial time hierarchy, relativization, parity and the polynomial methods, Valiant-Vazirani theorem, counting problems and classes, Toda's theorem, probabilistic computations, isolation lemma and parallel algorithms for matching, circuit identity testing, graph isomorphism and interactive proofs.

literature & reading

Arora, Barak: Computational Complexity – A Modern Approach, Cambridge University Press
Oded Goldreich: Computational Complexity – A Conceptual Approach, Cambridge University Press
Dexter Kozen: Theory of Computation, Springer
Schöning, Pruim: Gems of Theoretical Computer Science, Springer

Computer Algebra CA

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Frank-Olaf Schreyer

lecturers Prof. Dr. Frank-Olaf Schreyer

entrance requirements For graduate students: none

assessments / exams
- Regular attendance of classes and tutorials
- Solving the exercises, passing the midterm and the final exam.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

Solving problems occuring in computer algebra praxis
The theory behind algorithms

content

Arithmetic and algebraic systems of equations in geometry, engineering and natural sciences

- integer and modular arithmetics, prime number tests
- polynomal arithmetics and factorization
- fast Fourier-transformation, modular algorithms
- resultants, Gröbnerbasen
- homotopy methods for numerical solving
- real solutions, Sturm chains and other rules for algebraic signs Arithmetic and algebraic systems of equations in geometry, engineering and natural sciences
- integer and modular arithmetics, prime number tests
- polynomal arithmetics and factorization
- fast Fourier-transformation, modular algorithms
- resultants, Gröbnerbasen
- homotopy methods for numerical solving
- real solutions, Sturm chains and other rules for algebraic signs

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Computer Graphics CG

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Philipp Slusallek

lecturers Prof. Dr. Philipp Slusallek

entrance requirements Solid knowledge of linear algebra is recommended.

assessments / exams
- Successful completion of weekly exercises (30% of final grade)
- Successful participation in rendering competition (10%)
- Mid-term written exam (20%, final exam prerequisite)
- Final written exam (40%)
- In each of the above a minimum of 50% is required to pass

A re-exam typically takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade The grade is derived from the above assessments. Possible changes will be announced at the beginnning of each semester.

language English

aims / competences to be developed

This course provides the theoretical and practical foundation for computer graphics. It gives a wide overview of topics, techniques, and approaches used in various aspects of computer graphics but has some focus on image synthesis or rendering. The first part of the course uses ray tracing as a driving applications to discuss core topics of computer graphics, from vector algebra all the way to sampling theory, the human visual system, sampling theory, and spline curves and surfaces. A second part then uses rasterization approach as a driving example, introducing the camera transformation, clipping, the OpenGL API and shading langue, plus advanced techniques.

As part of the practical exercises the students incrementally build their own ray tracing system. Once the basics have been covered, the students participate in a rendering competition. Here they can implement their favorite advanced algorithm and are asked to generate a high-quality rendered image that shows their techniques in action.

content

- Introduction
- Overview of Ray Tracing and Intersection Methods
- Spatial Index Structures
- Vector Algebra, Homogeneous Coordinates, and Transformations
- Light Transport Theory, Rendering Equation
- BRDF, Materials Models, and Shading
- Texturing Methods
- Spectral Analysis, Sampling Theory
- Filtering and Anti-Aliasing Methods

- Recursive Ray Tracing & Distribution Ray-Tracing
- Human Visual System & Color Models
- Spline Curves and Surfaces
- Camera Transformations & Clipping
- Rasterization Pipeline
- OpenGL API & GLSL Shading
- Volume Rendering (opt.)

literature & reading

Will be announced in the lecture.

Cryptography — Crypto

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Dr. Nico Döttling

lecturers Prof. Dr. Cas Cremers
Dr. Nico Döttling
Dr. Antoine Joux
Dr. Lucjan Hanzlik
Dr. Julian Loss

entrance requirements For graduate students: Basic knowledge in theoretical computer science required, background knowledge in number theory and complexity theory helpful

assessments / exams
- Oral / written exam (depending on the number of students)
- A re-exam is normally provided (as written or oral examination).

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

The students will acquire a comprehensive knowledge of the basic concepts of cryptography and formal definitions. They will be able to prove the security of basic techniques.

content

- Symmetric and asymmetric encryption
- Digital signatures and message authentication codes
- Information theoretic and complexity theoretic definitions of security, cryptographic reduction proofs
- Cryptographic models, e.g. random oracle model
- Cryptographic primitives, e.g. trapdoor-one-way functions, pseudo random generators, etc.
- Cryptography in practice (standards, products)
- Selected topics from current research

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Data Networks DN

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr.-Ing. Holger Hermanns

lecturers Prof. Dr.-Ing. Holger Hermanns
Prof. Dr. Anja Feldmann

entrance requirements For graduate students: none

assessments / exams
- Regular attendance of classes and tutorials
- Qualification for final exam through mini quizzes during classes
- Possibility to get bonus points through excellent homework
- Final exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

After taking the course students have

- a thorough knowledge regarding the basic principles of communication networks,
- the fundamentals of protocols and concepts of protocol,
- Insights into fundamental motivations of different pragmatics of current network solutions,
- Introduction to practical aspects of data networks focusing on internet protocol hierarchies

content

Introduction and overview

Cross section:

- Stochastic Processes, Markov models,
- Fundamentals of data network performance assessment
- Principles of reliable data transfer
- Protokols and their elementary parts
- Graphs and Graphalgorithms (maximal flow, spanning tree)
- Application layer:
- Services and protocols
- FTP, Telnet
- Electronic Mail (Basics and Principles, SMTP, POP3, ..)
- World Wide Web (History, HTTP, HTML)

- Transport Layer:
- Services and protocols
- Addressing
- Connections and ports
- Flow control
- QoS
- Transport Protocols (UDP, TCP, SCTP, Ports)
- Network layer:
- Services and protocols
- Routing algorithms
- Congestion Control
- Addressing
- Internet protocol (IP)
- Data link layer:
- Services and protocols
- Medium access protocols: Aloha, CSMA (-CD/CA), Token passing
- Error correcting codes
- Flow control
- Applications: LAN, Ethernet, Token Architectures, WLAN, ATM
- Physical layer
- Peer-to-Peer and Ad-hoc Networking Principles

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Database Systems — DBS

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Jens Dittrich

lecturers Prof. Dr. Jens Dittrich

entrance requirements especially Saarland University CS department's undergraduate lecture *Big Data Engineering* (former *Informationssysteme*), *Programmierung 1* and *2*, *Algorithmen und Datenstrukturen* as well as *Nebenläufige Programmierung*

For graduate students:

- motivation for databases and database management systems;
- the relational data model;
- relational query languages, particularly relational algebra and SQL;
- **solid** programming skills in Java and/or C++
- undergrad courses in algorithms and data structures, concurrent programming

assessments / exams

- Passing a two-hour written exam at the end of the semester
- Successful demonstration of programming project (teams of up to three students are allowed); the project may be integrated to be part of the weekly assignments

Grades are based on written exam; 50% in weekly assignments (in paper and additionally paper or electronic quizzes) must be passed to participate in the final and repetition exams.

A repetition exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours

```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

This class may be run as a flipped classroom, i.e. 2 hours of lectures may be replaced by self-study of videos/papers; the other 2 hours may be used to run a group exercice supervised by the professor called "the LAB")

total workload

```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined based on project, midterm and best of endterm and reexam.

language English

aims / competences to be developed

Database systems are the backbone of most modern information systems and a core technology without which today's economy – as well as many other aspects of our lifes – would be impossible in their present forms. The course teaches the architectural and algorithmic foundations of modern database management systems (DBMS), focussing on database systems internals rather than applications. Emphasis is made on robust and time-tested techniques that have led databases to be considered a mature technology and one of the greatest success stories in computer science. At the same time, opportunities for exciting research in this field will be pointed out.

In the exercise part of the course, important components of a DBMS will be treated and where possible implemented and their performance evaluated. The goal this is to work with the techniques introduced in the lecture and to understand them and their practical implications to a depth that would not be attainable by purely theoretical study.

content

The course ”Database Systems” will introduce students to the internal workings of a DBMS, in particular:

- storage media (disk, flash, main memory, caches, and any other future storage medium)
- data managing architectures (DBMS, streams, file systems, clouds, appliances)
- storage management (DB-file systems, raw devices, write-strategies, differential files, buffer management)
- data layouts (horizontal and vertical partitioning, columns, hybrid mappings, compression, defragmentation)
- indexing (one- and multidimensional, tree-structured, hash-, partition-based, bulk-loading and external sorting, differential indexing, read- and write-optimized indexing, data warehouse indexing, main-memory indexes, sparse and dense, direct and indirect, clustered and unclustered, main memory versus disk and/or flash-based)
- processing models (operator model, pipeline models, push and pull, block-based iteration, vectorization, query compilation)
- processing implementations (join algorithms for relational data, grouping and early aggregation, filtering)
- query processing (scanning, plan computation, SIMD)
- query optimization (query rewrite, cost models, cost-based optimization, join order, join graph, plan enumeration)
- data recovery (single versus multiple instance, logging, ARIES)
- parallelization of data and queries (horizontal and vertical partitioning, shared-nothing, replication, distributed query processing, NoSQL, MapReduce, Hadoop and/or similar and/or future systems)
- read-optimized system concepts (search engines, data warehouses, OLAP)
- write-optimized system concepts (OLTP, streaming data)
- management of geographical data (GIS, google maps and similar tools)
- main-memory techniques

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Digital Transmission & Signal Processing — DTSP

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr.-Ing. Thorsten Herfet

lecturers Prof. Dr.-Ing. Thorsten Herfet

entrance requirements The lecture requires a solid foundation of mathematics (differential and integral calculus) and probability theory. The course will, however, refresh those areas indispensably necessary for telecommunications and potential intensification courses and by this open this potential field of intensification to everyone of you.

assessments / exams Regular attendance of classes and tutorials
Passing the final exam in the 2nd week after the end of courses.
Eligibility: Weekly exercises / task sheets, grouped into two blocks corresponding to first and second half of the lecture. Students must provide min. 50% grade in each of the two blocks to be eligible for the exam.

course types / weekly hours

```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload

```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Final exam mark

language English

aims / competences to be developed

Digital Signal Transmission and Signal Processing refreshes the foundation laid in ”Signals and Systems” [Modulkennung]. Including, however, the respective basics so that the various facets of the introductory study period (Bachelor in Computer Science, Vordiplom Computer- und Kommunikationstechnik, Elektrotechnik or Mechatronik) and the potential main study period (Master in Computer Science, Diplom-Ingenieur Computer- und Kommunikationstechnik or Mechatronik) will be paid respect to.

content

As the basic principle, the course will give an introduction into the various building blocks that modern telecommunication systems do incorporate. Sources, sinks, source and channel coding, modulation and multiplexing are the major keywords, but we will also deal with dedicated pieces like A/D- and D/A-converters and quantizers in a little bit more depth.

The course will refresh the basic transformations (Fourier, Laplace) that give access to system analysis in the frequency domain, it will introduce derived transformations (z, Hilbert) for the analysis of discrete systems and modulation schemes and it will briefly introduce algebra on finite fields to systematically deal with error correction schemes that play an important role in modern communication systems.

literature & reading

Will be announced before the start of the course on the course page on the Internet.

additional information

This module was formerly also known as *Telecommunications I*.

Distributed Systems DS

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Peter Druschel, Ph.D.

lecturers Prof. Peter Druschel, Ph.D.
Allen Clement, Ph.D

entrance requirements *Operating Systems* or *Concurrent Programming*

assessments / exams
- Regular attendance at classes and tutorials.
- Successful completion of a course project in teams of 2 students. (Project assignments due approximately every 2 weeks.)
- Passing grade on 2 out of 3 written exams: midterm, final exam, and a re-exam that takes place during the last two weeks before the start of lectures in the following semester.
- Final course grade: 50% project, 50% best 2 out of 3 exams.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

Introduction to the principles, design, and implementation of distributed systems.

content

- Communication: Remote procedure call, distributed objects, event notification, Inhalt dissemination, group communication, epidemic protocols.
- Distributed storage systems: Caching, logging, recovery, leases.
- Naming. Scalable name resolution.
- Synchronization: Clock synchronization, logical clocks, vector clocks, distributed snapshots.
- Fault tolerance: Replication protocols, consistency models, consistency versus availability trade-offs, state machine replication, consensus, Paxos, PBFT.
- Peer-to-peer systems: consistent hashing, self-organization, incentives, distributed hash tables, Inhalt distribution networks.
- Data centers. Architecture and infrastructure, distributed programming, energy efficiency.

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Embedded Systems ES

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Bernd Finkbeiner, Ph.D

lecturers Prof. Bernd Finkbeiner, Ph.D
Prof. Dr. Martina Maggio

entrance requirements none

assessments / exams
- Written exam at the end of the course.
- Demonstration of the implemented system.
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours

```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

The course is accompanied by a laboratory project, in which a non-trivial embedded system has to be realized.

total workload

```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

The students should learn methods for the design, the implementation, and the validation of safety-critical embedded systems.

content

Embedded Computer Systems are components of a technical system, e.g. an air plane, a car, a household machine, a production facility. They control some part of this system, often called the plant, e.g. the airbag controller in a car controls one or several airbags. Controlling means obtaining sensor values and computing values of actuator signals and sending them.

Most software taught in programming courses is transformational, i.e. it is started on some input, computes the corresponding output and terminates. Embedded software is reactive, i.e. it is continuously active waiting for signals from the plant and issuing signals to the plant.

Many embedded systems control safety-critical systems, i.e. malfunctioning of the system will in general cause severe damage. In addition, many have to satisfy real-time requirements, i.e. their reactions to input have to be produced within fixed deadlines.

According to recent statistics, more than 99
markdownRendererInterblockSeparator The course will cover most aspects of the design and implementation of embedded systems, e.g. specification mechanisms, embedded hardware, operating systems, scheduling, validation methods.

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Geometric Modelling GM

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Hans-Peter Seidel

lecturers Prof. Dr. Hans-Peter Seidel
Dr. Rhaleb Zayer

entrance requirements calculus and basic programming skills

assessments / exams
- Regular attendance and participation.
- Weekly Assignments (10% bonus towards the course grade; bonus points can only improve the grade; they do not affect passing)
- Passing the written exams (mid-term and final exam).
- The mid-term and the final exam count for 50% each, but 10% bonus from assignments will be added.
- A re-exam takes place at the end of the semester break or early in the next semester.

course types / weekly hours

```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

Practical assignments in groups of 3 students (practice)
Tutorials consists of a mix of theoretical + practical assignments.

total workload

```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be based on the performance in exams, exercises and practical tasks. The detailed terms will be announced by the module coordinator.

language English

aims / competences to be developed

Gaining knowledge of the theoretical aspect of geometric modelling problems, and the practical solutions used for modelling and manipulating curves and surfaces on a computer. From a broader perspective: Learning how to represent and interact with geometric models in a discretized, digital form (geometric representations by functions and samples; design of linear function spaces; finding “good” functions with respect to a geometric modelling task in such spaces).

content

- Differential geometry Fundamentals
- Interpolation and Approximation
- Polynomial Curves
- Bezier and Rational Bezier Curves
- B-splines, NURBS
- Spline Surfaces
- Subdivision and Multiresolution Modelling
- Mesh processing
- Approximation of differential operators
- Shape Analysis and Geometry Processing

literature & reading

Will be announced before the term begins on the lecture website.

Human Computer Interaction HCI

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Jürgen Steimle

lecturers Prof. Dr. Jürgen Steimle

entrance requirements undergraduate students: *Programmierung 1* and *2*
graduate students: none

assessments / exams Regular attendance of classes and tutorials
Successful completion of exercises and course project
Final exam
A re-exam takes place (as written or oral examination).

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

This course teaches the theoretical and practical foundations for human computer interaction. It covers a wide overview of topics, techniques and approaches used for the design and evaluation of modern user interfaces.

The course covers the principles that underlie successful user interfaces, provides an overview of input and output devices and user interface types, and familiarizes students with the methods for designing and evaluating user interfaces. Students learn to critically assess user interfaces, to design user interfaces themselves, and to evaluate them in empirical studies.

content

- Fundamentals of human-computer interaction
- User interface paradigms, input and output devices
- Desktop & graphical user interfaces
- Mobile user interfaces
- Natural user interfaces
- User-centered interaction design
- Design principles and guidelines
- Prototyping

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Image Processing and Computer Vision — IPCV

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Joachim Weickert

lecturers Prof. Dr. Joachim Weickert

entrance requirements Undergraduate mathematics (e.g. Mathematik für Informatiker I-III) and elementary programming knowledge in C

assessments / exams
- For the homework assignments one can obtain up to 24 points per week. Actively participating in the classroom assignments gives 12 more points per week, regardless of the correctness of the solutions. To qualify for both exams one needs 2/3 of all possible points.
- Passing the final exam or the re-exam.
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from the performance in the exam or the re-exam. The better grade counts.

language English

aims / competences to be developed

Broad introduction to mathematical methods in image processing and computer vision. The lecture qualifies students for a bachelor thesis in this field. Together with the completion of advanced or specialised lectures (9 credits at least) it is the basis for a master thesis in this field.

content

```
Inhalt

1. Basics
   1.1 Image Types and Discretisation
   1.2 Degradations in Digital Images
2. Colour Perception and Colour Spaces
3. Image Transformations
   3.1 Continuous Fourier Transform
   3.2 Discrete Fourier Transform
   3.3 Image Pyramids
   3.4 Wavelet Transform
4. Image Compression
5. Image Interpolation
6. Image Enhancement
   6.1 Point Operations
```

```
   6.2 Linear Filtering and Feature Detection
   6.3 Morphology and Median Filters
   6.3 Wavelet Shrinkage, Bilateral Filters, NL Means
   6.5 Diffusion Filtering
   6.6 Variational Methods
   6.7 Deconvolution Methods
7. Texture Analysis
8. Segmentation
   8.1 Classical Methods
   8.2 Variational Methods
9. Image Sequence Analysis
   9.1 Local Methods
   9.2 Variational Methods
10. 3-D Reconstruction
   10.1 Camera Geometry
   10.2 Stereo
   10.3 Shape-from-Shading
11. Object Recognition
   11.1 Hough Transform
   11.2 Invariants
   11.3 Eigenspace Methods
```

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Information Retrieval and Data Mining — IRDM

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Gerhard Weikum

lecturers Prof. Dr. Gerhard Weikum

entrance requirements Good knowledge of undergraduate mathematics (linear algebra, probability theory) and basic algorithms.

assessments / exams
- Regular attendance of classes and tutor groups
- Presentation of solutions in tutor groups
- Passing 2 of 3 written tests (after each third of the semester)
- Passing the final exam (at the end of the semester)

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined by the performance in written tests, tutor groups, and the final exam. Details will be announced on the course web site.

language English

aims / competences to be developed

The lecture teaches models and algorithms that form the basis for search engines and for data mining and data analysis tools.

content

Information Retrieval (IR) and Data Mining (DM) are methodologies for organizing, searching and analyzing digital Inhalts from the web, social media and enterprises as well as multivariate datasets in these contexts. IR models and algorithms include text indexing, query processing, search result ranking, and information extraction for semantic search. DM models and algorithms include pattern mining, rule mining, classification and recommendation. Both fields build on mathematical foundations from the areas of linear algebra, graph theory, and probability and statistics.

literature & reading

Will be announced on the course web site.

Internet Transport

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr.-Ing. Thorsten Herfet

lecturers Prof. Dr.-Ing. Thorsten Herfet

entrance requirements

- Motivation for networks and communication
- Practical experience (e.g. through *Hands on Networking*) is recommended
- Knowledge of the fundamentals of communication (e.g. through *Digital Transmission & Signal Processing*) is recommended

assessments / exams

- Regular attendance of classes and tutorials
- Eligibility for exam through quizzes and assignments
- Final Exam
- A re-exam typically takes place during the last two weeks before the start of lectures in the following semester

course types / weekly hours

```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload

```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, quizzes and assigments. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

Today the majority of all services is available via Internet-connections. Other than in the past this comprises not only data- but also media-services (like Voice Over IP or Video Streaming) and even Cyber-Physical Systems with their networked control loops.

The course introduces the basic characteristics of Internet-based communication (packetization on different layers, packet error detection and correction). It shows how existing protocols like HTTP, TCP and UDP can be shaped and evolved to fulfill the service requirements and how new protocols should be designed to serve the large variety of services.

content

- Introcudion of *EverythingoverIP* and *IPoverEverything*
- Theory of erasure channels (i.i.d, Gilbert-Elliott, channel capacity, minimum redundancy information)
- Wireless link layers (WiFi, PHY-bursts, Logical Link Control with DCF & EDCA, aggregation and ACK-techniques)
- Frame Check Sums, Cyclic Redundancy Checks
- Time Sensitive Networking
- Transport Layer services (flow control, congestion control, error control, segmentation and reassembly)
- QUIC media transport
- Error Coding under predictable reliability and latency (MDS-codes, binary codes)
- Upper layer protocols (HTTP, RTP/RTSP, DASH)

literature & reading

The course will come with a self-contained interactive manuscript. Complementary material will be announced before the start of the course on the course page on the Internet.

additional information

This module was formerly also known as *Future Media Internet* and *Multimedia Transport*.

Introduction to Computational Logic — ICL

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Gert Smolka

lecturers Prof. Dr. Gert Smolka

entrance requirements none

assessments / exams
- Regular attendance of classes and tutorials.
- Passing the midterm and the final exam.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

- structure of logic languages based on type theory
- distinction notation / syntax / semantics
- structure and formal representation of mathematical statements
- structure and formal representation of proofs (equational and natural deduction)
- solving Boolean equations
- proving formulas with quantifiers
- implementing syntax and deduction

content

Type Theory:

- functional representation of mathematical statements
- simply typed lambda calculus, De Bruijn representation and substitution, normalization, elimination of lambdas
- Interpretations and semantic consequence
- Equational deduction, soundness and completeness
- Propositional Logic
- Boolean Axioms, completeness for 2-valued interpretation
- resolution of Boolean equations, canonical forms based on decision trees and resolution

Predicate Logic (higher-order):

- quantifier axioms
- natural deduction
- prenex and Skolem forms

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Machine Learning — ML

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Isabel Valera

lecturers Prof. Dr. Isabel Valera

entrance requirements The lecture gives a broad introduction into machine learning methods. After the lecture the students should be able to solve and analyze learning problems.

assessments / exams
- Regular attendance of classes and tutorials.
- 50% of all points of the exercises have to be obtained in order to qualify for the exam.
- Passing 1 out of 2 exams (final, re-exam).

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Determined from the results of the exams, exercises and potential projects. The exact grading modalities are announced at the beginning of the course.

language English

aims / competences to be developed

The lecture gives a broad introduction into machine learning methods. After the lecture the students should be able to solve and analyze learning problems.

content

- Bayesian decision theory
- Linear classification and regression
- Kernel methods
- Bayesian learning
- Semi-supervised learning
- Unsupervised learning
- Model selection and evaluation of learning methods
- Statistical learning theory
- Other current research topics

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Operating Systems — OS

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Peter Druschel, Ph.D.

lecturers Prof. Peter Druschel, Ph.D.
Björn Brandenburg, Ph.D

entrance requirements For graduate students: none

assessments / exams Regular attendance at classes and tutorials
Successful completion of a course project in teams of 2 students
Passing 2 written exams (midterm and final exam)
A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours

```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload

```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

Introduction to the principles, design, and implementation of operating systems

content

Process management:

- Threads and processes, synchronization
- Multiprogramming, CPU Scheduling
- Deadlock

Memory management:

- Dynamic storage allocation
- Sharing main memory
- Virtual memory

I/O management:

- File storage management
- Naming
- Concurrency, Robustness, Performance

Virtual machines

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Optimization — Opti

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Kurt Mehlhorn

lecturers Prof. Dr. Kurt Mehlhorn
Dr. Andreas Karrenbauer

entrance requirements For graduate students: none

assessments / exams
- Regular attendance of classes and tutorials
- Solving accompanying exercises, successful partcipation in midterm and final exam
- Grades: Yes
- The grade is calculated from the above parameters according to the following scheme: 20%, 30%, 50%
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
  90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

The students learn to model and solve optimization problems from theory as from the real world

content

Linear Programming: Theory of polyhedra, simplex algorithm, duality, ellipsoid method * Integer linear programming: Branch-and-Bound, cutting planes, TDI-Systems * Network flow: Minimum cost network flow, minimum mean cycle cancellation algorithm, network simplex method * Matchings in graphs: Polynomial matching algorithms in general graphs, integrality of the matching polytope, cutting planes * Approximation algorithms: LP-Rounding, greedy methods, knapsack, bin packing, steiner trees and forests, survivable network design

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Security — Sec

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Michael Backes

lecturers Prof. Dr. Michael Backes
Prof. Dr. Cas Cremers

entrance requirements For graduate students: none

assessments / exams
- Regular attendance of classes and tutorials
- Passing the final exam
- A re-exam is normally provided (as written or oral examination).

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined by the performance in exams, tutor groups, and practical tasks. Details will be announced by the lecturer at the beginning of the course.

language English

aims / competences to be developed

Description, assessment, development and application of security mechanisms, techniques and tools.

content

- Basic Cryptography,
- Specification and verification of security protocols,
- Security policies: access control, information flow analysis,
- Network security,
- Media security,
- Security engineering

literature & reading

Will be announced on the course website

Semantics

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Gert Smolka

lecturers Prof. Dr. Gert Smolka

entrance requirements For graduate students: core lecture Introduction to Computational Logic

assessments / exams
- Regular attendance of classes and tutorials.
- Passing the midterm and the final exam

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

Understanding of

- Logical structure of programming languages
- Formal models of programming languages
- Type and module systems for programming languages

content

Theory of programming languages, in particular:

- Formal models of functional and object-oriented languages
- Lambda Calculi (untyped, simply typed, System F, F-omega, Lambda Cube, subtyping, recursive types, Curry-Howard Correspondence)
- Algorithms for type checking and type reconstruction

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Software Engineering SE

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr. Sven Apel

lecturers Prof. Dr. Sven Apel

entrance requirements
- Knowledge of programming concepts (as taught in the lectures *Programmierung 1* and *Programmierung 2*)
- Basic knowledge of software processes, design, and testing (as taught and applied in the lecture *Softwarepraktikum*)

assessments / exams Beside the lecture and weekly practical exercises, there will be a number of assignments in the form of mini-projects for each student to work on (every two to three weeks). The assignments will be assessed based on the principles covered in the lecture. Passing all assignments is a prerequisite for taking the final written exam. The final grade is determined only by the written exam. Further examination details will be announced by the lecturer at the beginning of the course. In short:
- Passing all assignments (prerequisite for the written exam)
- Passing the written exam

course types / weekly hours
```
  4 h lectures
+ 2 h exercises
= 6 h (weekly)
```

total workload
```
   90 h of classes and exercises
+ 180 h private study and assignments
= 270 h (= 9 ECTS)
```

grade The grade is determined by the written exam. Passing all assignments is a prerequisite for taking the written exam. The assignments do not contribute to the final grade. Further examination details will be announced by the lecturer at the beginning of the course.

language English

aims / competences to be developed

- The students know and apply modern software development techniques.
- They are aware of key factors contributing to the complexity of real-world software systems, in particular, software variability, configurability, feature interaction, crosscutting concerns, and how to address them.
- They know how to apply established design and implementation techniques to master software complexity.
- They are aware of advanced design and implementation techniques, including collaboration-based design, mixins/traits, aspects, pointcuts, advice.
- They are aware of advanced quality assurance techniques that take the complexity of real-world software systems into account: variability-aware analysis, sampling, feature-interaction detection, predictive performance modeling, etc.
- They appreciate the role of non-functional properties and know how to predict and optimize software systems regarding these properties.
- They are able to use formal methods to reason about key techniques and properties covered in the lecture.

content

- Domain analysis, feature modeling
- Automated reasoning about software configuration using SAT solvers
- Runtime parameters, design patterns, frameworks
- Version control, build systems, preprocessors
- Collaboration-based design
- Aspects, pointcuts, advice
- Expression problem, preplanning problem, code scattering & tangling, tyranny of the dominant decomposition, inheritance vs. delegation vs. mixin composition
- Feature interaction problem (structural, control- & data-flow, behavioral, non-functional feature interactions)
- Variability-aware analysis and variational program representation (with applications to type checking and static program analysis)
- Sampling (random, coverage)
- Machine learning for software performance prediction and optimization

literature & reading

- Feature-Oriented Software Product Lines: Concepts and Implementation. S. Apel, et al., Springer, 2013.
- Generative Programming: Methods, Tools, and Applications: Methods, Techniques and Applications. K. Czarnecki, et al., Addison-Wesley, 2000.
- Mastering Software Variability with FeatureIDE. J. Meinicke, et al., Springer, 2017.

Verification — Veri

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**6**	**9**

responsible Prof. Dr.-Ing. Holger Hermanns

lecturers Prof. Dr.-Ing. Holger Hermanns
Prof. Bernd Finkbeiner, Ph.D

entrance requirements For graduate students: none

assessments / exams
- Regular attendance of classes and tutorials
- Passing the final exam
- A re-exam takes place during the last two weeks before the start of lectures in the following semester.

course types / weekly hours
```
  4 h lectures
+ 2 h tutorial
= 6 h (weekly)
```

total workload
```
   90 h of classes
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

The students become familiar with the standard methods in computer-aided verification. They understand the theoretical foundations and are able to assess the advantages and disadvantages of different methods for a specific verification project. The students gain first experience with manual correctness proofs and with the use of verification tools.

content

- models of computation and specification languages: temporal logics, automata over infinite objects, process algebra
- deductive verification: proof systems (e.g., Floyd, Hoare, Manna/Pnueli), relative completeness, compositionality
- model checking: complexity of model checking algorithms, symbolic model checking, abstraction case studies

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Module Category 2

Advanced Lectures Cybersecurity (Vertiefungsvorlesungen)

Advanced Public Key Cryptography — APC

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Dr. Nico Döttling

lecturers Dr. Nico Döttling

entrance requirements Cryptography

assessments / exams Mündliche Prüfung oder Abschlussklausur

course types / weekly hours
```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload
```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.

language English

aims / competences to be developed

Students will be obtaining a basic understanding of advanced concepts of modern cryptography, such as how to modeling security of complex systems, advanced encryption schemes like fully homomorphic encryption and functional encryption, as well as zero-knowledge proofs and multiparty computation.

content

- Modelling Security for Encryption Schemes
- Proving Security of Encryption Schemes
- Tools and Paradigms for designing Encryption Schemes
- Advanced notions of encryption such as homomorphic encryption, identity based encryption, attribute-based encryption and functional encryption

literature & reading

The teaching material will be in English and it will be announced at the beginning of the lecture.

Algorithms in Cryptanalysis

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible	Dr. Antoine Joux
lecturers	Dr. Antoine Joux
entrance requirements	Good working knowledge of algebra and algorithms
assessments / exams	Written exam.
course types / weekly hours	
total workload	
grade	Determined by the performance in exams.
language	

aims / competences to be developed

The goal of this course is to familiarise the students with the variety of algorithmic techniques that are used in cryptanalysis and with the mathematical background underlying these techniques.

content

The course will be arranged around three main directions:

- Presentation of the cryptographic motivation
- Description of relevant algorithmic techniques
- Application of the algorithms in the cryptographic context

The techniques covered in the course will range from fundamental algorithms such as sorting which are essential in many cryptanalyses to advanced factorisation and discrete logarithm algorithms on finite field and elliptic curves, requiring a working knowledge of number theory.

literature & reading

Automated Debugging

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**at least every two years**	**1 semester**	**4**	**6**

responsible Prof. Dr. Andreas Zeller

lecturers Prof. Dr. Andreas Zeller

entrance requirements *Programmierung 1*, *Programmierung 2* and *Softwarepraktikum*

assessments / exams Projects and mini-tests

course types / weekly hours

```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload

```
  60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade The module is passed in its entirety if the examination performance has been passed.

language English

aims / competences to be developed

Finding and fixing software bugs can involve lots of effort. This course addresses this problem by automating software debugging, specifically identifying failure causes, locating bugs, and fixing them. Students learn the basics of systematic debugging, and explore tools and techniques for automated debugging.

content

- Tracking Problems
- The Scientific Method
- Cause-Effect Chains
- Building a Debugger
- Tracking Inputs
- Assertions and Sanitizers
- Detecting Anomalies
- Statistical Fault Localization
- Generating Tests
- Reducing Failure-Inducing Inputs
- Mining Software Archives
- Fixing the Defect
- Repairing Bugs Automatically
- Managing Bugs

literature & reading

The teaching material consists of text, Python code, and Jupyter Notebooks from the textbook “The Debugging Book” (https://www.debuggingbook.org/), also in English.

Ethics for Nerds — E4N

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional / summer semester**	**1 semester**	**4**	**6**

responsible Prof. Dr.-Ing. Holger Hermanns

lecturers Prof. Dr.-Ing. Holger Hermanns
Kevin Baum
Sarah Sterz

entrance requirements We expect basic knowledge of propositional and first-order logic, an open mind, and interest to look at computer science in ways you probably are not used to.

assessments / exams The details of exam admission and grading are announced at the beginning of each iteration. Typically, participant are graded based on

- an exam or a re-exam (the better mark counts),
- a short essay where the participant has to argue for or against a moral claim in a topic from computer science.

To get the exam admission, participants usually have to get 50% of the points on weekly exercise sheets.

course types / weekly hours

```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

(may be adjusted before the start of each iteration of the course)

total workload

```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Will be determined based on exam performance, essay performance, and possibly exercise outcomes. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

Many computer scientists will be confronted with morally difficult situations at some point in their career – be it in research, in business, or in industry. This module equips participants with the crucial assets enabling them to recognize such situations and to devise ways to arrive at a justified moral judgment regarding the question what one is permitted to do and what one should better not do. For that, participants will be made familiar with moral theories from philosophy, as well as different Codes of Ethics for computer scientists. Since one can quickly get lost when talking about ethics and morals, it is especially important to talk and argue clearly and precisely. In order to do prepare for that, the module offers substantial training regarding formal and informal argumentation skills enabling participants to argue beyond the level of everyday discussions at bars and parties. In the end, succesful participants are able to assess a morally controversial topic from computer science on their own and give a convincing argument for their respective assessments.

The module is intended to always be as clear, precise, and analytic as possible. What you won't find here is the meaningless bla-bla, needlessly poetic language, and vague and wordy profundity that some people tend to associate with philosophy.

content

This course covers:

- an introduction to the methods of philosophy, argumentation theory, and the basics of normative as well as applied ethics;
- relevant moral codices issued by professional associations like the ACM, the IEEE, and more;
- starting points to evaluate practices and technologies already in use or not that far away, including for instance: filter bubbles and echo chambers, ML-algorithms as predictive tools, GPS-tracking, CCTV and other tools from surveillance, fitness trackers, big data analysis, autonomous vehicles, lethal autonomous weapons systems and so on;
- an outlook on more futuristic topics like machine ethics, roboethics, and superintelligences;
- and more.

The content of the course is updated regularly to always be up-to-date and cover the currently most relevant topics, technologies, policies, and developments.

literature & reading

Will be announced before the start of the course on the course page.

Foundations of Web Security — FoWS

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Dr. Ben Stock

lecturers Dr. Ben Stock

entrance requirements *Security* or *Foundations of Cybersecurity 1 and 2*

assessments / exams Projekt und schriftliche Abschlussklausur

course types / weekly hours

```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload

```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.

language English

aims / competences to be developed

The students will acquire a practical understanding of the security threats a modern Web application is faced with. The students fully comprehend the attack surface of applications and know the necessary countermeasures and mitigations for a wide range of attacks.

content

- Historical evolution of the Web
- Client-side security (e.g., Cross-Site Scripting, Cross-Site Script Inclusion, Cross-Site Request Forgery)
- User-centric security (e.g., Clickjacking & Phishing)
- Server-side security (e.g., SQL injections, command injections)
- Infrastructure security (e.g., HTTPS & attacks against it)

literature & reading

The teaching material will be in English and it will be announced at the beginning of the lecture.

Generating Software Tests GST

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Prof. Dr. Andreas Zeller

lecturers Prof. Dr. Andreas Zeller

entrance requirements Programming 1, Programming 2, Softwarepraktikum

assessments / exams Projekte und Mini-Tests

course types / weekly hours
```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload
```
  60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.

language English

aims / competences to be developed

Software has bugs and catching bugs can involve lots of effort. Yet, finding bugs is important especially when these bugs are critical vulnerabilities. This course addresses this problem by automating software testing, specifically by generating tests automatically. Students learn the basics of general testing and security testing and explore the most important tools and techniques for generating software tests.

content

- Introduction to Software Testing
- Fuzzing: Breaking Things with Random Inputs
- Mutation-Based Fuzzing
- Greybox Fuzzing
- Search-Based Fuzzing
- Fuzzing with Grammars
- Parsing Inputs
- Probabilistic Grammar Fuzzing
- Fuzzing with Generators
- Reducing Failure-Inducing Inputs
- Mining Input Grammars
- Concolic Fuzzing
- Symbolic Fuzzing
- Testing APIs
- Testing Web Applications
- Testing Graphical User Interfaces
- When To Stop Fuzzing

literature & reading

The teaching material consists of text, Python code, and Jupyter Notebooks from the textbook “The Fuzzing Book” (https://www.fuzzing-book.org/) in English.

IT Founders's Lab 1 - Launchpad ITFL1

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**2**	**3**

responsible Prof. Dr. Andreas Zeller

lecturers Prof. Dr. Andreas Zeller
Dr. Sven Bugiel
Dr. Giancarlo Pellegrino

entrance requirements none

assessments / exams The examination consists of four mandatory parts, which must be submitted and presented during the course.

course types / weekly hours

```
  1 h lectures
+ 1 h tutorial
= 2 h (weekly)
```

total workload

```
  30 h of classes
+ 60 h private study
= 90 h (= 3 ECTS)
```

grade

- The grade is determined from the contribution of each group member for each part of the examination. Unless otherwise specified at the beginning of the course, the final grade is determined as follows:
  Part 1 – Summaries. (25% of final grade)
  Part 2 – Initial business plan. (25% of final grade)
  Part 3 – Mock-up. (25% of final grade)
  Part 4 – Sales pitch. (25% of final grade)
- The lecturer may consult with other jury members to determine the grade.

language English

aims / competences to be developed

Students learn to develop a business idea from current research topics, refine it in a market-oriented way, and pitch it to experts as a mock-up with the aim of a proof-of-concept demonstration that can lead to the development of a prototype and a subsequent start-up.

content

In this advanced lecture course, students attend a *series of lectures* (by the lecturers and guests; attendance is mandatory) to eventually develop a business idea in the field of computer science with the aim of founding a company based on it. Lecture topics include

- developing a business idea
- current research topics that could be turned into a business
- rapid prototyping
- writing a business plan
- pitching a business idea
- experience reports from founders.

The business idea should implement current research topics in a marketable way; it is developed and refined by the students during the project with the help of lecturers and technology transfer experts.

During the course, students provide four items:

Part 1 – Summaries. The students must provide written summaries of the lectures offered.

Part 2 – Initial business plan. The students compose an initial business plan. This document should be structured to address the following mandatory questions:

```
1. Idea
    1.1. What is the objective of the company/project?
    1.2. What is the innovative character of the service and product portfolio?
    1.3. Which characteristics give the product or service a unique position?
2. Team
    2.1. How is the project team composed?
    2.2. What complementary skills does the founding team have?
3. Market and Competition
    3.1. Is the addressed market big enough?
    3.2. What are the success factors in the market?
    3.3. What role do innovation and technical progress play?
    3.4. Which companies are competitors within the market?
4. Business Model
    4.1. How is your service new and useful?
5. Marketing and Sales
    5.1. What sales (volume) and turnover (value) is the planned company aiming for?
```

Part 3 – Mock-up. This mock-up, wireframe, or prototype describes the idea from a customer's perspective.

Part 4 – Sales pitch. The students present their business idea and their mock-up to a jury. The jury comprises lecturers, other examiners, and, if applicable, guests from technology transfer and industry. The presentation should convince potential investors to commit to the business idea. Annotated slides and the initial business plan must be made available to the jury before the sales pitch.

Students can work in *teams* of up to five members. Students in a team must submit a joint declaration stating which group member contributed to which parts and how.

literature & reading

additional information

This *Launchpad* course lays the groundwork for the subsequent IT Founder's Lab courses *Bootcamp* and *Accelerator*.

IT Founders's Lab 2 - Bootcamp — ITFL2

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**2**	**9**

responsible Prof. Dr. Andreas Zeller

lecturers Prof. Dr. Andreas Zeller
Dr. Sven Bugiel
Dr. Giancarlo Pellegrino

entrance requirements The course requires students to pass an earlier *IT Founder's Lab 1 - Launchpad* course with a grade of 1.3 or better. Capacity is limited. If there are more applicants than slots available, students will be selected by lots, keeping groups from *IT Founder's Lab 1 - Launchpad* intact if possible.

assessments / exams The examination consists of three mandatory parts, which must be submitted and presented during the course.

course types / weekly hours
```
2 h lectures
```

total workload
```
 30 h of classes
+ 60 h project work
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade
- The grade is determined from the contribution of each group member for each part of the examination. Unless otherwise specified at the beginning of the course, the final grade is determined as follows:
  Part 1 – Prototype. (60% of final grade)
  Part 2 – Refined business plan. (20% of final grade)
  Part 3 – Sales pitch. (20% of final grade)
- The lecturer may consult with other jury members to determine the grade.

language English

aims / competences to be developed

Students learn to implement a demonstrator of a business idea, demonstrating its feasibility and usefulness for customers. They refine their business idea such that it can lead to the development of a full-fledged implementation and eventual founding of a company. They pitch their demonstrator and business idea to experts and thus train their presentation skills. Students will learn systematic planning of technical and financial aspects, sustainable development of sophisticated concepts, and responsible work and organization in a team.

content

In this advanced lecture course, students implement and refine a prototype demonstrating their business idea. They attend a series of lectures (by the lecturers as well as guests) to demonstrate their business idea with the aim of subsequently / concurrently founding a company based on it. Lecture topics include

- building and refining minimal viable products (requirements, rapid prototyping)
- agile software development for start-ups,
- refining an initial business idea,
- writing a convincing business plan
- advanced experience reports from founders.

The business idea and demonstrators should implement current research topics in a marketable way; they are developed and refined by the students during the project with the help of lecturers and technology transfer experts.

During the course, students provide three items:

Part 1 – Prototype. Students implement a prototype demonstrating their business idea. The focus is on demonstrating the feasibility of critical parts of their idea, typically via a vertical prototype.

The *deliverable* includes the source code, the version history, build instructions, and a status report on the work done with a focus on technical implementation.

Part 2 – Refined business plan. The students compose a refined business plan. This document must

- extend the questions listed for the "Launchpad" course, referring to the current prototype. Differences must be highlighted and provided with a rationale (e.g., expert feedback from the jury).
- address the additional questions listed below.

The document should be structured along with the questions. All answers are mandatory.

```
1. Idea
    1.4. Which legal regulations, norms, or standards have to be fulfilled?
    1.5. What is the warranty and service policy?
2. Team
    2.3. Which positions still need to be filled?
3. Market and Competition
    3.5. What goals/strategies are competitors pursuing?
    3.6. Which expertise do competitors have?
    3.7. How big are the competitors' financial resources?
    3.8. What are the reasons for the successes and failures of competitors?
    3.9. How will the competitors react to the market entry?
4. Business Model
    4.2. What is the company‘s vision, goals, and strategy?
    4.3. What is the customer‘s need?
    4.4. What is the core competence of the company?
5. Marketing and Sales
    5.2. What prices should be achieved?
    5.3. According to which criteria are the prices formed?
    5.4. How high should the profit margin be?
    5.5. Which target groups are best reached through which sales channels?
    5.6. How is the attention of target group customers drawn to the product?
    5.7. What kind of marketing channels should be used?
6. Status Report
    6.1. What have your business activities to date been?
    6.2. Which experts and potential customers have you contacted, and with which results?
```

Part 3 – Sales pitch. The students present their business idea and their prototype (from a customer's perspective) to a jury. The jury comprises lecturers, other examiners, and, if applicable, guests from technology transfer and industry. The presentation should convince potential investors to commit to the business idea. Annotated slides and the refined business plan must be available to the jury before the sales pitch.

literature & reading

additional information

An IT Founder's Lab project topic proposed by one group can only be worked on by another group with the consent of the former group.

IT Founders's Lab 3 - Accelerator — ITFL3

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**2**	**9**

responsible Prof. Dr. Andreas Zeller

lecturers Prof. Dr. Andreas Zeller
Dr. Sven Bugiel
Dr. Giancarlo Pellegrino

entrance requirements The course requires students to pass an earlier *IT Founder's Lab 2 - Bootcamp* course with a grade of 2.3 or better.

assessments / exams The examination consists of three mandatory parts, which must be submitted and presented during the course.

course types / weekly hours
```
2 h lectures
```

total workload
```
 30 h of classes
+ 60 h project work
+ 180 h private study
= 270 h (= 9 ECTS)
```

grade
- The grade is determined from the contribution of each group member for each part of the examination. Unless otherwise specified at the beginning of the course, the final grade is determined as follows:
  Part 1 – Refined Prototype. (60% of final grade)
  Part 2 – Refined business plan. (20% of final grade)
  Part 3 – Sales pitch. (20% of final grade)
- The lecturer may consult with other jury members to determine the grade.

language English

aims / competences to be developed

Students learn to refine a demonstrator of a business idea, demonstrating its feasibility and usefulness for customers. They further refine their business idea such that it can lead to the development of a full-fledged implementation and eventual founding of a company. They pitch their demonstrator and business idea to experts and thus train and refine their presentation skills. Students will learn advanced systematic planning of technical and financial aspects, sustainable development of sophisticated concepts, and responsible work and organization in a team.

content

In this advanced lecture course, students refine a prototype demonstrating their business idea. They attend a series of lectures (by the lecturers as well as guests) to refine their business idea with the aim of subsequently / concurrently founding a company based on it. Lecture topics include

- further refining minimal viable products (requirements, rapid prototyping)
- further refining an initial business idea,
- further refining a convincing business plan
- further advanced experience reports from founders.

The business idea and demonstrators should implement current research topics in a marketable way; they are developed and refined by the students during the project with the help of lecturers and technology transfer experts.

During the course, students provide three items:

Part 1 – Refined prototype. Students refine the prototype demonstrating their business idea. The focus is on demonstrating the breadth of functionality, typically via a horizontal prototype.

The *deliverable* includes the source code, the version history, build instructions, and a status report on the work done with a focus on technical implementation.

Part 2 – Refined business plan. The students compose a refined business plan. This document must

- extend the questions listed for the *IT Founder's Lab 2 – Bootcamp* course, referring to the current prototype. Differences must be highlighted and provided with a *rationale* (e.g., expert feedback from the jury).
- address the additional questions listed below.

The document should be structured along with the questions. All answers are mandatory.

1. Idea
    1.6. What is the warranty and service policy?
    1.7. Have any intellectual property rights already been registered?
    1.8. At what stage of development are the products and services?
    1.9. What further development steps are planned?
    1.10. What resources are available for further development?
    1.11. Which areas may carry development risks, and how are these met?
2. Team
    2.4. What are the key positions in the project?
    2.5. What qualifications and experiences do the employees in key positions bring?
    2.6. Which business partners are involved in the service creation process?
    2.7. Can capacities be adjusted at short notice?
    2.8. Are quality assurance measures in place?
    2.9. What is the planned organizational structure of the company/project?
    2.10. Is there a need to optimize the current organizational structure? How?
3. Market and Competition
    3.10. To what extent is the intended company dependent on individual suppliers and customers?
    3.11. How big are the current and expected returns within market segments?
    3.12. How has the industry developed in the past and what are the forecasts?
    3.13. What market potential and volume are estimated for the individual market segments?
4. Business Model
    4.5. How is the core competence protected?
    4.6. Can the planned success be achieved with the proposed business model?
    4.7. Can the business model be easily adapted to changes in the economic environment?
    4.8. Which services are provided by the company itself, and what is outsourced?
5. Marketing and Sales
    5.8. Which sales/promotion measures will be used?
    5.9. How high is the cost to permanently retain a customer?
    5.10. How high are the customer acquisition costs?
    5.11. Which requirements (number, qualification, and equipment of the employees) must be fulfilled by the sales department to implement the marketing strategy successfully? What expenses are planned for this?
    5.12. How will sales and earnings be distributed among the individual sales channels?
    5.13. What market share per channel can be achieved?
    5.14. What expenses will be incurred when introducing the products and services?
6. Status Report
    6.1. What have your business activities to date been?
    6.2. Which experts and potential customers have you contacted, and with which results?

Part 3 – Sales pitch. The students present their business idea and their prototype (from a customer's perspective) to a jury. The jury comprises lecturers, other examiners, and, if applicable, guests from technology transfer and industry. The presentation should convince potential investors to commit to the business idea. Annotated slides and the refined business plan must be available to the jury before the sales pitch.

Students are expected to work in a team of up to five members. Teams must submit a joint declaration stating which group member contributed to which parts and how.

literature & reading

additional information

An IT Founder’s Lab project topic proposed by one group can only be worked on by another group with the consent of the former group.

Machine Learning in Cybersecurity — MLCySec

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Prof. Dr. Mario Fritz

lecturers Prof. Dr. Mario Fritz

entrance requirements *Data Science/Statistics Course*

assessments / exams Übungen, Projekt und mündliche Prüfung

course types / weekly hours
```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload
```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Das Modul ist insgesamt bestanden, wenn die Prüfungsleistungen bestanden wurden.

language English

aims / competences to be developed

Students know about the opportunities and risks of applying machine learning in cyber security. They understand a range of attacks and defense strategies and are capable of implementing such techniques. Students are aware of privacy risks of machine learning methods and understand how such risks can be mitigated.

content

- Machine learning methodology in the context of cyber security
- Applications and opportunities of learning in cyber security
- Risks and attacks on machine learning in cyber security
- Malware classification
- Anomaly detection
- Intrusion detection
- Evasion attacks
- Model stealing
- Privacy risks and attacks
- Privacy protection

literature & reading

The teaching material will be in English and it will be announced at the beginning of the lecture.

Mobile Security — MOS

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Dr. Sven Bugiel

lecturers Dr. Sven Bugiel

entrance requirements *Foundations of Cybersecurity 1 and 2*, *Programmierung 2* (recommended)

assessments / exams Schriftliche Abschlussklausur

course types / weekly hours

```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload

```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.

language English

aims / competences to be developed

This advanced lecture deals with different, fundamental aspects of mobile operating systems and application security, with a strong focus on the popular, open-source Android OS and its ecosystem. In general, the awareness and understanding of the students for security and privacy problems in this area is increased. The students learn to tackle current security and privacy issues on smartphones from the perspectives of different security principals in the smartphone ecosystem: end-users, app developers, market operators, system vendors, third parties (like companies).

Central questions of this course are:

- What is the threat model from the different principals' perspectives?
- How are the fundamental design patterns of secure systems and security best practices realized in the design of smartphone operating systems? And how does the multi-layered software stack (i.e., middleware on top of the OS) influence this design?
- How are hardware security primitives, such as Trusted Execution Environments, and trusted computing concepts integrated into those designs?
- What are the techniques and solutions market operators have at hand to improve the overall ecosystem's hygiene?
- Which problems and solutions did security research in this area identify in the past half-decade?
- Which techniques have been developed to empower the end-users to protect their privacy?

The lectures are accompanied by exercises to re-enforce the theoretical concepts and to provide an environment for hands-on experience for mobile security on the Android platform. Additionally, a short course project should give hands-on experience in extending Android's security architecture with a simple custom mechanism for access control enforcement.

content

- Security concepts and introduction to Android's security architecture
- Access control and permissions
- Role of Binder IPC in the security architecture
- Mandatory access control
- Compartmentalization
- Advanced attacks and problems

- SSL and WebViews
- Application-layer security extensions
- Smart Home IoT
- Hardware-based mobile platform security
- Course project: Security extension to the Android Open Source Project

literature & reading

The teaching material will be in English and it will consist of slides as well as book chapters.

Obfuscation

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**usually every year**	**1 semester**	**4**	**6**

responsible Dr. Nico Döttling

lecturers Dr. Nico Döttling

entrance requirements While there are no strict requirements to attend this course beyond being interested in the topic, having taking the core-lecture cryptography is recommended.

assessments / exams Passing a usually oral exam

course types / weekly hours

```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload

```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Determined by the performance in exams

language English

aims / competences to be developed

Obtain a fundamental understanding of how obfuscation can be defined and constructed using cryptographic notions and techniques. Study the mathematical structures and underlying hardness assumptions on which current obfuscation candidates are based.

content

In software design, obfuscation generally refers to various techniques which make computer code unintelligible, or make it hard to reverse engineer program code. Such techniques have been used for decades in an attempt to protect proprietary algorithms in commercial software. Unfortunately, commercially available obfuscation tools are typically broken within a very short time of their introduction.

From a scientific perspective, this raises the question whether the task of obfuscation is possible at all, or whether any conceivable obfuscation scheme can be broken. To approach this question, we first need to agree on a suitable notion of what it means to break an obfuscation scheme. This question was first addressed by a seminal work of Barak et al. (CRYPTO 2001) who considered several ways of defining security for obfuscation schemes.

In this course, we will take a comprehensive tour through the realm of cryptographically secure obfuscation. We will start by surveying the initial impossibility results, and see how they can be circumvented by weakening the security requirements in a meaningful way. We will proceed to show how obfuscation became a central hub of modern cryptography, on which essentially any advanced notion of proof systems and encryption can be based.

literature & reading

Parameterized Verification

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Dr. Swen Jacobs

lecturers Dr. Swen Jacobs

entrance requirements The course picks up on some of the topics of the core lecture ”Verification”, which is a recommended prerequisite for this course.

assessments / exams Passing a written exam (re-exam can be oral, if any)

course types / weekly hours

```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload

```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Determined by the performance in exams

language

aims / competences to be developed

The course is aimed at students interested in the theoretical concepts behind parameterized verification, which generalize system models, specification formalisms and proof methods from standard verification approaches.

content

We consider the problem of providing correctness and security guarantees for systems that scale with some parameter, e.g., the number of nodes in a network, the number of concurrent processes in a multi-threaded program, or the size of a data structure that a program operates on. Most systems are expected to scale in one or several parameters, but correctness and security guarantees are usually only given for fixed parameter values. In contrast, parameterized verification is the problem of obtaining correctness guarantees for all parameter values. In this course, we will look at methods for parameterized verification and investigate their capabilities and limitations.

literature & reading

The course is based on ”Decidability of Parameterized Verification” by Bloem et al., augmented with selected research papers.

Physical-Layer Security PHYSEC

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Dr. Nils-Ole Tippenhauer

lecturers Dr. Nils-Ole Tippenhauer

entrance requirements *Security* or *Foundations of Cyber Security I + II*

assessments / exams Übungen und schriftliche Abschlussklausur

course types / weekly hours
```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload
```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.

language English

aims / competences to be developed

- Classify and describe common physical-layer attacks and countermeasures
- Apply known side-channel attacks, e.g., simple power analysis
- Model, analyze, and simulate physical-layer attacks and defenses for wireless communications (e.g., eavesdropping, jamming, manipulation)
- Classify and describe countermeasures such as distance bounding protocols to prevent relay attacks
- Evaluate the security of existing cyber-physical systems against physical-layer attacks
- Classify and describe security issues and solutions for industrial control systems

content

The lecture will cover three main topic areas: attacks (and countermeasures) that leverage physical channels (e.g., side-channel attacks), attacks (and countermeasures) involving wireless communications (e.g., jamming, manipulation, and forwarding), and security for cyber-physical systems (such as industrial control systems).

Selected list of topics:

- Relay attacks
- Distance Bounding
- Physical-Layer Identification
- Wireless eavesdropping and manipulations
- GPS spoofing and countermeasures
- Industrial Control System security, attacks and countermeasures
- Security issues related to PLC logic applications, proprietary industrial protocols and end devices

literature & reading

The teaching material will be in English and will be announced at the beginning of the lecture.

additional information

While the lecture will touch physical-layer concepts such as (wireless) signal processing, no background in that area is assumed. Exercises will require students to run Linux applications (e.g., via a virtual machine).

Privacy-Enhancing Technologies PETS

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Dr. Wouter Lueks

lecturers Dr. Wouter Lueks

entrance requirements A basic understanding of security and cryptography (as taught for example in *Foundations of Cybersecurity 1* and *2* or *Security*) is essential to be able to follow the material in this course. A larger course in cryptography (for example the core lecture *Cryptography*) would help.

assessments / exams
- Programming projects
- Final exam (written or oral)
- Midterm

course types / weekly hours
```
  2 h lectures
+ 2 h tutorial and office hours
= 4 h (weekly)
```

total workload
```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Will be determined from performance in exams, exercises and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

Digital technologies have become an essential part of our day to day live. While often beneficial, these technologies also bring great privacy risks. In this course you will learn how to mitigate these risks by design privacy-friendly systems and how to evaluate the privacy-protections offered by systems.

To reason about the privacy of systems you will learn how to define desirable privacy properties and how to reason about privacy attackers. Privacy can be violated both at the application level (i.e., what data parties exchange) as well as on the meta-data level (i.e., how parties exchange data). You will learn about techniques to offer protection at both of these layers.

On the application layer, we'll discuss cryptographic techniques such as secure multi-party computation, homomorphic encryption and anonymous authentication that together can be used to ensure privacy at the application layer. We will also discuss data anonymisation techniques such as k-anonymity and differential privacy to enable privacy-friendly data publishing. On the meta-data level, we'll explore techniques for anonymous communication, censorship resistance, (browser) tracking and location privacy.

At the end of this course you will be able to:

- Explain basic building-blocks for designing privacy-friendly systems
- Combine these building blocks to solve simple problems while maintaining privacy
- Evaluate the privacy of simple proposed systems.

content

- Introduction to privacy
- Secure Multi-party computation

- (Fully) Homomorphic encryption
- Privacy-preserving authentication
- Anonymous communication
- Censorship resistance
- Protected data release and differential privacy
- Tracking
- Location Privacy

literature & reading

The teaching material will be in English and it will be announced at the beginning of the lecture.

Reactive Synthesis RS

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Dr. Swen Jacobs

lecturers Dr. Swen Jacobs

entrance requirements *Grundzüge der Theoretischen Informatik*

assessments / exams Projekt und schriftliche Abschlussklausur

course types / weekly hours
```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload
```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.

language English

aims / competences to be developed

Students will gain an understanding of reactive synthesis in its full breadth, ranging from its theoretical formalization as an infinite game to efficient algorithms and data structures to solve the synthesis problem, and in the implementation of state-of-the-art algorithms for practically relevant and challenging problems.

content

- State of the art in reactive synthesis
- Formalization of reactive synthesis problems as an infinite game
- Different types of infinite games
- Solving infinite games
- Efficient algorithms and data structures for solving games
- Implementation of reactive synthesis tools/game solvers

literature & reading

The teaching material will be in English and it will be announced at the beginning of the lecture.

Recht der Cybersicherheit – Datenschutzrechtliche Aspekte RdC-DS

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**i.d.R. jedes Wintersemester**	**1 Semester**	**4**	**6**

responsible Prof. Dr. Christoph Sorge

lecturers Prof. Dr. Christoph Sorge

entrance requirements Keine

assessments / exams Abschlussklausur bzw. mündliche (Nach-)Prüfung

course types / weekly hours
```
  2 h Vorlesungen
+ 2 h Übungen
= 4 h (wöchentlich)
```

total workload
```
   60 h Präsenzstudium
+ 120 h Eigenstudium
= 180 h (= 6 ECTS)
```

grade Wird aus Leistung in Abschlussklausur bzw. Nachprüfung ermittelt.

language i.d.R. Deutsch; wird zu Beginn der Veranstaltung bekannt gegeben

aims / competences to be developed

- Erarbeitung grundlegender juristischer Methodenkenntnisse, daraus ableitend grundlegende Befähigung sich weiteres juristisches Grundlagenwissen mit Hilfe von Literatur anzueignen
- Vermittlung von Kenntnissen in rechtlichen Teilbereichen, schwerpunktmäßig im Datenschutzrecht, aber auch von einzelnen Aspekten des Urheber-, Patent- und IT-Sicherheitsrechts

content

- Grundlagen juristischer Methodik
- Einführung in das europäische Datenschutzrecht
- Grundlagen des IT-Sicherheitsrechts
- Grundlagen des Urheber- und Patentrechts

literature & reading

Bekanntgabe im Rahmen der Vorlesung, sowie auf der Website der Vorlesung.

Recht der Cybersicherheit – Strafrechtliche Aspekte RdC-SR

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**i.d.R. jedes Sommersemester**	**1 Semester**	**4**	**6**

responsible Dr. Stephanie Vogelgesang

lecturers Dr. Stephanie Vogelgesang

entrance requirements Keine

assessments / exams Abschlussklausur bzw. mündliche (Nach-)Prüfung

course types / weekly hours
```
  2 h Vorlesungen
+ 2 h Übungen
= 4 h (wöchentlich)
```

total workload
```
   60 h Präsenzstudium
+ 120 h Eigenstudium
= 180 h (= 6 ECTS)
```

grade Wird aus Leistung in Abschlussklausur bzw. Nachprüfung ermittelt.

language i.d.R. Deutsch; wird zu Beginn der Veranstaltung bekannt gegeben

aims / competences to be developed

Die Vorlesung soll Informatikern und Studierenden verwandter Fächer einen Einblick in das juristische Denken und Arbeiten geben. Neben allgemeinen Konzepten werden exemplarisch Rechtsgebiete, die für berufliche Tätigkeiten im Bereich Cybersicherheit besonders relevant sein dürften, behandelt.

Die Vorlesung dient auch der Umsetzung des Anspruchs, den die Gesellschaft für Informatik in ihren ethischen Leitlinien formuliert: „Vom Mitglied wird erwartet, dass es die einschlägigen rechtlichen Regelungen kennt, einhält und gegebenenfalls an ihrer Fortschreibung mitwirkt." Sie hat hingegen nicht den Anspruch, den Besuch von Rechtsvorlesungen zu ersetzen (etwa im Nebenfach Rechtsinformatik). Sie kann jedoch auch aufzeigen, welche Rechtsgebiete für eine Vertiefung von Interesse sein könnten und wann es sich in der Praxis lohnt oder angebracht ist, sich einen Rechtsbeistand zu besorgen.

Nach einer allgemeineren Einführung wird ein umfassender Einblick in das Strafrecht vermittelt. Neben allgemeinen strafrechtlichen Normen werden insbesondere Delikte des sogenannten „Cyberstrafrechts" betrachtet. Dabei wird ein Teil der Veranstaltung die spezifisch strafrechtliche Bewertung von Cyberangriffen darstellen. Abschließend wird das Strafprozessrecht beleuchtet (u.a. Aspekte der Beschlagnahmung und Durchsuchung).

content

- Überblick über Rechtsgebiete
- Grundlagen juristischer Methodik
- Einführung in das Strafrecht und Strafprozessrecht
- Überblick über Cyberangriffe sowie deren strafrechtliche Bewertung

literature & reading

Bekanntgabe im Rahmen der Vorlesung, sowie auf der Website der Vorlesung.

Reverse Engineering and Exploit Development for Embedded Systems RExp4Em-Sys

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**2 weeks**	**BLOCK**	**6**

responsible Dr. Ali Abbasi

lecturers Dr. Ali Abbasi

entrance requirements An extensive background on software security and a background on embedded systems are recommended.

assessments / exams
- Regular attendance at classes and tutorials.
- Successful completion of a course final project (Project due approximately 2 weeks).
- Score at least 50% on the final oral exam (which is based on your final project).
- To be admitted to the exam, you must achieve at least 50% of the points from the exercises.

course types / weekly hours Daily lectures followed by daily tutorial and excercises

total workload
```
  60 h of lectures and exercises
+ 120 h project work
= 180 h (= 6 ECTS)
```

grade Will be determined from performance in oral exam, exercise tasks and final project. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

In this course, we will work toward understanding the fundamentals of developing software/hardware exploits against embedded systems. We will cover topics such as firmware extraction modification, and different hardware serial protocols. We also cover topics such as exploit development for embedded devices and write exploits for vulnerabilities such as uninitialized stack variables, off-by-one bugs, Use-after-free, and utilize techniques such as ROP, Signal-oriented programming, to attack embedded systems. We also attack micro-controllers and try to extract secrets from them by utilizing reverse-engineering techniques and firmware patching. Finally, we perform fuzz-testing on embedded firmware via re-hosting.

content

1. Software security vulnerabilities in embedded systems 1
2. Software security vulnerabilities in embedded systems 2
3. Software security vulnerabilities in embedded systems 3
4. Introduction to Firmware and Peripheral Register Configuration
5. Embedded Hardware Peripherals
6. Binary and Firmware Emulation
7. Ghidra and Reverse Engineering
8. Fuzzing and Firmware Patching
9. Fuzzing 2
10. Real-World Firmware Exploitation and Project QA

literature & reading

Will be announced before the start of the course on the course page on the Internet.

Secure Web Development — SecWebDev

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Dr. Nils-Ole Tippenhauer

lecturers Dr. Nils-Ole Tippenhauer

entrance requirements none

assessments / exams Projekt und schriftliche Abschlussklausur

course types / weekly hours

```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload

```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Das Modul ist insgesamt bestanden, wenn die Prüfungsleistung bestanden wurde.

language English

aims / competences to be developed

Students will learn principles, best-practices, and tools to build secure web applications. Also, Students will acquire deep understanding of existing vulnerabilities and security threats.

content

- Basics on secure software engineering and development life-cycle
- Architecture of modern web application
- Secure coding and coding patterns
- Security of the HTTP message processing pipeline
- Known threats and vulnerabilities
- (Mini) BiBiFi challenges (Build it, Break it, Fix it)

literature & reading

Teaching material and notes will be in English and announced at the beginning of the lecture.

additional information

Given the limited resources available for this lecture, the course is limited to 20 seats.

Side-Channels Attacks & Defenses — SCAD

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**occasional**	**1 semester**	**4**	**6**

responsible Dr. Michael Schwarz

lecturers Dr. Michael Schwarz

entrance requirements A background in the basics of operating systems and in programming C is recommended

assessments / exams project and written exam

course types / weekly hours
```
  2 h lectures
+ 2 h tutorial
= 4 h (weekly)
```

total workload
```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Will be determined from performance in exams, exercises, and practical tasks. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

Students will acquire both a theoretical and practical understanding of microarchitectural attacks, such as side-channel attacks, transient-execution attacks, and software-based fault attacks. The students will understand the attack surface for these types of attacks and learn how such attacks can be mitigated on the hardware, operating system, and software layer. Moreover, students will acquire a more in-depth understanding of how modern CPUs work internally.

The lectures are accompanied by exercises to apply the theoretical concepts in a practical setting and get hands-on experiences with side-channel attacks and their mitigations.

content

- Basic introduction to the CPU microarchitecture and side channels
- Software-based side-channel attacks (e.g., cache attacks, timing attacks)
- Trusted execution environments and their attack surface (e.g., controlled-channel attacks)
- Transient execution attacks (e.g., Meltdown, Spectre, ZombieLoad)
- Software-based fault attacks (e.g., Rowhammer, Plundervolt)
- Overview of various other types of side channels
- Mitigation strategies in software and hardware

literature & reading

The teaching material will be in English and it will be announced at the beginning of the lecture.

Usable Security and Privacy — USP

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
1-3	**4**	**every summer semester**	**1 semester**	**4**	**6**

responsible Dr. Katharina Krombholz

lecturers Dr. Katharina Krombholz

entrance requirements *Foundations of Cybersecurity 1* and *2* or the core lecture *Security* and knowledge in statistics are highly recommended. A deep understanding of the topics covered in these lectures as well as a good understanding of statistics is necessary.

assessments / exams

- Regular attendance of classes and tutorials & office hours.
- Assignments during the semester
- Final exam
- A re-exam takes place before the start of lectures in the following semester for students who failed in the final exam or did not participate in the final exam

course types / weekly hours

```
  2 h lectures
+ 2 h tutorial or office hours
= 4 h (weekly)
```

total workload

```
   60 h of classes
+ 120 h private study
= 180 h (= 6 ECTS)
```

grade Will be determined from performance in the exam and the exercises. The exact modalities will be announced at the beginning of the module.

language English

aims / competences to be developed

In this lecture, students will learn about human-centric aspects of IT security. Besides research and design methods, students will learn about hot topics in usable security such as authentication, confidentiality and privacy. In particular, they will learn to

- design user studies to study how humans interact with security & privacy technology with respect to threat models,
- collect, understand, evaluate qualitative & quantitative data,
- interpret results and draw conclusions based on your data,
- design new security and privacy technology that is better tied to the users' needs and values.

content

- Qualitative Research Methods
- Quantitative Research Methods
- Statistics
- User Study Design and Ethics
- Design Methods
- Surveillance
- Privacy
- Authentication
- Encryption

literature & reading

Will be announced before the start of the course on the course page on the internet.

additional information

Occasionally, this course is offered as a three week block course before the lectures of the summer semester. In this case, lecture and tutorial distribution as well as examination will vary and be communicated with the students on the course page on the internet. This module was formerly also known as *Usable Security*.

Module Category 3

Seminar Cybersecurity

Seminar

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
2	**4**	**jedes Semester**	**1 Semester**	**2**	**7**

responsible Studiendekan der Fakultät Mathematik und Informatik
Studienbeauftragter der Informatik

lecturers Dozent/inn/en der Fachrichtung

entrance requirements Grundlegende Kenntnisse im jeweiligen Teilbereich des Studienganges.

assessments / exams
- Thematischer Vortrag mit anschließender Diskussion
- Aktive Teilnahme an der Diskussion
- Gegebenenfalls schriftliche Ausarbeitung oder Projekt

course types / weekly hours `2 SWS Seminar`

total workload
```
  30 h Präsenzstudium
+ 180 h Eigenstudium
= 210 h (= 7 ECTS)
```

grade Wird aus den Leistungen im Vortrag und der schriftlichen Ausarbeitung und/oder dem Seminarprojekt ermittelt. Die genauen Modalitäten werden von dem/der jeweiligen Dozenten/in bekannt gegeben.

language Deutsch oder Englisch

aims / competences to be developed

Die Studierenden haben am Ende der Veranstaltung vor allem ein tiefes Verständnis aktueller oder fundamentaler Aspekte eines spezifischen Teilbereiches der Informatik erlangt.

Sie haben weitere Kompetenz im eigenständigen wissenschaftlichen Recherchieren, Einordnen, Zusammenfassen, Diskutieren, Kritisieren und Präsentieren von wissenschaftlichen Erkenntnissen gewonnen.

content

Weitgehend selbstständiges Erarbeiten des Seminarthemas:

- Lesen und Verstehen wissenschaftlicher Arbeiten
- Analyse und Bewertung wissenschaftlicher Aufsätze
- Diskutieren der Arbeiten in der Gruppe
- Analysieren, Zusammenfassen und Wiedergeben des spezifischen Themas
- Erarbeiten gemeinsamer Standards für wissenschaftliches Arbeit
- Präsentationstechnik

Spezifische Vertiefung in Bezug auf das individuelle Thema des Seminars.

Der typische Ablauf eines Seminars ist üblicherweise wie folgt:

- Vorbereitende Gespräche zur Themenauswahl
- Regelmäßige Treffen mit Diskussion ausgewählter Beiträge
- ggf. Bearbeitung eines themenbegleitenden Projekts
- Vortrag und ggf. Ausarbeitung zu einem der Beiträge

literature & reading

Material wird dem Thema entsprechend ausgewählt.

additional information

Die jeweils zur Verfügung stehenden Seminare werden vor Beginn des Semesters angekündigt und unterscheiden sich je nach Studiengang.

Module Category 4

Master Seminar and Thesis

Master Seminar

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
3	**4**	**every semester**	**1 semester**	**2**	**12**

responsible Dean of Studies of the Faculty of Mathematics and Computer Science
Study representative of computer science

lecturers Professors of the department

entrance requirements Acquisition of at least 30 CP

assessments / exams
- Preparation of the relevant scientific literature
- Written elaboration of the topic of the master thesis
- Presentation about the planned topic with subsequent discussion
- Active participation in the discussion

course types / weekly hours `2 h seminar (weekly)`

total workload
```
 30 h seminar
+ 40 h contact with supervisor
+ 290 h private study
= 360 h (= 12 ECTS)
```

grade graded

language English or German

aims / competences to be developed

The Master seminar sets the ground for carrying out independent research within the context of an appropriately demanding research area. This area provides sufficient room for developing own scientific ideas.

At the end of the Master seminar, the basics ingredients needed to embark on a successful Master thesis project have been explored and discussed with peers, and the main scientific solution techniques are established.

The Master seminar thus prepares the topic of the Master thesis. It does so while deepening the students' capabilities to perform a scientific discourse. These capabilities are practiced by active participation in a reading group. This reading group explores and discusses scientifically demanding topics of a coherent subject area.

content

The methods of computer science are systematically applied, on the basis of the "state-of-the-art".

literature & reading

Scientific articles corresponding to the topic area in close consultation with the lecturer.

Master Thesis

st. semester	std. st. sem.	cycle	duration	SWS	ECTS
4	**4**	**every semester**	**6 months**	**-**	**30**

responsible Dean of Studies of the Faculty of Mathematics and Computer Science
Study representative of computer science

lecturers Professors of the department

entrance requirements Successful completion of the *Master Seminar*

assessments / exams Written elaboration in form of a scientific paper. It describes the scientific findings as well as the way leading to these findings. It contains justifications for decisions regarding chosen methods for the thesis and discarded alternatives. The student's own substantial contribution to the achieved results has to be evident. In addition, the student presents his work in a colloquium, in which the scientific quality and the scientific independence of his achievements are evaluated.

course types / weekly hours none

total workload
```
  50 h contact with supervisor
+ 850 h private study
= 900 h (= 30 ECTS)
```

grade Grading of the Master Thesis

language English or German

aims / competences to be developed

In the master thesis the student demonstrates his ability to perform independent scientific work focusing on an adequately challenging topic prepared in the master seminar.

content

In the master thesis the student demonstrates his ability to perform independent scientific work focusing on an adequately challenging topic prepared in the master seminar.

literature & reading

According to the topic